ModelArts

故障排除

文档版本 01

发布日期 2025-11-18

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。如没有权限，请参见**在Notebook中，如何访问其他账号的OBS桶？**。

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

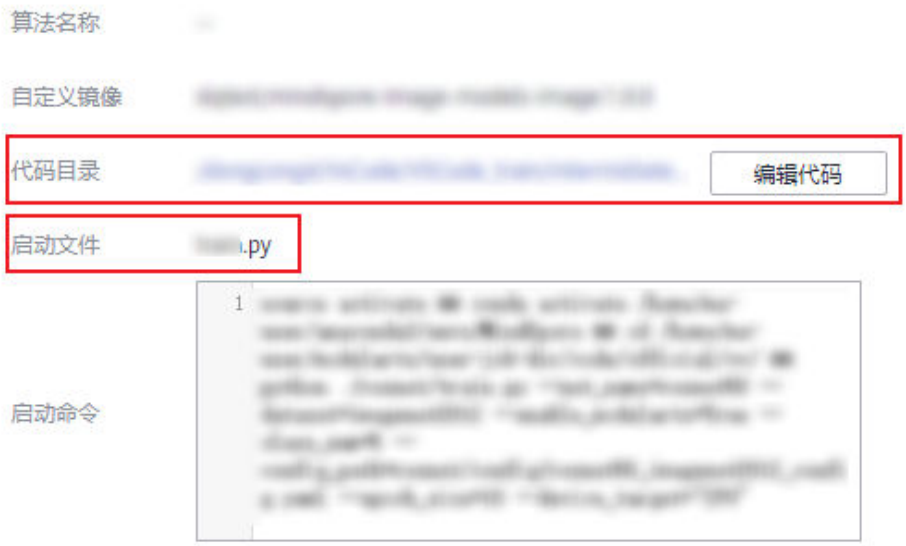
1. 进入OBS管理控制台，查找对应的OBS桶，单击桶名称进入概览页。
2. 在左侧菜单栏选择“访问权限控制>桶ACLs”，检查当前账号是否具备读写权限，如果没有权限，请联系桶的拥有者配置权限。
3. 在左侧菜单栏选择“访问权限控制>桶策略”，检查当前OBS桶是否允许子用户访问。

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。

1.2 ModelArts 中提示 ModelArts.7211: 账号已受限

问题现象

在ModelArts控制台提示“ModelArts.7211: 账号已受限”。

原因分析

可能是账号欠费导致。

处理办法

查看华为云账号是否欠费，可以通过以下步骤进行：

1. 登录华为云官网的**费用中心**。
2. 在“总览”页面，您可以查看可用额度。如果可用额度为负数，则表示账号存在欠费，建议充值。

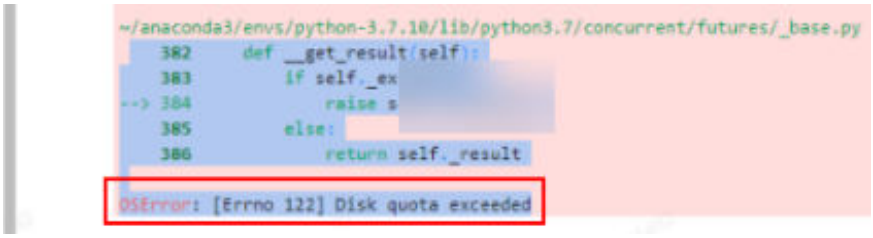
2 开发环境

2.1 环境配置故障

2.1.1 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。

2.1.2 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。

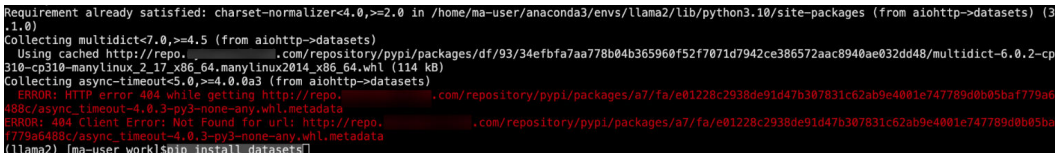
```
!pip install keras==2.3.1
```

```
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
```

2.1.3 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
```

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

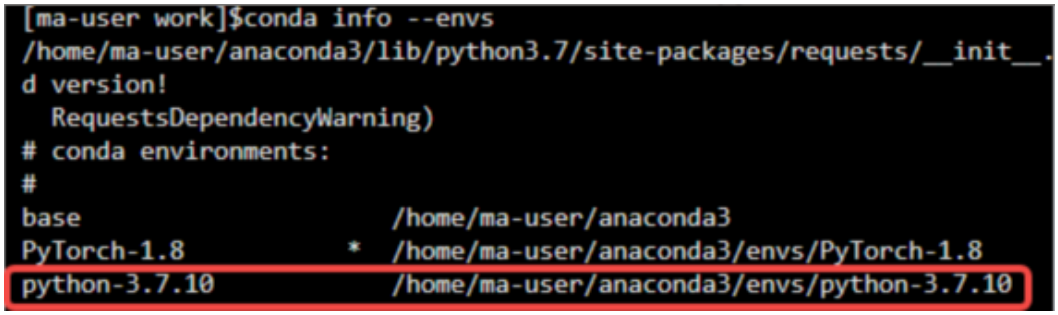
问题现象

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2-1所示。

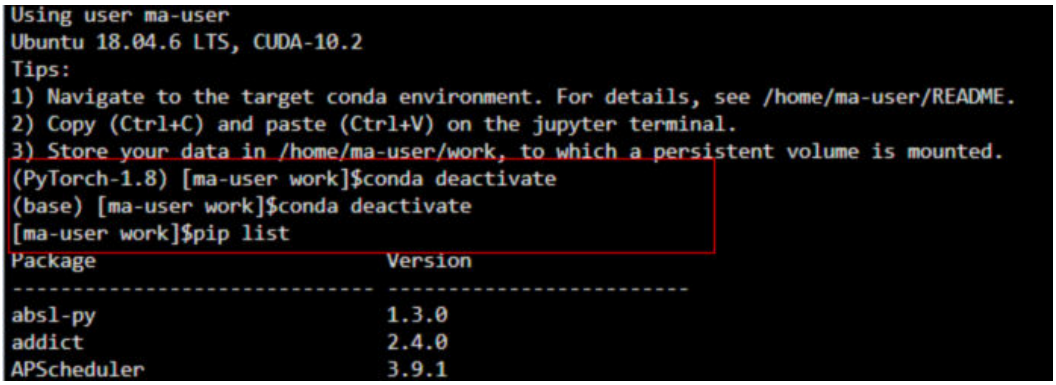
图 2-1 查询创建的虚拟环境



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。

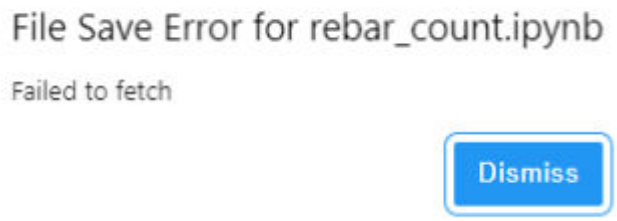
图 2-2 命令示例



2.1.5 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。

2.1.6 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 2-3 报错 Server Connection Error 截图

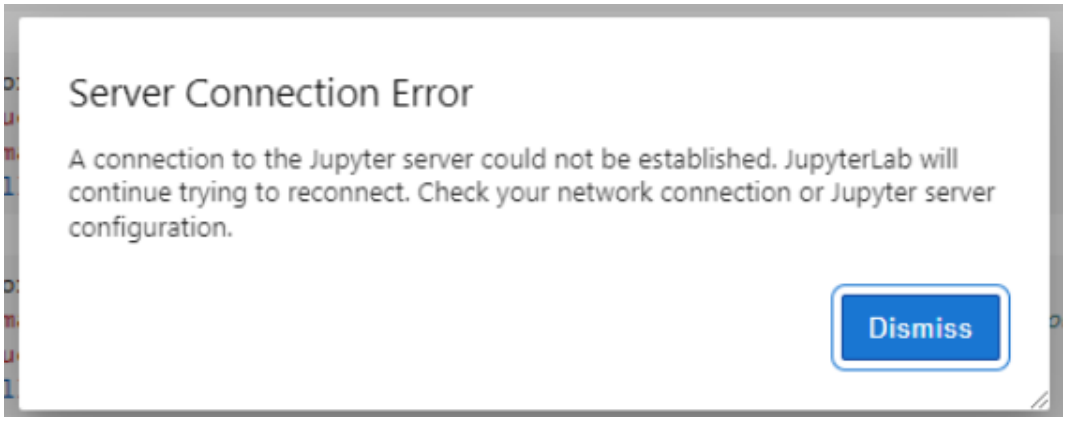
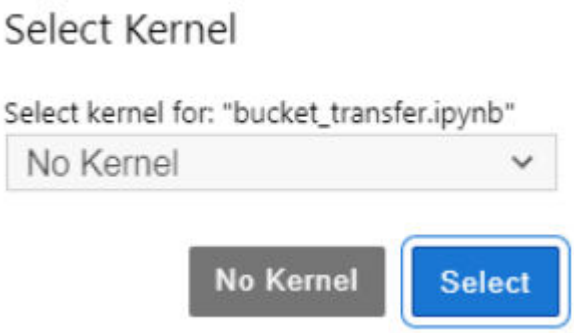


图 2-4 选不到 Kernel



原因分析

用户误操作引起的。

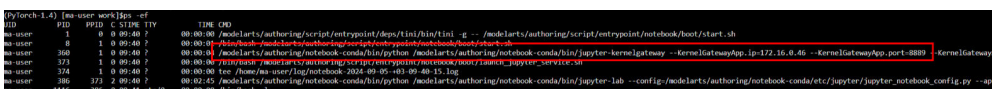
解决方案

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"

${CONDA_DIR}/bin/jupyter kernelgateway --KernelGatewayApp.ip=${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 2-5 检查进程是否启动



2.2 实例故障

2.2.1 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。

图 2-6 查看事件

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。具体操作，请参见**Notebook的自定义镜像制作方法**。

2.2.2 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

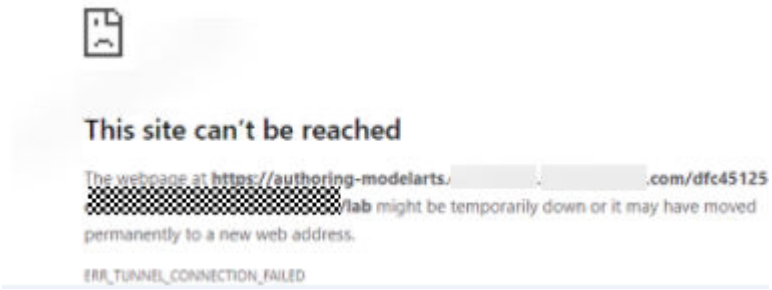
如果报此错误时，请提工单或拨打热线电话协助解决。提工单和热线电话请参见：**https://www.huaweicloud.com/service/contact.html**。

报错 500

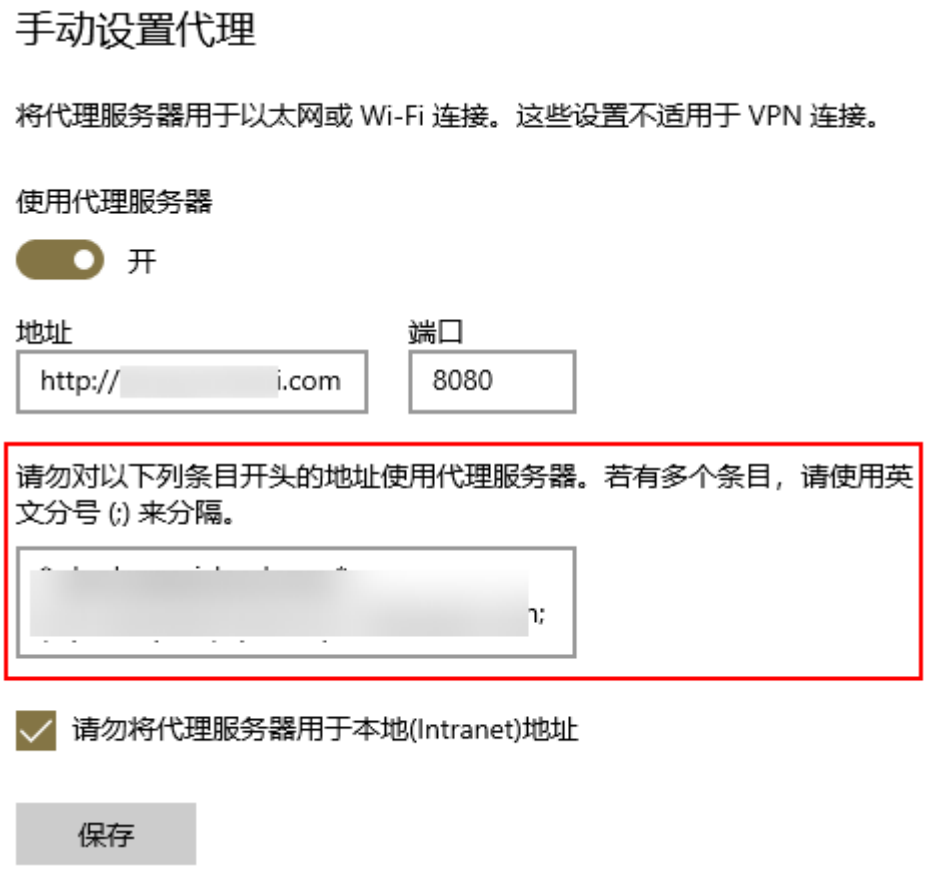
Notebook JupyterLab页面无法打开，报错500，可能是工作目录work下的磁盘空间满了，请参考**Notebook提示磁盘空间已满**排查并清理磁盘空间。

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。



2.2.3 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。

2.2.4 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。

2.2.5 单击 Notebook 的打开按钮时报“请求超时”错误?

问题现象

单击Notebook的打开按钮时出现“请求超时”错误。

原因分析

当Notebook容器因内存溢出等原因导致崩溃时，如果此时单击Notebook的打开按钮时，将会出现“请求超时”错误。

处理方法

请耐心等待容器恢复，约几十秒，再重新单击打开按钮即可。

2.2.6 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。

2.2.7 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末尾加上Notebook实例的token。出于安全考虑，不建议通过此方式传播Notebook实例，防止实例被恶意利用。

URL参考示例：https://example.com/11136b81-4da9-49f3-a2c4-a41434f*****/lab?token=****

2.3 代码运行故障

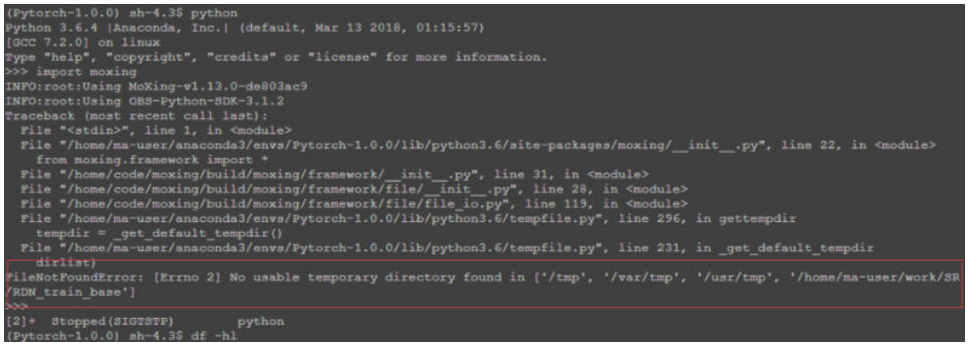
2.3.1 Notebook 运行代码报错，在'/tmp'中找不到文件

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 2-7 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
   ```

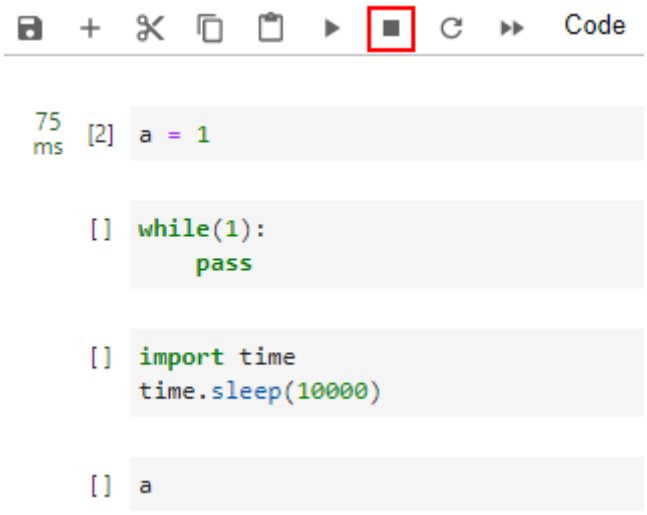
2. 请删除不用的大文件。
   a. 删除示例文件“test.txt”： **rm -f /home/ma-user/work/data/test.txt**
   b. 删除示例文件夹“data”： **rm -rf /home/ma-user/work/data/**

2.3.2 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2-8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 2-8 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。

2.3.3 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。

2.3.4 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。

2.3.5 开发环境提示空间不足，如何解决?

问题现象

开发环境提示空间不足。

原因分析

开发环境空间不足。

解决方法

当提示空间不足时，推荐使用EVS类型的Notebook实例。

参考**如何在Notebook中上传下载OBS文件?** 操作指导，针对原有的Notebook，首先将代码和数据上传至OBS桶中。然后创建一个EVS类型的Notebook，将此OBS中的文件下载至Notebook本地（指新建的EVS类型Notebook）。

2.3.6 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()
```

2.3.7 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
```

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

问题现象

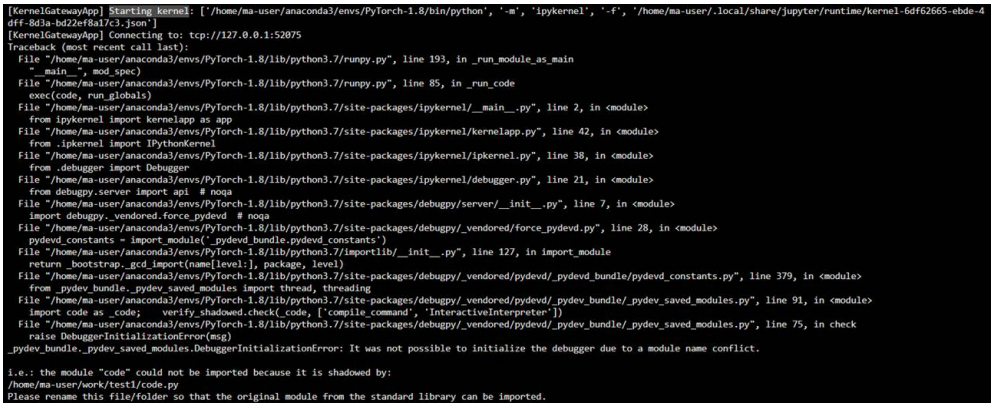
现象：创建Notebook文件后，右上角的Kernel状态为“No Kernel”。

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：



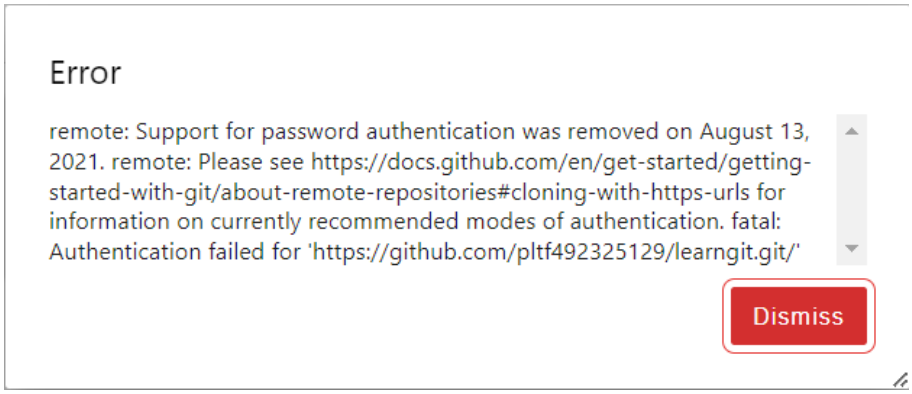
2. 重命名当前工作目录下和创建kernel依赖的库文件冲突的文件名称。
   常见容易冲突的文件：code.py、select.py。

2.4 JupyterLab 插件故障

2.4.1 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：

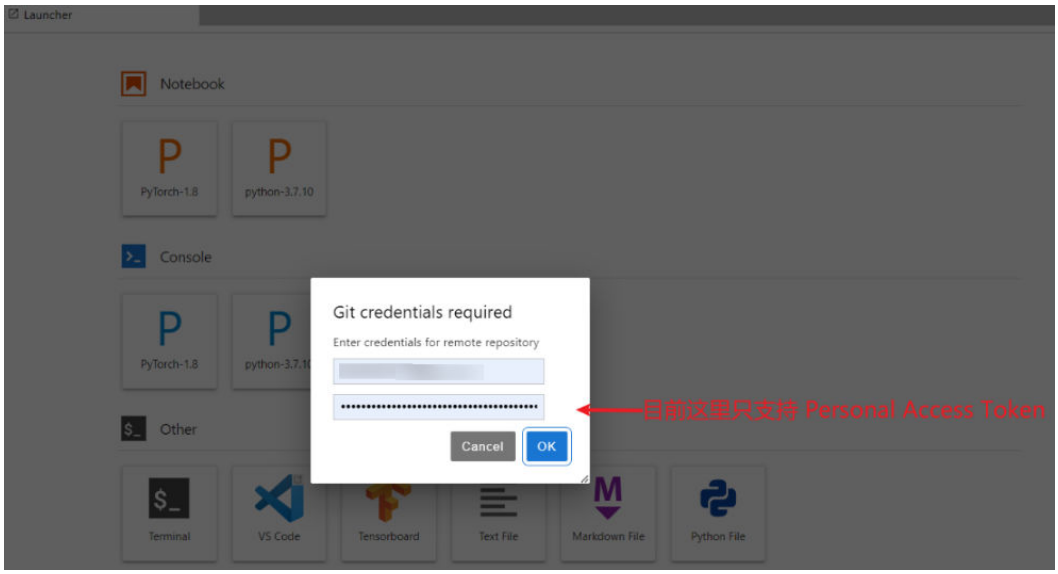


原因分析

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**。



2.5 VS Code 连接开发环境失败故障处理

2.5.1 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

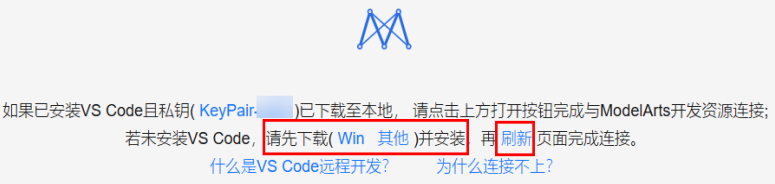
原因分析

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。更多信息，请参见**VS Code一键连接Notebook**。

图 2-9 VS Code



2.5.2 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

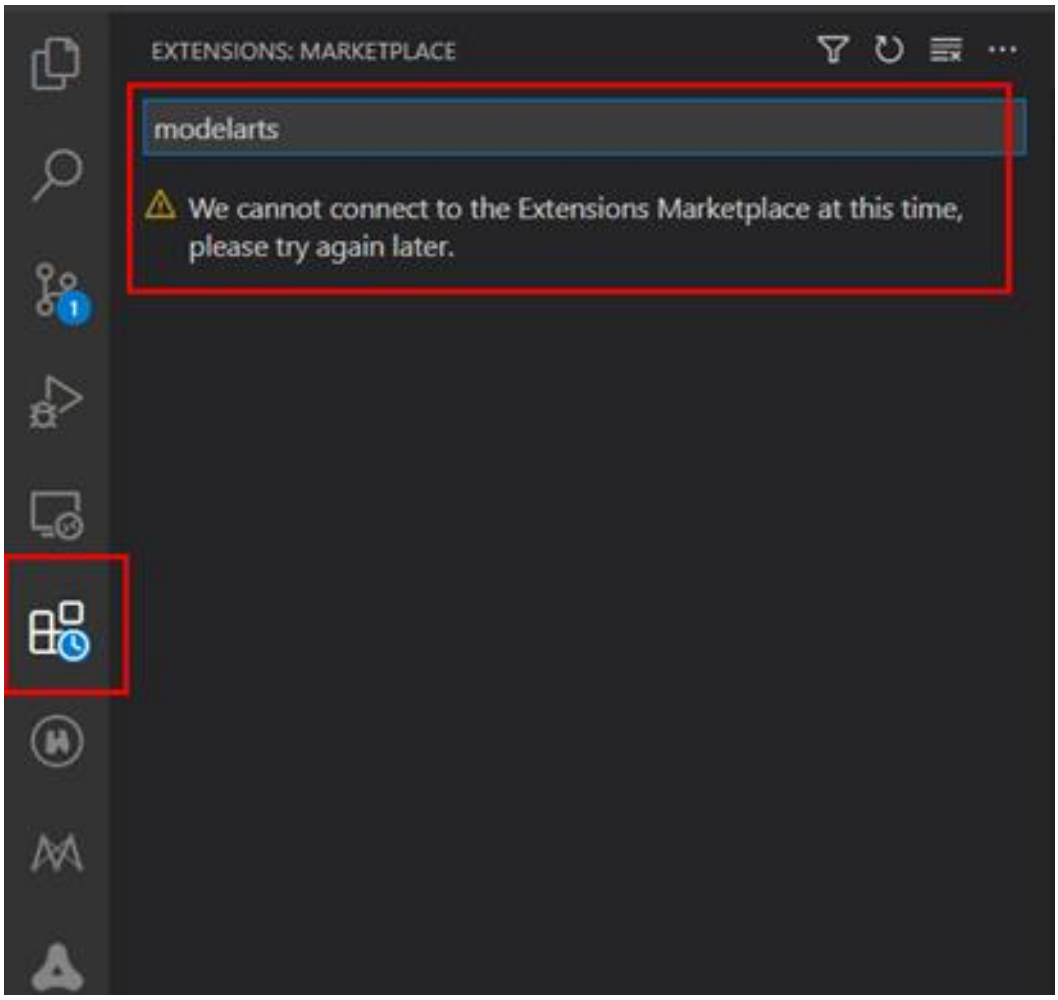
如果本地为Linux系统，见原因分析二。

原因分析一

自动安装VS Code插件ModelArts-HuaweiCloud失败。

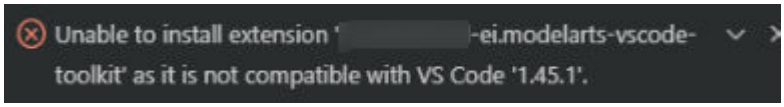
解决方法一

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。



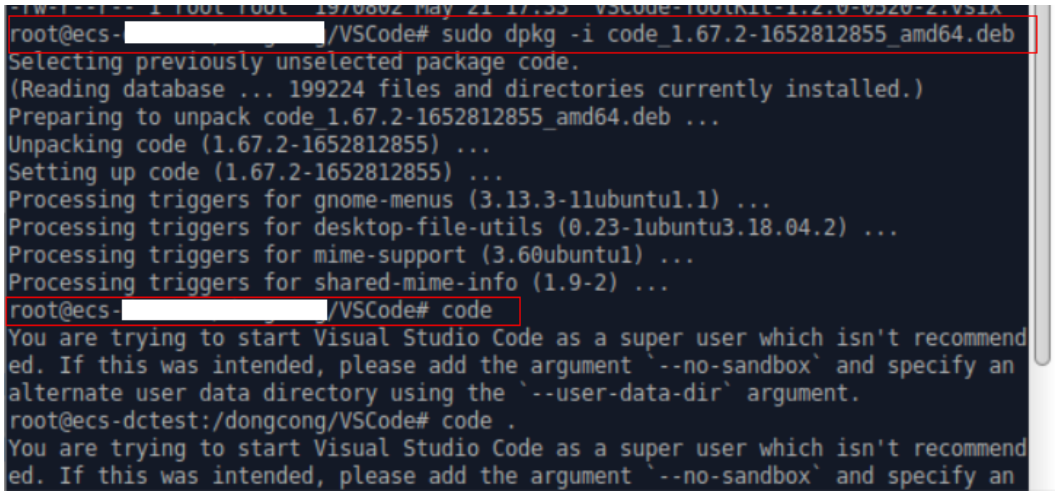
操作完成后再次执行搜索，如果显示如下则网络正常，请回到ModelArts控制台界面再次单击界面上的“VS Code接入”按钮。

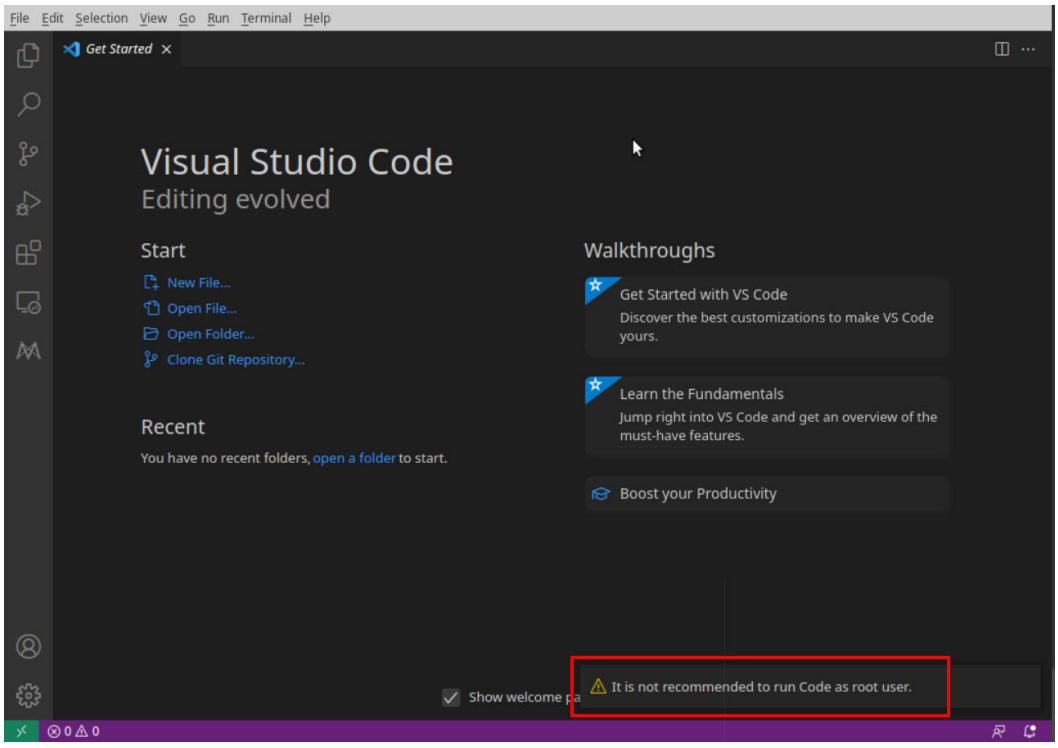
方法二：出现如下图报错，是由于VS Code版本过低，建议升级VS Code版本为1.57.1或者最新版。



原因分析二

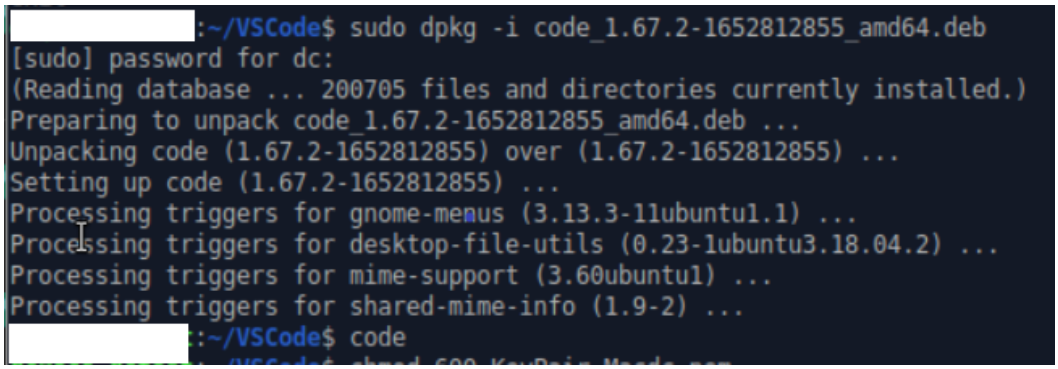
本地系统为Linux，由于使用root用户安装VS Code，打开VS Code显示信息It is not recommended to run Code as root user

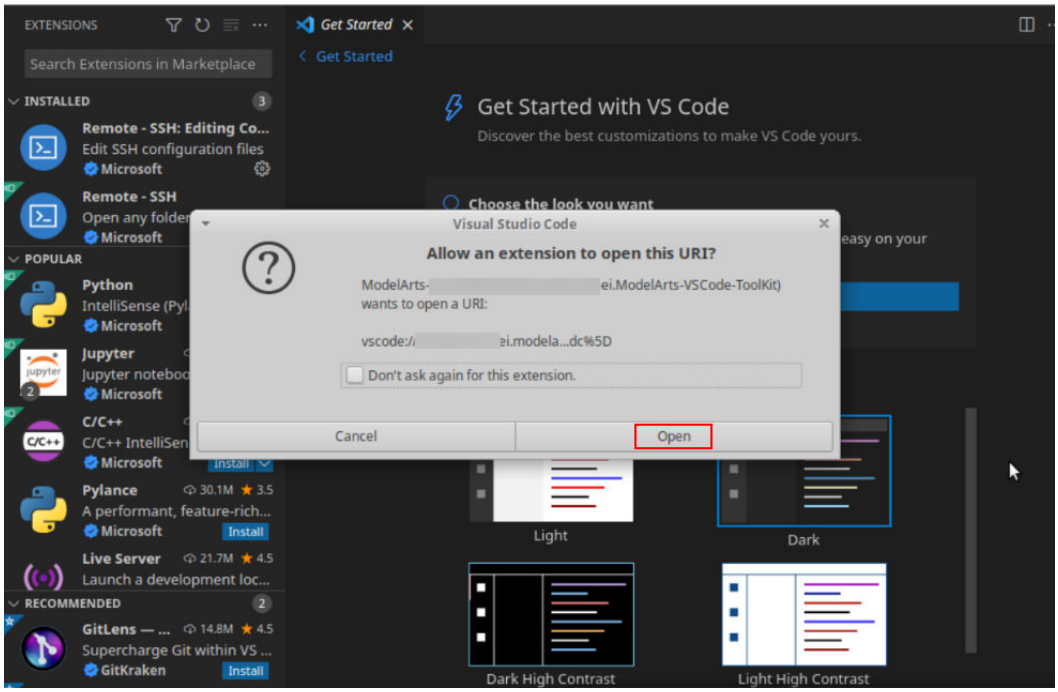




解决方法二

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。





2.5.3 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。

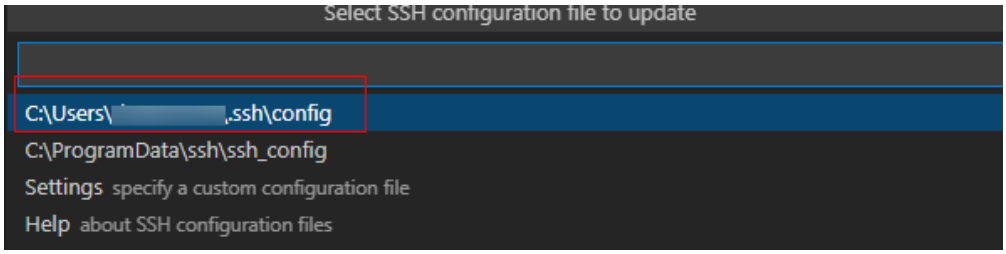
远端排查

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

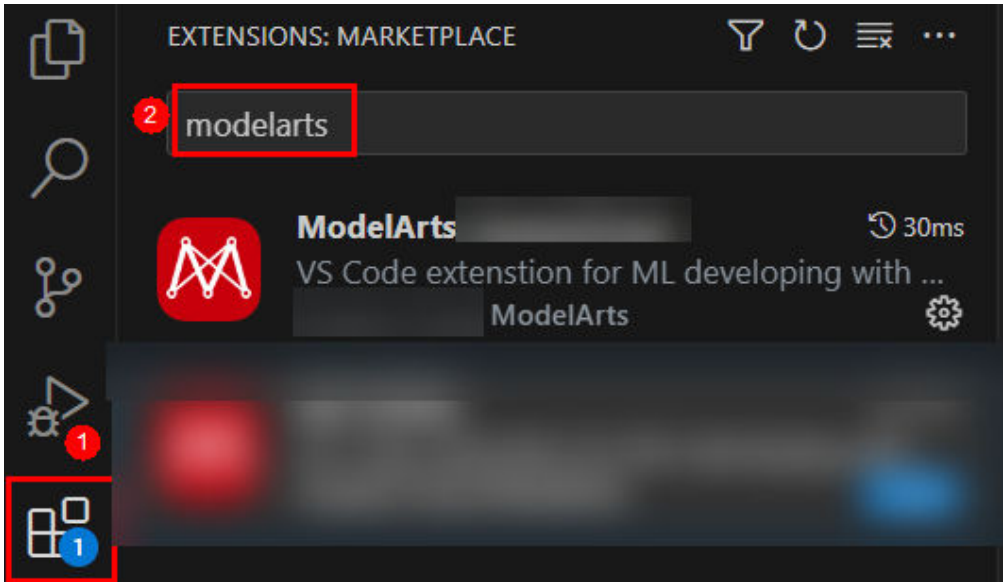
1. 检查配置是否正确。

   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
```

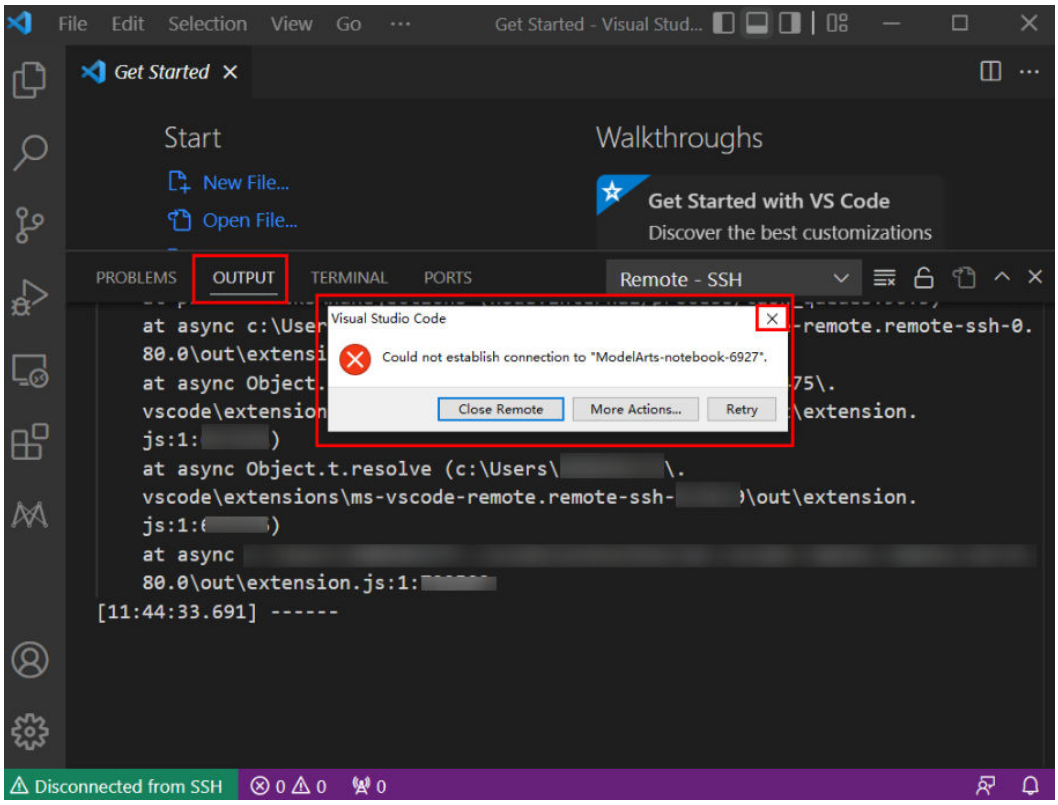
   - 如果正确请按继续排查。
   - 如果不正确请按上面格式修改后继续排查。
2. 查看密钥文件的路径，建议放在C:\Users\{user}\.ssh下，并确保密钥文件无中文字符。
3. 排查插件包是否为最新版：在extensions中搜索，看是否需要升级。检查Remote-SSH三方插件是否兼容。



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。

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

问题现象



原因分析

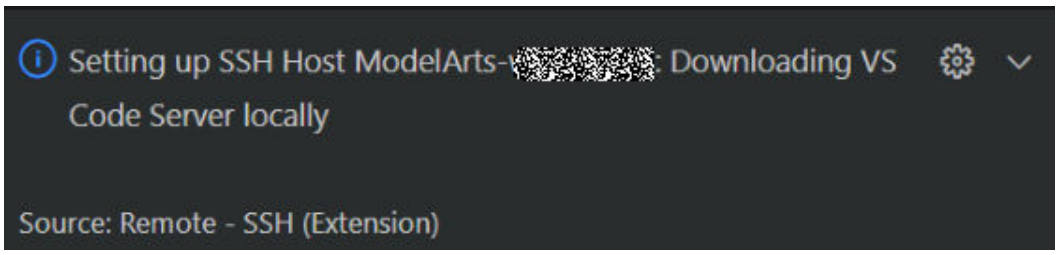
执行VS Code Remote SSH连接失败。

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。

2.5.5 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

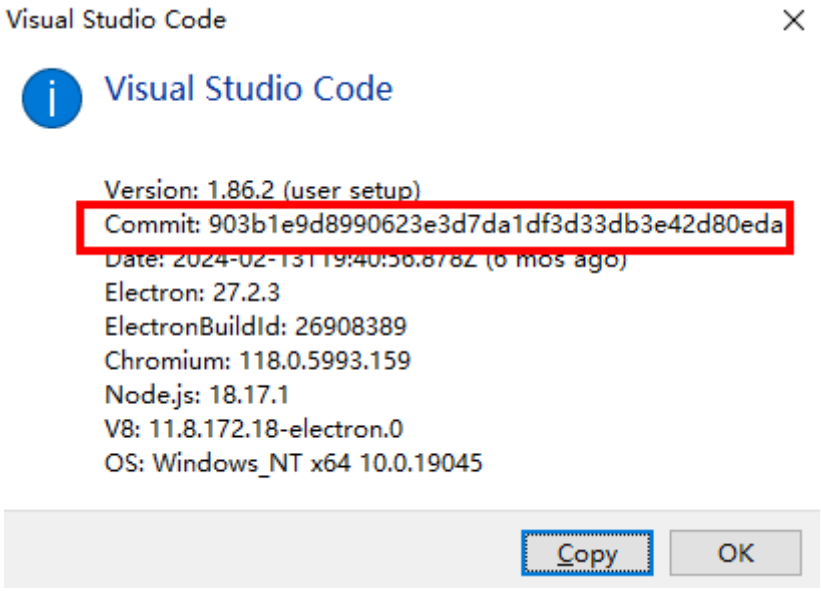


原因分析

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”，请下载其他版本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AArch64的，通过下面的链接，手动修改commit-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）

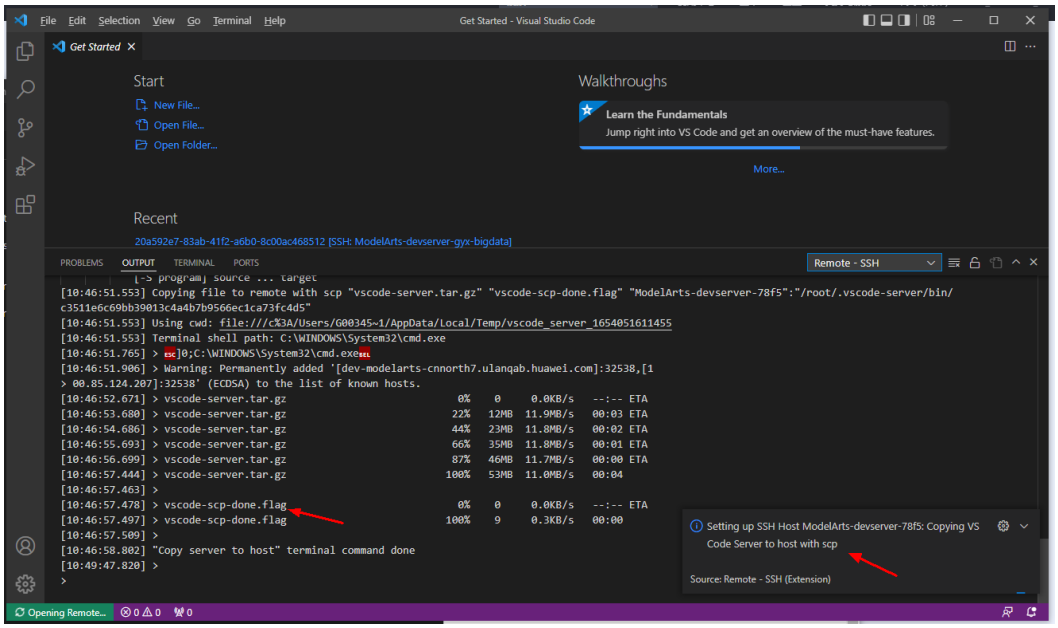
   ```
   commitId=<提交的ID码>
   mkdir -p /home/ma-user/.vscode-server/bin/$commitId
   tar -zxvf vscode-server-linux-x64.tar.gz -C /home/ma-user/.vscode-server/bin/$commitId  --strip=1
   chmod 750 -R /home/ma-user/.vscode-server/bin/$commitId
   ```

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）。

2.5.6 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

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。



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

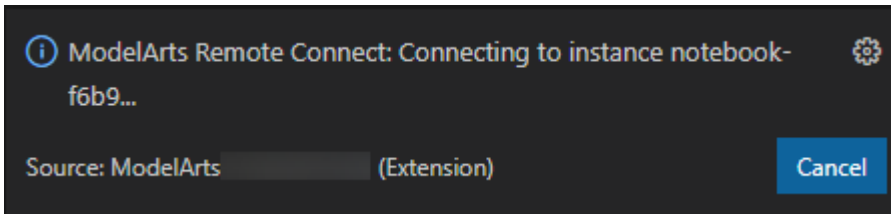
关闭VS Code所有窗口后，回到ModelArts控制台界面再次单击界面上的“VS Code接入”按钮。

2.5.7 连接远端开发环境时，一直处于"ModelArts Remote Connect: Connecting to instance xxx..."超过 10 分钟以上，如何解决?

问题现象

连接远端开发环境时，一直处于"ModelArts Remote Connect: Connecting to instance xxx..."超过10分钟以上。

报错示例如下：

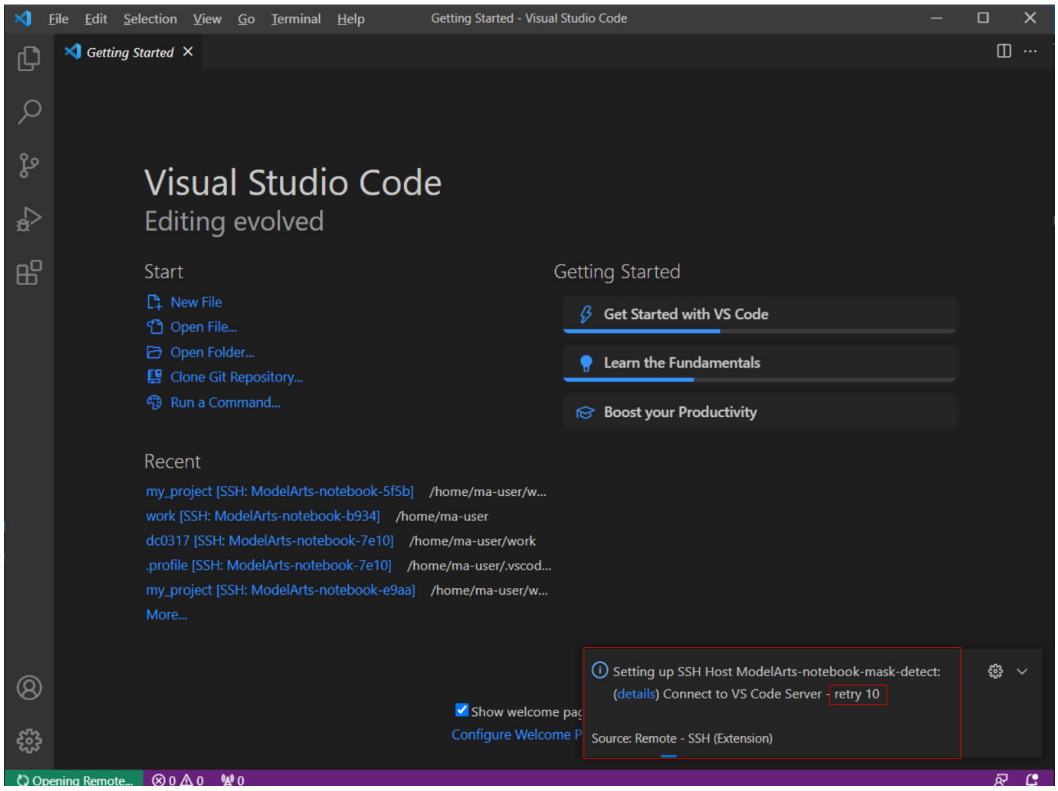


解决方法

单击“Cancel”，并回到ModelArts控制台界面再次单击界面上的“VS Code接入”按钮。

2.5.8 远程连接处于 retry 状态如何解决?

问题现象



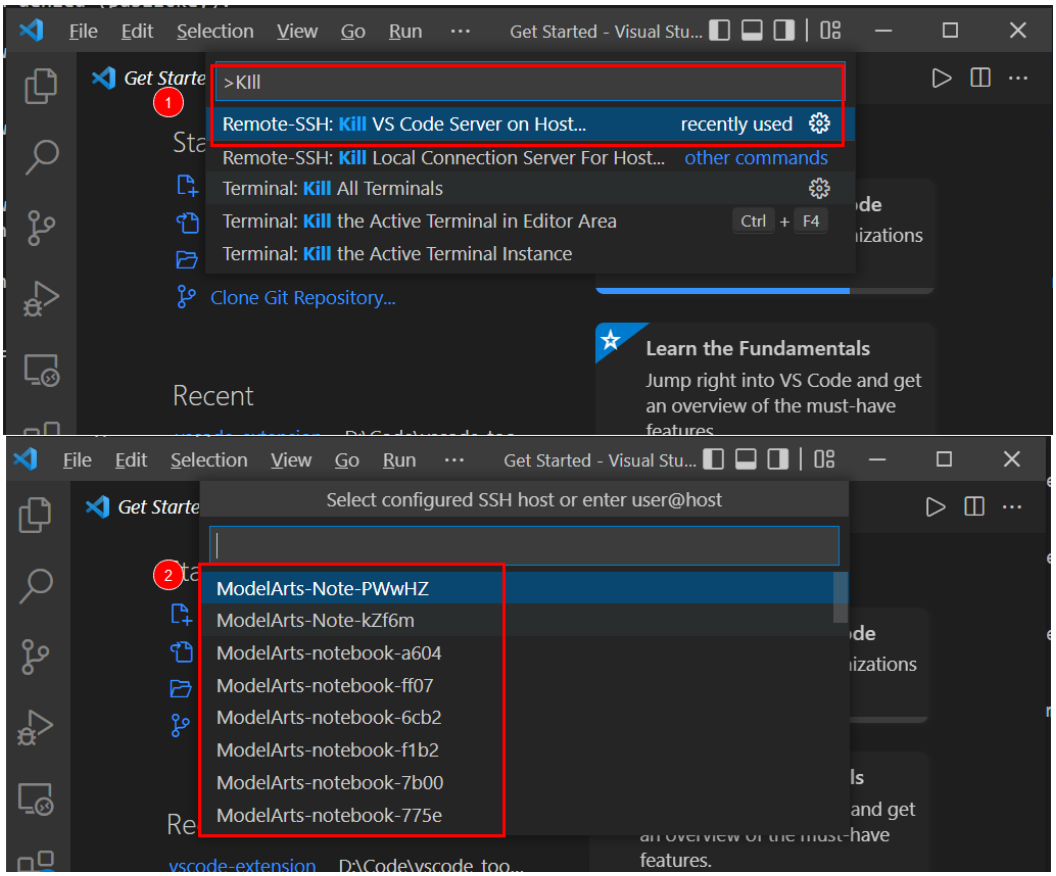
原因分析

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 2-10 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：

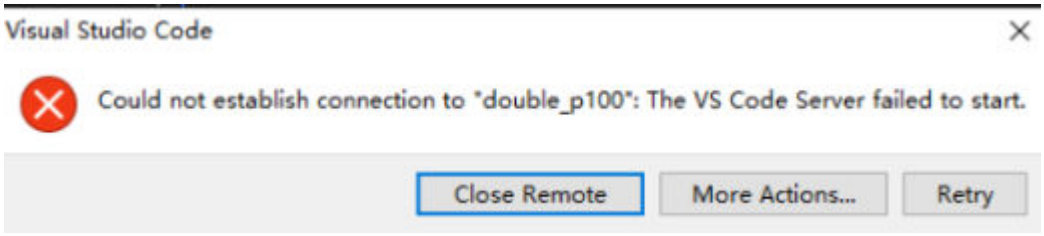
- IdentityFile：本地密钥路径
- User：用户名，例如：ma-user
- HostName：IP地址
- Port：端口号

说明

vscode-server相关问题也可以使用上述的解决方法。

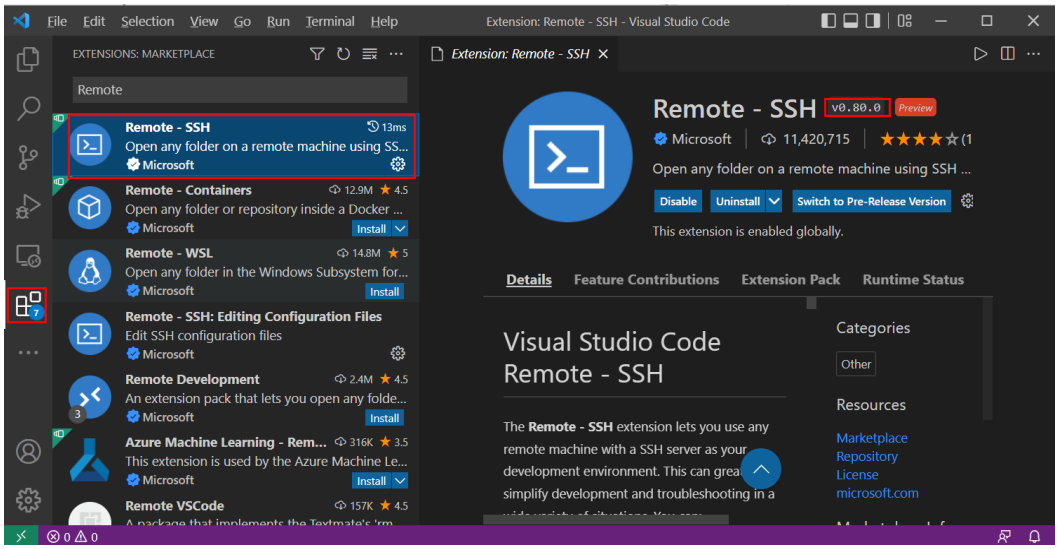
2.5.9 报错“The VS Code Server failed to start”如何解决?

问题现象



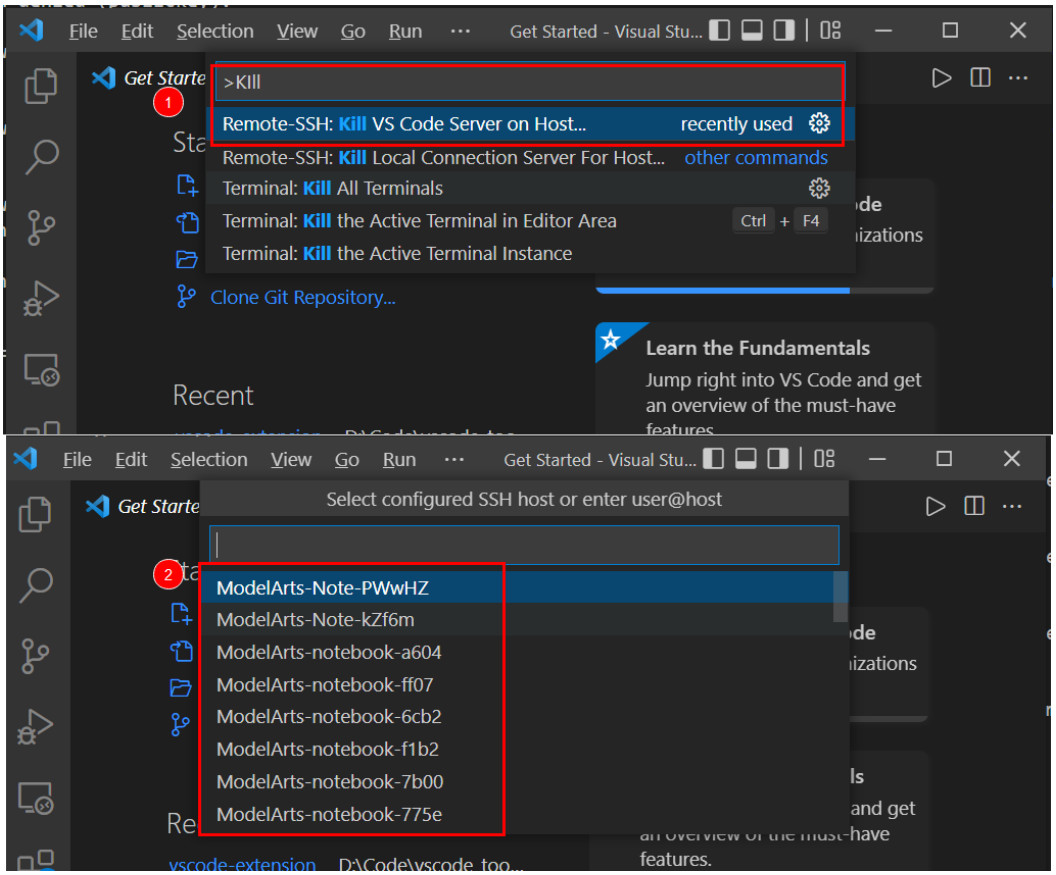
解决方法

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。

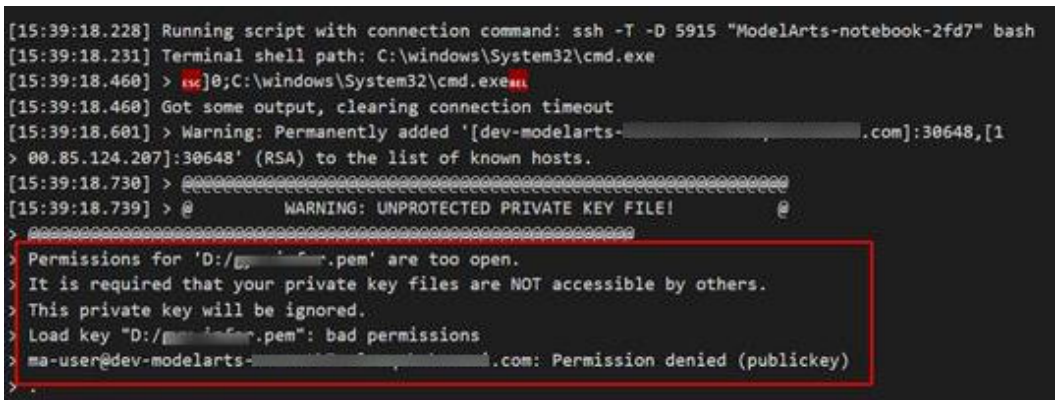
图 2-11 清除异常的实例



----结束

2.5.10 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.

2.5.11 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

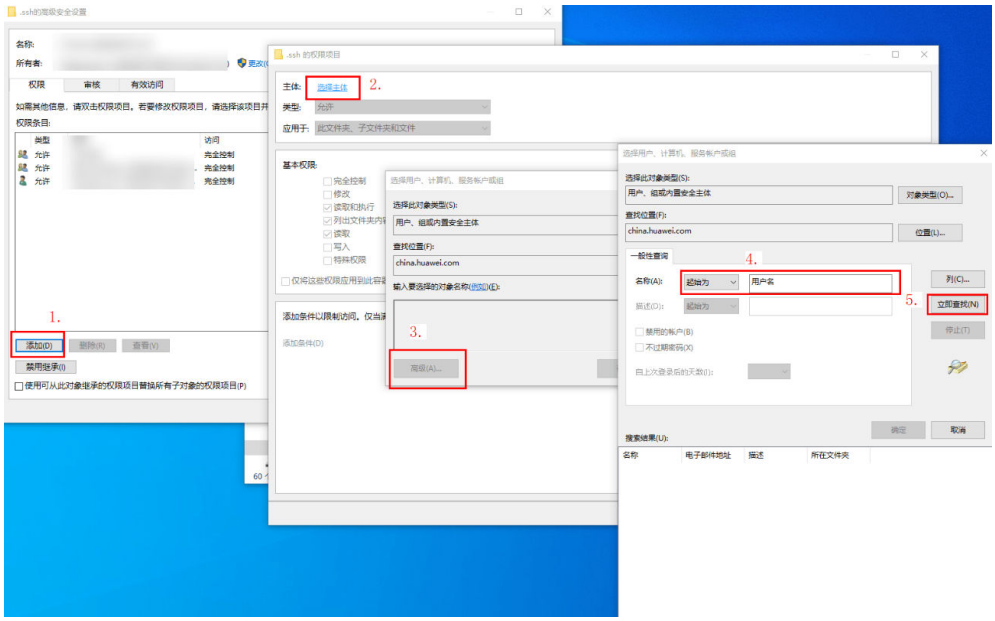
1. 找到.ssh文件夹。一般位于“C:\Users”，例如“C:\Users\xxx”。

   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 2-12 添加所有者

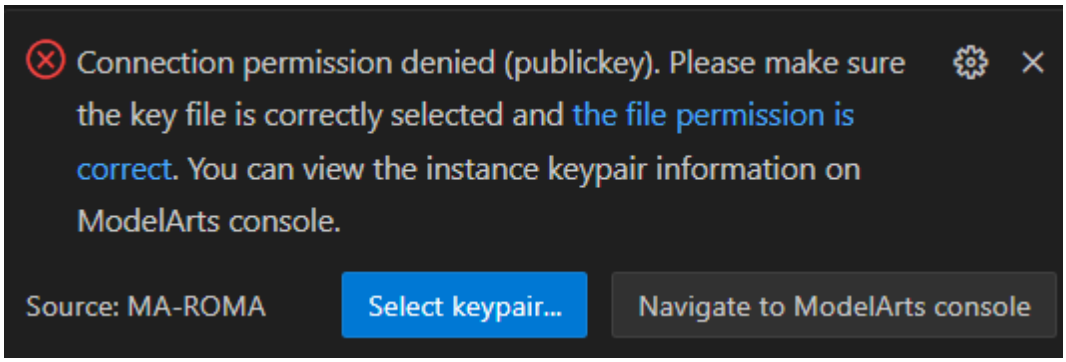
   

5. 完成所有操作后，再次关闭并打开VS Code并尝试连接到远程SSH主机。备注：此时密钥需放到.ssh文件夹中。

2.5.12 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。

解决方案

请参考以下文档进行解决。

- **本地通过SSH连接Notebook实例时，报错：Bad permissions/Permission denied (publickey)**
- **在ModelArts打开Notebook访问正常，使用SSH/VS Code访问时权限报错：Permission denied (publickey)**
- **用户修改/home/ma-user/.ssh目录权限导致SSH无法使用，报错：Permission denied (publickey)**

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

问题现象

原因分析

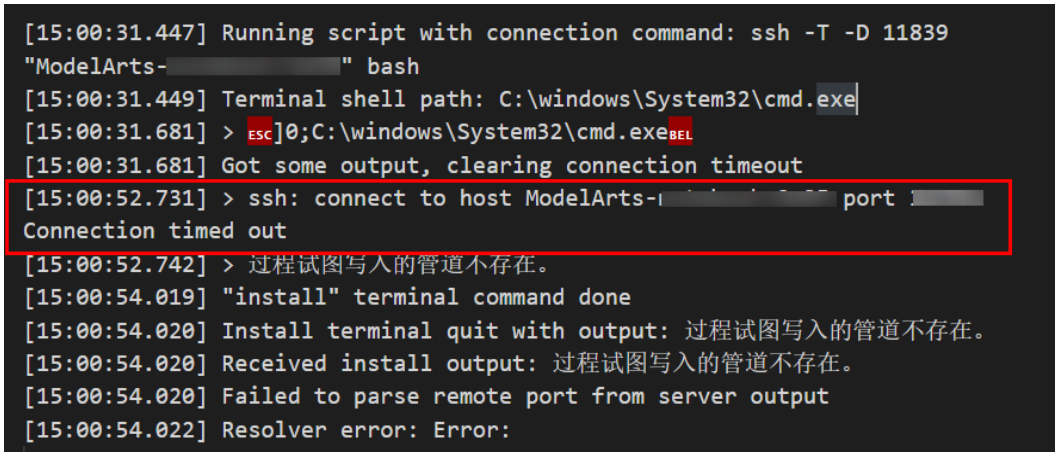
实例处于非运行状态。

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。

2.5.14 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。

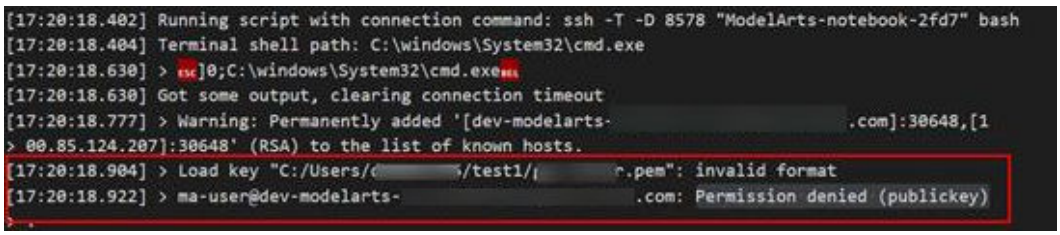
解决方法：请**修改白名单**为本地网络访问IP或者去掉白名单配置。

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。

2.5.15 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。

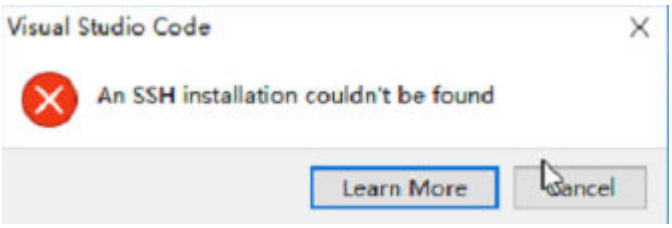
解决方法

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：

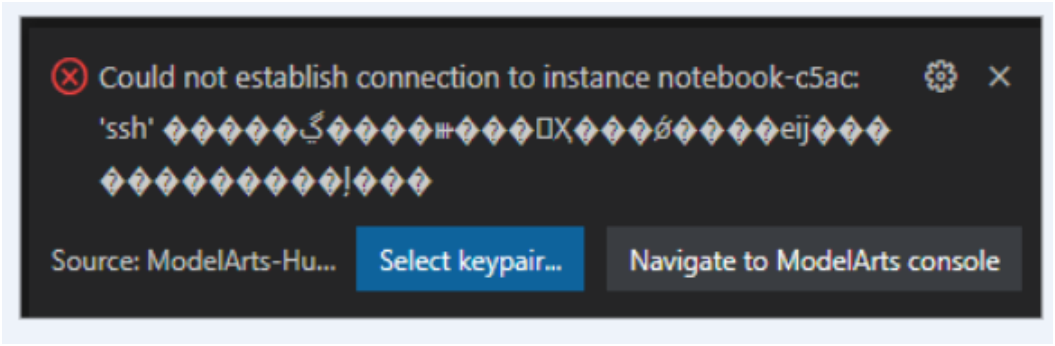
1. 登录控制台，搜索“密码安全中心 DEW”，选择“密钥对管理 > 账号密钥对”页签，查看并下载正确的密钥文件。
2. 如果密钥不支持下载且已无法找到之前下载的密钥，建议创建新的开发环境实例并创建新的密钥文件。

2.5.16 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**。

```
C:\windows\system32>ssh
usage: ssh [-46AaCfGgKkMNnqsTtVvXxYy] [-B bind_interface]
           [-b bind_address] [-c cipher_spec] [-D [bind_address:]port]
           [-E log_file] [-e escape_char] [-F configfile] [-I pkcs11]
           [-i identity_file] [-J [user@]host[:port]] [-L address]
           [-l login_name] [-m mac_spec] [-O ctl_cmd] [-o option] [-p port]
           [-Q query_option] [-R address] [-S ctl_path] [-W host:port]
           [-w local_tun[:remote_tun]] destination [command]
```

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：

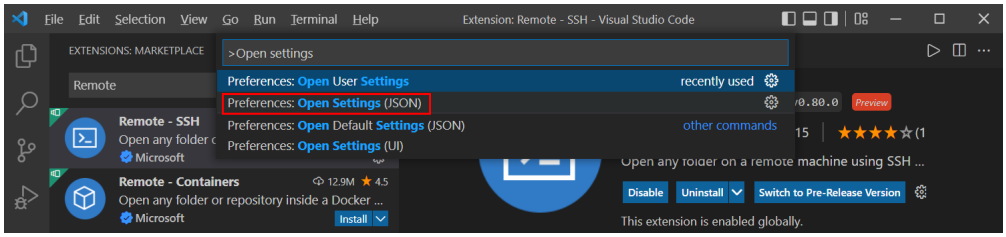
```
netsh advfirewall firewall add rule name=sshd dir=in action=allow protocol=TCP localport=22
```

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*"

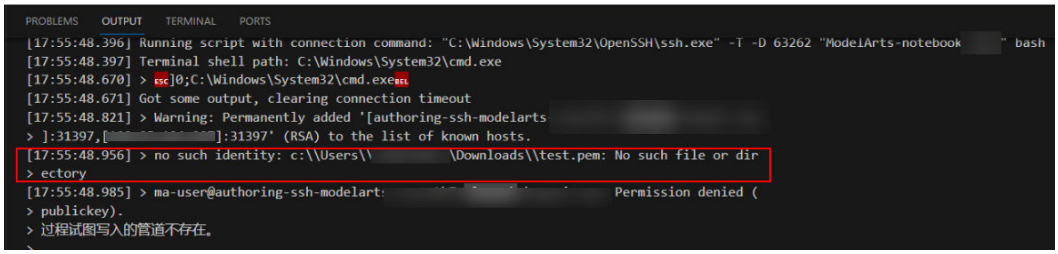
```
{
    "extensions.autoCheckUpdates": false,
    "extensions.autoUpdate": false,
    "remote.SSH.remotePlatform": {
        "ModelArts-notebook-          ": "linux"
    },
    "remote.SSH.path": "D:/OpenSSH-Win64/ssh.exe"
}
```

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

问题现象

报错“no such identity: C:/Users/xx/test.pem: No such file or directory”。

图 2-13 报错示例



原因分析

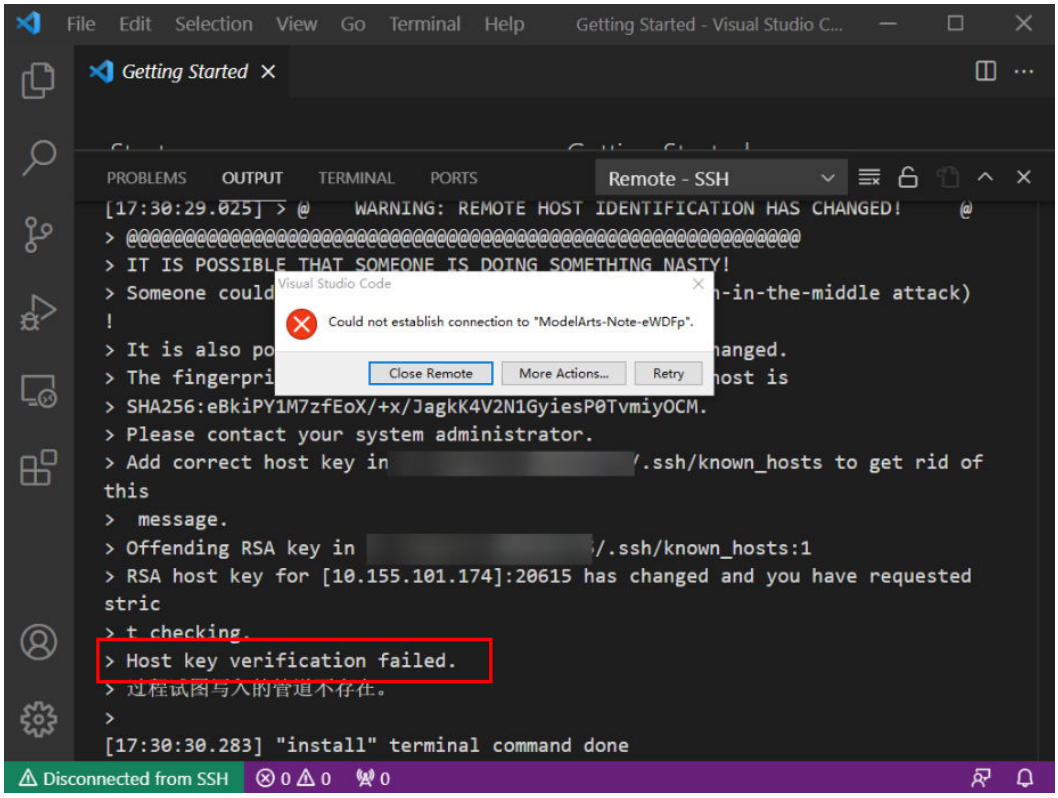
密钥文件不存在于该路径下，或者该路径下密钥文件名被修改。

解决方法

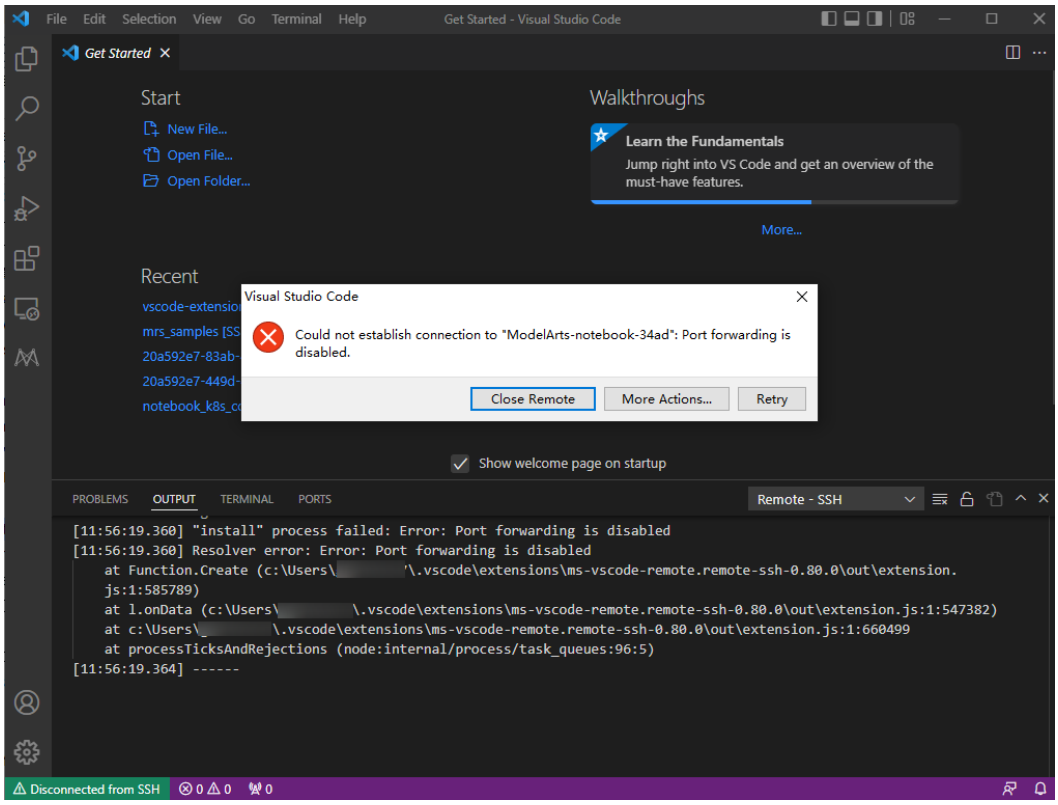
重新选择密钥路径。

2.5.18 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

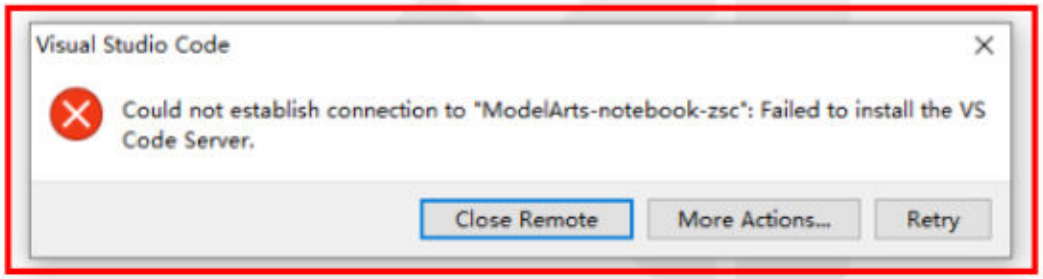
- 在VS Code中手工配置远程连接时，在本地的ssh config文件中增加配置参数“StrictHostKeyChecking no”和“UserKnownHostsFile=/dev/null”

  ```
  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。

2.5.19 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。

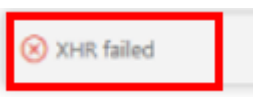
解决方法

删除/home/ma-user/work路径下无用文件。

2.5.20 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”。

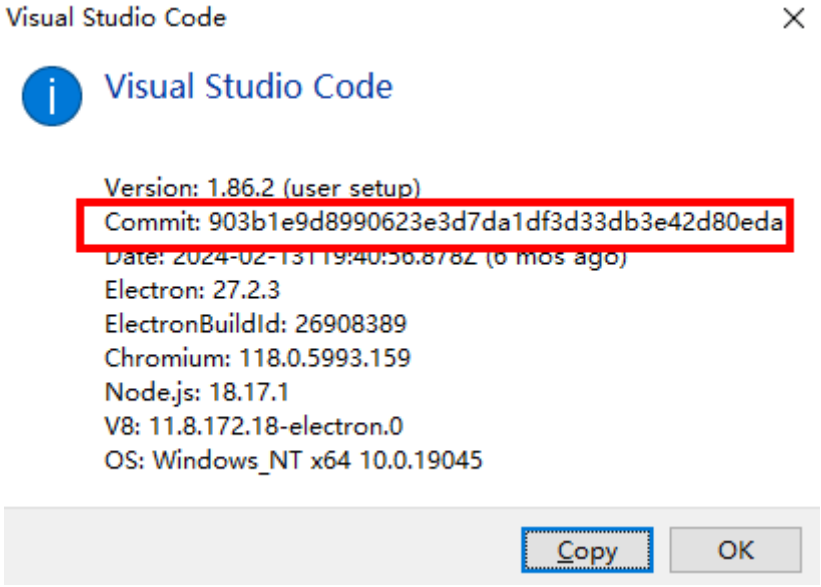


原因分析

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
     ```

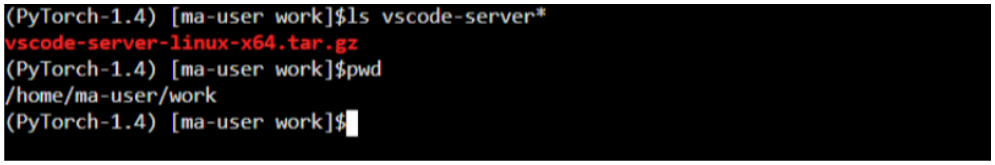
   - 如果实例的架构是AArch的，通过下面的链接，手动修改commit-id（commit-id替换时去掉尖括号），使用浏览器下载vscode-server-linux-arm64.tar.gz文件。下载完成后，将下载的vscode-server-linux-arm64.tar.gz文件重命名为“vscode-server-linux-x64.tar.gz”。

     ```
     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）

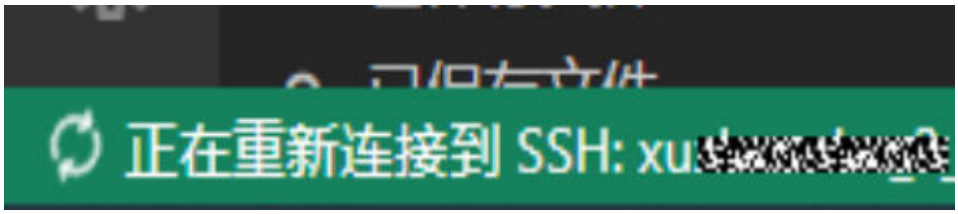
```
commitId=<提交的ID码>
mkdir -p /home/ma-user/.vscode-server/bin/$commitId
tar -zxvf vscode-server-linux-x64.tar.gz -C /home/ma-user/.vscode-server/bin/$commitId  --strip=1
chmod 750 -R /home/ma-user/.vscode-server/bin/$commitId
```

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）。

2.5.21 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 2-14 打开 VS Code ssh config 配置文件

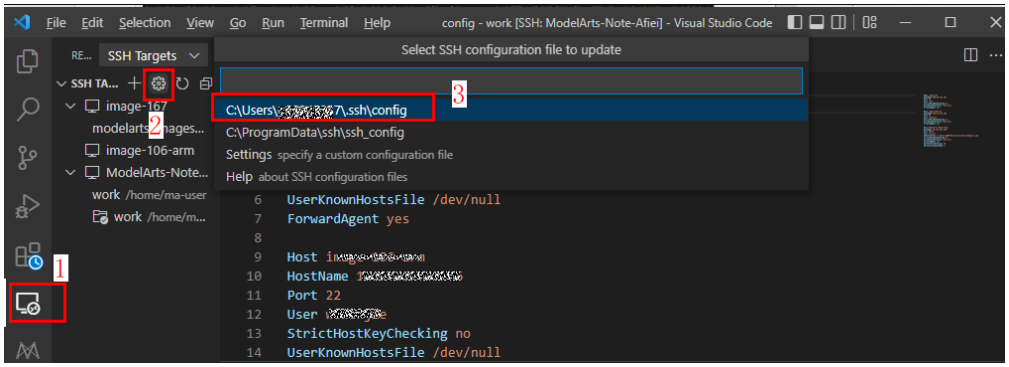
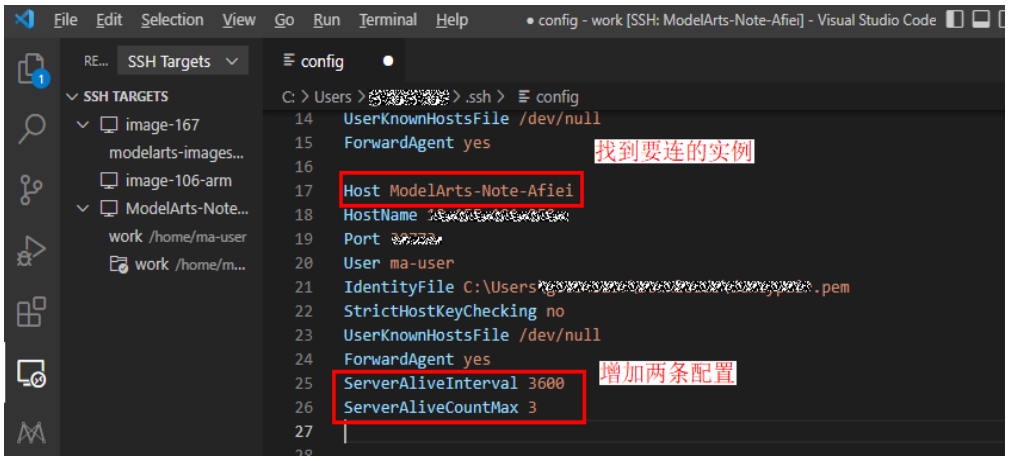


图 2-15 增加配置信息



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

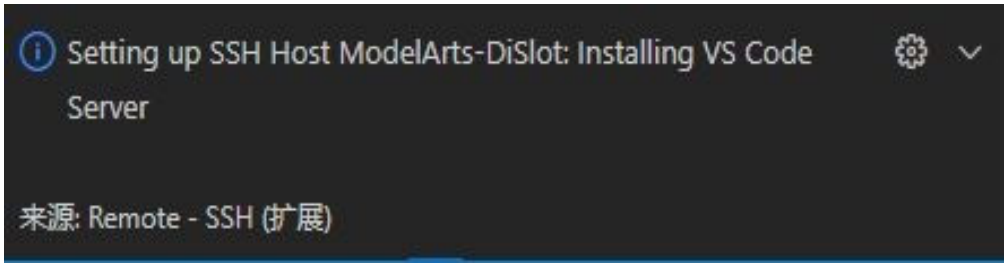
参考：https://unix.stackexchange.com/questions/3026/what-do-options-serveraliveinterval-and-clientaliveinterval-in-sshd-config-d

- 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
```

2.5.22 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 2-16 打开 Settings

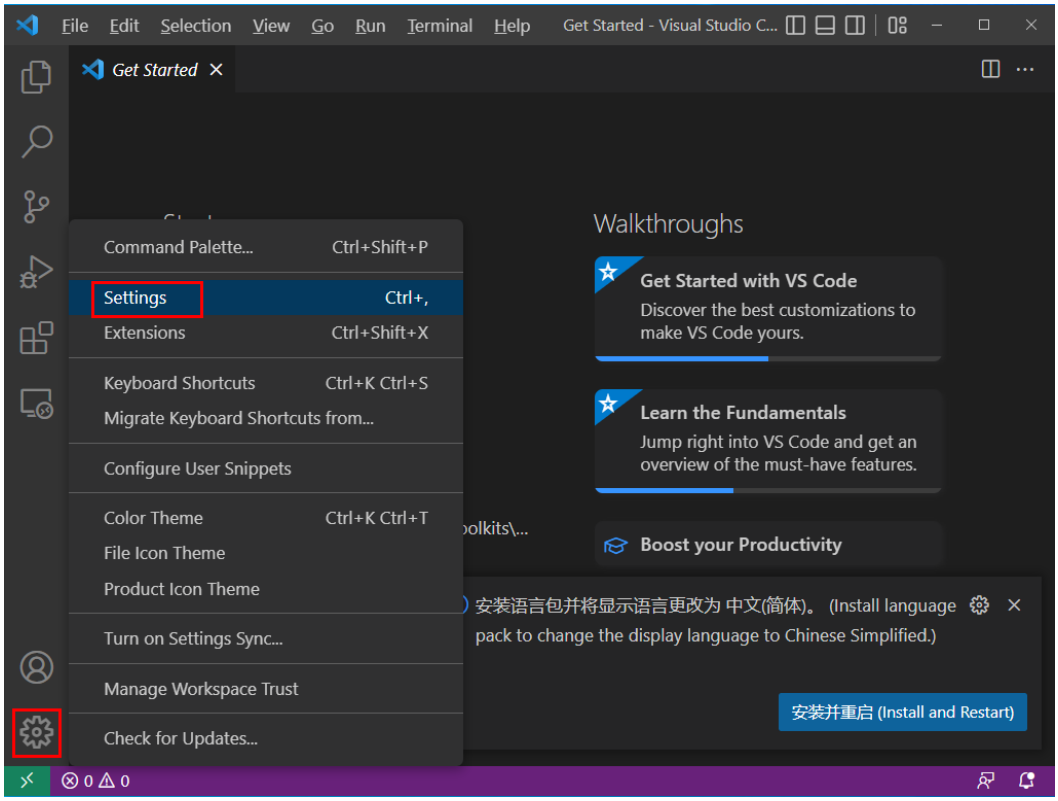
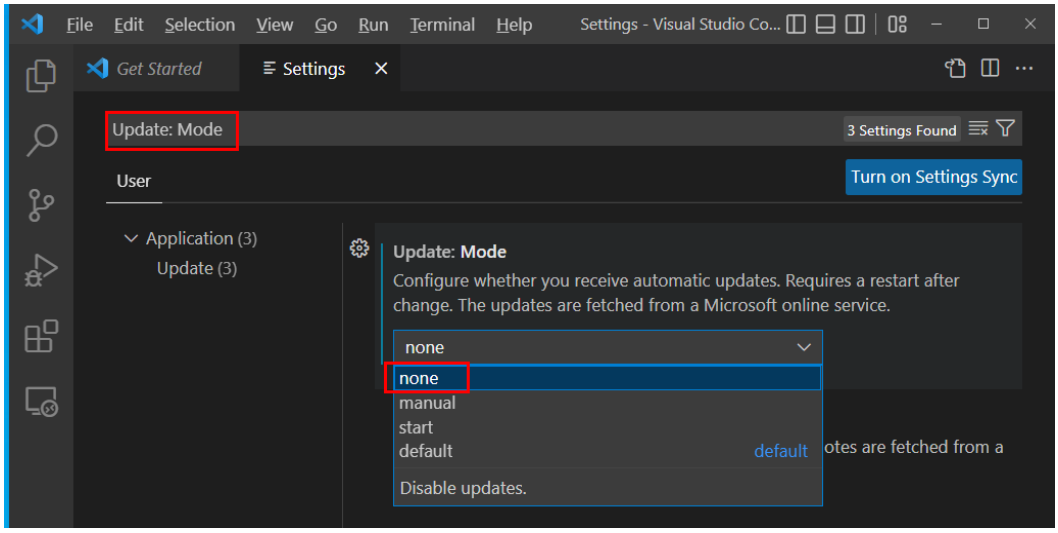
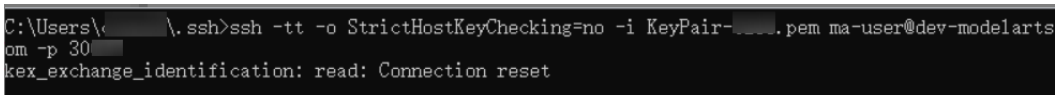


图 2-17 设置“Update: Mode”为“none”



2.5.23 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。

2.5.24 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。

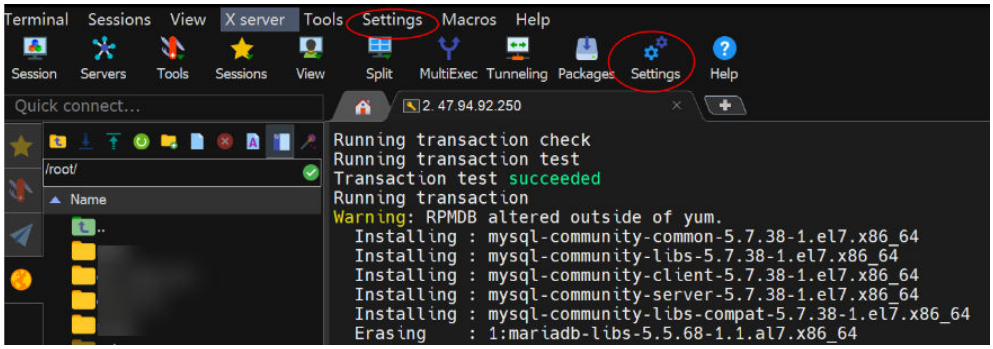
可能原因

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

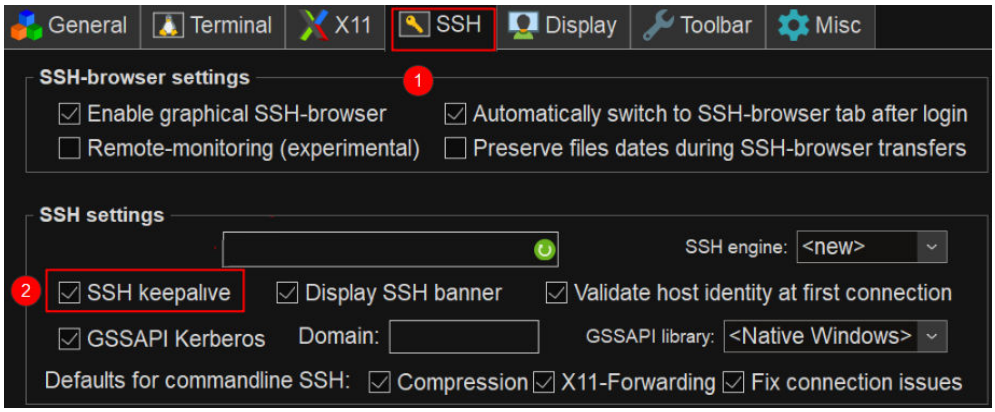
步骤1 打开MobaXterm，单击菜单栏“Settings”，如**图1 打开“Settings”**所示。

图 2-18 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 2-19 勾选“SSH keepalive”

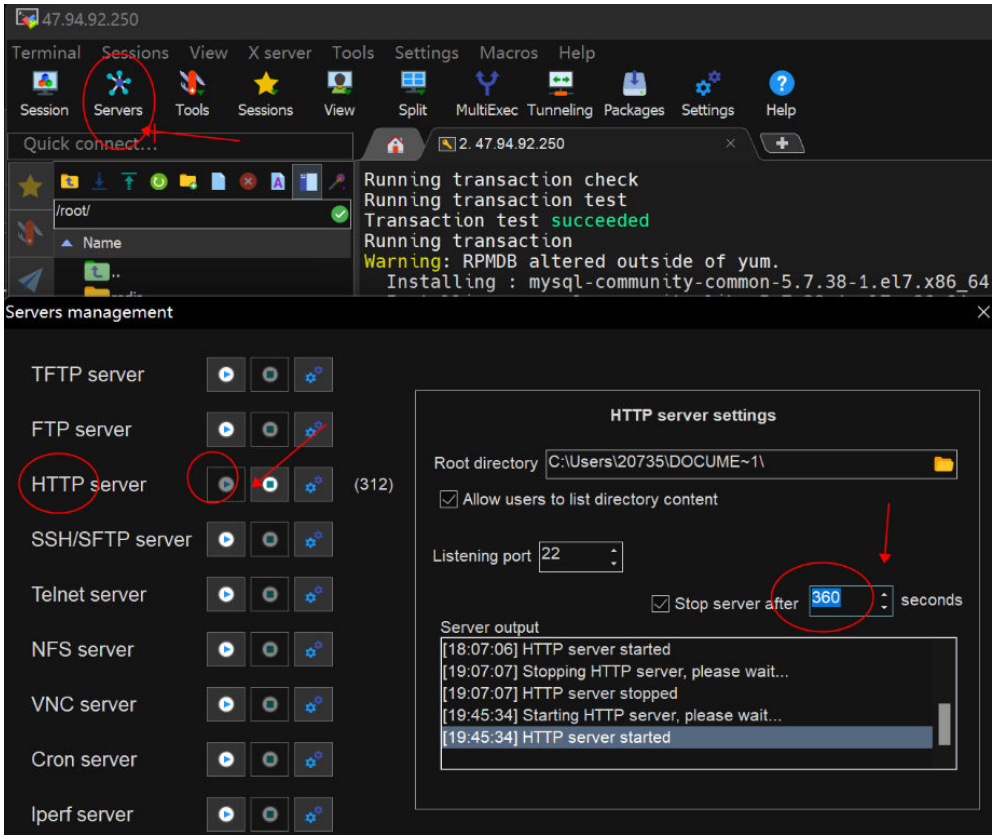


说明

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 2-20 设置“Stop server after”



----结束

2.5.25 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**。

2.5.26 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。

2.5.27 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
```

2.6 SSH 故障

2.6.1 本地通过 SSH 连接 Notebook 实例的故障排查

当SSH出现故障时，可能由以下原因导致：

表 2-1 SSH 故障原因

故障类型	故障可能原因	相关文档
权限相关	用户密钥不匹配或本地密钥文件权限不正确。	本地通过SSH连接Notebook实例时，报错：Bad permissions/ Permission denied (publickey)
	用户被锁定，导致SSH/VS Code无法访问。	在ModelArts打开Notebook访问正常，使用SSH/VS Code访问时权限报错：Permission denied (publickey)

故障类型	故障可能原因	相关文档
	用户修改“/home/ma-user/.ssh”目录权限导致SSH无法使用。	用户修改/home/ma-user/.ssh目录权限导致SSH无法使用，报错：Permission denied (publickey)
连接被拒绝	用户使用SSH连接Notebook实例时，同时建立的连接数超过10个，导致连接被拒绝。	用户使用SSH连接Notebook实例时，同时建立的连接数超过10个，报错：ssh_exchange_identification: Connection closed by remote host
	SSH偶现拒绝访问问题，报错：Not allowed at this time	SSH偶现拒绝访问问题，报错：Not allowed at this time
	用户自定义镜像使用远程SSH功能，OpenSSH版本不兼容或低于8.0，导致连接被拒绝。	用户使用自定义镜像创建Notebook，本地通过SSH连接Notebook时，报错：The OS version does not match

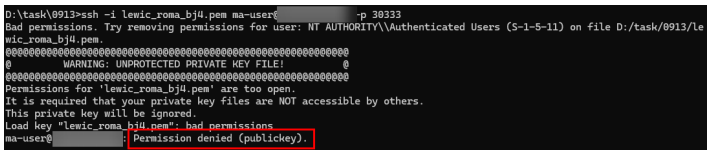
2.6.2 本地通过 SSH 连接 Notebook 实例时，报错：Bad permissions/Permission denied (publickey)

问题现象

本地通过SSH连接Notebook实例时，报错：

```
Bad permissions/Permission denied (publickey)
```

图 2-21 报错示例



原因分析

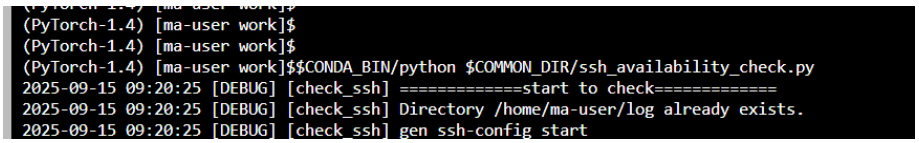
本地密钥不匹配或本地密钥文件权限不正确。

解决方案

1. 在JupyterLab执行以下命令，检测服务端SSH是否正常。

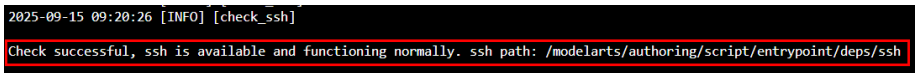
```
$CONDA_BIN/python $COMMON_DIR/ssh_availability_check.py
```

图 2-22 terminal 命令示例



- 如果执行结果如下，表示远端SSH服务正常，应该是密钥存在问题。请执行**步骤2**。

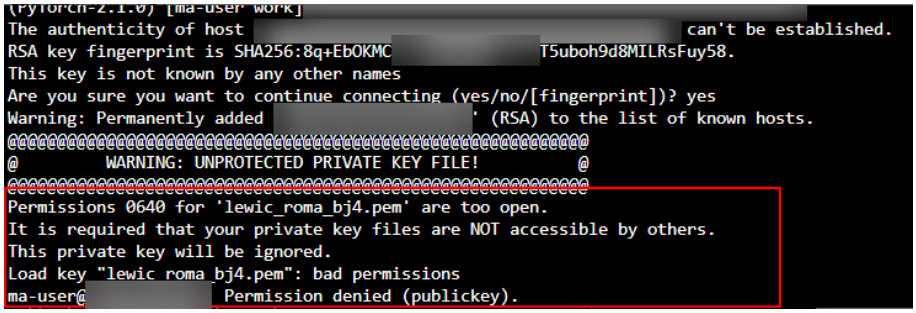
  图 2-23 terminal 结果示例

- 如果未出现上述结果，请参考以下文档进行排查。
  - **在ModelArts打开Notebook访问正常，使用SSH/VS Code访问时权限报错：Permission denied (publickey)**
  - **用户修改/home/ma-user/.ssh目录权限导致SSH无法使用，报错：Permission denied (publickey)**

2. 检查是否为自己本地密钥文件权限不正确。
   - Mac系统：

     图 2-24 连接报错示例

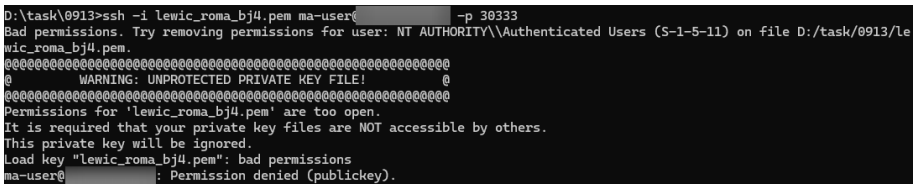
     

     出现上述报错，请执行以下命令修改权限。

     ```
     chmod 600 ${密钥文件路径}
     ```

   - Windows系统：

     图 2-25 连接报错示例

     出现上述报错，请将密钥放到“C:/user/{用户名}”的目录或子目录下，仅允许当前用户有访问权限。

   如果修改权限后仍报错，请执行**步骤3**。

3. 检查密钥是否为自己配置的密钥或者是否为重名密钥。

   您可以先停止Notebook实例，重新创建并绑定新的密钥，重启Notebook实例，然后重新使用SSH连接Notebook实例。如果连接正常，可能是因为密钥不匹配。您可以通过以下操作，判断密钥不匹配的原因。

   a. 查看密钥名称：在**ModelArts管理控制台**的Notebook实例详情页，查看“认证”值。

图 2-26 Notebook 实例详情页

b. 查看是否为重名密钥：在**密码安全中心控制台**的“密钥对管理 > 账号密钥对”页面，检查其他区域是否存在与上一步重名的密钥。

图 2-27 账号密钥对

如果密钥不对应或存在重名，请尝试更换正确的密钥。

2.6.3 在 ModelArts 打开 Notebook 访问正常，使用 SSH/VS Code 访问时权限报错：Permission denied (publickey)

问题现象

通过**ModelArts管理控制台**打开的Notebook访问正常，使用SSH/VS Code访问时权限报错：

```
Permission denied (publickey)
```

图 2-28 报错示例

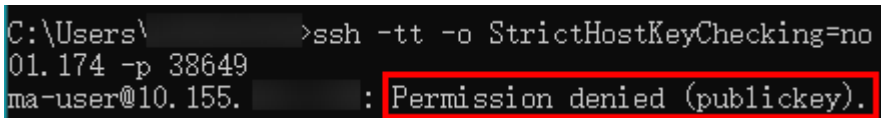
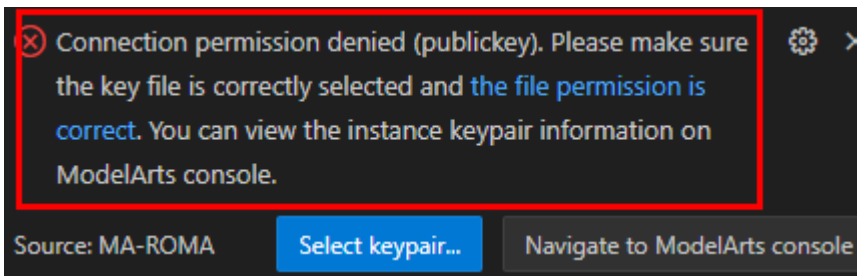


图 2-29 报错示例

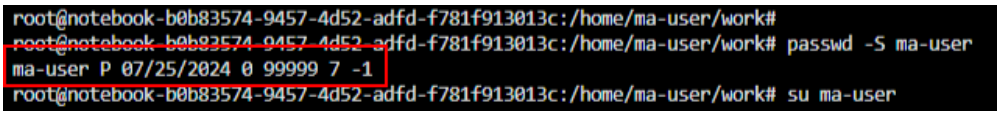


原因分析

执行passwd -S ma-user命令后：

- 如果结果中出现L，表示用户被锁定，需要使用passwd命令解锁用户。
- 如果结果中出现P，表示用户状态正常，可以排查其它可能原因。

图 2-30 解锁用户



解决方案

在Notebook中打开Terminal窗口，依次执行以下命令，使用passwd命令解锁用户。

```
passwd -S ma-user
passwd ma-user 回车输入密码
passwd -S ma-user
```

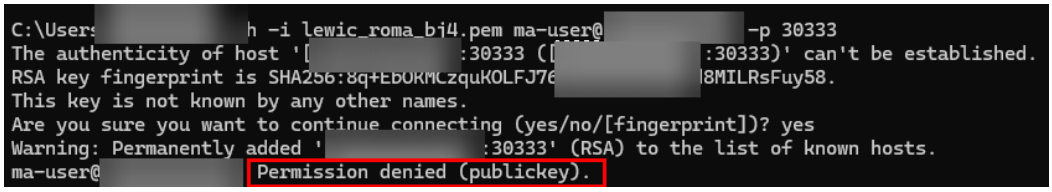
2.6.4 用户修改/home/ma-user/.ssh 目录权限导致 SSH 无法使用，报错：Permission denied (publickey)

问题现象

用户修改/home/ma-user/.ssh目录权限导致SSH连接Notebook无法使用，报错：

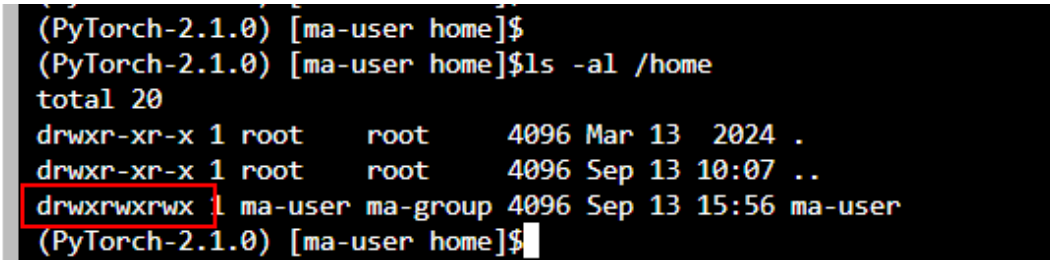
```
Permission denied (publickey)
```

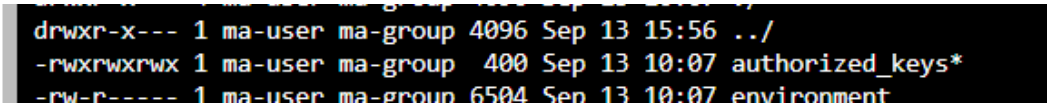
图 2-31 报错示例



原因分析

服务端/home/ma-user目录、SSH目录权限不正确。权限不正确示例如下：





解决方案

解决方案有以下两种，请按需选择：

- 在制作镜像时直接删除“/home/ma-user/.ssh”目录（用户无需配置SSH，Notebook启动时会自动生成）。
- 参考以下命令，修改.ssh相关文件权限。

```
# 建议将/home/ma-user目录权限设置为750
chmod 750 /home/ma-user

chmod 750 ~/.ssh
chmod 644 ~/.ssh/authorized_keys
chmod 640 ~/.ssh/environment
chmod 750 ~/.ssh/etc
chmod 640 ~/.ssh/known_hosts
chmod 750 ~/.ssh/var
chmod 600 ~/.ssh/etc/ssh_host_rsa_key
chmod 640 ~/.ssh/etc/ssh_host_rsa_key.pub
chmod 750 ~/.ssh/etc/sshd_config
chmod 750 ~/.ssh/var/run/sshd.pid
```

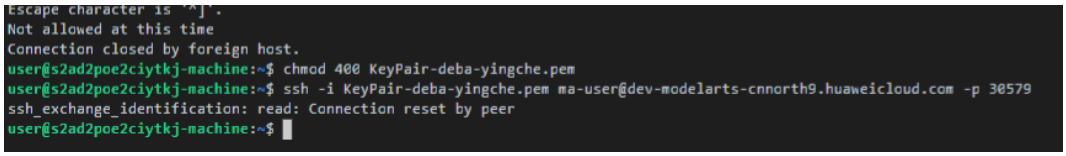
2.6.5 用户使用 SSH 连接 Notebook 实例时，同时建立的连接数超过 10 个，报错：ssh_exchange_identification: Connection closed by remote host

问题现象

用户使用SSH连接Notebook实例时，同时建立的连接数超过10个，导致链接被拒绝，报错：

```
ssh_exchange_identification: Connection closed by remote host
```

图 2-32 报错示例

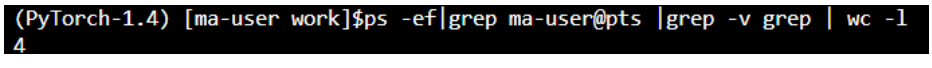


原因分析

同时建立的连接数超过10个，导致链接被拒绝。您可以执行以下命令，查看连接数。

```
ps -ef|grep ma-user@pts |grep -v grep | wc -l
```

图 2-33 连接数查询



解决方案

1. 断开不使用的SSH连接。
2. 重启Notebook。具体操作，请参见**启动/停止/删除实例**。

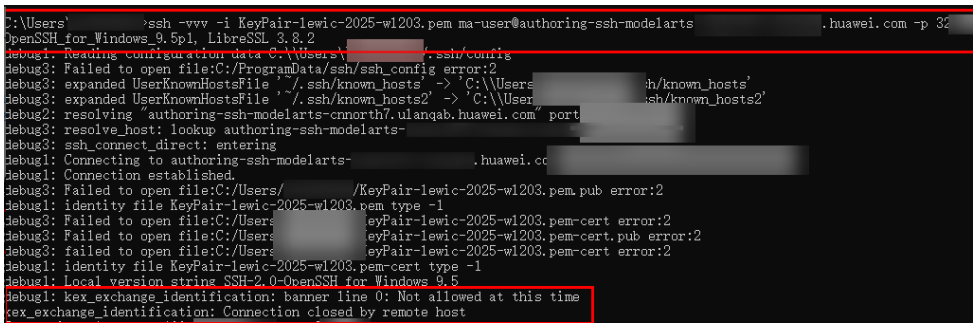
2.6.6 SSH 偶现拒绝访问问题，报错：Not allowed at this time

问题现象

执行SSH命令连接Notebook时，一直报错：Not allowed at this time。

```
debug1: kex_exchange_identification: banner line 0: Not allowed at this time
kex_exchange_identification: Connection closed by remote host
```

图 2-34 报错示例



原因分析

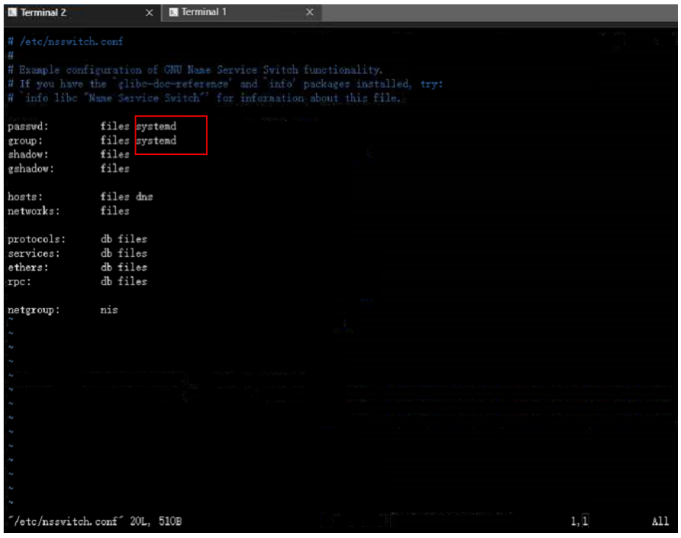
存在攻击者通过持续暴力破解SSH服务端触发段错误（libnss_systemd.so.2），导致服务端进入90秒拒绝服务状态，正常用户无法访问。具体流程如下：

1. 用户的Notebook SSH地址被其他用户频繁访问，且访问时未携带用户名，例如：ssh hosts.com -p 3333。
2. 用户使用的自定义镜像中包含不兼容的libnss_systemd.so.2文件，且/etc/nsswitch.conf配置了使用systemd方式进行用户信息校验。当SSH服务端在处理未携带用户名的连接请求时，触发了段错误。
3. 频繁执行SSH连接命令，SSH服务端在段错误后进入90秒的拒绝服务状态，导致服务长时间无法访问。

解决方案

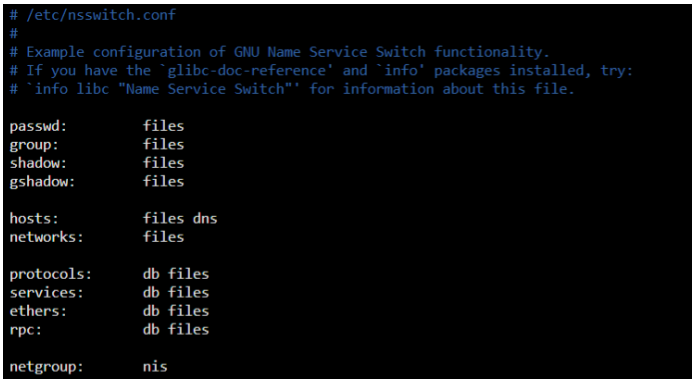
1. 修改自定义镜像中/etc/nsswitch.conf配置文件，删除systemd并保存。
   - 修改前：

图 2-35 配置文件（含 systemd）示例



- 修改后：

图 2-36 配置文件（不含 systemd）示例



修改内容说明：

passwd: files：指定系统从本地文件（如/etc/passwd）中读取用户信息。

group: files：指定系统从本地文件（如/etc/group）中读取组信息。

2. 90秒后重试SSH连接，即可恢复正常。

2.6.7 用户使用自定义镜像创建 Notebook，本地通过 SSH 连接 Notebook 时，报错：The OS version does not match

问题现象

用户使用自定义镜像创建Notebook，并开启远程SSH。本地通过SSH连接Notebook实例时，连接被拒绝，报错：

```
The OS version does not match
```

原因分析

镜像内置的OpenSSH版本不兼容或低于8.0。

解决方案

您可以直接使用Notebook预置镜像，或者使用预置镜像制作自定义镜像。更多信息，请参见**创建Notebook实例（新版页面）**和**Notebook的自定义镜像制作方法**。

图 2-37 预置镜像

2.7 自定义镜像故障

2.7.1 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，可能由以下原因导致：

- 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；
- 用户的Anaconda环境中是否出现了以下问题：
  - 在“{python_env}/lib”目录下存在以python开头的非法目录（例如“pythonNone”），正常目录名应该是python+版本号（例如“python3.7”），这可能是由于环境配置错误或意外操作导致的。
  - 用户可能手动在Anaconda环境目录“{conda}/envs”下创建了空目录或在环境的“lib”目录下创建了非法目录，这种操作会破坏Anaconda的目录结构。
  - 用户可能手动清空了某个环境目录内的文件，而这些文件是Anaconda环境所必需的，导致环境无法正常工作。
- SSH故障排查请参见**本地通过SSH连接Notebook实例的故障排查**。

2.7.2 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。

2.7.3 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。

2.7.4 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。

2.7.5 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。

2.7.6 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

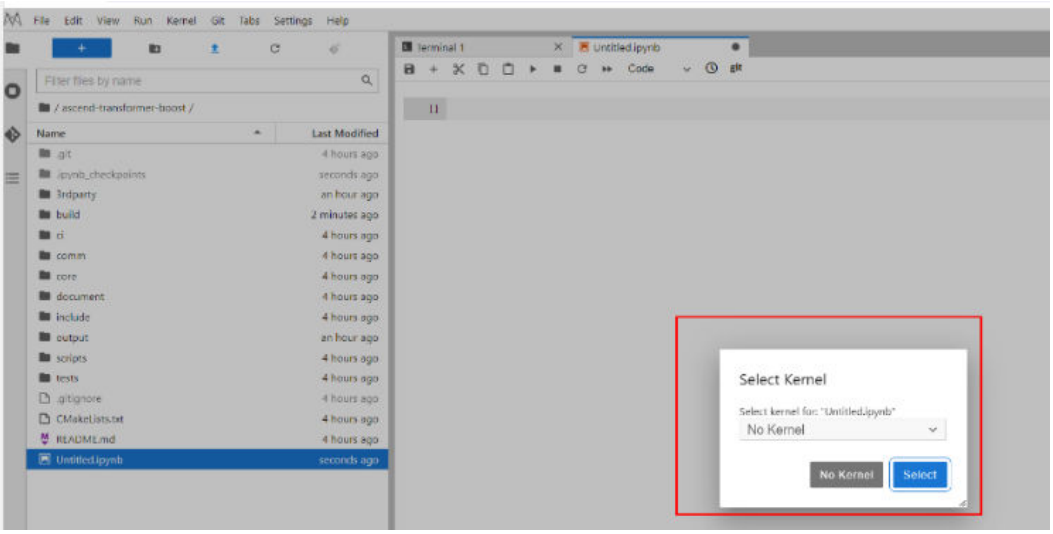
以对应租户的华为云账号登录SWR服务，查看镜像是否已经推送成功。

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

2.7.7 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。

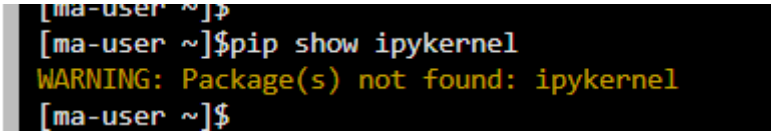


原因分析

自定义镜像的Python环境没有注册。

解决方案

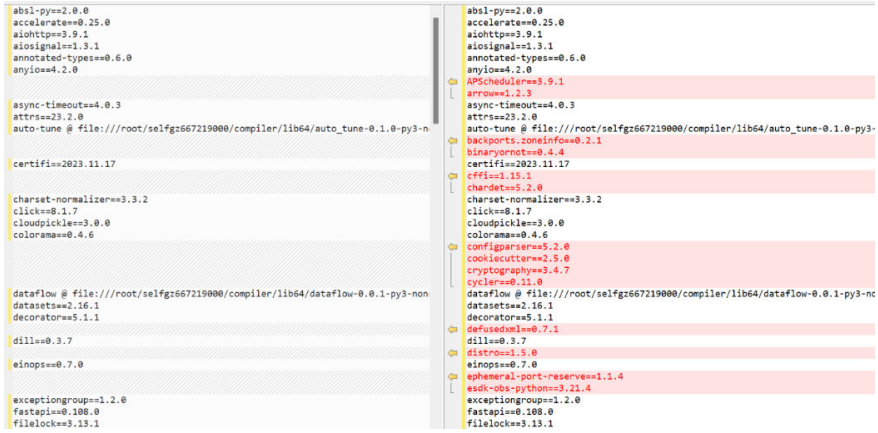
1. 在Terminal里执行命令排查实例存在几个Conda环境。
   ```
   conda env list
   ```
2. 执行如下命令分别切换到对应环境查看是否有ipykernel包。
   ```
   conda activate base # base替换为实际使用的Python环境
   pip show ipykernel
   ```
3. 对应conda环境没有ipykernel，直接**在Notebook中添加自定义IPython Kernel**安装。



2.7.8 用户自定义镜像自建的 conda 环境会检测到一些额外的包，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里会检测到一些额外的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。

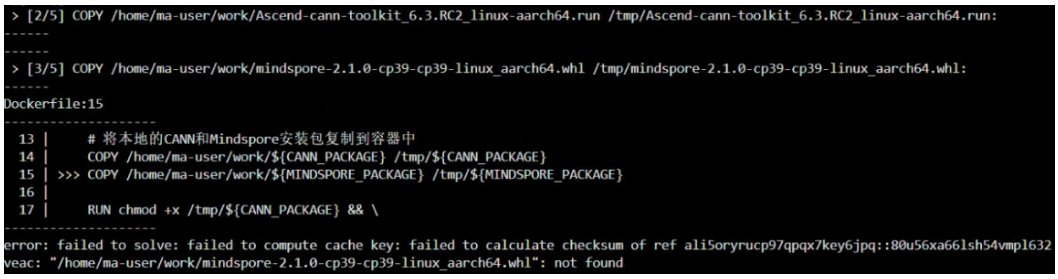
1. 执行如下命令，在用户运行的Conda环境下查找modelarts.pth。
   ```
   # /home/ma-user/anaconda3指用户的Python环境
   find /home/ma-user/anaconda3 -name modelarts.pth
   ```
2. 执行如下命令，删除用户使用的Python环境中的modelarts.pth文件。
   ```
   # /xxx/modelarts.pth 指用户通过第一步查出来的文件路径
   rm -rf /xxx/modelarts.pth
   ```

2.7.9 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 2-38 报错 xxx not found



原因分析

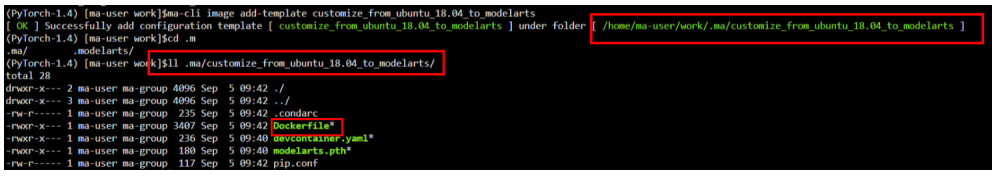
复制的文件需要放在Dockerfile同级文件夹或者子目录中，不能放在Dockerfile上层目录。

图 2-39 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 2-40 查询 Dockerfile 的路径

2. Dockerfile命令修改为相对路径，例如：

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


2.7.10 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。

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。

   例如，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*

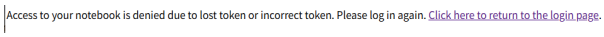
2.7.11 旧版镜像启动后无权限访问

问题现象

用户使用2023年06月之前注册的镜像，创建Notebook实例后运行正常，SSH可以连接，但是网页端打开后无法访问，报错如下：

```
Access to your notebook is denied due to lost token or incorrect token. Please log in again. Click here to return to the login page.
```

图 2-41 报错图片示例



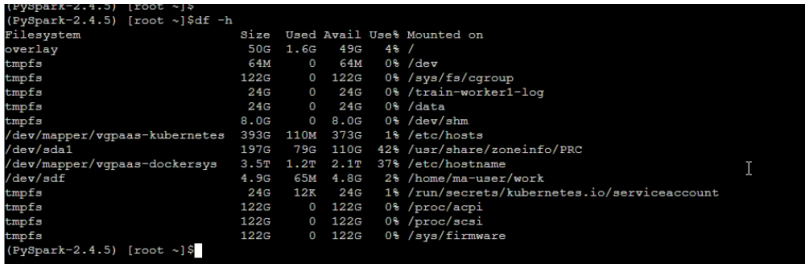
问题原因

2023年06月之前的Notebook为1.0版本，该版本已下线，需要使用新版Notebook。

解决方案

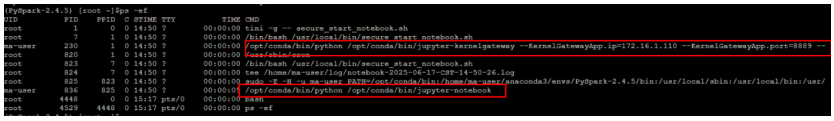
1. 确认实例是否为Notebook 1.0的启动方式。
   a. 创建Notebook实例时开启“SSH远程开发”。更多信息，请参见**创建Notebook实例**。
   b. 通过SSH连接Notebook后，任选以下方式确认是否为Notebook 1.0的启动方式。
      - 方式一：执行df -h命令。
        如下图所示，如果不存在/modelarts目录，则为Notebook 1.0方式启动方式。

        图 2-42 不存在/modelarts 目录示例

      - 方式二：查看启动进程，执行ps -ef命令。
        如果存在下图中的进程，则为Notebook 1.0方式启动方式。

        图 2-43 查看启动进程示例

2. 停止Notebook实例。
   a. 登录ModelArts管理控制台，在左侧导航栏选择“开发空间 > Notebook ”。
   b. 在“Notebook”页面的“操作”列，单击“停止”，在弹出的对话框，单击“确定”。更多信息，请参见**启动/停止/删除实例**。
3. 在镜像管理中删除历史镜像。
   a. 登录ModelArts管理控制台，在左侧导航栏单击“镜像管理”。
   b. 在“镜像管理”页面单击镜像名称，在“操作”列单击历史镜像对应的“删除”。
   c. 在弹出的对话框，输入DELETE，单击“确定”。
4. 重新注册镜像。

a. 登录ModelArts管理控制台，在左侧导航栏单击“镜像管理”。
b. 在“镜像管理”页面右上角，单击“注册镜像”。
c. 在“注册镜像”页面，配置相关信息，然后单击“立即注册”。更多信息，请参见**制作自定义镜像用于创建Notebook**。

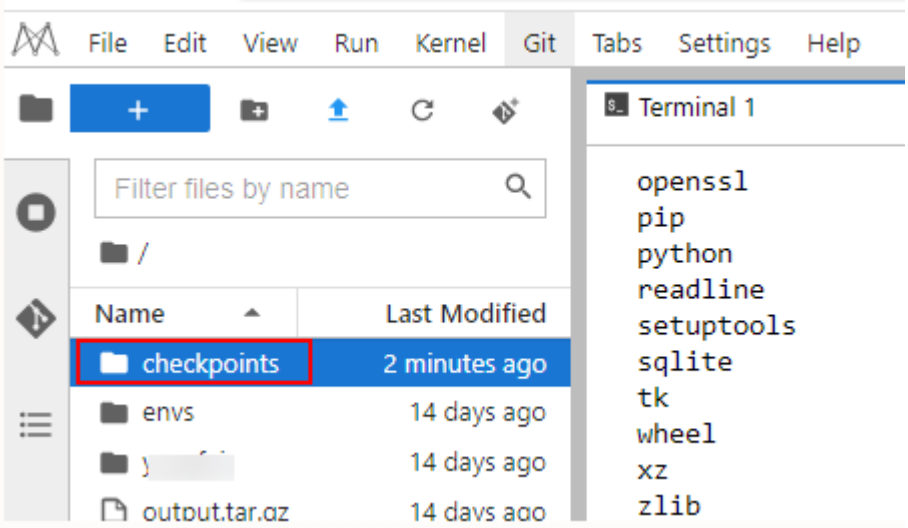
5. 根据新注册的镜像，重新创建Notebook。具体操作，请参见**创建Notebook实例**。

2.8 其他故障

2.8.1 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 2-44 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
```

2.8.2 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

2.8.3 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……。

```
(PyTorch-1.8) [ma-user work]$tensorboard --logdir ./
/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/site-packages/requests/__init__.py:104: RequestsDependencyWarning: urllib3 (1.26.12) or chardet (5.1.0)/charset_normalizer
ed version!
  RequestsDependencyWarning)
TensorFlow installation not found - running with reduced feature set.
Serving TensorBoard on localhost; to expose to the network, use a proxy or pass --bind_all
TensorBoard 2.1.1 at http://localhost:6006/ (Press CTRL+C to quit)
Exception in thread Reloader:
Traceback (most recent call last):
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/threading.py", line 926, in _bootstrap_inner
    self.run()
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/threading.py", line 870, in run
    self._target(*self._args, **self._kwargs)
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/site-packages/tensorboard/backend/application.py", line 586, in _reload
    multiplexer.AddRunsFromDirectory(path, name)
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/site-packages/tensorboard/backend/event_processing/plugin_event_multiplexer.py", line 199, in AddRunsFromDirectory
    for subdir in io_wrapper.GetLogdirSubdirectories(path):
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/site-packages/tensorboard/backend/event_processing/io_wrapper.py", line 200, in <genexpr>
    subdir
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/site-packages/tensorboard/backend/event_processing/io_wrapper.py", line 155, in ListRecursivelyViaWalking
    for dir_path, _, filenames in tf.io.gfile.walk(top, topdown=True):
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/site-packages/tensorboard/compat/tensorflow_stub/io/gfile.py", line 687, in walk
    for subitem in walk(joined_subdir, topdown, onerror=onerror):
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/site-packages/tensorboard/compat/tensorflow_stub/io/gfile.py", line 687, in walk
    for subitem in walk(joined_subdir, topdown, onerror=onerror):
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/site-packages/tensorboard/compat/tensorflow_stub/io/gfile.py", line 687, in walk
    for subitem in walk(joined_subdir, topdown, onerror=onerror):
  [Previous line repeated 1 more time]
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/site-packages/tensorboard/compat/tensorflow_stub/io/gfile.py", line 664, in walk
    listing = listdir(top)
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/site-packages/tensorboard/compat/tensorflow_stub/io/gfile.py", line 626, in listdir
    return get_filesystem(dirname).listdir(dirname)
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/site-packages/tensorboard/compat/tensorflow_stub/io/gfile.py", line 184, in listdir
    entries = os.listdir(compat.as_str_any(dirname))
PermissionError: [Errno 13] Permission denied: './.lsp symlink/etc/ssl/private'
```

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
```

3 训练作业

3.1 OBS 操作相关故障

3.1.1 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): Unsuccessful 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.system('pip install torch')
   import torch
   ```

3.1.2 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
```

3.1.3 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()
   ```

3.1.4 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()
```

3.1.5 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**。

3.1.6 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**。

3.1.7 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**。

3.1.8 tensorboard 显示 502 bad gateway

问题现象

启动tensorboard后，打开tensorboard提示502 bad gateway，或者偶现502 bad gateway。

原因分析

出现该问题的可能原因如下:

- 启动tensorboard对应的summary目录错误，导致tensorboard启动失败。
- 启动tensorboard对应的summary目录过大，导致tensorboard加载summary导致OOM。

处理方法

检查summary目录是否存在其他文件，如有请删除。

检查summary目的文件是否过大（比如大于5GB），如果有请减小summary。

3.2 云上迁移适配故障

3.2.1 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')
  ```

3.2.2 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**。

3.2.3 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')
   ```

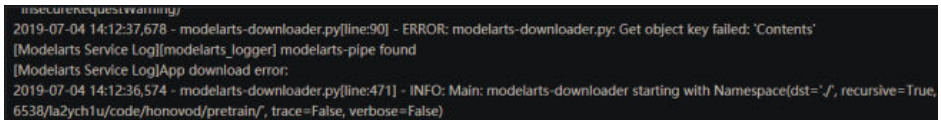
3.2.4 下载代码目录失败

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3-1：

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

图 3-1 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/')
  ```

3.2.5 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**。

3.2.6 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**。

3.2.7 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()
```

3.2.8 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。

3.2.9 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**。

3.2.10 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**。

3.2.11 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._write_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**。

3.2.12 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**。

3.2.13 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**。

3.2.14 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GP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。

3.2.15 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**。

3.2.16 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:00000183C6C4010C66D399E000COE3xx
   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。

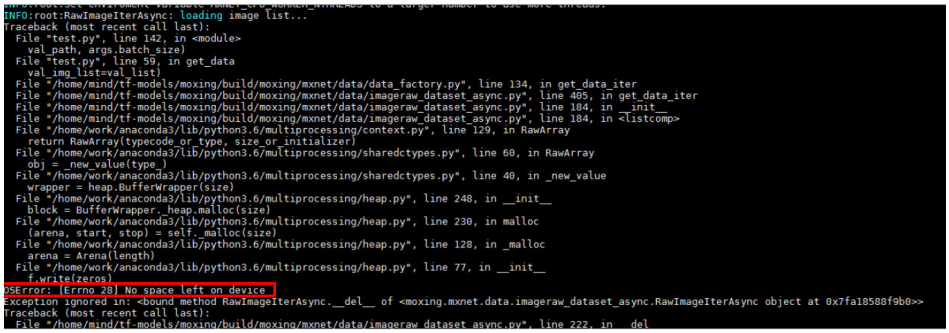
3.3 硬盘限制故障

3.3.1 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 3-2 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**。

3.3.2 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，GP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GP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GP资源。
3. 请在代码中添加环境变量来解决。

   ```
   import os
   os.system('export TMPDIR=/cache')
   ```

3.3.3 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.....................................
```

3.3.4 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。

3.3.5 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，具体规格的空间大小可参见**训练环境中不同规格资源“/cache”目录的大小**。

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。

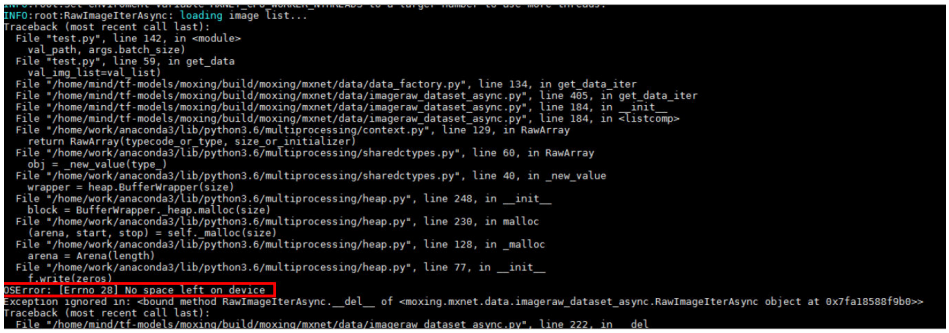
- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

3.3.6 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 3-3 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**。

3.3.7 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 3-4 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**。

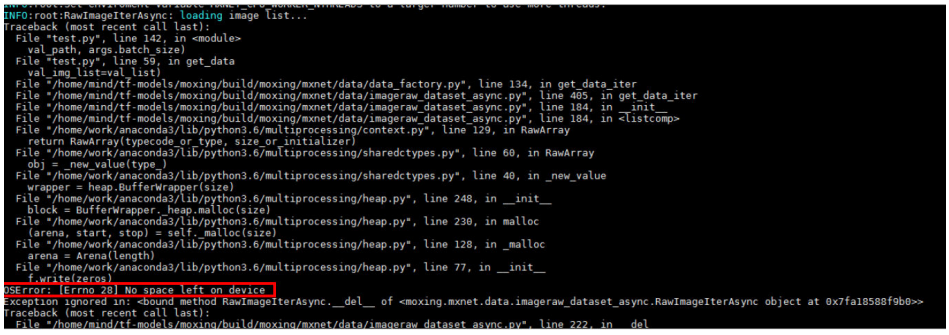
3.3.8 常见的磁盘空间不足的问题和解决办法

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 3-5 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/modelarts/目录，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。也可能是用户单目录下写入文件过多，同时每个文件名还字段超长，超过欧拉系统的限制。

  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。具体规格的空间大小可参见**训练环境中不同规格资源“/cache”目录的大小**
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/modelarts/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")
   ```
7. 控制单个目录下的最大文件数在约5,200,000 个文件以下，同时考虑到达这个上限的50%的时候性能就会显著下降，所以推荐实际使用上限为200W-250W。

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**。

3.4 外网访问限制

3.4.1 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**。

3.4.2 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。

3.5 权限问题

3.5.1 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**

3.5.2 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**。

3.6 GP 相关问题

3.6.1 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**。

3.6.2 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**。

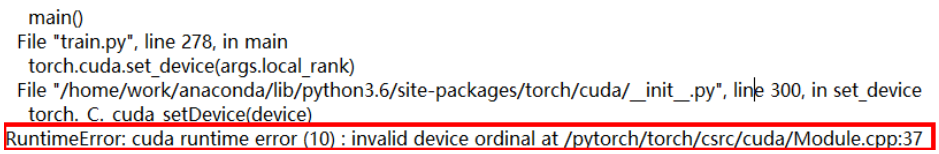
3.6.3 日志提示“cuda runtime error (10) : invalid device ordinal at xxx”

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 3-6 错误日志



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GP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GP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GP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**。

3.6.4 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**。

3.6.5 训练作业找不到 GP

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GP。

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GP：

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

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

3.7 业务代码问题

3.7.1 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**。

3.7.2 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**。

3.7.3 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

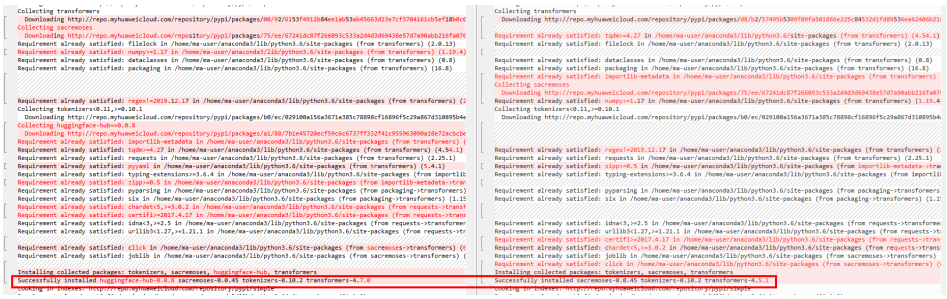
出现该问题的可能原因如下

- 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，建议先尝试复制作业，复制后仍然失败，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 3-7 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**。

3.7.4 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**。

3.7.5 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**。

3.7.6 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**。

3.7.7 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**。

3.7.8 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**。

3.7.9 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**。

3.7.10 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(selF.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。

3.7.11 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。

3.7.12 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')
```

3.7.13 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。

3.7.14 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”。

3.7.15 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。

3.7.16 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。

3.7.17 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)
```

3.7.18 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。

3.7.19 训练作业状态显示“审核作业初始化”

问题现象

当创建训练作业的“算法来源”选择“自定义”镜像创建训练作业时，训练作业状态显示审核作业初始化。

原因分析

自定义镜像首次运行时，需要先审核镜像。

通过审核之后才可创建作业，即当前状态为审核作业初始化。

3.7.20 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。

3.7.21 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。

3.8 预置算法运行故障

3.8.1 日志提示“label_map.pbtxt cannot be found”

问题现象

使用目标检测算法训练时，训练作业日志运行出现如下报错：
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。

原因分析

算法要求标注框为矩形标注框，提供的数据标注为非矩形，因此导致该错误发生。

处理方法

请您将数据的标注改为矩形的标注框。

建议与总结

在训练作业前，推荐您检查数据的标注是否符合算法要求（如物体检测类算法的标注框为矩形标注框）。

3.8.2 日志提示“root: XXX valid number is 0”

问题现象

日志提示“root: XXX valid number is 0”，表示训练集/验证集/测试集的有效样本量为0，例如：

```
INFO: root: Train valid number is 0.
INFO: root: Eval valid number is 0.
INFO: root: Predict valid number is 0.
```

原因分析

该日志表示数据集中的有效样本量为0，可能有如下原因：

- 数据未标注。
- 标注的数据是不符合规格的（如目标检测算法要求标注为矩形框，但是提供数据标注为非矩形框）。

处理方法

请您检查数据是否已标注，或检查数据标注是否符合算法要求。

3.8.3 日志提示“ValueError: label_map not match”

问题现象

日志提示“ValueError: label_map not match”，且打印出标签数据，如：

```
ValueError: label_map not match. {1:'apple', 2:'orange', 3:'banana', 4:'pear'} & {1:'apple', 2:'orange', 3:'banana'}
```

原因分析

训练集中的标签个数与验证集中的个数不一致，导致该错误发生。

例如，训练集中的标签共有4个，验证集中的标签只有3个。

处理方法

请您保持数据中训练集和验证集的标签数量一致。

3.8.4 日志提示“Please set the train_url to an empty obs directory”

问题现象

日志提示“Please set the train_url to an empty obs directory”。

原因分析

对于不支持断点训练的模型，如果选择训练输出路径不是空目录，会出现该报错。

处理方法

对于不支持断点训练的模型，请您将模型的输出路径train_url设置为空目录。

3.8.5 日志提示“UnboundLocalError: local variable 'epoch'”

问题现象

使用YOLOv5算法增量训练时出现如下报错：UnboundLocalError: local variable 'epoch' referenced before assignment。

原因分析

增量训练作业设置的epochs参数有误，该问题是由YOLOv5的增量训练机制引起：

- 如果第二次增量训练的epochs数值和第一次常规训练的epochs数值设置一样，则会报错。
- 如果第二次增量训练的epochs数值小于第一次常规训练的epochs数值，则增量训练会出现少训练一个epoch的现象。

处理方法

第二次增量训练设置的epochs数值需要大于第一次常规训练设置的epochs数值。

举例：对一个已经完成的训练作业（假设训练了50个epochs），想要训练更多的epochs（追加30个epochs），假设上一个训练作业的输出目录为“obs://my_bucket/train_url”，则设置参数“checkpoint_url=obs://my_bucket/train_url/last.pt”，并设置参数epochs=80（如果第二次设置参数epochs=30则增量训练只会训练29个epochs）。

3.8.6 使用订阅算法训练结束后没有显示模型评估结果

问题现象

AI Gallery中的YOLOv5算法，训练结束后没有显示模型评估结果。

原因分析

未标注的图片过多，导致没有模型评估结果。

处理方法

对所有训练数据进行标注。

3.8.7 使用 python3.6-torch1.4 版本镜像环境安装 MMCV 报错

问题现象

日志报错中存在AssertionError: MMCV==1.2.5 is used but incompatible. Please install mmcv>=1.3.1, <=1.5.0。

原因分析

MMCV的依赖与PyTorch版本不匹配。

处理方法

可参考**链接**的内容，根据PyTorch和CUDA版本安装对应版本的MMCV。

3.9 训练作业卡死

3.9.1 训练作业卡死检测定位

什么是训练作业卡死检测

训练作业在运行中可能会因为某些未知原因导致作业卡死，如果不能及时发现，就会导致无法及时释放资源，从而造成极大的资源浪费。为了节省训练资源成本，提高使用体验，ModelArts提供了卡死检测功能，能自动识别作业是否卡死，并在日志详情界面上展示，同时能配置通知及时提醒用户作业卡死。

检测规则

卡死检测主要是通过监控作业进程的状态和资源利用率来判定作业是否卡死，会启动一个协程来周期性地监控上述两个指标的变化情况。卡死检测有单实例和全实例两种检测规则，是同时生效的。

- **单实例检测**
  - 进程状态：只要训练作业单实例中的进程IO存在变化，就进入下一个检测周期。如果在多个检测周期内，所有进程IO都没有变化，则进入资源利用率检测阶段。
  - 资源利用率：在作业单实例进程IO没有变化的情况下，采集一定时间段内的GPU利用率或NPU利用率，并根据这段时间内的GPU利用率或NPU利用率的方差和中位数来判断资源使用率是否有变化。如果没有变化，则判定作业卡死。
- **全实例检测**

  资源利用率：当作业在一段时间内所有运行中的实例的GPU利用率或者NPU利用率没有变化，同时每个实例的CPU使用也低于1核，则判定作业卡死。

系统预置了卡死检测的环境变量“MA_HANG_DETECT_TIME=30”，表示检测到指标异常并持续30分钟则判定作业卡死。如果需要修改卡死检测时间，则可以修改环境变量“MA_HANG_DETECT_TIME”的值，具体操作指导请参见**管理训练容器环境变量**。

⚠ 注意

- 由于检测规则的局限性，当前卡死检测存在一定的误检率。如果是作业代码本身逻辑（如长时间sleep）导致的卡死，请忽略。
- 如果对于误检有疑问或者卡死问题无法自行解决，您可以前往**ModelArts开发者论坛**进行提问或者搜索问题。

约束限制

卡死检测仅支持资源类型为GPU和NPU的训练作业。

操作步骤

卡死检测无需额外配置，作业运行中会自动执行检测。检测到作业卡死后会在训练作业详情页提示作业疑似卡死。如需检测到卡死后发送通知（短信、邮件等）请在作业创建页面配置事件通知。

常见案例

1. **复制数据卡死**

   问题现象

   调用mox.file.copy_parallel复制数据时卡死。

   解决方案

   - 复制文件和文件夹均可采用：
     ```
     import moxing as mox
     mox.file.set_auth(is_secure=False)
     ```
   - 复制单个大文件5G以上时可采用：
     ```
     from moxing.framework.file import file_io
     ```
     查看当前moxing调用的接口版本：file_io._LARGE_FILE_METHOD，如果输出值为1则为V1版本，如果输出值为2，则为V2版本。

     V1版本修改：file_io._NUMBER_OF_PROCESSES=1

     V2版本修改：file_io._LARGE_FILE_METHOD = 1，将模式设置成V1然后用V1的方式修改规避，也可以直接file_io._LARGE_FILE_TASK_NUM=1。
   - 复制文件夹时可采用：
     ```
     mox.file.copy_parallel(threads=0,is_processing=False)
     ```

2. **训练前卡死**

   作业为多节点训练，且还未开始训练时发生卡死，可以在代码中加入os.environ["NCCL_DEBUG"] = "INFO"，查看NCCL DEBUG信息。

   - **问题现象**1

     日志中还未出现NCCL DEBUG信息时已卡死。

     解决方案1

     检查代码，检查是否有参数中未传入“master_ip”和“rank”参数等问题。
   - **问题现象2**

     分布式训练的日志中，发现有的节点含有GDR信息，而有的节点无GDR信息，导致卡死的原因可能为GDR。
     ```
     # 节点A日志
     modelarts-job-a7305e27-d1cf-4c71-ae6e-a12da6761d5a-worker-1:1136:1191 [2] NCCL INFO Channel 00 : 3[5f000] -> 10[5b000] [receive] via NET/IB/0/GDRDMA
     modelarts-job-a7305e27-d1cf-4c71-ae6e-a12da6761d5a-worker-1:1140:1196 [6] NCCL INFO Channel 00 : 14[e1000] -> 15[e9000] via P2P/IPC
     modelarts-job-a7305e27-d1cf-4c71-ae6e-a12da6761d5a-worker-1:1141:1187 [7] NCCL INFO Channel 00 : 15[e9000] -> 11[5f000] via P2P/IPC
     modelarts-job-a7305e27-d1cf-4c71-ae6e-a12da6761d5a-worker-1:1138:1189 [4] NCCL INFO Channel 00 : 12[b5000] -> 14[e1000] via P2P/IPC
     modelarts-job-a7305e27-d1cf-4c71-ae6e-a12da6761d5a-worker-1:1137:1197 [3] NCCL INFO Channel 00 : 11[5f000] -> 16[2d000] [send] via NET/IB/0/GDRDMA

     # 节点B日志
     modelarts-job-a7305e27-d1cf-4c71-ae6e-a12da6761d5a-worker-2:1139:1198 [2] NCCL INFO
     ```

```
Channel 00 : 18[5b000] -> 19[5f000] via P2P/IPC
modelarts-job-a7305e27-d1cf-4c71-ae6e-a12da6761d5a-worker-2:1144:1200 [7] NCCL INFO
Channel 00 : 23[e9000] -> 20[b5000] via P2P/IPC
modelarts-job-a7305e27-d1cf-4c71-ae6e-a12da6761d5a-worker-2:1142:1196 [5] NCCL INFO
Channel 00 : 21[be000] -> 17[32000] via P2P/IPC
modelarts-job-a7305e27-d1cf-4c71-ae6e-a12da6761d5a-worker-2:1143:1194 [6] NCCL INFO
Channel 00 : 22[e1000] -> 21[be000] via P2P/IPC
modelarts-job-a7305e27-d1cf-4c71-ae6e-a12da6761d5a-worker-2:1141:1191 [4] NCCL INFO
Channel 00 : 20[b5000] -> 22[e1000] via P2P/IPC
```

解决方案2

在程序开头设置“os.environ["NCCL_NET_GDR_LEVEL"] = '0'”关闭使用GDR，或者寻找运维人员将机器添加GDR。

- **问题现象3**

  NCCL信息中报出Got completion with error 12, opcode 1, len 32478, vendor err 129等通信信息时，说明当前网络不是很稳定。

  解决方案3

  可加入3个环境变量。

  - NCCL_IB_GID_INDEX=3： 使用RoCE v2协议，默认使用RoCE v1，但是v1在交换机上没有拥塞控制，可能丢包，而且后面的交换机不会支持v1，就无法启动。
  - NCCL_IB_TC=128：数据包走交换机的队列4通道，这是RoCE协议标准。
  - NCCL_IB_TIMEOUT=22：把超时时间设置长一点，正常情况下网络不稳定会有5秒钟左右的间断，超过5秒就返回timeout了，改成22预计有二十秒左右，算法为4.096 μs * 2 ^ timeout。

3. **训练中途卡死**
   - **问题现象1**

     检测每个节点日志是否有报错信息，某个节点报错但作业未退出导致整个训练作业卡死。

     解决方案1

     查看报错原因，解决报错。

   - **问题现象2**

     作业卡在sync-batch-norm中或者训练速度变慢。pytorch如果开了sync-batch-norm，多机会慢，因开了sync-batch-norm以后，每一个iter里面每个batch-norm层都要做同步，通信量很大，而且要所有节点同步。

     解决方案2

     关掉sync-batch-norm，或者升pytorch版本，升级pytorch到1.10。

   - **问题现象3**

     作业卡在tensorboard中，出现报错：

     ```
     writer = Sumarywriter('./path/to/log')
     ```

     解决方案3

     存储路径设为本地路径，如cache/tensorboard，不要使用OBS路径。

   - **问题现象4**

     使用pytorch中的dataloader读数据时，作业卡在读数据过程中，日志停在训练的过程中并不再更新日志。

     解决方案4

     用dataloader读数据时，适当减小num_worker。

4. **训练最后一个epoch卡死**

   问题现象

   通过日志查看数据切分是否对齐，如果未对齐，容易导致部分进程完成训练退出，而部分训练进程因未收到其他进程反馈卡死，如下图同一时间有的进程在epoch48，而有的进程在epoch49。

   ```
   loss exit lane:0.12314446270465851
   step loss is 0.29470521211624146
   [2022-04-26 13:57:20,757][INFO][train_epoch]:Rank:2 Epoch:[48][20384/all] Data Time 0.000(0.000) Net Time 0.705(0.890) Loss 0.3403(0.3792)LR 0.00021887
   [2022-04-26 13:57:20,757][INFO][train_epoch]:Rank:1 Epoch:[48][20384/all] Data Time 0.000(0.000) Net Time 0.705(0.891) Loss 0.3028(0.3466) LR 0.00021887
   [2022-04-26 13:57:20,757][INFO][train_epoch]:Rank:4 Epoch:[49][20384/all] Data Time 0.000(0.147) Net Time 0.705(0.709) Loss 0.3364(0.3414)LR 0.00021887
   [2022-04-26 13:57:20,758][INFO][train_epoch]:Rank:3 Epoch:[49][20384/all] Data Time 0.000 (0.115) Net Time 0.706(0.814) Loss 0.3345(0.3418) LR 0.00021887
   [2022-04-26 13:57:20,758][INFO][train_epoch]:Rank:0 Epoch:[49][20384/all] Data Time 0.000(0.006) Net Time 0.704(0.885) Loss 0.2947(0.3566) LR 0.00021887
   [2022-04-26 13:57:20,758][INFO][train_epoch]:Rank:7 Epoch:[49][20384/all] Data Time 0.001 (0.000) Net Time 0.706 (0.891) Loss 0.3782(0.3614) LR 0.00021887
   [2022-04-26 13:57:20,759][INFO][train_epoch]:Rank:5 Epoch:[48][20384/all] Data Time 0.000(0.000) Net Time 0.706(0.891) Loss 0.5471(0.3642) LR 0.00021887
   [2022-04-26 13:57:20,763][INFO][train_epoch]:Rank:6 Epoch:[49][20384/all] Data Time 0.000(0.000) Net Time 0.704(0.891) Loss 0.2643(0.3390)LR 0.00021887
   stage 1 loss 0.4600560665130615 mul_cls_loss loss:0.01245919056236744 mul_offset_loss 0.44759687781333923 origin stage2_loss 0.048592399805784225
   stage 1 loss:0.4600560665130615 stage 2 loss:0.048592399805784225 loss exit lane:0.10233864188194275
   ```

   解决方案

   使用tensor的切分操作对齐数据。

3.9.2 复制数据卡死

问题现象

调用mox.file.copy_parallel复制数据时卡死。

解决方案

- 复制文件和文件夹均可采用：

  ```
  import moxing as mox
  mox.file.set_auth(is_secure=False)
  ```

- 复制单个大文件5G以上时可采用：

  ```
  from moxing.framework.file import file_io
  ```

  查看当前moxing调用的接口版本：file_io._LARGE_FILE_METHOD，如果输出值为1则为V1版本，如果输出值为2，则为V2版本。

  V1版本修改：file_io._NUMBER_OF_PROCESSES=1

  V2版本修改：可以file_io._LARGE_FILE_METHOD = 1，将模式设置成V1然后用V1的方式修改规避，也可以直接file_io._LARGE_FILE_TASK_NUM=1。

- 复制文件夹时可采用：

  ```
  mox.file.copy_parallel(threads=0,is_processing=False)
  ```

3.9.3 训练前卡死

作业为多节点训练，且还未开始训练时发生卡死，可以在代码中加入os.environ["NCCL_DEBUG"] = "INFO"，查看NCCL DEBUG信息。

问题现象 1

日志中还未出现NCCL DEBUG信息时已卡死。

解决方案 1

检查代码，检查是否有参数中未传入“master_ip”和“rank”参数等问题。

问题现象 2

分布式训练的日志中，发现有的节点含有GDR信息，而有的节点无GDR信息，导致卡死的原因可能为GDR。

```
# 节点A日志
modelarts-job-a7305e27-d1cf-4c71-ae6e-a12da6761d5a-worker-1:1136:1191 [2] NCCL INFO Channel 00 : 3[5f000] -> 10[5b000] [receive] via NET/IB/0/GDRDMA
modelarts-job-a7305e27-d1cf-4c71-ae6e-a12da6761d5a-worker-1:1140:1196 [6] NCCL INFO Channel 00 : 14[e1000] -> 15[e9000] via P2P/IPC
modelarts-job-a7305e27-d1cf-4c71-ae6e-a12da6761d5a-worker-1:1141:1187 [7] NCCL INFO Channel 00 : 15[e9000] -> 11[5f000] via P2P/IPC
modelarts-job-a7305e27-d1cf-4c71-ae6e-a12da6761d5a-worker-1:1138:1189 [4] NCCL INFO Channel 00 : 12[b5000] -> 14[e1000] via P2P/IPC
modelarts-job-a7305e27-d1cf-4c71-ae6e-a12da6761d5a-worker-1:1137:1197 [3] NCCL INFO Channel 00 : 11[5f000] -> 16[2d000] [send] via NET/IB/0/GDRDMA

# 节点B日志
modelarts-job-a7305e27-d1cf-4c71-ae6e-a12da6761d5a-worker-2:1139:1198 [2] NCCL INFO Channel 00 : 18[5b000] -> 19[5f000] via P2P/IPC
modelarts-job-a7305e27-d1cf-4c71-ae6e-a12da6761d5a-worker-2:1144:1200 [7] NCCL INFO Channel 00 : 23[e9000] -> 20[b5000] via P2P/IPC
modelarts-job-a7305e27-d1cf-4c71-ae6e-a12da6761d5a-worker-2:1142:1196 [5] NCCL INFO Channel 00 : 21[be000] -> 17[32000] via P2P/IPC
modelarts-job-a7305e27-d1cf-4c71-ae6e-a12da6761d5a-worker-2:1143:1194 [6] NCCL INFO Channel 00 : 22[e1000] -> 21[be000] via P2P/IPC
modelarts-job-a7305e27-d1cf-4c71-ae6e-a12da6761d5a-worker-2:1141:1191 [4] NCCL INFO Channel 00 : 20[b5000] -> 22[e1000] via P2P/IPC
```

解决方案 2

在程序开头设置“os.environ["NCCL_NET_GDR_LEVEL"] = '0'”关闭使用GDR，或者寻找运维人员将机器添加GDR。

问题现象 3

NCCL信息中报出Got completion with error 12, opcode 1, len 32478, vendor err 129等通信信息时，说明当前网络不是很稳定。

解决方案 3

可加入3个环境变量。

- NCCL_IB_GID_INDEX=3： 使用RoCE v2协议，默认使用RoCE v1，但是v1在交换机上没有拥塞控制，可能丢包，而且后面的交换机不会支持v1，就无法启动。
- NCCL_IB_TC=128：数据包走交换机的队列4通道，这是RoCE协议标准。
- NCCL_IB_TIMEOUT=22：把超时时间设置长一点，正常情况下网络不稳定会有5秒钟左右的间断，超过5秒就返回timeout了，改成22预计有二十秒左右，算法为4.096 μs * 2 ^ timeout。

3.9.4 训练中途卡死

问题现象 1

检测每个节点日志是否有报错信息，某个节点报错但作业未退出导致整个训练作业卡死。

解决方案 1

查看报错原因，解决报错。

问题现象 2

作业卡在sync-batch-norm中或者训练速度变慢。pytorch如果开了sync-batch-norm，多机会慢，因开了sync-batch-norm以后，每一个iter里面每个batch-norm层都要做同步，通信量很大，而且要所有节点同步。

```
from sync_batchnorm import SynchronizedBatchNorm1d, DataParallelWithCallback

sync_bn = SynchronizedBatchNorm1d(10, eps=1e-5, affine=False)
sync_bn = DataParallelWithCallback(sync_bn, device_ids=[0, 1])
```

解决方案 2

关掉sync-batch-norm，或者升pytorch版本，升级pytorch到1.10。

问题现象 3

作业卡在tensorboard中，出现报错：

```
writer = Sumarywriter('./path/to/log')
```

解决方案 3

存储路径设为本地路径，如cache/tensorboard，不要使用OBS路径。

问题现象 4

使用pytorch中的dataloader读数据时，作业卡在读数据过程中，日志停在训练的过程中并不再更新日志。

解决方案 4

用dataloader读数据时，适当减小num_worker。

3.9.5 训练最后一个 epoch 卡死

问题现象

通过日志查看数据切分是否对齐，如果未对齐，容易导致部分进程完成训练退出，而部分训练进程因未收到其他进程反馈卡死，如下图同一时间有的进程在epoch48，而有的进程在epoch49。

```
loss exit lane:0.12314446270465851
step loss is 0.29470521211624146
[2022-04-26 13:57:20,757][INFO][train_epoch]:Rank:2 Epoch:[48][20384/all] Data Time 0.000(0.000) Net Time 0.705(0.890) Loss 0.3403(0.3792)LR 0.00021887
[2022-04-26 13:57:20,757][INFO][train_epoch]:Rank:1 Epoch:[48][20384/all] Data Time 0.000(0.000) Net Time 0.705(0.891) Loss 0.3028(0.3466) LR 0.00021887
[2022-04-26 13:57:20,757][INFO][train_epoch]:Rank:4 Epoch:[49][20384/all] Data Time 0.000(0.147) Net Time 0.705(0.709) Loss 0.3364(0.3414)LR 0.00021887
[2022-04-26 13:57:20,758][INFO][train_epoch]:Rank:3 Epoch:[49][20384/all] Data Time 0.000 (0.115) Net Time 0.706(0.814) Loss 0.3345(0.3418) LR 0.00021887
[2022-04-26 13:57:20,758][INFO][train_epoch]:Rank:0 Epoch:[49][20384/all] Data Time 0.000(0.006) Net Time 0.704(0.885) Loss 0.2947(0.3566) LR 0.00021887
[2022-04-26 13:57:20,758][INFO][train_epoch]:Rank:7 Epoch:[49][20384/all] Data Time 0.001 (0.000) Net Time 0.706 (0.891) Loss 0.3782(0.3614) LR 0.00021887
[2022-04-26 13:57:20,759][INFO][train_epoch]:Rank:5 Epoch:[48][20384/all] Data Time 0.000(0.000) Net Time 0.706(0.891) Loss 0.5471(0.3642) LR 0.00021887
[2022-04-26 13:57:20,763][INFO][train_epoch]:Rank:6 Epoch:[49][20384/all] Data Time 0.000(0.000) Net Time 0.704(0.891) Loss 0.2643(0.3390)LR 0.00021887
stage 1 loss 0.4600560665130615 mul_cls_loss loss:0.01245919056236744 mul_offset_loss 0.44759687781333923 origin stage2_loss 0.048592399805784225
stage 1 loss:0.4600560665130615 stage 2 loss:0.048592399805784225 loss exit lane:0.10233864188194275
```

解决方案

使用tensor的切分操作对齐数据。

3.10 训练作业运行失败

3.10.1 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。

3.10.2 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GP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“复制”，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“复制”，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。

3.10.3 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。具体请参见**示例：从 0 到 1 制作自定义镜像并用于训练（MindSpore+Ascend）**。
   - 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。

3.10.4 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X86 CPU/X86 GP规格的资源。

3.10.5 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
```

3.10.6 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。

3.10.7 订阅算法物体检测 YOLOv3_ResNet18(Ascend)训练失败报错 label_map.pbtxt cannot be found

问题现象

使用订阅算法物体检测YOLOv3_ResNet18(Ascend) 进行训练作业，训练失败报错label_map.pbtxt cannot be found。

原因分析

该报错信息表示验证集中有label在训练集中不存在，可能由于在发布数据集版本进行数据切分时，训练集比例填写为0导致发布的数据全部为验证集，所以出现上述报错。

处理方法

重新发布数据，切分比例为0.8 或者0.9重新创建训练作业进行训练。

3.10.8 训练作业训练失败报错：TypeError: unhashable type: ‘list’

问题现象

使用订阅算法**图像分类-EfficientNetB4**进行训练报错：TypeError: unhashable type: ‘list’。

原因分析

可能由于使用了多标签分类导致（即一个图片用了1个以上的标签）。

处理方法

使用单标签分类的数据集进行训练。

3.11 专属资源池创建训练作业

3.11.1 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。

3.11.2 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]”。

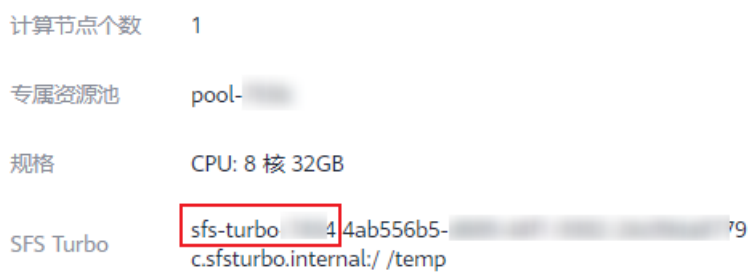
原因分析

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 3-8 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。

3.12 训练作业性能问题

3.12.1 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。

3.13 Ascend 相关问题

3.13.1 Cann 软件与 Ascend 驱动版本不匹配

问题现象

训练失败并提示“Cann软件与Ascend驱动版本不匹配”。

原因分析

当昇腾规格的训练作业在ModelArts训练平台上运行时，会自动对Cann软件与Ascend驱动的版本匹配情况进行检查。如果平台发现版本不匹配，则会立即训练失败，避免后续无意义的运行时长。

解决方案

专属资源池的**Ascend驱动版本**需与训练基础镜像中的**Cann软件版本**匹配。

ModelArts上支持的**Ascend驱动版本**可以在ModelArts专属资源池（NEW）的列表页查看“加速卡驱动”获取。

Ascend驱动版本与Cann软件版本的兼容关系如下表所示：

表 3-1 Ascend 驱动版本与 Cann 软件版本的兼容关系

Ascend 驱动版本	支持Cann软件版本	基础镜像
c81-22.0.0.3	5.1.0	mindspore_1.7.0-cann_5.1.0-py_3.7-euler_2.8.3-aarch64
		tensorflow_1.15.0-cann_5.1.0-py_3.7-euler_2.8.3-aarch64
		pytorch_1.8.1-cann_5.1.0-py_3.7-euler_2.8.3-aarch64

3.13.2 训练作业的日志出现 detect failed（昇腾预检失败）

问题现象

训练启动的日志出现如下相关错误：

```
time="2023-05-27T07:07:08Z" level=error msg="detect failed, error: dsmi-checker detect failed, error: fork/exec /home/ma-user/modelarts/bin/detect/ascend_check: no such file or directory" file="ascend_check.go:56" Command=bootstrap/run Component=ma-training-toolkit Platform=ModelArts-Service
time="2023-05-27T07:07:13Z" level=error msg="[detect] ascend-check error, exiting..." file="run_train.go:94" Command=bootstrap/run Component=ma-training-toolkit Platform=ModelArts-Service
```

原因分析

出现该问题的可能原因如下：

- 用户的自定义镜像中无ascend_check工具，导致启动预检失败。
- 用户的自定义镜像中的ascend相关工具不可用，导致预检失败。

处理方法

通过给训练作业加环境变量“MA_DETECT_TRAIN_INJECT_CODE”并将对应的值设置成0，就可以将预检功能关闭。环境变量说明参考**查看训练容器环境变量**。

4 推理部署

4.1 模型管理

4.1.1 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 4-1 查看构建日志

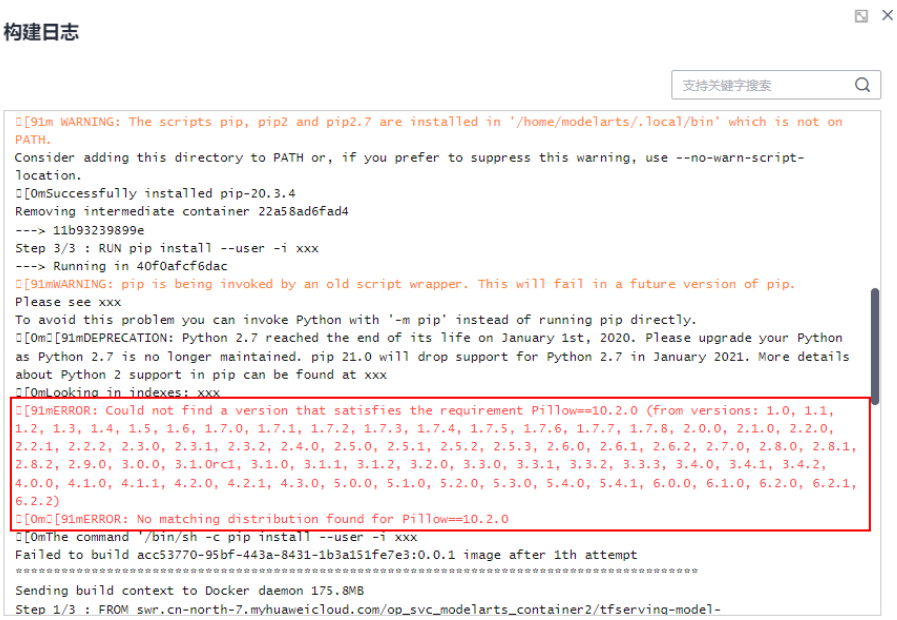
常见问题

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 4-2 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 4-3 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。基础镜像的系统架构详情可以参考**推理基础镜像列表**。

4.1.2 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。

4.1.3 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 4-4 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 4-5 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

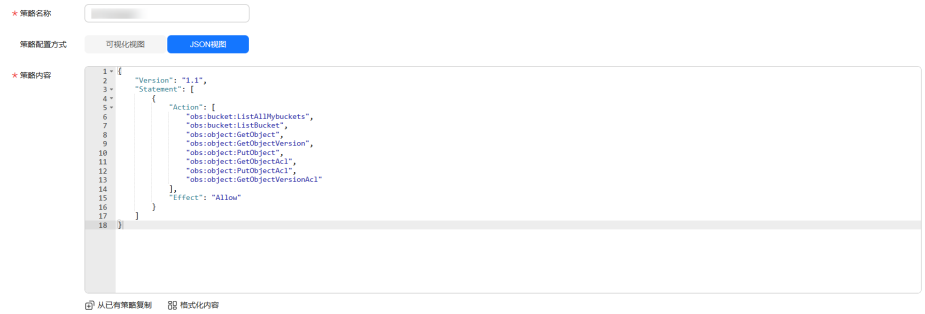
了解ModelArts依赖的OBS权限自定义策略，请参见**ModelArts依赖的OBS权限自定义策略样例**。

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 4-6 统一身份认证服务添加权限

图 4-7 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 4-8 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 4-9 子用户添加权限

4.1.4 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**。

4.1.5 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")
```

4.1.6 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。

4.1.7 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 4-10 权限管理

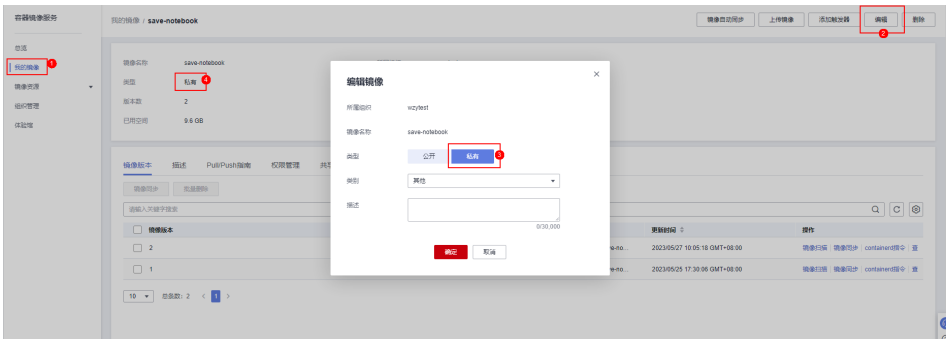
图 4-11 查看权限详情和去 IAM 修改委托权限

图 4-12 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 4-13 修改镜像类型为私有



4.1.8 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。

4.1.9 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。
2. 如果模型启动失败根本没有日志，则考虑使用推理模型调试功能，具体参见：**在开发环境中构建并调试推理镜像**。

4.1.10 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
```

4.1.11 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。

4.1.12 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。

4.1.13 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 4-14 使用动态加载

4.1.14 订阅的模型一直处于等待同步状态

问题现象

订阅的模型一直处于等待同步状态。

原因分析

订阅的模型一直处于等待同步状态，可能原因如下：

- 由于ModelArts的数据存储、模型导入以及部署上线等功能依赖OBS、SWR等服务，需获取依赖服务的授权后，才能正常使用ModelArts的相关功能。
- 您未被授权执行该操作。执行同步操作时报错：ModelArts.0108: 您未被授权执行该操作。
- 订阅已过期。执行同步操作时报错：ModelArts.5055: 订阅已过期。

处理方法

- 在权限管理页面进行依赖服务的授权。完成委托授权请参考**了解ModelArts权限配置**。
- 检查是否有OBS权限或者接口操作权限。

- 订阅已过期，可以在AI Gallery确认可以续订后，重新订阅。

4.1.15 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 4-15 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。

4.2 服务部署

4.2.1 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。

4.2.2 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 。

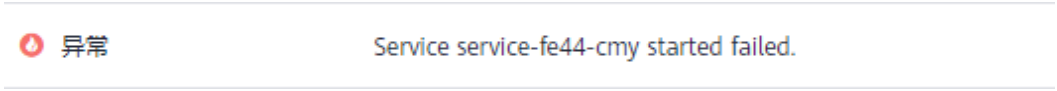
   如果您曾经找过运维人员删除过异常的实例pod，事件中可能会出现告警“服务异常，不正常的实例数为XXX”。在出现这种告警后，服务会自动拉起新的正常实例，从而恢复到正常运行状态。请您耐心等待。

4.2.3 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 4-16 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**
- **缺少OBS桶读取权限，导致无法获取模型权重信息**

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 4-17 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 4-18 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 4-19 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?**

缺少 OBS 桶读取权限，导致无法获取模型权重信息

缺少OBS桶读取权限，导致无法获取模型权重信息，请参考**ModelArts Studio（MaaS）模型服务部署失败，报错：jod failed: real time create service failed**。

4.2.4 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。

4.2.5 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。

4.2.6 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。

4.2.7 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 4-20 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）。

4.2.8 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**。

4.2.9 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**。

4.2.10 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。

4.2.11 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
```

4.2.12 IEF 节点边缘服务部署失败

问题现象

部署边缘服务时，出现“异常”状态。

原因分析 1

部署边缘服务时，使用到IEF纳管的边缘节点，就需要用户给ModelArts的委托赋予**Tenant Administrator**权限，否则将无法成功部署边缘服务。具体可参见**IEF的权限说明**。

处理方法 1

1. 在ModelArts管理控制台，选择“权限管理”。
2. 在用户名对应的“授权内容”列，单击“查看权限”，确认用户的委托权限是否已包含**Tenant Administrator**。

图 4-21 查看委托权限详情

   - 是，重新“启动”边缘服务，如果还是“异常”则联系技术支持处理。
   - 否，执行下一步，给用户添加委托权限。
3. 添加委托权限。

   说明

   如果是IAM子账号，没有修改委托权限，请联系管理员添加**Tenant Administrator**委托权限。

   a. 登录统一身份认证服务IAM管理控制台。
   b. 单击导航栏的“委托”，进入委托页面。
   c. 搜索ModelArts使用的委托，例如“modelarts_agency”，单击委托名称进入“基本信息”页面。

d. 单击“授权”，添加**Tenant Administrator**权限，按操作指引完成授权。

4. 授权完成后，重新“启动”边缘服务，观察状态是否正常。

原因分析 2

部署边缘服务时，使用到IEF纳管的边缘节点。如果部署失败，需要在IEF侧进行故障处理。

处理方法 2

请参考**IEF边缘应用常见问题**进行故障排查。

4.2.13 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 4-22 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。

4.2.14 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GP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GP，取决于的模型需要用到CPU还是GP，以及推理脚本如何编写。

4.2.15 使用 AI 市场物体检测 YOLOv3_Darknet53 算法训练后部署在线服务报错

问题现象

使用AI市场物体检测YOLOv3_Darknet53算法进行训练，将数据集切分后进行部署在线服务报错，日志如下：TypeError: Cannot interpret feed_dict key as Tensor: The name 'images:0' refers to a Tensor which does not exist. The operation, 'images', does not exist in the graph。

处理方法

- 如果切分了数据集，需要删除推理代码中“Yolov3Service”类中的如下代码：

```
self.model_inputs = {'images': 'images:0'}
self.model_outputs = {'feat_1': 'feat_1:0',
                      'feat_2': 'feat_2:0',
                      'feat_3': 'feat_3:0'}
```

- 不做数据集切分操作。如果选择未切分的数据集，算法将做纯训练场景；

4.2.16 使用预置 AI 算法部署在线服务报错 gunicorn：error：unrecognized arguments

问题现象

使用预置AI算法部署在线服务报错gunicorn：error：unrecognized arguments...

图 4-23 在线服务报错

```
[2020-12-09 02:01:03 +0000] [944] [INFO] Booting worker with pid: 944
usage: gunicorn [-h] [--model_path MODEL_PATH] [--model_name MODEL_NAME]
[--pt_server_name PT_SERVER_NAME]
[--service_file SERVICE_FILE]
gunicorn: error: unrecognized arguments: /home/mind/model/V0432/cdn_short.pt /home/mind/model/cdn_long.pt
[2020-12-09 02:01:03 +0000] [944] [INFO] Worker exiting (pid: 944)
[2020-12-09 02:01:03 +0000] [950] [INFO] Booting worker with pid: 950
usage: gunicorn [-h] [--model_path MODEL_PATH] [--model_name MODEL_NAME]
[--pt_server_name PT_SERVER_NAME]
[--service_file SERVICE_FILE]
gunicorn: error: unrecognized arguments: /home/mind/model/V0432/cdn_short.pt /home/mind/model/cdn_long.pt
[2020-12-09 02:01:04 +0000] [953] [INFO] Worker exiting (pid: 953)
usage: gunicorn [-h] [--model_path MODEL_PATH] [--model_name MODEL_NAME]
[--pt_server_name PT_SERVER_NAME]
[--service_file SERVICE_FILE]
```

原因分析

根据报错日志分析，模型目录下存在多余文件“/home/mind/model/v0432/cdn_short.pt”。

处理方法

在模型目录中删除“/home/mind/model/v0432/cdn_short.pt”文件，重新导入模型后进行部署在线服务即可正常预测。

4.2.17 内存不足如何处理?

问题现象

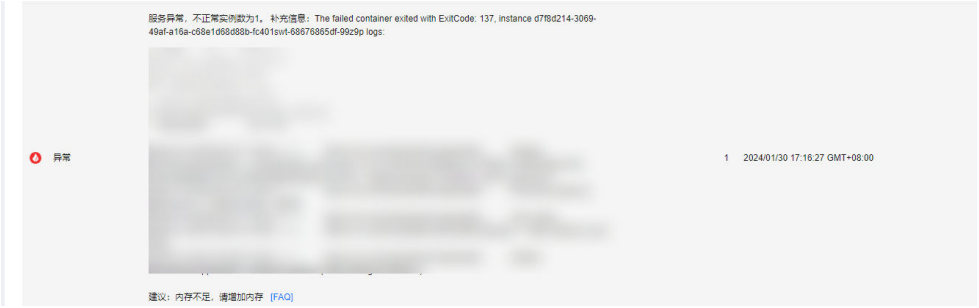
- 在部署或升级在线服务时，如果部署或升级失败，并且在事件中出现如下类似提示。

  图 4-24 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 4-25 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 4-26 选择计算节点规格

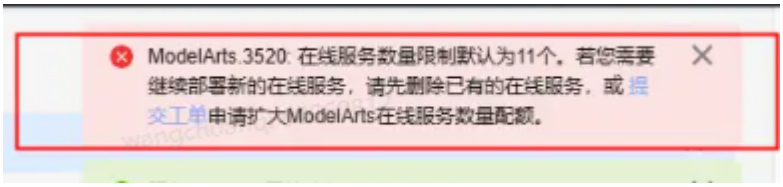
  

- 运行中服务出现告警时，需要分析是您的代码是否出现漏洞导致内存溢出、是否因为业务使用量太大需要更多的内存。如果因业务原因需要更多内存，请升级在线服务选择更大内存规格的计算节点。

4.2.18 在线服务数量限制默认为 11 个：ModelArts.3520

问题现象

ModelArts.3520:在线服务数量限制默认为11个。若您需要继续部署新的在线服务，请先删除已有的在线服务，或提交工单申请扩大ModelArts在线服务数量配额。



原因分析

在线服务数量限制默认为11个，超出了此数量。

处理方法

1. 先删除已有的在线服务，再重试。
2. 提交工单申请扩大ModelArts在线服务数量配额。

4.2.19 部署服务时报错 pod has unbound immediate PersistentVolumeClaims

问题现象

服务配置存储挂载，在部署服务时事件中有异常：资源不足，服务调度失败。pod has unbound immediate PersistentVolumeClaims。

图 4-27 存储挂载报错

问题原因

如果服务配置了存储挂载，在创建对应的存储卷时，存储卷声明(PersistentVolumeClaims)关联存储卷(PersistentVolume)有一定的时延，当检测服务检测到未绑定的存储卷声明时，会报该异常。

解决方法

一般情况下，不用处理，短时间内会恢复正常。如果长时间未恢复正常，请联系技术支持。

4.3 服务预测

4.3.1 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 4-28 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。

4.3.2 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。

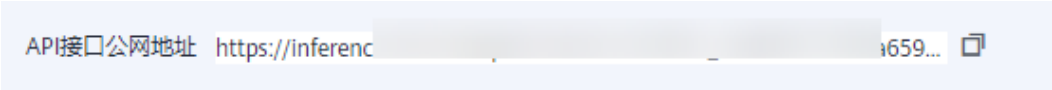
APIG.0101 预测地址错误

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 4-29 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 4-30 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）**。

4.3.3 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。

4.3.4 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。

4.3.5 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 4-31 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GP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：

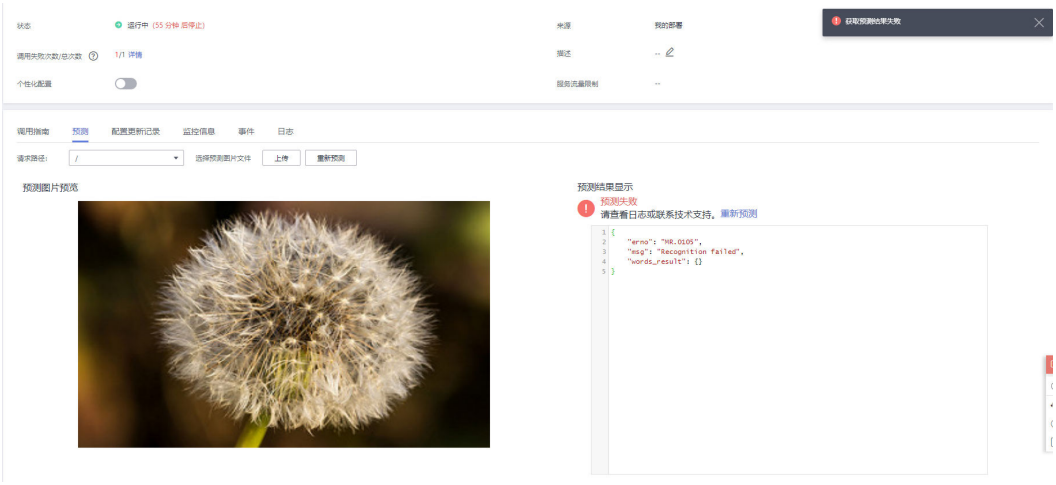
   - 缩小预测请求数量看是否问题还复现，如果不复现是因为负载过大导致服务进程退出，需要扩容实例数量或者提升规格。
   - 推理代码本身存在错误，请排查推理代码解决。

4.3.6 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": {}}。

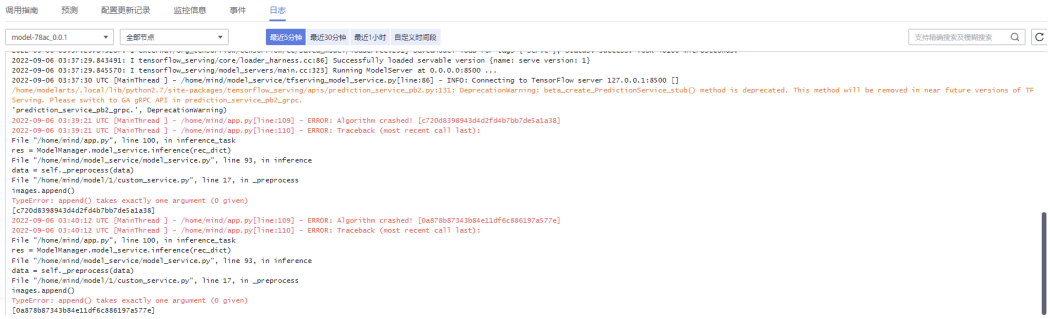
图 4-32 预测报错



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 4-33 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**。

4.3.7 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。

4.3.8 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。所有预测接口在APIG网关存在默认超时时间40秒或者60秒，具体依据APIG实例初始化配置。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。

4.3.9 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”。

4.3.10 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。

4.3.11 时序预测-time_series_v2 算法部署在线服务预测报错

问题现象

在线服务预测报错：ERROR: data is shorter than windows。

原因分析

该报错说明预测使用的数据行数小于window超参值。

在使用订阅算法时序预测-time_series_v2训练时，超参：window设置为60。训练完成并创建模型后，部署在线服务，进行预测，当预测的数据行数小于window超参值时，日志中有报错信息：ERROR: data is shorter than windows 。

处理方法

- 增加预测数据行数大于训练作业window超参值。
- 复制训练作业，修改window超参值。

5 MoXing

5.1 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
   - ModelArts使用MoXing复制报错：**some parts are failed when download**
   - 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')
   - 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。
- 训练解密组件异常

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. 报错timeout时，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. 出现错误码403时检查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，请更换为一个可用的OBS路径。
4. 出现报错some part are failed when download。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。
6. 出现错误码503或报错的requestid为空时联系OBS服务查看对应桶的后台监控，确认是否出现带宽或并发打满导致的流控现象，调整后重试即可。
7. 出现**can't find xxx.temp** 或**FileNotFoundError**报错信息，检测是否多个进程同时下载了同一文件并避免该场景出现。

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。

5.2 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"
```

5.3 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')
  ```


5.4 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')
```

5.5 MoXing 安装失败或使用卡死

问题现象

1. 安装MoXing报错Invalid version。
2. 使用MoXing出现卡死现象。

原因分析

1. 24.1版本以上的pip限制了Python库的版本号命名规则，旧版本MoXing不符合。
2. 旧版本MoXing并发下载时调用了Python原生库queue，该库存在卡死风险。

处理方法

升级MoXing版本到2.3.11及以上，对上述问题进行了修复。

6 API/SDK

6.1 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
```

6.2 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。

6.3 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存在“flavor_id”参数，CPU规格的专属资源池不支持使用“flavor_id”参数。

6.4 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
```


7 资源池

7.1 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。

例如创建Standard专属资源池时，创建失败，提示配额不足，请单击“申请最大配额”，在“新建工单”页面，根据您的需求，填写相关参数。其中，“问题描述”项请填写需要调整的内容和申请原因。填写完毕后，勾选协议并单击“提交”。

图 7-1 配额不足资源创建失败

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“资源管理 > 标准算力集群（Standard Cluster）”，进入标准算力集群（Standard Cluster）页面。
2. 您可以通过单击集群列表右上角的“失败记录”右侧数字，查看当前处于失败状态的资源池订单信息。

   鼠标移至“失败”状态可查看失败原因。

图 7-2 创建失败资源池信息

3. 在弹框中找到创建失败/变更失败任务，单击操作列的“查看详情”，即可看到该操作失败的具体原因。

说明

列表中查看订单记录（不包括逻辑子池等），每条记录最多保留90天

7.2 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 7-1 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。 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。
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。

隔离码	分类	子类	异常中文描述	检测方法
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网络连接异常。
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。
A20 0001	节点管理	驱动升级	GPU升级。	节点正在执行GPU驱动升级。
A20 0002	节点管理	驱动升级	NPU升级。	节点正在执行NPU驱动升级。
A20 0008	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。

隔离码	分类	子类	异常中文描述	检测方法
A05 0933	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。
A05 0931	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。
A05 0932	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。
A05 0804	硬件故障	硬件故障	通过硬件告警发现。	硬件告警监控发现。 请在事件中心授权修复，详细请参考**事件中心授权运维**。

配置节点事件类告警通知

节点故障事件会上报到AOM，您可以在AOM配置短信、邮件等通知方式。

说明

以下步骤基于AOM2.0配置。

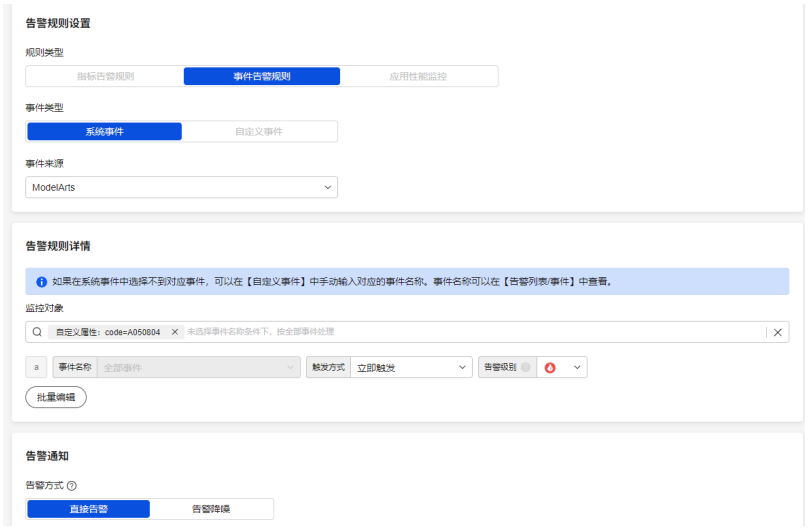
步骤1 登录AOM控制台

步骤2 在左侧导航栏选择“告警中心 > 告警规则”，在右上角单击“创建告警规则”。

步骤3 设置告警规则（以故障码A050804为例）。

- 规则类型：选择事件告警规则。
- 事件类型：选择系统事件。
- 事件来源：选择ModelArts。
- 监控对象：监控对象通过自定义属性进行筛选，格式为code=${故障码}。
  本示例中选择“code=A050804”事件，触发方式选择“立即触发”。

图 7-3 告警规则设置



- 告警方式：选择“直接告警”。
- 告警通知（可选）：如果需要将告警通过邮件、手机方式通知您，可在告警通知处，为此告警规则配置行动规则。如果此处无行动规则，请新建告警行动规则。

----结束

7.3 资源池推理服务一直初始化中如何解决

问题现象

创建资源池时作业类型选择了推理服务，资源池创建成功后推理一直显示“环境初始化。

原因分析

专属池网段和推理微服务dispatcher网段冲突，导致专属池上的VPCEP终端节点无法创建，该region无法使用此网段创建包含推理服务的资源池。

处理方法

选择其他网段的ModelArts网络重建资源池即可解决网段冲突问题。

7.4 专属资源池关联 SFS Turbo 显示异常

问题现象 1

专属资源池关联SFS Turbo时显示异常，关联失败。

图 7-4 关联异常

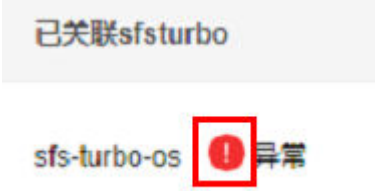
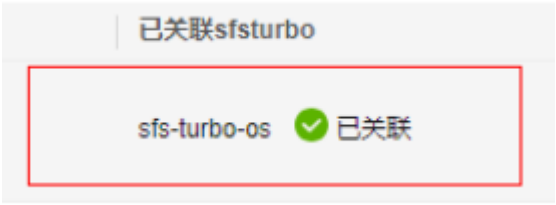


图 7-5 报错提示

问题现象 2

网络操作解除关联SFS Turbo后状态仍显示已关联且无报错信息，而解除关联按钮置灰不可操作。同时该网络的解除关联SFS Turbo按钮置灰不可操作。

图 7-6 关联 SFS Turbo 状态



原因分析

ModelArts缺少SFS Turbo委托权限导致关联或解除关联失败。

处理方法

需要您给ModelArts配置SFS Turbo委托权限，配置步骤请参考最佳实践的“委托授权ModelArts云服务使用SFS Turbo”章节。

7.5 公共池训练作业排队超过设置的配额后，无法提交作业

问题现象

公共池训练作业排队超过设置的配额后，无法提交作业。

图 7-7 报错示例

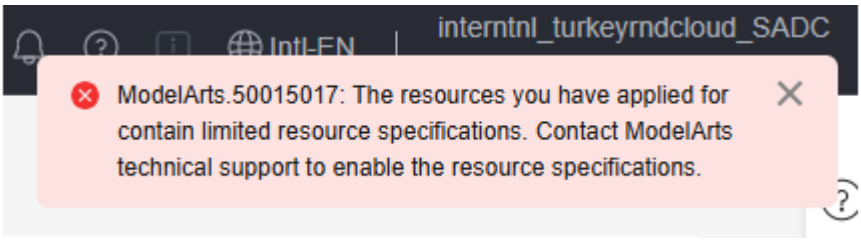
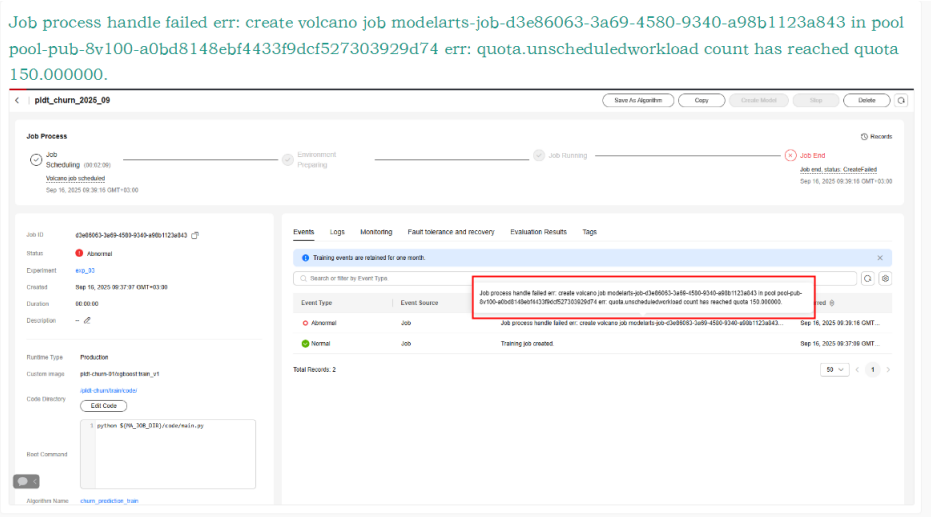


图 7-8 报错提示



问题原因

存在默认排队任务配额（如上图的为150），超过默认排队配额训练作业会报错。

解决方法

公共资源池排队任务存在默认配额，超过默认排队配额，训练作业会报错。用户可以删除不使用的训练作业或者提工单申请修改配额。

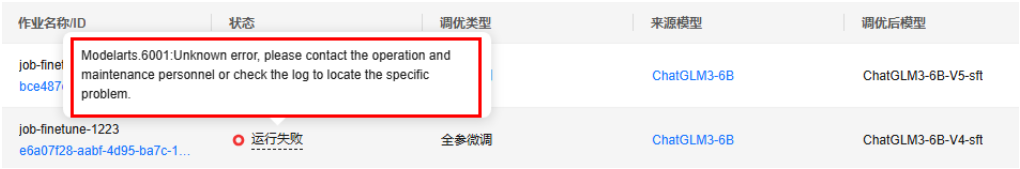
8 ModelArts Studio（MaaS）

8.1 ModelArts Studio（MaaS）模型调优作业运行失败，报错：Modelarts.6001

问题现象

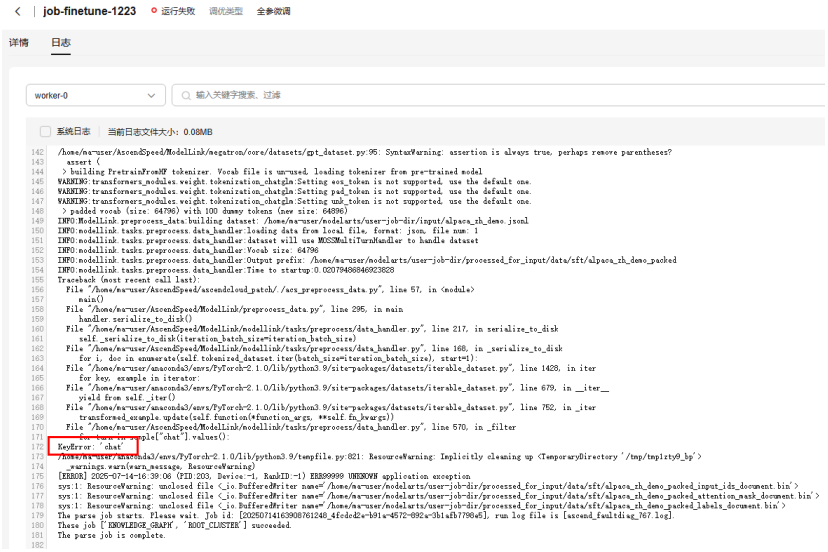
在“模型调优”页面使用Qwen2.5-14B模型创建调优作业，“状态”显示为“运行失败”，报错：Modelarts.6001:Unknown error, please contact the operation and maintenance personnel or check the log to locate the specific problem.

图 8-1 调优作业运行失败



在“模型调优”页面的“作业名称/ID”列，单击作业ID，在“作业详情 > 日志”页签，日志报错：KeyError: 'chat'。

图 8-2 日志报错



原因分析

调优数据集格式选择错误：控制台选择数据集格式为MOSS，但实际上传的数据集格式为Alpaca。

处理方法

重新创建调优任务，选择正确的数据集格式。

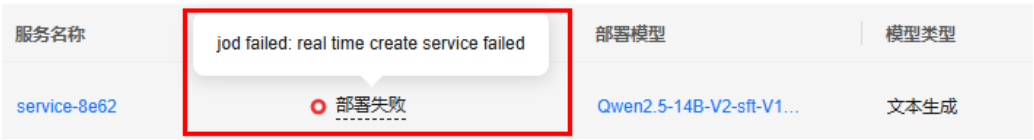
- 关于调优数据集异常的日志说明，请参见**ModelArts Studio（MaaS）调优数据集异常日志说明**。
- 关于模型与数据集格式说明，请参见**使用ModelArts Studio（MaaS）调优模型**。

8.2 ModelArts Studio（MaaS）模型服务部署失败，报错：jod failed: real time create service failed

问题现象

使用调优后的Qwen2.5-14B模型部署模型服务，在“在线服务 > 我的服务”页面，服务状态显示为“部署失败”，报错：jod failed: real time create service failed。

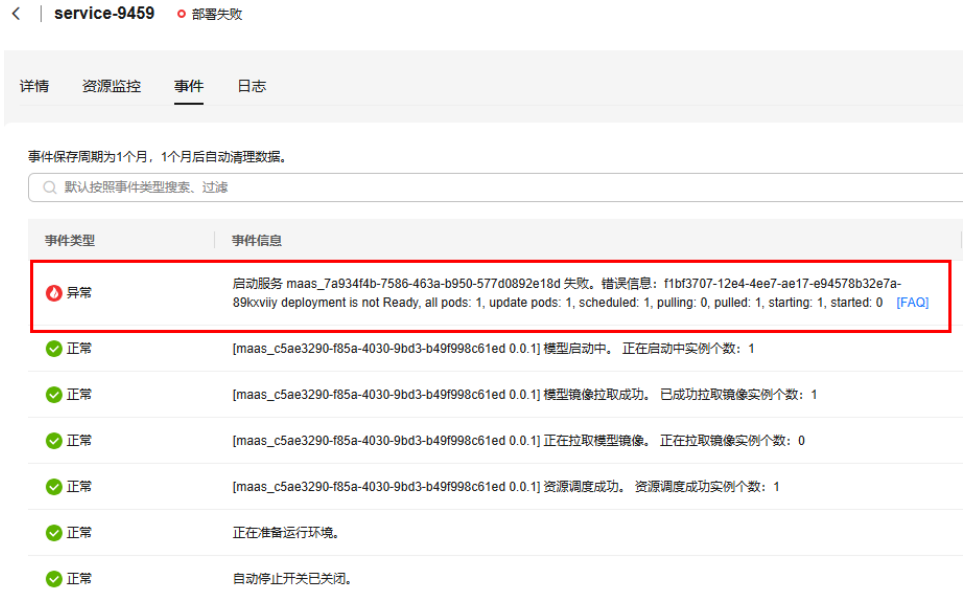
图 8-3 部署失败



在“我的服务”页面，单击服务名称，在“服务详情 > 事件”页签，事件类型为“异常”，事件信息如下：

启动服务 maas_7a934f4b-7586-463a-b950-577d0892e18d 失败。错误信息：f1bf3707-12e4-4ee7-ae17-e94578b32e7a-89kxviiy deployment is not Ready, all pods: 1, update pods: 1, scheduled: 1, pulling: 0, pulled: 1, starting: 1, started: 0

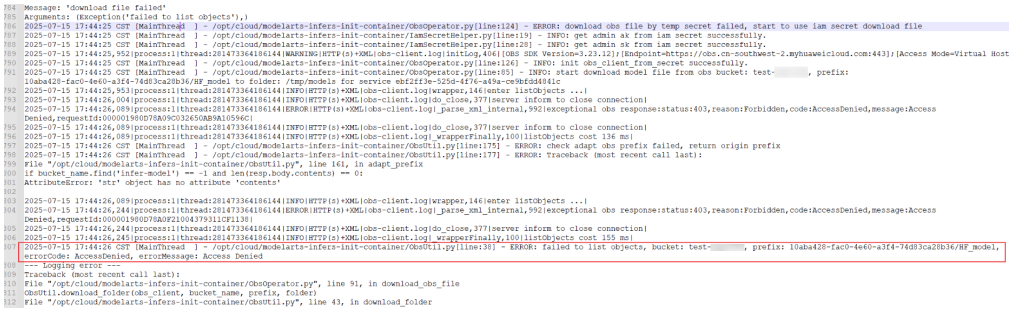
图 8-4 事件异常



在“我的服务”页面，单击服务名称，在“服务详情 > 日志”页签，日志报错如下：

2025-07-15 17:44:26 CST [MainThread] - /opt/cloud/modelarts-infers-init-container/ObsUtil.py[line:38] - ERROR: failed to list objects, bucket: test-***-***, prefix: 10aba428-fac0-4e60-a3f4-74d83ca28b36/HF_model, errorcode: AccessDenied, errorMessage: Access Denied

图 8-5 日志报错



原因分析

缺少OBS桶读取权限，导致无法获取模型权重信息。

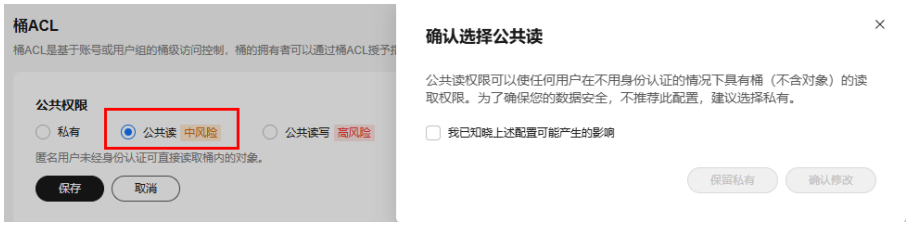
处理方法

通过桶ACL授予指定账号或用户组特定的访问权限。更多信息，请参见**配置桶ACL**。

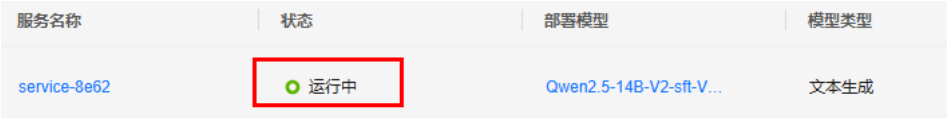
1. 在**OBS管理控制台**左侧导航栏，选择“桶列表”。
2. 在“桶列表”页面，单击桶名称，在左侧导航栏，选择“权限控制 >桶ACL”。
3. 在“桶ACL”页面，按需设置访问权限。下文以“公共权限”设置为“公共读”为例进行说明。

“公共权限”选择“公共读”，单击“保存”，在弹出的对话框，勾选“我已知晓上述配置可能产生的影响”，单击“确认修改”。

图 8-6 选择公共读



4. 重新启动部署失败的服务。
   a. 在**ModelArts Studio（MaaS）控制台**左侧导航栏，选择“在线推理”。
   b. 在“在线推理 > 我的服务”页面，在“操作”列单击部署失败服务对应的“启动”。
      服务状态显示为“运行中”，表示部署成功。



8.3 在 ModelArts Studio（MaaS）创建 Qwen2-0.5B 或 Qwen2-1.5B 模型的 LoRA 微调类型的调优任务，显示创建失败

问题现象

创建LoRA调优任务，选择支持Modellink框架类型的模型Qwen2-0.5B，数据集选择MOSS格式的JSONL数据，添加超参设置，创建调优任务失败。

关键日志报错：

```
AttributeError: 'Parameter' object has no attribute 'main_grad'
```

原因分析

Qwen2-0.5B或Qwen2-1.5B模型不支持也不建议PP切分。

问题影响

训练无法进行。

处理方法

对于Qwen2-0.5B或Qwen2-1.5B模型，LoRA微调时不支持PP切分。请将切分策略PP设置为1。

8.4 在 ModelArts Studio（MaaS）创建训练任务，显示创建失败

问题现象

创建训练任务时，选择Qwen2.5-7B、Qwen2.5-14B、Qwen2.5-32B、Qwen2.5-72B-1K或者Qwen2-VL-7B模型，创建训练任务失败。

关键日志报错（出现以下任意报错）：

- 报错1：
  ```
  [INFO|trainer.py:2278] 2025-01-09 20:49:47,170 >>  Will skip the first 5 epochs then the first 0 batches
  ```
- 报错2：
  ```
  [rank0]: RuntimeError: Cannot find sufficient samples, consider increasing dataset size.
  ```

原因分析

数据集过少，导致训练失败。

说明

其中，增量预训练会packing，将短sample拼成seq_len长度进行训练，因此原数据条数多不意味着处理后samples多。

问题影响

训练失败或者训练结果与预期不符。

处理方法

增加数据集数量。

9 Lite Server

9.1 GPU 裸金属服务器使用 EulerOS 内核误升级如何解决

问题现象

GP Vnt1裸金属服务器，操作系统为EulerOS 2.9(基于CentOS制作的Linux发行版)，经常遇到服务器重启后，操作系统内核无故升级，导致系统上原安装的nvidia-driver等软件无法使用，只能卸载重新安装。

原因分析

分析EulerOS内核是如何在不知情的情况下升级的：

1. 首先查看当前操作系统内核。
   ```
   [root@Server-ddff ~]# uname -r
   4.18.0-147.5.1.6.h934.eulerosv2r9.x86_64
   ```
2. 一般执行如下升级命令，就会导致自动下载和安装高级内核版本。
   ```
   yum update -y
   ```
   执行后查看当前可用内核，发现已经新增了内核h998：
   ```
   [root@Server-ddff ~]#
   [root@Server-ddff ~]# cat /boot/grub2/grub.cfg |grep "menuentry "
   menuentry 'EulerOS (4.18.0-147.5.1.6.h998.eulerosv2r9.x86_64) 2.0 (SP9x86_64)' --class euleros --
   class gnu-linux --class gnu --class os --unrestricted $menuentry_id_option 'gnulinux-4.18.0-147.5.
   1.6.h934.eulerosv2r9.x86_64-advanced-f6aefacb-f2d3-4809-b708-6ad0357037f5' {
   menuentry 'EulerOS (4.18.0-147.5.1.6.h934.eulerosv2r9.x86_64) 2.0 (SP9x86_64)' --class euleros --
   class gnu-linux --class gnu --class os --unrestricted $menuentry_id_option 'gnulinux-4.18.0-147.5.
   1.6.h934.eulerosv2r9.x86_64-advanced-f6aefacb-f2d3-4809-b708-6ad0357037f5' {
   menuentry 'EulerOS (0-rescue) 2.0 (SP9x86_64)' --class euleros --class gnu-linux --class gnu --class os
   --unrestricted $menuentry_id_option 'gnulinux-0-rescue-advanced-f6aefacb-f2d3-4809-b708-6ad
   0357037f5' {
   [root@Server-ddff ~]#
   ```
3. 查看假如reboot（尚未reboot）后默认选择的内核版本：
   ```
   [root@Server-ddff ~]# grub2-editenv list
   saved_entry=EulerOS (4.18.0-147.5.1.6.h998.eulerosv2r9.x86_64) 2.0 (SP9x86_64)
   boot_success=0
   [root@Server-ddff ~]#
   ```
   发现默认系统内核已经变为h998，reboot后就会生效。 此时如果重启那么内核版本就被升级了。

处理方法

下文中假设当前服务器的内核版本是为4.18.0-147.5.1.6.h934.eulerosv2r9.x86_64，介绍如何避免操作系统内核自动升级。

1. 操作系统内核升级生效，必然需要服务器重启， 因此重启reboot前需要查看当前默认选择的内核版本：
   ```
   [root@Server-ddff ~]# grub2-editenv list
   saved_entry=EulerOS (4.18.0-147.5.1.6.h998.eulerosv2r9.x86_64) 2.0 (SP9x86_64)
   boot_success=0
   [root@Server-ddff ~]#
   ```
   如上发现reboot后内核为4.18.0-147.5.1.6.h998.eulerosv2r9.x86_64，和当前内核版本h934不一致，则需要重新设置内核版本与当前版本一致。
2. 查看当前内核版本，并且锁定reboot后默认启动的内核版本，执行如下命令：
   ```
   grub2-set-default 'EulerOS (4.18.0-147.5.1.6.h934.eulerosv2r9.x86_64) 2.0 (SP9x86_64)'
   ```
3. 执行后查看默认启动的内核版本是否和上述设置的相同：
   ```
   [root@Server-ddff ~]# grub2-editenv list
   saved_entry=EulerOS (4.18.0-147.5.1.6.h934.eulerosv2r9.x86_64) 2.0 (SP9x86_64)
   boot_success=0
   [root@Server-ddff ~]#
   ```
   发现和当前内核一致，因此即使reboot也不会更改服务器的内核版本。

如果希望升级指定的操作系统内核，也可以执行grub2-set-default进行设置默认启动内核版本。但操作系统内核升级可能带来的问题。例如在操作系统内核4.18.0-147.5.1.6.h934.eulerosv2r9.x86_64 下安装的nvidia-driver-515，由于执行了yum update并reboot服务器， 发现再次执行nvidia命令时报错：

```
[root@Server-ddff ~]# nvidia-smi
NVIDIA-SMI has failed because it couldn't communicate with the NVIDIA driver. Make sure that the latest NVIDIA driver is installed and running.
[root@Server-ddff ~]#
```

此时只能安装nvidia-driver-515以及配套的cuda版本。

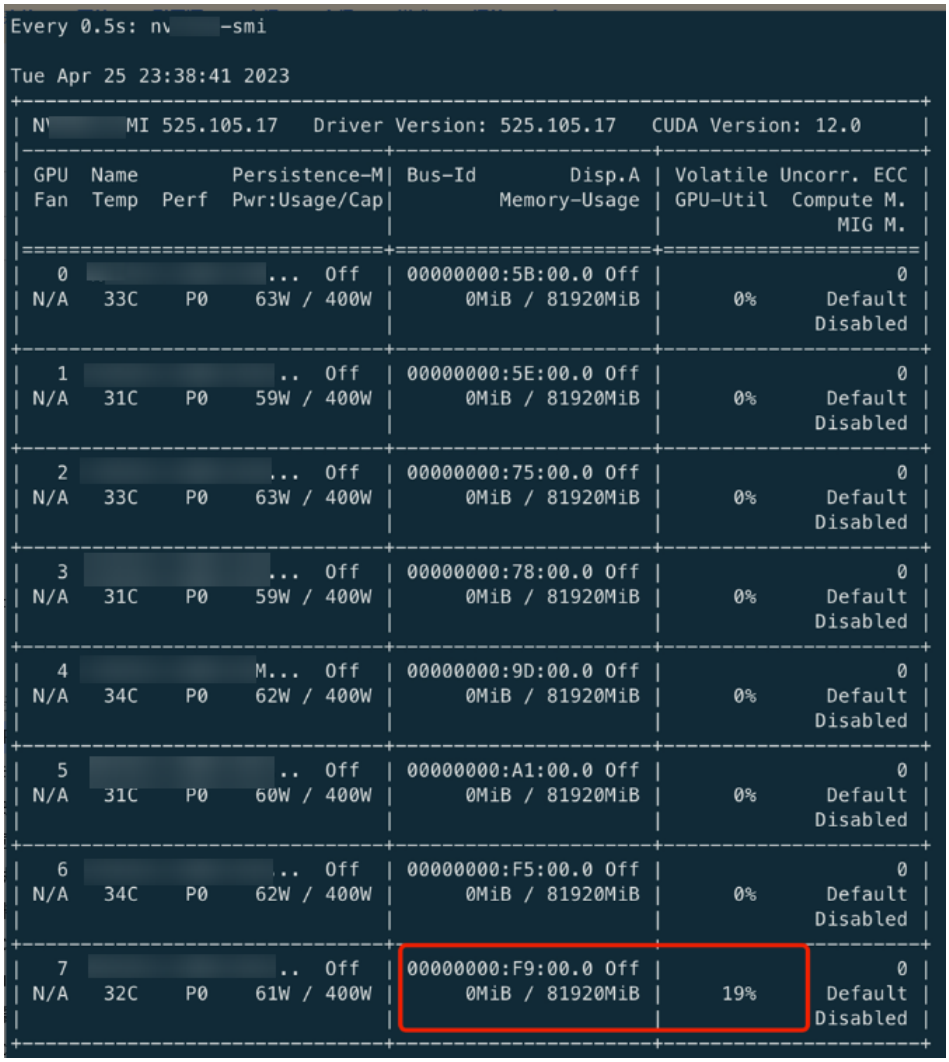
9.2 GPU A 系列裸金属服务器没有任务但 GPU 被占用如何解决

问题现象

服务器没有任务，但GPU显示被占用。

截图示例如下：

图 9-1 显卡运行状态



处理方法

```
nvidia-smi -pm 1
```

9.3 GPU A 系列裸金属服务器无法获取显卡如何解决

问题现象

在A系列裸金属服务器上使用PyTorch一段时间后，出现获取显卡失败的现象，报错如下：

```
> torch.cuda.is_available()
/usr/local/lib/python3.8/dist-packages/torch/cuda/__init__.py:107: UserWarning: CUDA initialization: Unexpected error from cudaGetDeviceCount(). Did you run some cuda functions before calling NumCudaDevices() that might have already set an error? Error 802: system not yet initialized (Triggered internally at ../c10/cuda/CUDAFunctions.cpp:109.)
  return torch._C._cuda_getDeviceCount() > 0
False
```


原因分析

Error 802原因为缺少fabricmanager，可能由于以下原因导致nvidia-fabricmanager.service不工作：

- 可能系统资源不足、如内存不足、内存泄露。
- 硬件故障、如IB网络或者GPU互联设备故障等。
- 没安装nvidia-fabricmanager组件或被误卸载。

处理方法

- 如果未安装fabricmanager，则需安装该组件。
- 如果已安装fabricmanager，运行以下命令重启fabricmanager.service。

```
systemctl restart  nvidia-fabricmanager.service
```

说明

建议您进一步定位出nvidia-fabricmanager不工作原因，避免该问题再次发生。

9.4 GPU 裸金属服务器无法 Ping 通如何解决

问题现象

在华为云使用GPU裸金属服务器时， 服务器绑定EIP（华为云弹性IP服务）后，出现无法ping通弹性公网IP现象。

原因分析

1. 查看当前GPU裸金属服务器的安全组的入方向规则的配置，发现仅开通了TCP协议的22端口。
2. ping命令是一种基于ICMP协议（Internet Control Message Protocol）的网络诊断工具，利用ICMP协议向目标主机发送数据包并接收返回的数据包来判断网络连接质量。当安全组的入方向规则中没有包含ICMP协议，就会出现ping不通的问题。

处理方法

在当前安全组的入方向规则中添加一条规则，基本协议选择ICMP协议，详细配置如下表所示，添加规则步骤请参考**添加安全组规则**。

表 9-1 入方向规则

方向	协议/应用	端口	源地址
入方向	ICMP	全部	0.0.0.0/0

华为云安全组支持的协议参考可参考下表。

表 9-2 入方向规则

协议	端口	说明
协议	端口	说明
FTP	21	FTP服务上传和下载文件。
SSH	22	远程连接Linux弹性云服务器。
Telnet	23	使用Telnet协议访问网站。
SMTP	25	SMTP服务器所开放的端口，用于发送邮件。 基于安全考虑，TCP 25端口出方向默认被封禁，申请解封请参考**TCP 25端口出方向无法访问时怎么办？** 。
HTTP	80	使用HTTP协议访问网站。
POP3	110	使用POP3协议接收邮件。
IMAP	143	使用IMAP协议接收邮件。
HTTPS	443	使用HTTPS协议访问网站。
SQL Server	1433	SQL Server的TCP端口，用于供SQL Server对外提供服务。
SQL Server	1434	SQL Server的TCP端口，用于返回SQLServer使用了哪个TCP/IP端口。
Oracle	1521	Oracle通信端口，弹性云服务器上部署了Oracle SQL需要放行的端口。
MySQL	3306	MySQL数据库对外提供服务的端口。
Windows Server Remote Desktop Services	3389	Windows远程桌面服务端口，通过这个端口可以连接Windows弹性云服务器。
代理	8080	8080端口常用于WWW代理服务，实现网页浏览，实现网页浏览。如果您使用8080端口，访问网站或使用代理服务器时，需要在IP地址后面加上：8080。安装Apache Tomcat服务后，默认服务端口为8080。
NetBIOS	137、138、139	NetBIOS协议常被用于Windows文件、打印机共享和Samba。 • 137、138：UDP端口，通过网上邻居传输文件时使用的端口。 • 139：通过这个端口进入的连接试图获得NetBIOS/SMB服务。

9.5 GPU A 系列裸金属服务器 RoCE 带宽不足如何解决?

问题现象

GP Ant8支持RoCE网卡， Ubuntu20.04场景，在进行nccl-tests时，总线带宽理论峰值可达90GB/s，但实际测试下来的结果只有35GB/s。

原因分析

“nv_peer_mem”是一个Linux内核模块，它允许支持P2P(Peer-to-Peer)的NV GPU直接进行内存访问(DMA)。这意味着数据可以直接在多个GPU之间传输，而无需经过CPU或系统内存，这可以显著降低延迟并提高带宽。

所以既然nccl-tests能正常测试， 但是达不到预期，可能是nv_peer_mem异常。

处理方法

- 查看nv_peer_mem是否已安装。
  ```
  dpkg -i | grep peer
  ```
  如果未安装则需要安装，如果已安装则进入下一检测项。
- 查看该软件是否已经加载至内核。
  ```
  lsmod | grep peer
  ```
  如果没有则需要重新加载至内核，执行如下命令进行加载：
  ```
  /etc/init.d/nv_peer_mem start
  ```
  如果执行失败，可能是未加载nv_peer_mem.conf至/etc/infiniband/中或nv_peer_mem不在/etc/init.d/中。
- 如果是找不到相关文件的问题，可以搜索相关文件在哪里，然后复制到指定目录，例如可执行如下命令：
  ```
  cp /tmp/nvidia-peer-memory-1.3/nv_peer_mem.conf  /etc/infiniband/
  cp /tmp/nvidia-peer-memory-1.3/debian/tmp/etc/init.d/nv_peer_mem   /etc/init.d/
  ```

9.6 使用 SFS 盘出现报错 rpc_check_timeout:939 callbacks suppressed

问题现象

弹性文件服务（Scalable File Service，SFS）提供按需扩展的高性能文件存储（NAS），可以在裸金属服务器中通过网络协议挂载使用，SFS支持NFS和CIFS的网络协议。在使用裸金属服务器时， 将数据放在SFS盘中， 并发建立多个NFS链接、并发的读写数据、做大模型训练。 但有时候会出现读取速度变慢的现象，并且SFS提示报错"rpc_check_timeout:939 callbacks suppressed"。

原因分析

根据SFS客户端日志分析出现问题的时间点发现，SFS盘连接的客户端个数较多，在问题的时间点并发读取数据，I/O超高；当前SFS服务端的机制是：当SFS盘的性能到上限时，就会出现I/O排队。I/O排队造成处理时间超过 1 分钟时，客户端内核会打印"rpc_check_timeout:939 callbacks suppressed"日志。这个日志只是说明某个I/O处理

时间超过 1 分钟了，不会造成数据丢失。客户端有重试机制，等峰值过去后，所有I/O最终都会正确处理。所以理论上，出现该错误日志， 并不会造成数据丢失， 只是SFS客户端I/O速度变慢或卡顿，但最终会争取处理。

处理方法

- 结合当前购买的SFS盘性能规划业务， 建议不要运行到性能上限。
- 可以购买多个SFS Turbo实例分担业务压力， 或者更换高性能的SFS盘。
- 一个SFS实例容量建议不要太大，建议以同样的成本换成购买多个SFS实例。

9.7 华为云 CCE 集群纳管 GPU 裸金属服务器由于 CloudInit 导致纳管失败的解决方案

问题现象

创建出3台GPU裸金属服务器，使用A节点制作镜像，用于在CCE纳管裸金属服务器时，使用该镜像，但是纳管后发现服务器A纳管失败，剩下两台服务器纳管成功。

原因分析

在CCE纳管过程中，需要通过cloudinit userdata机制拉取cce-agent，但是在服务器上查看没有拉cce-agent的动作，理论上该动作是cloudinit中的脚本在创建时自动执行的，可能是由于安装脚本没有注入userdata或者注入了但未执行。

经查看是由于userdata未执行，可能原因为服务器A制作镜像时没有清理残留目录导致，即：

镜像里面“/var/lib/cloud/instances”残留了制作镜像机器（后面称模板机）的实例ID信息，如果制作镜像不清理“/var/lib/cloud/*”就会导致用该镜像再重装模板机时，cloud-init根据残留目录（含实例ID）判断已经执行过一次，进而不会再执行user-data里面的脚本。

而使用该镜像的服务器B和C，由于实例ID信息和镜像中残留的服务器A实例ID不同，就会执行user-data，所以CCE能纳管成功。

处理方法

制作镜像前，清理“/var/lib/cloud/”目录下的所有信息，请参考**清理临时文件**步骤对文件进行清理，然后再制作镜像。CCE重新纳管服务器A时， 使用最新制作的镜像即可。

9.8 裸金属服务器 EulerOS 升级 NetworkManager-config-server 导致 SSH 连接故障解决方案

问题现象

裸金属服务器EulerOS 2.8系统下，使用yum update -y命令，导致软件NetworkManagre-config-server升级到高版本，出现SSH连接故障无法访问。

原因分析

1. 查看yum命令历史，发现执行了“yum update -y”，“yum update -y”命令是用于在Linux操作系统上更新软件包的命令。其中，选项-y表示在更新时自动确认所有提示信息，而不需要手动输入“y”确认。

   请注意，使用此命令将会检查您系统中已安装的软件包并更新至最新版本。

   图 9-2 yum 命令历史

   ```
   [root@devserver-ei-cto-office-cn9-4 ~]# yum history
   ID     | Command line             | Date and time    | Action(s)      | Altered
   -------------------------------------------------------------------------------
        9 | install lrzsz -y         | 2023-05-27 11:35 | Install        |    1
        8 | install -y vim gcc gcc-c | 2023-05-23 19:28 | Install        |   70
        7 | update -y                | 2023-05-23 19:23 | I, U           |  437 E<
        6 | install cloud-utils-grow | 2020-02-17 17:48 | Install        |    1 >
        5 | install gcc perl strace  | 2020-02-17 17:46 | Install        |    3  <
        4 | install python2-requests | 2020-02-17 17:17 | Install        |    8 >
        3 | install python2-six      | 2020-02-17 17:17 | Install        |    1
        2 | install cloud-init       | 2020-02-17 17:14 | Install        |   22
        1 |                          | 2020-02-17 16:24 | Install        |  809 EE
   [root@devserver-ei-cto-office-cn9-4 ~]#
   ```

2. 查看NetworkManager配置：

   ```
   NetworkManager --print-config
   ```

   配置内容如下：

   ```
   # NetworkManager configuration: /etc/NetworkManager/NetworkManager.conf (lib: 00-server.conf)

   [main]
   # plugins=ifcfg-rh,ibft
   # rc-manager=symlink
   # auth-polkit=true
   # dhcp=dhclient
   no-auto-default=*
   ignore-carrier=*

   [logging]
   # backend=journal
   # audit=false
   ```

   发现“no-auto-default=*”是打开的状态，“no-auto-default=*”含义是关闭DH Client，无法使用DHCP获取IP。正常情况下裸金属服务器这个参数是被注释的状态。

   - 当服务器有网卡配置文件， NetworkManager.service实现将VPC子网分配的私有IP写入网卡配置文件中。NetworkManager.service会优先读取网卡配置文件中的IP设置为主机IP， 此时无论DH Cient是否关闭，服务器都可以获取分配IP。
   - 当服务器没有网卡配置文件时，DH Client开启，此时服务器会分配私有IP。如果关闭DH Client，则服务器无法获取私有IP。

图 9-3 查看 NetworkManager 配置

```
[root@xb-test- ~]# ll /etc/sysconfig/network-scripts/
total 224
-rw-r--r--. 1 root root   254 Jul 22  2019 ifcfg-lo
-rwxr-xr-x. 1 root root  2062 Jul 22  2019 ifdown
-rwxr-xr-x. 1 root root   649 Jul 22  2019 ifdown-bnep
-rwxr-xr-x. 1 root root  6423 Jul 22  2019 ifdown-eth
-rwxr-xr-x. 1 root root   772 Jul 22  2019 ifdown-ippp
-rwxr-xr-x. 1 root root  4540 Jul 22  2019 ifdown-ipv6
lrwxrwxrwx. 1 root root    11 Jul 22  2019 ifdown-isdn -> ifdown-ippp
-rwxr-xr-x. 1 root root  2068 Jul 22  2019 ifdown-post
-rwxr-xr-x. 1 root root   873 Jul 22  2019 ifdown-routes
-rwxr-xr-x. 1 root root  1462 Jul 22  2019 ifdown-sit
-rwxr-xr-x. 1 root root  1625 Jul 22  2019 ifdown-Team
-rwxr-xr-x. 1 root root  1560 Jul 22  2019 ifdown-TeamPort
-rwxr-xr-x. 1 root root  1466 Jul 22  2019 ifdown-tunnel
-rwxr-xr-x. 1 root root  5413 Jul 22  2019 ifup
-rwxr-xr-x. 1 root root 12206 Jul 22  2019 ifup-aliases
-rwxr-xr-x. 1 root root   909 Jul 22  2019 ifup-bnep
-rwxr-xr-x. 1 root root 13656 Jul 22  2019 ifup-eth
-rwxr-xr-x. 1 root root 12078 Jul 22  2019 ifup-ippp
-rwxr-xr-x. 1 root root 11895 Jul 22  2019 ifup-ipv6
lrwxrwxrwx. 1 root root     9 Jul 22  2019 ifup-isdn -> ifup-ippp
-rwxr-xr-x. 1 root root   654 Jul 22  2019 ifup-plip
-rwxr-xr-x. 1 root root  1068 Jul 22  2019 ifup-plusb
-rwxr-xr-x. 1 root root  5004 Jul 22  2019 ifup-post
-rwxr-xr-x. 1 root root  2004 Jul 22  2019 ifup-routes
-rwxr-xr-x. 1 root root  3307 Jul 22  2019 ifup-sit
-rwxr-xr-x. 1 root root  1759 Jul 22  2019 ifup-Team
-rwxr-xr-x. 1 root root  1880 Jul 22  2019 ifup-TeamPort
-rwxr-xr-x. 1 root root  2825 Jul 22  2019 ifup-tunnel
-rwxr-xr-x. 1 root root  1840 Jul 22  2019 ifup-wireless
-rwxr-xr-x. 1 root root  5425 Jul 22  2019 init.ipv6-global
-rw-r--r--. 1 root root 20603 Jul 22  2019 network-functions
-rw-r--r--. 1 root root 31025 Jul 22  2019 network-functions-ipv6
[root@xb-test- ~]# NetworkManager --print-config
# NetworkManager configuration: /etc/NetworkManager/NetworkManager.conf (lib: 00-server.conf)

[main]
# plugins=ifcfg-rh,ibft
# rc-manager=symlink
# auth-polkit=true
# dhcp=dhclient
no-auto-default=*
ignore-carrier=*

[logging]
# backend=journal
# audit=false
[root@xb-test- ~]#
```

图 9-4 查看网络配置

```
[root@xb-test- ~]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp69s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether fa:16:3e:7c:d4:d0 brd ff:ff:ff:ff:ff:ff
3: enp189s0f0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN group default qlen 1000
    link/ether 8c:2a:8e:9a:8d:72 brd ff:ff:ff:ff:ff:ff
4: enp189s0f1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN group default qlen 1000
    link/ether 8c:2a:8e:9a:8d:73 brd ff:ff:ff:ff:ff:ff
5: enp189s0f2: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN group default qlen 1000
    link/ether 8c:2a:8e:9a:8d:74 brd ff:ff:ff:ff:ff:ff
6: enp189s0f3: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN group default qlen 1000
    link/ether 8c:2a:8e:9a:8d:75 brd ff:ff:ff:ff:ff:ff
[root@xb-test- ~]# ifconfig
enp189s0f0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 8c:2a:8e:9a:8d:72  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp189s0f1: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 8c:2a:8e:9a:8d:73  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp189s0f2: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 8c:2a:8e:9a:8d:74  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp189s0f3: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 8c:2a:8e:9a:8d:75  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp69s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        ether fa:16:3e:7c:d4:d0  txqueuelen 1000  (Ethernet)
        RX packets 12  bytes 780 (780.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@xb-test- ~]#
```

命令“yum update -y”或“yum update NetworkManagre-config-server”，都会将NetworkManagre-config-server软件升级，高版本的NetworkManagre-config-server会将参数no-auto-default=*是打开的状态，又因当前镜像无网卡配置文件导致ip获取不到，从而使得SSH连接失败。

处理方法

在EulerOS2.8操作系统，NetworkManagre-config-server是一个无用的软件包，无需安装 。执行以下命令卸载NetworkManagre-config-server，并重启NetworkManager服务，重新尝试SSH连接，验证网络是否恢复。

```
# 卸载 NetworkManagre-config-server
rpm -e NetworkManager-config-server
# 重启 NetworkManager 服务
systemctl restart NetworkManager
```

10 Lite Cluster

10.1 资源池创建失败的原因与解决方法?

本文主要介绍在ModelArts资源池创建失败时，如何查找失败原因，并解决问题。

问题定位

您可以参考以下步骤，查看资源池创建失败的报错信息，并根据相应的解决方法解决问题：

1. 登录ModelArts控制台，左侧导航栏单击“资源管理 > 轻量算力集群（Lite Cluster）”，单击资源池列表右上角的“失败记录”右侧数字，查看创建失败的资源池订单信息。
   鼠标移至“失败”状态可查看失败原因。
2. 在弹框中找到创建失败/变更失败任务，单击操作列的“查看详情”，即可看到该操作失败的具体原因。

说明

列表中查看订单记录（不包括逻辑子池等），每条记录最多保留90天

解决方法

- **ModelArts权限管理的委托权限不足，导致创建失败?**
  解决方法请参见**ModelArts创建委托授权**。
- **申请的资源中包含受限购买的资源规格，导致购买失败?**
  当前modelarts.bm.npu.arm.8snt9b3.d为受限购买，需要提前联系ModelArts运营或提工单申请开通资源规格。

图 10-1 报错信息

- **ECS、EVS配额不足，导致创建失败?**

  集群所需的ECS实例数、内存大小、CPU核数和EVS硬盘大小资源会超出华为云默认提供的资源配额，因此需要申请扩大配额。解决方法请参见**申请扩大资源配额**。

  图 10-2 报错信息（1）

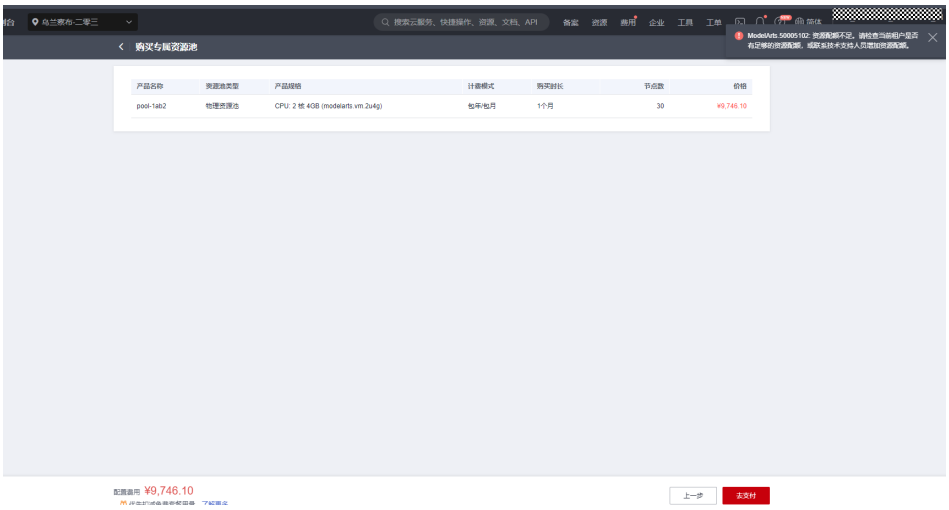
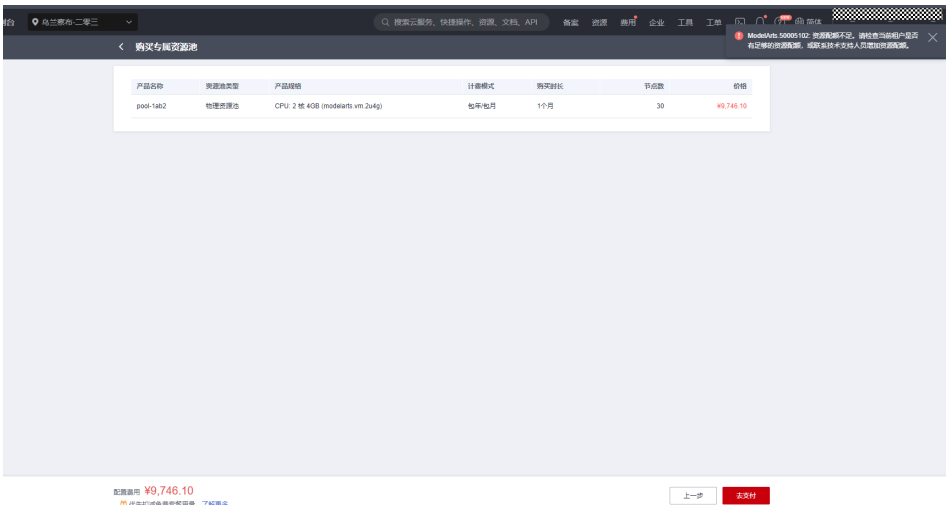
  

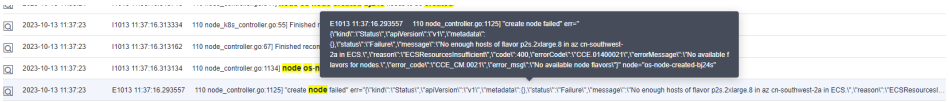
  图 10-3 报错信息（2）

- **资源售罄或容量不足，导致创建失败?**

  减少资源池节点数量，或提工单给ModelArts申请更多资源。

  图 10-4 报错信息

- **ECS、BMS节点创建失败?**

  查看资源池失败报错信息：

  - 包含错误码，如：Ecs.0000时，可查看**弹性云服务器 ECS_错误码**查看详细的错误信息及处理措施。

  - 包含错误码，如：BMS.0001时，可查看**裸金属服务器 BMS_错误码**查看详细的错误信息及处理措施。
  - 包含错误码，如：CCE.01400001时，可查看**云容器引擎 CCE_错误码**查看详细的错误信息及处理措施
  - 其他报错请提工单联系ModelArts运维进一步定位解决。
- **集群纳管节点失败?**

  查看资源池失败报错信息：
  - 查看资源池失败报错信息，包含错误码，如：CCE.01400001时，可查看**云容器引擎 CCE_错误码**查看详细的错误信息及处理措施。
  - 其他报错请提工单联系ModelArts运维进一步定位解决。
- **集群容器网段不足，导致创建失败?**

  图 10-5 报错信息

  用户可根据实际业务场景和节点规模，自定义配置容器网段，配置方式如下：

  a. ModelArts Standard池，资源池创建阶段指定容器网段，根据实际需要设置更大的容器网段。

  图 10-6 设置容器网段

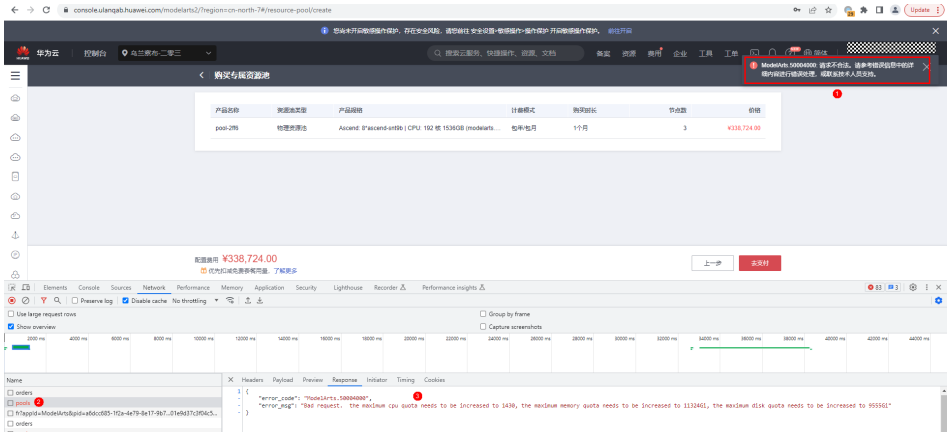
  b. ModelArts Lite池，选择/创建具有更大容器网段的CCE集群。CCE容器网段配置参见**网络规划**。
- **账号冻结导致创建失败?**

  查看资源池失败报错信息，存在"frozen deposit fail"，表示账号冻结导致资源创建失败。检查账号状态和资源欠费情况，账号解冻后重新购买资源。
- **订单取消导致资源创建失败?**

  查看资源池失败报错信息，存在"the operation is canceled by customer"，表示资源池对应订单已取消，取消原因可能为超时未支付、用户自主取消，需重新购买。
- **其他错误**

可通过F12查看浏览器请求信息，选择标红的pools接口，查看响应里的详细报错信息，如下图所示。通过错误提示修正输入参数后再次提交订单。

图 10-7 报错信息



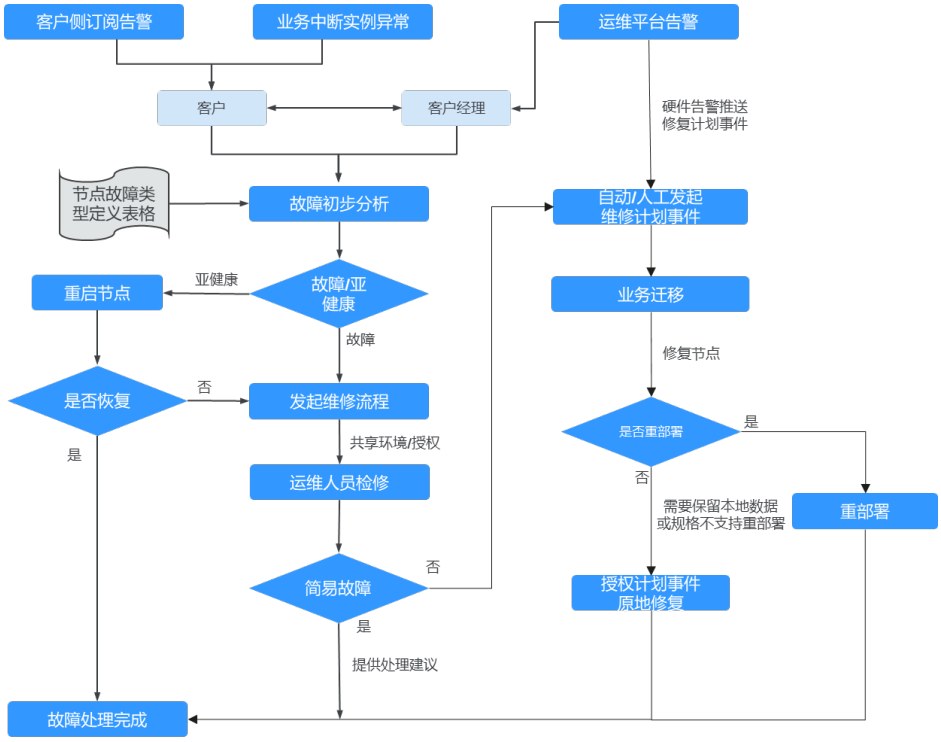
如CCE集群不可用，请检查CCE集群版本和状态。报错信息如下：

```
{
    "error_code": "ModelArts.50004000",
    "error_msg": "Bad request. spec.clusters[0].providerId: Invalid value: \"77f6f112-a631-11eb-8dae-0255ac100b0d\": the cluster 77f6f112-a631-11eb-8dae-0255ac100b0d is not available"
}
```

10.2 如何定位和处理 Cluster 资源池节点故障

故障说明和处理建议

图 10-8 Lite 池故障处理流程



对于ModelArts Lite资源池，每个节点会以DaemonSet方式部署node-agent组件，该组件会检测节点状态，并将检测结果写到K8S NodeCondition中。同时，节点故障指标默认会上报到AOM，您可在AOM配置告警通知。

当发生节点异常时，在故障初步分析阶段，您可先按**表10-1**识别是否为亚健康并自助进行处理，如果不是，则为故障，请联系客户经理发起维修流程（如果无客户经理可提交工单）。

表 10-1 节点故障指标定义

NodeCondition Type	分类	子类	异常中文描述	检测方法	处理建议
NT_NPU_DEVICE	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。	可能是亚健康，建议先重启节点，如果重启节点后未恢复，发起维修流程。
NT_NPU_NET	NPU	链路	npu dcmi net异常。	NPU网络连接异常。	可能是亚健康，建议先重启节点，如果重启节点后未恢复，发起维修流程。
NT_NPU_CARD_LOSE	NPU	掉卡	NPU卡丢失。	节点规格的NPU卡数和k8sNode中可调度的卡数不一致。	可能是亚健康，建议先重启节点，如果重启节点后未恢复，发起维修流程。
NT_NPU_OTHER	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。	发起维修流程。
NT_NPU_ECC_COUNT	NPU	显存	NPU ECC次数达到维修阈值。	NPU的HBM总的多Bit Ecc隔离地址记录达到64个。	发起维修流程。
NT_NET_NTP_CHECK	Runtime	其他	ntp异常。	ntpd或者chronyd服务异常。	发起维修流程。
NT_KUBE_DISK_READONLY_CHECK	Runtime	其他	Kubelet硬盘只读	以下目录只读： /mnt/paas/kubernetes/kubelet	发起维修流程。

NodeCondition Type	分类	子类	异常中文描述	检测方法	处理建议
NT_GPU_SMI_ECC_CHECK	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。	可能是亚健康，建议先重启节点，如果重启节点后未恢复，发起维修流程。

NodeCondition Type	分类	子类	异常中文描述	检测方法	处理建议
NT_GPU_SMI_ERROR	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。	发起维修流程。
NT_GPU_SMI_RUNTIME	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。	发起维修流程。
NT_GPU_SMI_ECC_COUNT	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。	发起维修流程。
NT_GPU_CARD_LOSE	GPU	掉卡	GPU卡丢失。	节点规格的GPU卡数和以下任意值不相等： 1. lspci可见GPU卡数。 2. nvidia-smi可见卡数。 3. k8s可调度卡数不相等。	发起维修流程。
NT_GPU_SMI_INFOROM_ERROR	GPU	其他	infoROM告警。	执行nvidia-smi的返回信息中包含“infoROM is corrupted”告警。	发起维修流程。
NT_GPU_OTHER	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。	发起维修流程。
NT_NET_IB_CHECK	IB	链路	IB网卡异常。	ibstat查看网卡非Active状态。	可能是亚健康，建议先重启节点，如果重启节点后未恢复，发起维修流程。

部分故障模式通过华为云运维平台硬件告警监控发现，相关的故障定义和处理建议如**表10-2**所示。同时，这类故障产生时默认会上报AOM事件，您可在AOM配置告警通知。

表 10-2 节点故障事件定义

故障码	分类	子类	异常中文描述	检测方法	处理建议
A050804	硬件故障	硬件故障	通过硬件告警发现。	硬件告警监控发现。	请在事件中心授权修复，详细请参考**事件中心授权运维**。
A050202	Runtime	其他	k8s节点notReady	登录CCE集群查看告警节点状态	确认异常后，优先将此节点设置为不可调度，并将业务pod调度到其他节点

配置节点指标类告警通知

节点故障指标(nt_npg)默认会上报到AOM，您可以在AOM配置短信、邮件等通知方式。

说明

以下步骤基于AOM2.0配置。

nt_npg指标type=2是无效值，nt_npg{type="NT_NPU_CARD_LOSE"} !=2表示过滤掉无效值。

步骤1 登录AOM控制台

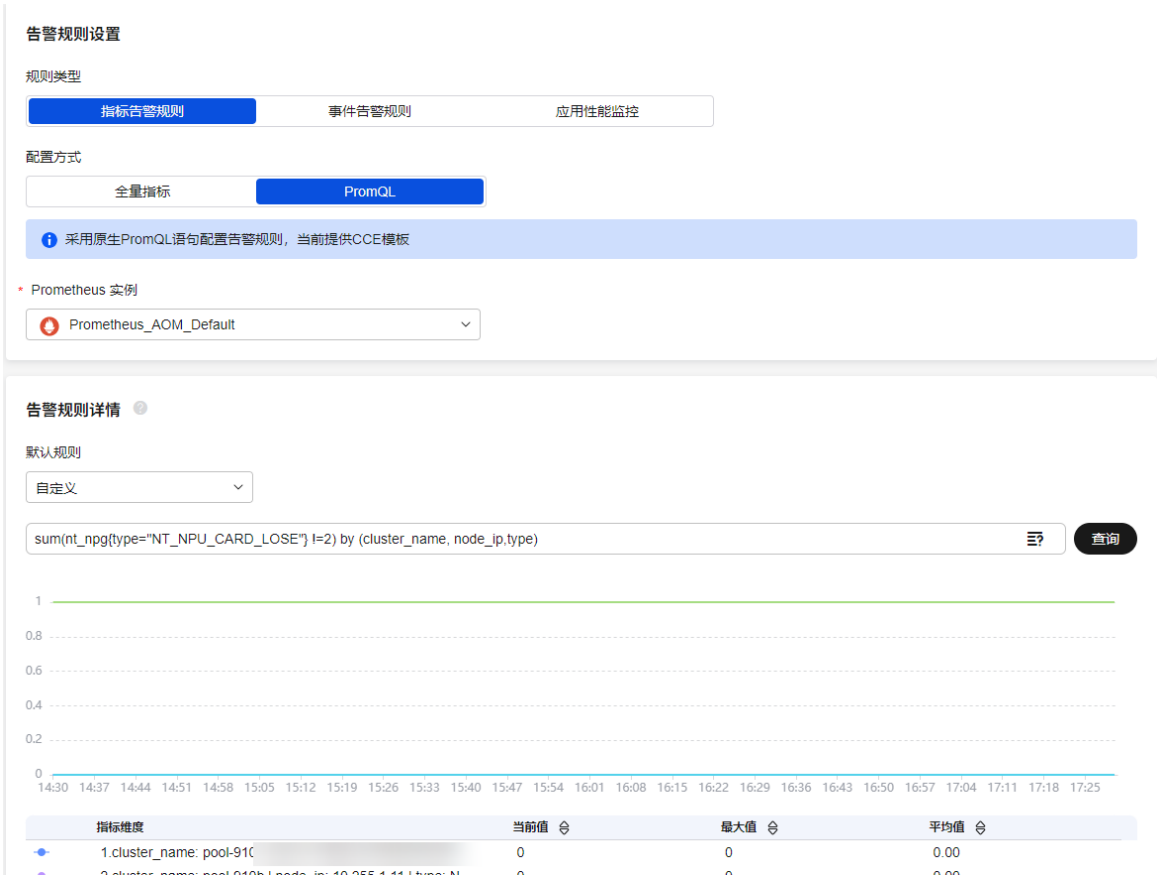
步骤2 在左侧导航栏选择“告警中心 > 告警规则”，单击“创建告警规则”。

步骤3 设置告警规则（以NPU掉卡为例）。

- 规则类型：选择指标告警规则。
- 配置方式：选择PromQL。
- 默认规则：选择自定义，命令行输入框：

```
sum(nt_npg{type="NT_NPU_CARD_LOSE"} !=2) by (cluster_name, node_ip,type)
```

图 10-9 告警规则设置



- 告警条件：选择触发条件为持续时间1分钟，产生重要告警。
- 告警通知（可选）：如果需要将告警通过邮件、手机方式通知您，可在告警通知处，为此告警规则配置行动规则。如果此处无行动规则，请新建告警行动规则。

----结束

配置节点事件类告警通知

节点故障事件会上报到AOM，您可以在AOM配置短信、邮件等通知方式。

说明

以下步骤基于AOM2.0配置。

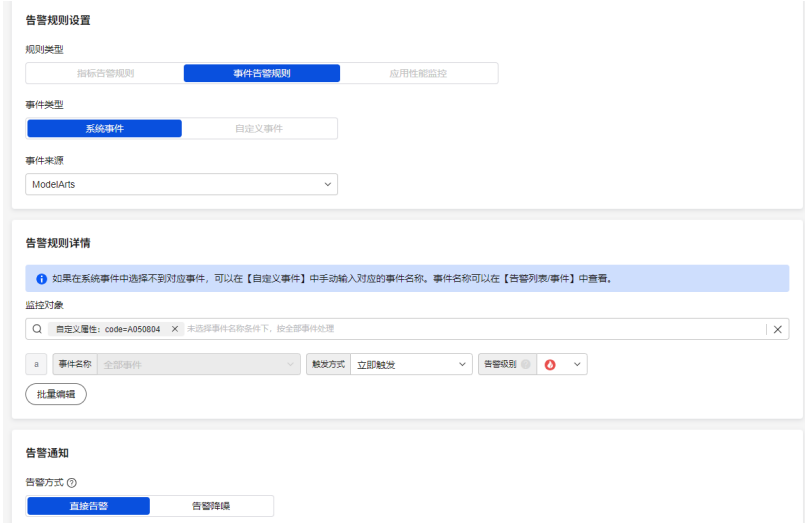
步骤1 登录AOM控制台

步骤2 在左侧导航栏选择“告警中心 > 告警规则”，在右上角单击“创建告警规则”。

步骤3 设置告警规则（以故障码A050804为例）。

- 规则类型：选择事件告警规则。
- 事件类型：选择系统事件。
- 事件来源：选择ModelArts。
- 监控对象：监控对象通过自定义属性进行筛选，格式为code=${故障码}。
  本示例中选择“code=A050804”事件，触发方式选择“立即触发”。

图 10-10 告警规则设置



- 告警方式：选择“直接告警”。
- 告警通知（可选）：如果需要将告警通过邮件、手机方式通知您，可在告警通知处，为此告警规则配置行动规则。如果此处无行动规则，请新建告警行动规则。

----结束

10.3 特权池信息数据显示均为 0%如何解决?

问题现象

特权池基本信息页面数据均显示为0%（如CPU使用率、内存使用率、加速卡使用率、加速卡显存使用率）。

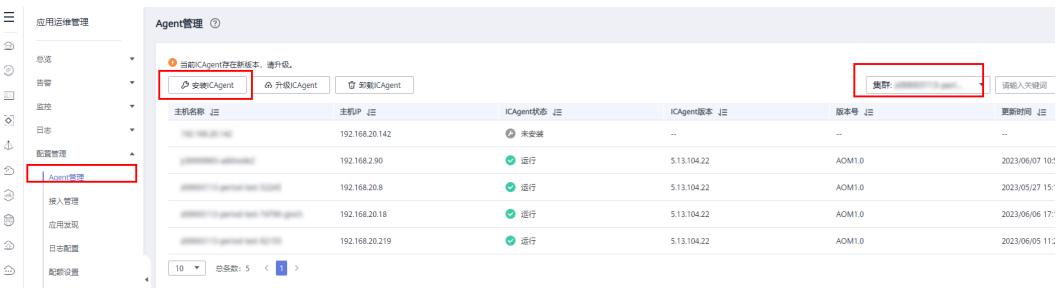
原因分析

原因是集群没有安装ICAgent。新建特权池时默认会安装ICAgent，可能由于用户自行卸载ICAgent，导致资源池数据显示异常。

处理方法

登录“应用运维管理”控制台，在“配置管理 > Agent管理”中，选择未安装ICAgent的集群，并单击“安装ICAgent”。

图 10-11 安装 ICAgent



说明

建议不要随意卸载ICAgent，否则会影响特权池详情页的参数显示。

10.4 重置节点后无法正常使用?

问题现象

当ModelArts Lite的CCE集群在资源池上只有一个节点，且用户设置了volcano为默认调度器时，在ModelArts侧进行重置节点的操作后，节点无法正常使用，节点上的POD会调度失败。

原因分析

在ModelArts侧进行节点重置后，modelarts-os会向节点添加准入污点，进行节点准入，而因为集群volcano没有污点容忍，且集群内只有一个节点，导致volcano无法启动，进而导致modelarts-os节点上管理污点的maos-node-agent容器无法启动，使得污点无法被自动清理。

处理方法

- （推荐）解决方案一（按需使用volcano调度器）：
  a. CCE页面上修改默认调度器为kube-scheduler。
  b. 删除maos-node-agent的pod（重启pod）。
  c. CCE页面上删除节点上的污点A200008。
  d. ModelArts页面上重置节点。

  该方案的缺点：用户新建负载时需要手动指定调度器为volcano，参考**指导**。
- 解决方案二（默认全部使用volcano调度器）：
  a. CCE页面上配置中心修改默认调度器为kube-scheduler。
  b. 删除maos-node-agent的pod（重启pod）。
  c. CCE页面上删除节点上的污点A200008。
  d. ModelArts上重置节点。
  e. CCE页面上配置中心修改默认调度器为volcano。

  该方案的缺点：后续对ModelArts的节点做相关操作如重置、升级驱动等可能会出现节点异常无法启动的情况。

10.5 如何根据 Cluster 节点故障自动恢复业务

AI服务器单点硬件故障不可避免，在大规模算力使用场景下，资源池规模越大存在硬件故障的可能性越高。当发生硬件故障时可能会影响节点上服务的正常运行。

Lite Cluster节点巡检到故障后，会上报故障信息到k8s节点和AOM云服务，详情请见**Cluster节点故障上报**。ModelArts提供故障感知、通知能力，配合业务在原生k8s中进行快速恢复。

针对故障节点信息，自动恢复业务的解决方案大致流程如下：

步骤1：获取故障节点信息（以Watch k8s节点方式为例）

步骤2：用Taint隔离集群中的故障节点

步骤3：重新下发业务

自动恢复业务后，处理故障节点请参见**如何定位和处理Cluster资源池节点故障**。

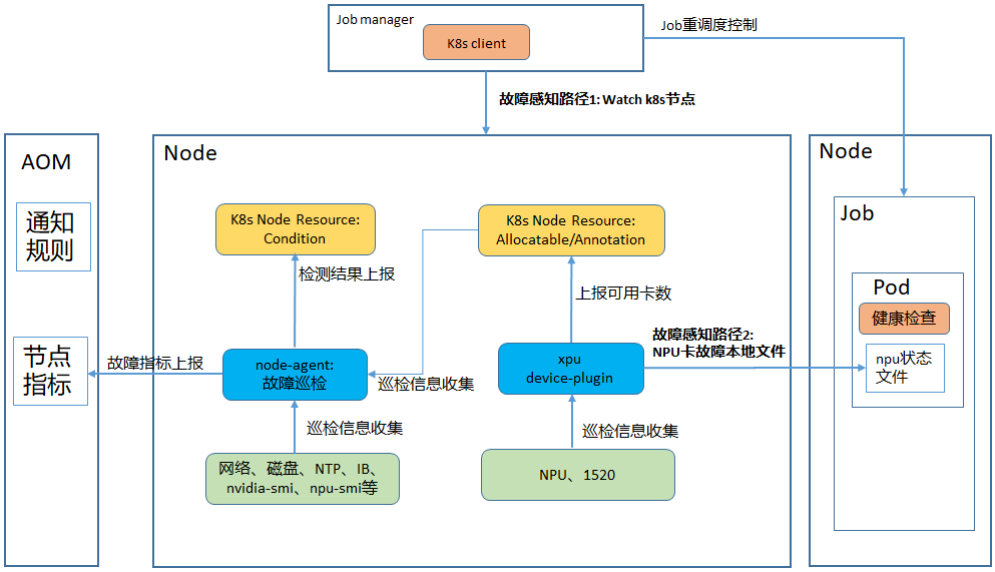
Cluster 节点故障上报

对于ModelArts Lite资源池，每个节点会以DaemonSet方式部署node-agent组件。

如果是GPU资源池会同时安装node-agent组件和gpu device-plugin组件。如果是NPU资源池则是会同时安装node-agent组件和npu device-plugin组件。

节点故障上报如**图10-12**所示。

图 10-12 节点故障上报



故障感知：device-plugin负责xpu卡的故障巡检，node-agent负责AI运行环境的巡检。 device-plugin组件会从驱动侧获取芯片故障并实时上报可用卡数到K8S NodeAllocatable，同时，对使用NPU卡的k8s Pod，device-plugin会自动为其挂载npu状态文件可用于pod内进行健康检查。node-agent组件则是通过巡检脚本收集各故障信息，同时将检测结果汇聚并写到K8S NodeCondition中。

故障通知：node-agent收集到巡检结果后，会周期性上报节点故障指标到AOM服务，配置及时通知方法可参见**如何定位和处理Cluster资源池节点故障**

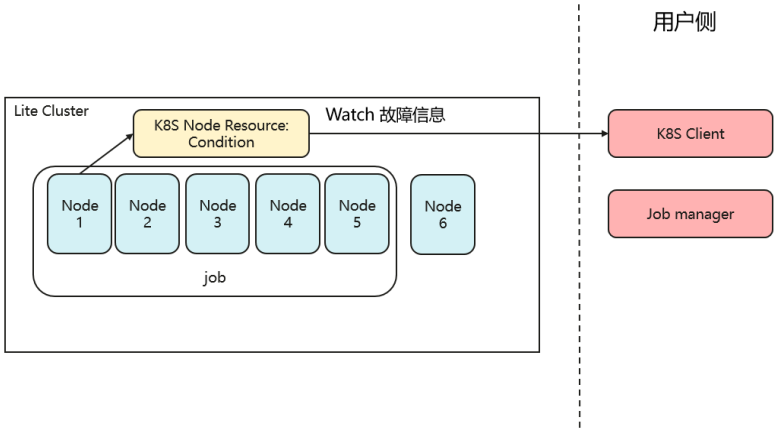
K8S Node Allocatable以及Condition详情可参考**Kubernetes节点状态**。

```
K8S NodeCondition样例：
{
"type": "NT_NPU_CARD_LOSE",
"status": "False",
"lastHeartbeatTime": null,
"lastTransitionTime": "2024-09-27T10: 45: 55Z",
"reason": "os_task_name:npu-card-lose",
"message": "ok"
}
```

status有3种值：False、True以及Unknown。以上述为例，当status为True时，代表节点发生了Type类型为“NT_NPU_CARD_LOSE”的故障，所有故障类型定义请参见**如何定位和处理Cluster资源池节点故障**。

步骤 1：获取故障节点信息

图 10-13 获取故障节点信息



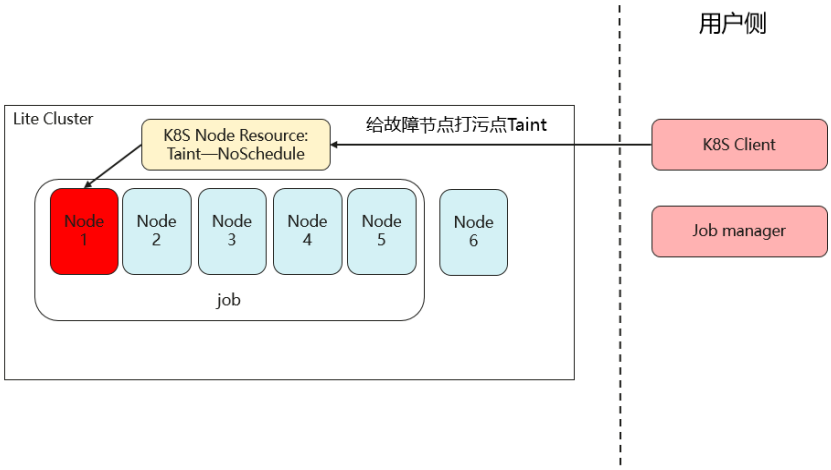
可通过**Kubernetes API访问集群**，然后获取集群中Nodes的详情，Go客户端样例如下：

```
package main

import (
    "context"
    "fmt"
    "time"

    "k8s.io/api/core/v1"
    metav1 "k8s.io/apimachinery/pkg/apis/meta/v1"
    "k8s.io/client-go/kubernetes"
    "k8s.io/client-go/tools/clientcmd"
)
func main() {
    // 通常可以使用clientcmd.RecommendedHomeFile，值是$HOME/.kube/config，根据自身情况可以修改
    config, _ := clientcmd.BuildConfigFromFlags("", clientcmd.RecommendedHomeFile)
    // 创建 clientset 客户端
    clientset, _ := kubernetes.NewForConfig(config)
    // 创建一个 watcher
    watcher, err := clientset.CoreV1().Nodes().Watch(context.TODO(), metav1.ListOptions{})
    if err != nil {
        fmt.Printf("Failed to create watcher: %v\n", err)
    }
    defer watcher.Stop()
    termCh := make(chan struct{}, 1)
    // 事件驱动获取节点故障信息
    go func() {
        for {
            select {
            case <-termCh:
                return
            case event, ok := <-watcher.ResultChan():
                if !ok {
                    fmt.Printf("Failed to get watcher chan: %v\n", err)
                    return
                }
                node := event.Object.(*v1.Node)
                fmt.Printf("Event type: %v, Node name: %s\n", event.Type, node.Name)
                for _, v := range node.Status.Conditions {
                    fmt.Printf("Node Condition Type: %v, Type Status: %v\n", v.Type, v.Status)
                }
            }
```

```
    }
  }()
  // 或者全量获取Nodes
  nodes, _ := clientset.CoreV1().Nodes().List(context.TODO(), metav1.ListOptions{})
  fmt.Printf("There are %d nodes in the cluster\n", len(nodes.Items))
  time.Sleep(10 * time.Second)
  // 定义结束信号
  termCh <- struct{}{}
}
```

步骤 2：用 Taint 隔离集群中的故障节点

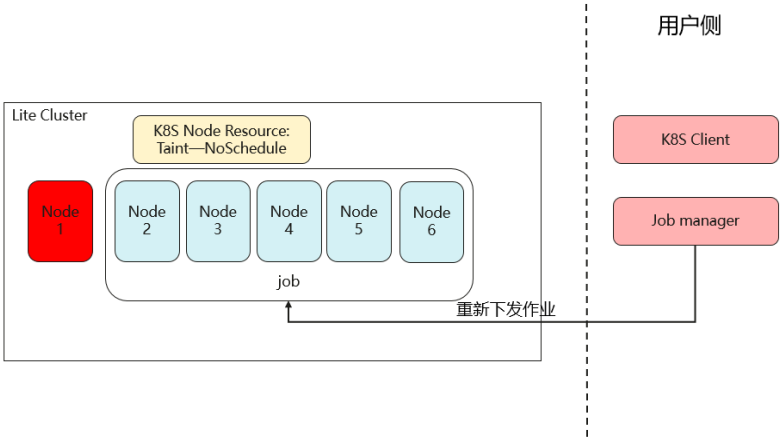
图 10-14 用 Taint 隔离集群中的故障节点



Taint相关资料可参考**污点和容忍度**。

步骤 3：重新下发业务

图 10-15 重新下发业务



当业务为训练作业时，可参考**设置断点续训练**。