ModelArts

SDK 参考

文档版本 01

发布日期 2025-02-25

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 文档导读

本文档指导您如何安装和配置开发环境、如何通过调用ModelArts SDK提供的接口函数进行二次开发。

章节	内容
SDK简介	简要介绍ModelArts SDK的概念。
快速开始	介绍如何使用ModelArts SDK进行二次开发。
（可选）本地服务器安装ModelArts SDK	介绍如何在本地安装ModelArts SDK。
（可选）Session鉴权	Session模块的主要作用是实现与公有云资源的鉴权，并初始化ModelArts SDK Client、OBS Client。
OBS管理概述	ModelArts SDK支持对OBS的SDK接口进行调用，包括创建OBS桶，上传/下载文件和文件夹，删除OBS对象和桶。
ModelArts SDK具体操作管理请参见如下章节： **数据管理** **训练管理** **模型管理** **服务管理**	介绍使用ModelArts SDK进行的常用操作。

2 SDK 简介

ModelArts服务软件开发工具包（ModelArts SDK）是对ModelArts服务提供的REST API进行的Python封装，以简化用户的开发工作。用户直接调用ModelArts SDK即可轻松启动AI训练以及生成模型并将其部署为在线服务。

ModelArts SDK目前只提供Python语言的SDK，同时支持大于3.7.x版本且小于3.10.x版本的Python版本，推荐使用3.9.x版本。

使用场景

ModelArts SDK目前仅支持在ModelArts开发环境Notebook和本地PC两种环境使用。

须知

ModelArts SDK不支持在训练作业和在线服务中使用。

- ModelArts SDK已经集成在ModelArts开发环境Notebook中，可以直接使用，无需进行Session鉴权。

  登录ModelArts控制台，在“开发空间 > Notebook”中创建Notebook实例，在Terminal或ipynb文件中直接调用ModelArts SDK的接口。在Notebook中调用SDK，可直接参考接口说明，执行OBS管理、作业管理、模型管理和服务管理等操作。
- ModelArts SDK支持在本地安装配置使用。使用时，需进行Session鉴权。
  a. 本地安装SDK。如果本地未安装SDK，可参考**（可选）本地服务器安装ModelArts SDK**安装；如果本地已安装，则无需再次安装。
  b. 进行Session鉴权。可参见**（可选）Session鉴权**完成鉴权。鉴权完成后，即可开始使用。

SDK 版本说明

表 2-1 ModelArts SDK 版本说明

发布时间	版本号	说明
2023-04	1.4.18	1.4.18版本在SDK旧版本基础上优化集成，主要新增DLI Spark任务提交能力，支持服务部署到推理新版专属资源池。

支持的区域

当前支持的“region_name”中国-香港（ap-southeast-1）、亚太-曼谷（ap-southeast-2）、亚太-新加坡（ap-southeast-3）、拉美-圣地亚哥（la-south-2）。

3 快速开始

ModelArts SDK目前仅支持在ModelArts开发环境Notebook和本地PC两种环境使用。

须知

ModelArts SDK不支持在训练作业和在线服务中使用。

- ModelArts SDK已经集成在ModelArts开发环境Notebook中，可以直接使用，无需进行Session鉴权。

  登录ModelArts控制台，在“开发环境 > Notebook”中创建Notebook实例，在Terminal或ipynb文件中直接调用ModelArts SDK的接口。在Notebook中调用SDK，可直接参考接口说明，执行OBS管理、作业管理、模型管理和服务管理等操作。
- ModelArts SDK支持在本地安装配置使用。使用时，需进行Session鉴权。
  a. 本地安装SDK。如果本地未安装SDK，可参考**（可选）本地服务器安装ModelArts SDK**安装；如果本地已安装，则无需再次安装。
  b. 进行Session鉴权。可参见**（可选）Session鉴权**完成鉴权。鉴权完成后，即可开始使用。

4 （可选）本地服务器安装 ModelArts SDK

如果需要在个人PC或虚拟机上使用ModelArts SDK，则需要在本地环境中安装ModelArts SDK，安装后可直接调用ModelArts SDK轻松管理数据集、创建ModelArts训练作业及创建AI应用，并将其部署为在线服务。

ModelArts SDK 使用限制

本地ModelArts SDK不支持进行**训练作业调测**、**模型调试**和**在开发环境中部署本地服务进行调试**，当前仅支持在开发环境Notebook中调试。

本地安装 ModelArts SDK 步骤

在本地安装ModelArts SDK，具体的配置步骤如下：

- **步骤一：下载ModelArts SDK**
- **步骤二：配置运行环境**
- **步骤三：安装ModelArts SDK**

说明

ModelArts SDK支持安装在Windows和Linux操作系统中。

如果在Windows上安装ModelArts SDK时出现报错，可参见**FAQ：安装ModelArts SDK报错**处理报错。

步骤一：下载 ModelArts SDK

1. **下载ModelArts SDK软件包**，获取最新版本的ModelArts SDK软件包。
2. （可选）完成软件包签名校验。
   a. **下载软件包签名校验文件**。
   b. 安装openssl并进行软件一致性验证，具体验签命令如下：

```
openssl cms -verify -binary -in D:\modelarts-latest-py2.py3-none-any.whl.cms -inform DER -content D:\modelarts-latest-py2.py3-none-any.whl  -noverify > ./test
```

说明

本示例以软件包在D:\举例，请根据软件包实际路径修改。

步骤二：配置运行环境

1. 检查本地环境是否已安装Python。如果环境中没有安装Python，可从**Python官网**下载并安装合适的Python版本。Python版本需大于3.7.x版本且小于3.10.x版本，推荐使用3.7.x版本。

   在本地环境执行命令**python --version**，显示如下内容说明Python已安装。

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

2. 检查是否已安装Python通用包管理工具pip。如果Python安装过程中没有安装通用包管理工具pip，则参见**pip官网**完成pip安装，推荐pip版本小于24.0。

   在本地环境执行命令**pip --version**，显示如下内容说明pip已安装。

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

   说明

   在Windows环境中，如果提示“不是内部或外部命令”，请您在“环境变量”中设置“Path”，增加Python和pip的安装路径，具体步骤如下。pip的安装路径一般为Python所在目录的Scripts文件夹。

   1. 快捷键“win+R”，在“运行”窗口中输入“sysdm.cpl”，单击“确定”。
   2. 在“系统属性”中切换到“高级”页签，单击“环境变量”。
   3. 在“环境变量”的“用户变量”中鼠标左键双击“Path”，在“编辑环境变量”窗口单击“新增”，新增Python和pip的安装路径。安装路径需定位到Scripts文件夹，例如“c:\python\python**\Scripts”。

3. 配置pip源。以Windows环境为例，配置pip源方法如下：

   a. 新建pip文件夹。启动cmd，输入set命令，查看APPDATA路径。并在APPDATA对应路径下创建pip文件夹。文件内容示例如下：

   ```
   C:\Users\xxx>set
   ALLUSERSPROFILE=C:\ProgramData
   APPDATA=C:\Users\xxx\AppData\Roaming
   ```

   如上所示，即需要在C:\Users\xxx\AppData\Roaming路径下创建pip文件夹。

   b. 在pip文件夹中创建一个名为pip的文本文件，并将后缀名由“.txt”改为“.ini”。文件内容示例如下：

   其中，index-url为pip源ip地址，使用时需自行替换。本示例以华为源为例，具体如下：

   ```
   [global]
   index-url = https://mirrors.huaweicloud.com/repository/pypi/simple
   trusted-host = mirrors.huaweicloud.com
   disable-pip-version-check = true
   timeout = 120
   [install]
   ignore-installed = true
   no-dependencies = yes
   ```

4. 启动cmd，执行如下命令下载需要的pip源中的包。

   ```
   C:\Users\xxx>pip install numpy      #numpy可替换为您需要下载的包
   ```

步骤三：安装 ModelArts SDK

启动cmd，执行如下命令安装ModelArts SDK。

pip install *{SDK软件包路径}*\modelarts-latest-py2.py3-none-any.whl

```
C:\Users\xxx>pip install C:\Users\xxx\Downloads\modelarts-latest-py2.py3-none-any.whl
......
Successfully installed Pillow-*.*.0 ... modelarts-*.*.* ...
```

在安装SDK时会默认同时安装所需的依赖包。当显示“Successfully installed”时，表示ModelArts SDK安装完成。如果安装失败，可参见**FAQ：安装ModelArts SDK报错**处理报错。

说明

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

pip install xxxx

其中，xxxx为依赖包的名称。

后续操作

本地安装ModelArts SDK后，需完成**Session鉴权**。Session鉴权完成后，您可直接调用ModelArts的SDK接口。

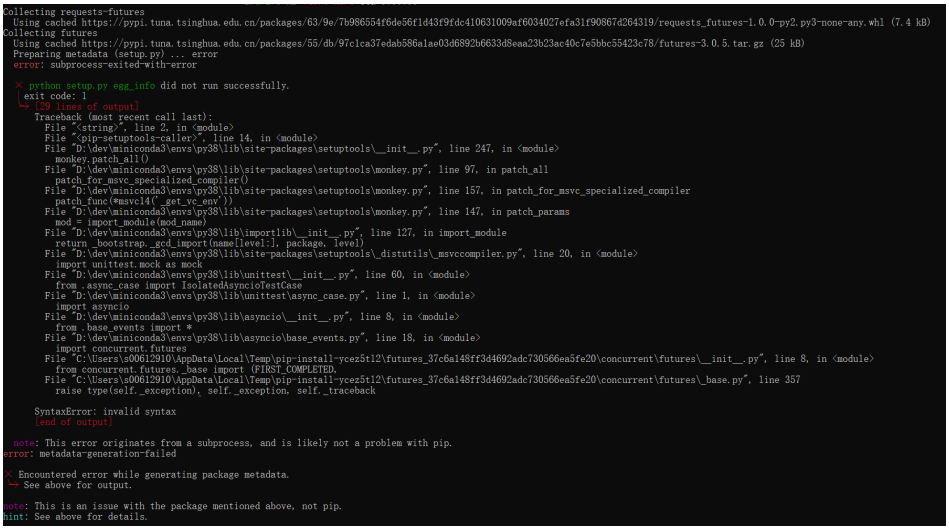
FAQ：安装 ModelArts SDK 报错

- 在Windows上安装ModelArts SDK，Python版本需≤3.10.x版本，推荐使用3.7.x版本。

  如果本地安装SDK时，出现如下图中的报错，需要先安装3.1.1版本的futures依赖包，然后再重新安装SDK。

  ```
  pip install futures==3.1.1
  ```

  图 4-1 安装 ModelArts SDK 报错信息

- 当pip版本>=24.1版本时，会对安装包的名称进行校验，可能会出现如下报错：

  ```
  ERROR: Invalid requirement: 'modelarts==latest': Expected end or semicolon (after name and no valid version specifier)
  modelarts==latest
  ```

  针对以上报错，可重新安装更低版本的pip，或将离线包名modelarts-latest-py2.py3-none-any.whl中的latest修改为任意版本号，例如modelarts-1.0.0-py2.py3-none-any.whl，修改后不影响实际安装SDK的真实版本号。

5 Session 鉴权

5.1 （可选）Session 鉴权

Session 鉴权概述

Session模块的主要作用是实现与公有云资源的鉴权，并初始化ModelArts SDK Client、OBS Client。当成功建立Session后，您可以直接调用ModelArts的SDK接口。

- ModelArts开发环境Notebook不需要Session鉴权，可以直接使用。示例代码如下：

```
from modelarts.session import Session
session = Session()
```

- 本地PC使用ModelArts SDK时，需要进行Session鉴权。鉴权方式可参考如下认证方式，选择其中一种方式进行认证即可。
  - **用户名密码认证模式**： 支持**OBS管理**、**数据管理**、**训练管理**、**模型管理**、**服务管理**的鉴权。
  - **用户AK-SK认证模式**： 支持**OBS管理**、**数据管理**、**训练管理**、**模型管理**、**服务管理**的鉴权。

用户名密码认证模式

本地安装完成ModelArts SDK后，可通过用户名密码认证模式进行Session鉴权。示例代码如下：

- 使用账号认证

  “username”填写您的账号名。

```
from modelarts.session import Session

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

- 使用IAM用户认证

  “account”填写您的账号名，“username”填写您的IAM用户名。

```
from modelarts.session import Session

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

说明

账号与用户的概念介绍，请参见**IAM基本概念**。获取您的账号、用户名等信息，请参见**获取用户名、用户ID、项目名称、项目ID**。

如果您的华为云账号已经升级为华为账号，则账号认证方式将不可用，请创建一个IAM用户，使用IAM用户认证。

用户 AK-SK 认证模式

本地安装完成ModelArts SDK后，可通过用户AK-SK认证模式进行Session鉴权。示例代码如下：

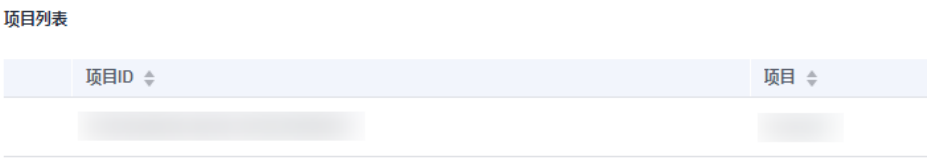
```
from modelarts.session import Session

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

其中，各参数说明如下：

- “access_key”和“secret_key”获取方式如下：
  a. 登录管理控制台，可单击控制台右上角的账户名，在菜单栏中单击“我的凭证”，进入“我的凭证”页面。
  b. 在“我的凭证 > 访问密钥”中，单击“新增访问密钥”。
  c. 在“新增访问密钥”弹窗中，填写该密钥的描述说明，单击“确定”。根据提示单击“立即下载”，下载密钥。密钥文件会直接保存到浏览器默认的下载文件夹中，文件名为“credentials.csv”，可打开文件查看访问密钥（Access Key Id和Secret Access Key）。
- “project_id”即项目ID，获取方式如下：
  在“我的凭证”页面，单击“API凭证”，在“项目列表”中可查看项目ID和名称（即“项目”）。多项目时，展开“所属区域”，从“项目ID”列获取子项目ID。

  图 5-1 查看项目 ID

- “region_name”即区域ID。获取方式请参见**获取区域ID**。

5.2 用户名密码认证模式

本模式支持**OBS管理**、**训练管理**、**模型管理**、**服务管理**的鉴权。

示例代码

账号与用户的概念介绍，请参见**IAM基本概念**。获取您的账号、用户名等信息，请参见**获取用户名、用户ID、项目名称、项目ID**。

- 使用账号认证

  “username”填写您的账号名。

```
from modelarts.session import Session
session = Session(username='***',  password='***', region_name='***', project_id='***')
```

  说明

  如果您的华为云账号已经升级为华为账号，则账号认证方式将不可用，请创建一个IAM用户，使用IAM用户认证。

- 使用IAM用户认证

  “account”填写您的账号名，“username”填写您的IAM用户名。

```
from modelarts.session import Session
session = Session(account='***', username='***',  password='***', region_name='***', project_id='***')
```

5.3 用户 AK-SK 认证模式

本模式支持**OBS管理**、**训练管理**、**模型管理**、**服务管理**模块的鉴权。

示例代码

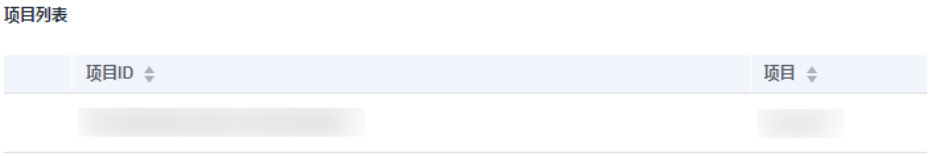
```
from modelarts.session import Session
session = Session(access_key='***',secret_key='***', project_id='***', region_name='***')
```

其中，各参数说明如下：

- “access_key”和“secret_key”获取方式如下：
  a. 登录管理控制台，可单击控制台右上角的账户名，在菜单栏中单击“我的凭证”，进入“我的凭证”页面。
  b. 在“我的凭证 > 访问密钥”中，单击“新增访问密钥”。
  c. 在“新增访问密钥”弹窗中，填写该密钥的描述说明，单击“确定”。根据提示单击“立即下载”，下载密钥。密钥文件会直接保存到浏览器默认的下载文件夹中，文件名为“credentials.csv”，可打开文件查看访问密钥（Access Key Id和Secret Access Key）。
- “project_id”即项目ID，获取方式如下：

  在“我的凭证”页面，单击“API凭证”，在“项目列表”中可查看项目ID和名称（即“项目”）。多项目时，展开“所属区域”，从“项目ID”列获取子项目ID。

图 5-2 查看项目 ID



- “region_name”即区域ID。获取方式请参见**获取区域ID**。

6 OBS 管理

6.1 OBS 管理概述

ModelArts SDK 1.1.3支持对OBS进行管理，主要涵盖上传下载文件和文件夹。具体包括如下操作：

- **上传文件至OBS**
- **上传文件夹至OBS**
- **从OBS下载文件**
- **从OBS下载文件夹**

6.2 文件传输（推荐）

说明

该接口支持上传本地文件和文件夹至OBS，支持下载OBS文件和文件夹至本地，推荐使用该接口。

示例代码

在ModelArts Notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

```
from modelarts.session import Session
session = Session()
# 1. 上传本地文件至OBS
session.obs.copy(src_path='/home/ma-user/file1.txt', dst_path='obs://bucket-name/dir1/file1.txt')

# 2. 下载OBS文件至本地
session.obs.copy(src_path='obs://bucket-name/dir1/file1.txt', dst_path='/home/ma-user/file1.txt')

# 3. 上传本地文件夹至OBS
session.obs.copy(src_path='/home/ma-user', dst_path='obs://bucket-name/dir1', keep_last_dir=True)

# 4. 下载OBS文件夹至本地
session.obs.copy(src_path='obs://bucket-name/dir1', dst_path='/home/ma-user', keep_last_dir=True)
```

表 6-1 请求参数说明

参数	是否必选	参数类型	描述
session	是	Object	会话对象。
src_path	是	String	源文件或源文件夹路径，当源路径为OBS路径时，必须以obs://作为路径前缀。
dst_path	是	String	目的文件或目的文件夹路径，当目的路径为OBS路径时，必须以obs://作为路径前缀。
keep_last_dir	否	Boolean	默认为True，复制文件夹时是否将源文件夹最后一级目录复制至目的文件夹下，仅对文件夹复制有效。

表 6-2 失败相应说明

参数	参数类型	描述
error_code	String	调用失败时的错误码。调用成功时无此字段。
error_msg	String	调用失败时的错误信息。调用成功时无此字段。

6.3 上传文件至 OBS

示例代码

在ModelArts Notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

```
from modelarts.session import Session
session = Session()
session.obs.upload_file(src_local_file='/home/ma-user/file1.txt', dst_obs_dir='obs://bucket-name/dir1/')
```

示例代码执行后，本地源文件“file1.txt”被上传至“bucket-name”桶的“dir1”文件夹下，路径为“obs://bucket-name/dir1/file1.txt”。其中，桶名称和文件夹的名称均可以按照业务需求自定义。

参数说明

表 6-3 请求参数说明

参数	是否必选	参数类型	描述
session	是	Object	会话对象。
src_local_file	是	String	本地需要上传的文件路径。

参数	是否必选	参数类型	描述
dst_obs_dir	是	String	上传的目标OBS桶地址，必须以“obs://”作为前缀，上传的目标文件夹后缀必须以“/”结尾。

表 6-4 失败响应参数说明

参数	参数类型	描述
error_code	String	调用失败时的错误码。 调用成功时无此字段。
error_msg	String	调用失败时的错误信息。 调用成功时无此字段。

6.4 上传文件夹至 OBS

示例代码

在ModelArts Notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参考**Session鉴权**。

```
from modelarts.session import Session
session = Session()
session.obs.upload_dir(src_local_dir='/home/ma-user/', dst_obs_dir='obs://bucket-name/dir1/')
```

示例代码执行后，本地源文件夹“/ma-user/”被上传至“bucket-name”桶的“dir1”文件夹下，路径为“obs://bucket-name/dir1/ma-user/”。其中，桶名称和文件夹的名称均可以按照业务需求自定义。

参数说明

表 6-5 请求参数说明

参数	是否必选	参数类型	描述
session	是	Object	会话对象。
src_local_dir	是	String	本地需要上传的文件夹路径。 当上传的文件夹下内容为空或者该文件夹下包含多个文件夹且有文件夹下内容有空时，OBS对应路径下不产生该空文件夹。
dst_obs_dir	是	String	上传的目标OBS桶地址，必须以“obs://”作为前缀，上传的目标文件夹后缀必须以“/”结尾。

表 6-6 失败响应参数说明

参数	参数类型	描述
error_code	String	调用失败时的错误码。 调用成功时无此字段。
error_msg	String	调用失败时的错误信息。 调用成功时无此字段。

6.5 从 OBS 下载文件

示例代码

在ModelArts Notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参考**Session鉴权**。

```
from modelarts.session import Session
session = Session()
session.obs.download_file(src_obs_file="obs://bucket-name/dir1/file1.txt", dst_local_dir="/home/ma-user/")
```

示例代码执行后，OBS源文件“file1.txt”被下载至“/home/ma-user/file1.txt”。

参数说明

表 6-7 请求参数说明

参数	是否必选	参数类型	描述
session	是	Object	会话对象。
src_obs_file	是	String	下载的源OBS文件，必须以“obs://”作为前缀。
dst_local_dir	是	String	下载目标的本地文件夹，下载的本地目标文件夹后缀必须以“/”结尾。

表 6-8 失败响应参数说明

参数	参数类型	描述
error_code	String	调用失败时的错误码。 调用成功时无此字段。
error_msg	String	调用失败时的错误信息。 调用成功时无此字段。

6.6 从 OBS 下载文件夹

示例代码

在ModelArts Notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参考**Session鉴权**。

```
from modelarts.session import Session
session = Session()
session.obs.download_dir(src_obs_dir="obs://bucket-name/dir1/", dst_local_dir="/home/ma-user/work/")
```

示例代码执行后，OBS源文件夹“dir1”被下载至本地“/home/ma-user/work/dir1/”。

⚠ 注意

下载到本地的路径需要有写权限。

参数说明

表 6-9 请求参数说明

参数	是否必选	参数类型	描述
session	是	Object	会话对象。
src_obs_dir	是	String	下载的源OBS文件夹，必须以“obs://”作为前缀，文件夹后缀必须以"/"结尾。当下载的文件夹下有文件夹且内容为空时，对应路径下不产生对应空文件夹。
dst_local_dir	是	String	下载的目标本地文件夹，下载的目标本地文件夹后缀必须以“/”结尾。

表 6-10 失败响应参数说明

参数	参数类型	描述
error_code	String	调用失败时的错误码。 调用成功时无此字段。
error_msg	String	调用失败时的错误信息。 调用成功时无此字段。

7 数据管理

7.1 数据集管理

7.1.1 查询数据集列表

分页查询用户的数据集列表。

```
list_datasets(session, dataset_type=None, dataset_name=None, offset=None, limit=None)
```

示例代码

- 示例一：查询数据集列表

```
from modelarts.session import Session
from modelarts.dataset import Dataset
session = Session()
# 查询数据集列表
dataset_list = Dataset.list_datasets(session)
print(dataset_list) # 打印出查询结果
```

- 示例二：根据数据集类型查询数据集列表

```
# 查询图像分类数据集
dataset_list = Dataset.list_datasets(session, dataset_type=0)
print(dataset_list)
```

- 示例三：根据数据集名称查询数据集列表

```
# 查询名称中包含dataset的数据集列表
dataset_list = Dataset.list_datasets(session, dataset_name="dataset")
print(dataset_list)
```

- 示例四：分页查询数据集列表

```
# 默认一次返回10条数据集记录，可通过设置limit和offset进行分页查询
dataset_list = Dataset.list_datasets(session, offset=0, limit=50) # 查询第1-50条记录
print(dataset_list)
dataset_list = Dataset.list_datasets(session, offset=1, limit=50) # 查询第51-100条记录
print(dataset_list)
```


参数说明

表 7-1 请求参数

参数	是否必选	参数类型	描述
session	是	Object	会话对象，初始化方法请参见**Session鉴权**。
dataset_type	否	Integer	根据数据集类型查询数据集列表，默认为空。可选值如下： ● 0：图像分类 ● 1：物体检测 ● 3: 图像分割 ● 100：文本分类 ● 101：命名实体 ● 102：文本三元组 ● 200：声音分类 ● 201：语音内容 ● 202：语音分割 ● 400：表格数据集 ● 600：视频标注 ● 900：自由格式
dataset_name	否	String	模糊匹配数据集名称，默认为空。
offset	否	Integer	分页列表的起始页，默认为0。
limit	否	Integer	指定每一页返回的最大条目数，取值范围[1,100]，默认为10。

7.1.2 创建数据集

创建数据集，支持从OBS中导入数据。

```
create_dataset(session, dataset_name=None, data_type=None, data_sources=None, work_path=None, dataset_type=None, **kwargs)
```

创建数据集支持两种用法：

- 根据**标注类型**创建数据集，一个数据集只能支持一种标注任务类型。
  ```
  create_dataset(session,dataset_name=None, dataset_type=None, data_sources=None, work_path=None, **kwargs)
  ```
- 根据**数据类型**创建数据集，用户可以在相同的数据集上创建不同类型的标注任务，如在图像数据集上创建图像分类、物体检测等标注任务。
  ```
  create_dataset(session,dataset_name=None, data_type=None, data_sources=None, work_path=None, **kwargs)
  ```

说明

推荐使用根据数据类型创建数据集，根据标注类型创建数据集的功能将会下线。

示例代码

- 示例一：根据数据类型创建图像数据集

```
from modelarts.session import Session
from modelarts.dataset import Dataset

session = Session()

dataset_name = "dataset-image"  # 数据集名称
data_type = "IMAGE"             # 数据集类型，图像类型数据集
data_sources = dict()           # 数据集数据来源
data_sources["type"] = 0        # 数据来源类型，0表示OBS
data_sources["path"] = "/obs-gaia-test/data/image/image-classification/" # 数据在OBS中的路径
work_path = dict()              # 数据集输出位置，用于存放输出的标注信息等文件
work_path['type'] = 0           # 数据集工作目录的类型，0表示OBS
work_path['path'] = "/obs-gaia-test/data/output/work_path/"  # 数据集工作目录在OBS中的路径
create_dataset_resp = Dataset.create_dataset(session, dataset_name=dataset_name,
data_type=data_type,
                                             data_sources=data_sources, work_path=work_path)
```

- 示例二：根据数据类型创建图像数据集（导入标注信息）

```
from modelarts.session import Session
from modelarts.dataset import Dataset

session = Session()

dataset_name = "dataset-image-with-annotations"
data_type = "IMAGE"
data_sources = dict()
data_sources["type"] = 0
data_sources["path"] = "/obs-gaia-test/data/image/image-classification/"
annotation_config = dict()      # 源数据的标注格式
annotation_config['scene'] = "image_classification" # 数据标注场景为图像分类标注
annotation_config['format_name'] = "ModelArts image classification 1.0" # 标注格式为ModelArts
image classification 1.0
data_sources['annotation_config'] = annotation_config
work_path = dict()
work_path['type'] = 0
work_path['path'] = "/obs-gaia-test/data/output/work_path/"
create_dataset_resp = Dataset.create_dataset(session, dataset_name=dataset_name,
data_type=data_type,
                                             data_sources=data_sources, work_path=work_path)
```

- 示例三：根据数据类型创建表格数据集

```
from modelarts.session import Session
from modelarts.dataset import Dataset

session = Session()

dataset_name = "dataset-table"
data_type = "TABLE"
data_sources = dict()
data_sources["type"] = 0
data_sources["path"] = "/obs-gaia-test/data/table/table0/"
data_sources['with_column_header'] = True
work_path = dict()
work_path['type'] = 0
work_path['path'] = "/obs-gaia-test/data/output/work_path/"
# 表格类型的数据集需要指定表格数据的schema信息
schema0 = dict()
schema0['schema_id'] = 0
schema0['name'] = "name"
schema0['type'] = "STRING"
schema1 = dict()
schema1['schema_id'] = 1
```



```
schema1['name'] = "age"
schema1['type'] = "STRING"
schema2 = dict()
schema2['schema_id'] = 2
schema2['name'] = "label"
schema2['type'] = "STRING"
schemas = []
schemas.append(schema0)
schemas.append(schema1)
schemas.append(schema2)
create_dataset_resp = Dataset.create_dataset(session, dataset_name=dataset_name,
data_type=data_type,
                                             data_sources=data_sources, work_path=work_path, schema=schemas)
```

- 示例四：根据标注类型创建图像分类数据集

```
from modelarts.session import Session
from modelarts.dataset import Dataset

session = Session()

dataset_name = "dataset-image-classification"
dataset_type = 0   # 数据集的标注类型，0表示图像分类标注类型
data_sources = dict()
data_sources["path"] = "/obs-gaia-test/data/image/image-classification/"
data_sources["type"] = "0"
work_path = dict()
work_path['type'] = 0
work_path['path'] = "/obs-gaia-test/data/output/work_path/"
create_dataset_resp = Dataset.create_dataset(session, dataset_name=dataset_name,
dataset_type=dataset_type, data_sources=data_sources, work_path=work_path)
```

- 示例五：根据标注类型创建文本三元组数据集

```
dataset_name = "dataset-text-triplet"
dataset_type = 102   # 数据集标注类型，102表示文本三元组标注类型
data_sources = dict()
data_sources['type'] = 0
data_sources['path'] = "/obs-gaia-test/data/text/text-classification/"
work_path = dict()
work_path['type'] = 0
work_path['path'] = "/obs-gaia-test/data/output/work_path/"

# 创建文本三元组标注类型的数据集，需要传入标签参数
label_entity1 = dict()    # 标签对象
label_entity1['name'] = "疾病"    # 标签名称
label_entity1['type'] = 101     # 标签类型，101表示实体类型标签
label_entity2 = dict()
label_entity2['name'] = "疾病别称"
label_entity2['type'] = 101
label_relation1 = dict()
label_relation1['name'] = "又称为"
label_relation1['type'] = 102    # 标签类型，102表示关系类型标签
property = dict()    # 关系类型标签需要在标签属性中指定起始实体标签和终止实体标签
property['@modelarts:from_type'] = "疾病"    # 起始实体标签
property['@modelarts:to_type'] = "疾病别称"    # 终止实体标签
label_relation1['property'] = property
labels = []
labels.append(label_entity1)
labels.append(label_entity2)
labels.append(label_relation1)
create_dataset_resp = Dataset.create_dataset(session, dataset_name=dataset_name,
dataset_type=dataset_type, data_sources=data_sources, work_path=work_path, labels=labels)
```

- 示例六：根据标注类型创建表格数据集

```
dataset_name = "dataset-table"
dataset_type = 400    # 数据集标注类型，400表示表格数据集
data_sources = dict()
data_sources['type'] = 0
data_sources['path'] = "/obs-gaia-test/data/table/table0/"
data_sources['with_column_header'] = True    # 用来指明表格数据中是否包含表头
work_path = dict()
```



```
work_path['type'] = 0
work_path['path'] = "/obs-gaia-test/data/output/work_path/"

# 表格数据集需要传入表格数据的表头参数
schema0 = dict()    # 表格的表头
schema0['schema_id'] = 0    # 第一列表头
schema0['name'] = "name"    # 表头名称，该列表头为name
schema0['type'] = "STRING"    # 表头数据类型，表示字符串
schema1 = dict()
schema1['schema_id'] = 1
schema1['name'] = "age"
schema1['type'] = "STRING"
schema2 = dict()
schema2['schema_id'] = 2
schema2['name'] = "label"
schema2['type'] = "STRING"
schemas = []
schemas.append(schema0)
schemas.append(schema1)
schemas.append(schema2)
create_dataset_resp = Dataset.create_dataset(session, dataset_name=dataset_name,
dataset_type=dataset_type, data_sources=data_sources, work_path=work_path, schema=schemas)
```

参数说明

表 7-2 请求参数

参数	是否必选	参数类型	描述
session	是	Object	会话对象，初始化方法请参见**Session鉴权**。
dataset_name	是	String	数据集名称。
data_type	否	String	数据集的数据类型，与dataset_type参数二选一，推荐使用data_type。可选值如下： • IMAGE：图像 • TEXT：文本 • AUDIO：音频 • TABLE：表格 • VIDEO：视频 • PLAIN：自由格式

参数	是否必选	参数类型	描述
dataset_type	否	Integer	根据数据集类型查询数据集列表，与data_type参数二选一。可选值如下： • 0：图像分类 • 1：物体检测 • 3: 图像分割 • 100：文本分类 • 101：命名实体 • 102：文本三元组 • 200：声音分类 • 201：语音内容 • 202：语音分割 • 400：表格数据集 • 600：视频标注 • 900：自由格式
data_sources	是	表7-3	数据集输入位置，用于将此目录及子目录下的源数据(如图片/文件/音频等)同步到数据集。对于表格数据集，该参数为导入目录。表格数据集的工作目录不支持为KMS加密桶下的OBS路径。
work_path	是	表7-7	数据集输出位置，用于存放输出的标注信息等文件。
labels	否	List of 表7-8	数据集标签列表，创建文本三元组标注类型的数据集时需要传递该参数。
schema	否	List of 表7-10	schema列表，用于表格数据集，指定表头的名称和类型。
description	否	String	数据集描述，默认为空，描述不能包含^!<>=&"'等特殊字符，长度为0-256。

表 7-3 DataSource 参数

参数	是否必选	参数类型	描述
type	是	Integer	数据类型。可选值如下： • 0：OBS桶（默认值） • 5：AI Gallery下载数据集

参数	是否必选	参数类型	描述
path	是	String	数据源所在路径。 ● 字符限制：不允许出现的特殊字符有换行符(\n)、回车符(\r)、制表符(\t)。
content_info	否	表7-4	从AI Gallery下载数据集时数据集资产的信息。
annotation_config	否	表7-5	数据标注格式的说明。目前支持的标注格式类型如下： ● 图像分类 ● 物体检测 ● 文本分类 ● 声音分类
with_column_header	否	Boolean	表格数据集必选参数，表格的第一行是否为表头。 ● True：第一行数据作为表头 ● False：第一行数据不作为表头，仅为样本数据

表 7-4 ContentInfo 参数

参数	是否必选	参数类型	描述
content_id	是	String	AI Gallery中数据集资产的ID。
version_id	是	String	AI Gallery中数据集资产的版本ID。

表 7-5 AnnotationConfig 参数

参数	是否必选	参数类型	描述
scene	是	String	支持的标注格式场景，可选值如下： ● image_classification：图像分类 ● object_detection：物体检测 ● text_classification：文本分类 ● audio_classification：声音分类

参数	是否必选	参数类型	描述
format_name	是	String	不同标注场景下的标注格式。可选值如下： • image_classification – ModelArts imageNet 1.0 – ModelArts image classification 1.0 • object_detection – ModelArts PASCAL VOC 1.0 – YOLO • text_classification – ModelArts text classfication 1.0 – ModelArts text classfication combine 1.0 • audio_classification – ModelArts audio classfication dir 1.0
parameters	否	表7-6	标注格式的高级参数，如样本分隔符等。

表 7-6 AnnotationConfigParam 参数

参数	是否必选	参数类型	描述
included_labels	否	List of 表7-8	只导入包含指定标签的样本。
sample_label_separator	否	String	文本和标签之间的分隔符。分隔符仅支持一个字符，必须为大小写字母，数字或@#￥%^&*_=|?/':.;,中的某一个字符，分隔符需要转义。
label_separator	否	String	标签和标签之间的分隔符。分隔符仅支持一个字符，必须为大小写字母，数字或@#￥%^&*_=|?/':.;,中的某一个字符，分割符需要转义。
difficult_only	否	Boolean	是否只导入难例。

表 7-7 WorkPath 参数

参数	是否必选	参数类型	描述
type	是	Integer	数据类型。可选值如下： • 0：OBS桶（默认值）

参数	是否必选	参数类型	描述
path	是	String	数据集输出位置，用于存放输出的标注信息等文件。 • 格式为 “/桶名称/文件路径”，例如“/obs-bucket/flower/rose/”（使用目录作为路径）。 • 不能直接使用桶作为路径。 • 输出位置不能与输入位置相同或者是输入位置的子目录。 • 长度限制：不少于3字符，不能超过700个字符。 • 字符限制：不允许出现的特殊字符有换行符(\n)、回车符(\r)、制表符(\t)。

表 7-8 Label 参数

参数	是否必选	参数类型	描述
name	是	String	标签名称。
type	是	Integer	标签类型，可选值如下： • 0：图像分类 • 1：物体检测 • 3: 图像分割 • 100：文本分类 • 101：命名实体标签 • 102：文本三元组关系标签 • 200：声音分类 • 201：语音内容 • 202：语音分割 • 600：视频标注
property	否	表7-9	标签基本属性键值对，如颜色。

表 7-9 LabelProperty 参数

参数	是否必选	参数类型	描述
@modelarts:color	否	String	内置属性：标签展示的颜色，为色彩的16进制代码，默认为空。例如：“#FFFFF0”。

参数	是否必选	参数类型	描述
@modelarts:from_type	否	String	内置属性：三元组关系标签的起始实体类型，创建关系标签时必须指定，该参数仅文本三元组数据集使用。
@modelarts:to_type	否	String	内置属性：三元组关系标签的指向实体类型，创建关系标签时必须指定，该参数仅文本三元组数据集使用。

表 7-10 Schema 参数

参数	是否必选	参数类型	描述
schema_id	否	Integer	Schema ID。
name	否	String	Schema名称。
type	否	String	Schema值类型，可选值如下： ● STRING ● SHORT ● INT ● LONG ● DOUBLE ● FLOAT ● BYTE ● DATE ● TIMESTAMP ● BOOLEAN
description	否	String	Schema描述。

7.1.3 查询数据集详情

查询数据集的详细信息，包括数据集的样本信息、版本信息等。

```
dataset.get_dataset_info()
```

示例代码

查询数据集详情

```
from modelarts.session import Session
from modelarts.dataset import Dataset
session = Session()

dataset = Dataset(session, dataset_id)
```

```
dataset_info = dataset.get_dataset_info()
print(dataset_info) # 输出数据集的详细信息
```

参数说明

无。

7.1.4 更新数据集

更新数据集的名称和描述信息。

```
dataset.update_dataset(dataset_name=None, description=None)
```

示例代码

更新数据集名称

```
from modelarts.session import Session
from modelarts.dataset import Dataset
session = Session()

dataset = Dataset(session, dataset_id)
dataset.update_dataset(dataset_name = "new-dataset-name")
```

参数说明

表 7-11 请求参数

参数	是否必选	参数类型	描述
dataset_name	否	String	新的数据集名称。
description	否	String	数据集描述信息。

7.1.5 删除数据集

根据数据集ID删除指定的数据集

```
delete_dataset(session, dataset_id)
```

示例代码

删除数据集

```
from modelarts.session import Session
from modelarts.dataset import Dataset
session = Session()

Dataset.delete_dataset(session, dataset_id="68ZXdK6CZwgvUICOOdC")
```

参数说明

表 7-12 请求参数

参数	是否必选	参数类型	描述
session	是	Object	会话对象，初始化方法请参见**Session鉴权**。
dataset_id	是	String	数据集的ID。

7.2 数据集版本管理

7.2.1 查询数据集版本列表

查询数据集的版本列表。

```
dataset.list_versions()
```

示例代码

查询数据集版本列表

```
from modelarts.session import Session
from modelarts.dataset import Dataset
session = Session()

dataset = Dataset(session, dataset_id)
version_list = dataset.list_versions()
print(version_list) # 打印数据集的版本列表
```

参数说明

无。

7.2.2 创建数据集版本

为数据集创建新的版本。

```
dataset.create_version(name=None, version_format=None, label_task_type=None, label_task_id=None,
**kwargs)
```

示例代码

示例一：为数据集创建新的版本

```
from modelarts.session import Session
from modelarts.dataset import Dataset
session = Session()

dataset = Dataset(session, dataset_id)
create_version_resp = dataset.create_version(name="V001", version_format="Default", label_task_type=0,
description="version 001")
```

示例二：基于标注任务创建数据集

```
from modelarts.session import Session
from modelarts.dataset import Dataset
```

```
session = Session()

dataset = Dataset(session, dataset_id)
create_version_resp = dataset.create_version(label_task_id="IbAhFai5KXWC3gthUfz", description="dataset version from label task")
```

参数说明

表 7-13 请求参数

参数	是否必选	参数类型	描述
name	否	String	版本名称，必须是中文、字母、数字、下划线或中划线组成的合法字符串，长度为1-32位。
version_format	否	String	数据集版本格式。可选值如下： ● Default：默认格式
label_task_type	否	Integer	版本数据对应的标注类型。可选值如下： ● 0：图像分类 ● 1：物体检测 ● 3：图像分割 ● 100：文本分类 ● 101：命名实体 ● 102：文本三元组 ● 200：声音分类 ● 201：语音内容 ● 202：语音分割 ● 400：表格数据集 ● 600：视频标注 ● 900：自由格式
label_task_id	否	String	基于标注任务创建数据集版本，标注任务ID。
description	否	String	版本描述信息，默认为空，长度为0-256位，不能包含!<>=&"'特殊字符。

7.2.3 查询数据集版本详情

根据版本ID查询数据集指定版本的详细信息。

```
dataset.get_version_info(version_id)
```

示例代码

查询数据集指定版本的详细信息


```
from modelarts.session import Session
from modelarts.dataset import Dataset
session = Session()

dataset = Dataset(session, dataset_id)
version_info = dataset.get_version_info(version_id="cSPuXPgnYp7ObRs6LaR")
print(version_info) # 打印数据集版本的详情
```

参数说明

表 7-14 请求参数

参数	是否必选	参数类型	描述
version_id	是	String	数据集版本ID。

7.2.4 删除数据集版本

删除数据集的指定版本。

```
dataset.delete_version(version_id)
```

示例代码

删除数据集指定版本

```
from modelarts.session import Session
from modelarts.dataset import Dataset
session = Session()

dataset = Dataset(session, dataset_id)
dataset.delete_version(version_id="cSPuXPgnYp7ObRs6LaR")
```

参数说明

表 7-15 请求参数

参数	是否必选	参数类型	描述
version_id	是	String	数据集版本ID。

7.3 样本管理

7.3.1 查询样本列表

查询数据集的样本列表，不支持表格类型数据集。

```
dataset.list_samples(version_id=None, offset=None, limit=None)
```

示例代码

- 示例一：查询数据集样本列表

```
from modelarts.session import Session
from modelarts.dataset import Dataset
session = Session()

dataset = Dataset(session, dataset_id)
list_samples_resp = dataset.list_samples()
print(list_samples_resp) # 打印样本列表的信息
```

- 示例二：查询数据集指定版本的样本列表

```
list_samples_resp = dataset.list_samples(version_id = "cSPuXPgnYp7ObRs6LaR")
print(list_samples_resp)
```

参数说明

表 7-16 请求参数

参数	是否必选	参数类型	描述
version_id	否	String	数据集版本ID。传入版本ID查询数据集相应版本的样本列表。
offset	否	Integer	分页列表的起始页，默认为0。
limit	否	Integer	指定每一页返回的最大条目数，取值范围[1,100]，默认为10。

7.3.2 查询单个样本详情

根据样本ID查询数据集中指定样本的详细信息。

```
dataset.get_sample_info(sample_id)
```

示例代码

根据ID查询数据集中样本的详细信息

```
from modelarts.session import Session
from modelarts.dataset import Dataset
session = Session()

dataset = Dataset(session, dataset_id)
sample_info = dataset.get_sample_info(sample_id="2551e78974aed9b60156d8376232f6bd")
print(sample_info) # 打印样本的详细信息
```

参数说明

表 7-17 请求参数

参数	是否必选	参数类型	描述
sample_id	是	String	样本ID。

7.3.3 批量删除样本

根据样本的ID列表批量删除数据集中的样本。

```
dataset.delete_samples(samples)
```

示例代码

批量删除数据集中的样本

```
from modelarts.session import Session
from modelarts.dataset import Dataset
session = Session()

dataset = Dataset(session, dataset_id)
samples = []
samples.append("2551e78974aed9b60156d8376232f6bd")
samples.append("0d315fec1efc7568de5cccf522c10a1b")
dataset.delete_samples(samples)
```

参数说明

表 7-18 请求参数

参数	是否必选	参数类型	描述
samples	是	List of String	待删除的样本ID列表。

7.4 导入任务管理

7.4.1 查询导入任务列表

查询数据集导入任务列表。

```
dataset.list_import_tasks()
```

示例代码

查询数据集导入任务列表

```
from modelarts.session import Session
from modelarts.dataset import Dataset
session = Session()

dataset = Dataset(session, dataset_id)
list_tasks_resp = dataset.list_import_tasks()
print(list_tasks_resp) # 打印导入任务列表
```

参数说明

无。

7.4.2 创建导入任务

支持从OBS中导入新的数据，导入方式包括**目录导入**和**Manifest文件导入**。

```
dataset.import_data(path=None,  anntation_config=None, **kwargs)
```

不同类型的数据集支持的导入方式如表7-19所示。

表 7-19 不同数据集支持的导入方式

数据集类型	OBS目录导入	Manifest文件导入	备注
图像分类	支持	支持	-
物体检测	支持	支持	-
图像分割	支持	支持	-
文本分类	支持	支持	-
命名实体	不支持	支持	-
文本三元组	不支持	支持	-
声音分类	支持	支持	-
语音内容	不支持	支持	-
语音分割	不支持	支持	-
表格数据集	支持	不支持	新导入的表格数据的schema和数据集一致。
视频标注	不支持	支持	-

示例代码

- 示例一：物体检测数据集目录导入

```
from modelarts.session import Session
from modelarts.dataset import Dataset
session = Session()

dataset = Dataset(session, dataset_id)
annotation_config = dict()
annotation_config['scene'] = "object_detection"
annotation_config['format_name'] = "ModelArts PASCAL VOC 1.0"
import_resp = dataset.import_data(path="/obs-gaia-test/data/image/image-detection/",
annotation_config=annotation_config)
```

- 示例二：物体检测数据集Manifest文件导入

```
annotation_config = dict()    # Manifest文件导入任务中，传入annotation_config参数可以导入标注信息
import_resp = dataset.import_data(
        path="/obs-gaia-test/data/output/work_path/dataset-5932-Qdd1RUZ3wqBQrwrTr3v/
annotation/V001/V001.manifest",annotation_config=annotation_config)
```

- 示例三：表格数据集目录导入

```
import_resp = dataset.import_data(
        path="/obs-gaia-test/data/table/table1/", with_column_header=True)
```

参数说明

表 7-20 请求参数

参数	是否必选	参数类型	描述
path	是	String	导入的OBS路径或Manifest路径。 • 导入Manifest时，path必须精确到具体Manifest文件。 • 导入为目录时，目前仅支持数据集类型为图片分类、物体检测、图像分割、文本分类、声音分类和表格数据集。 • 字符限制：不允许出现的特殊字符有换行符(\n)、回车符(\r)、制表符(\t)。
annotation_config	否	表7-5	数据标注格式的说明。该字段为None，则不导入标注信息。如果根据Manifest文件导入，可以传入一个内容为空的dict对象实现导入标注信息。目前支持的标注格式类型如下： • 图像分类 • 物体检测 • 语音分类 • 文本分类
with_column_header	否	Boolean	表格数据集必选参数，表格的第一行是否为表头。 • True：第一行数据作为表头 • False：第一行数据不作为表头，仅为样本数据

7.4.3 查询导入任务状态

根据任务ID查询数据集导入任务的状态和详情。

```
dataset.get_import_task_info(task_id)
```

示例代码

查询数据集导入任务的详情

```
from modelarts.session import Session
from modelarts.dataset import Dataset
session = Session()

dataset = Dataset(session, dataset_id)
task_info = dataset.get_import_task_info(task_id="r4R52nJ4VJKcivuioCU")
print(task_info) # 打印导入任务的详细信息
```

参数说明

表 7-21 请求参数

参数	是否必选	参数类型	描述
task_id	是	String	导入任务的任务ID。

7.5 导出任务管理

7.5.1 查询导出任务列表

查询数据集导出任务列表。

```
dataset.list_export_tasks()
```

示例代码

查询数据集导出任务列表

```
from modelarts.session import Session
from modelarts.dataset import Dataset
session = Session()

dataset = Dataset(session, dataset_id)
list_tasks_resp = dataset.list_export_tasks()
print(list_tasks_resp) # 打印导出任务列表
```

参数说明

无。

7.5.2 创建导出任务

将当前数据集的样本导出到指定的OBS路径下。仅支持图像分类、物体检测、图像分割和自由格式数据集。

```
dataset.export_data(path)
```

示例代码

导出数据集到OBS目录

```
from modelarts.session import Session
from modelarts.dataset import Dataset
session = Session()

dataset = Dataset(session, dataset_id)
export_resp = dataset.export_data("/obs-gaia-test/data/output/export-test/")
```

参数说明

表 7-22 请求参数

参数	是否必选	参数类型	描述
path	是	String	数据导出的OBS路径。

7.5.3 查询导出任务状态

根据任务ID查询数据集导出任务的状态和详情。

```
dataset.get_export_task_info(task_id)
```

示例代码

查询数据集导出任务状态

```
from modelarts.session import Session
from modelarts.dataset import Dataset
session = Session()

dataset = Dataset(session, dataset_id)
task_info = dataset.get_export_task_info(task_id="iuHALF6xdkSAGKVN2jD")
print(task_info) # 查询导出任务的详细信息
```

参数说明

表 7-23 请求参数

参数	是否必选	参数类型	描述
task_id	是	String	导出任务的任务ID。

7.6 Manifest 管理

7.6.1 Manifest 管理概述

在ModelArts使用过程中，需要做数据标注、模型训练、推理、数据集管理、市场发布等业务，这些业务都基于数据集进行的。为了规范对数据集的使用，适配各个使用场景，同时兼顾数据集管理的灵活性，本文档描述数据集管理的接口和描述规范——Manifest文件。

- Manifest文件中定义了标注对象和标注内容的对应关系。Manifest文件中也可以只有原始文件信息，没有标注信息，如生成的未标注的数据集。
- Manifest文件使用UTF-8编码，Manifest处理程序需具备UTF-8处理能力。
- Manifest文件中文本分类的source数值可以包含中文，其他字段不建议用中文。

- Manifest文件可以由用户、第三方工具或ModelArts标注系统生成。
- Manifest文件名没有特殊要求，可以为任意合法文件名。

7.6.2 解析 Manifest 文件

解析Manifest文件，支持本地和OBS。如果是OBS，需要Session信息。

```
manifest.parse_manifest(manifest_path, encoding='utf-8')
```

示例代码

通过Manifest路径来解析获取Manifest的信息。

```
from modelarts.session import Session
from modelarts.dataset.format.manifest import Manifest

path = "obs://your-obs-bucket/manifest/V001.manifest"
session = Session()
manifest_info= Manifest.parse_manifest(path,session=session)
```

参数说明

表 7-24 请求参数

参数	是否必选	参数类型	描述
manifest_path	是	String	Manifest文件路径，支持OBS和本地路径。如果是OBS，需要Session信息。
encoding	否	String	文件编码格式，默认为utf-8。

表 7-25 manifest_info 返回参数说明

参数	参数类型	描述
size	Long	样本数量。
samples	JSON Array	样本列表。样本属性请见表7-26。

表 7-26 sample 样本属性

参数	参数类型	描述
source	String	被标注对象的URI，支持OBS、HTTPS、Content。其中Content为文本内容，例如：“source”:“s3://path-to-jpg”，“source”:“content://I love machine learning”。

参数	参数类型	描述
annotations	JSON Array	样本标注信息。如果不给出，则是未标注对象。 annotations值为一个对象列表。标注属性请见表7-27。
usage	String	用途，可选值为TRAIN、EVAL、TEST、INFERENCE。指明该对象用于训练、评估、测试、推理，如果没有给出该字段，则使用者自行决定如何使用该对象。
inference_loc	String	当此Manifest文件由推理服务生成时会有该字段，表示推理输出的结果文件位置。
id	String	样本ID。
source_type	String	source的类型，比如csv。
source_property	String	source的属性。
hard	Boolean	是否是难例，true表示是难例，false为非难例。
hard_coefficient	Double	难度系数，范围为[0,1]。
hard_reasons	String	标签级别难例原因。通过中划线间隔单个难例原因ID。
source_map	String	source的映射。

表 7-27 annotation 标注属性

参数	参数类型	描述
name	String	标注名称。
type	String	标注类型。
id	String	标注ID。
annotation_loc	String	标注文件的云存储路径，对于物体检测是必选字段，对于其他类型是可选字段。
annotation_property	String	标注属性。
confidence	Double	置信度，数值类型，范围0<=confidence<=1，表示机器标注的置信度。
creation_time	String	创建该标注的时间。是用户写入标注的时间，不是Manifest生成时间。
annotated_by	String	标注人。

参数	参数类型	描述
annotation_format	String	描述标注文件的格式。默认为“PASCAL VOC”。
hard	Boolean	是否是难例。
hard_coefficient	Double	难度系数。
annotation_loc_map	String	标注文件路径的映射。

7.6.3 创建和保存 Manifest 文件

需要先创建包含Manifest信息的对象，然后保存。Manifest信息请见表7-25。路径支持本地和OBS，如果是OBS，需要Session信息。

```
manifest_info.save(path, session=None, save_mode="w")
```

示例代码

在保存Manifest文件之前需要先创建包含Manifest信息的对象，包括Sample样本信息及其标签信息Annotation，然后将若干个样本组成Manifest。保存的时候调用save接口，将session信息传入，即可保存到指定路径。

```
from modelarts.dataset.format.manifest.annotation import Annotation
from modelarts.dataset.format.manifest import Manifest
from modelarts.dataset.format.manifest.sample import Sample
from modelarts.session import Session

size = 0
sample_list = []
for i in range(19):
    size = size + 1
    source = "s3://obs-path/examples/image-classification/data/image_" + str(i) + ".jpg"
    usage = "TRAIN"
    inference_loc = "s3://obs-path/examples/image-classification/data/image_" + str(i) + ".txt"
    annotations_list = []

    for j in range(1):
        annotation_type = "modelarts/image_classification"
        if 0 == i % 2:
            annotation_name = "Bees"
        else:
            annotation_name = "Rabbits"
        annotation_creation_time = "2019-02-20 08:23:06"
        annotation_format = "manifest"
        annotation_property = {"color": "black"}
        annotation_confidence = 0.8
        annotated_by = "human"
        annotations_list.append(
            Annotation(name=annotation_name, type=annotation_type,
                       confidence=annotation_confidence,
                       creation_time=annotation_creation_time,
                       annotated_by=annotated_by, annotation_format=annotation_format,
                       annotation_property=annotation_property))
    sample_list.append(
        Sample(source=source, usage=usage, annotations=annotations_list, inference_loc=inference_loc))
manifest_info = Manifest(samples=sample_list, size=size)

path = "obs://your-obs-bucket/manifest/V001.manifest"
session = Session()
manifest_info.save(path, session=session, save_mode="a")
```

参数说明

表 7-28 请求参数

参数	是否必选	参数类型	描述
path	是	String	Manifest文件保存路径。
session	否	Object	会话对象，初始化方法请参见**Session鉴权**。当需要操作OBS时必填。
save_mode	否	String	保存模式。默认为w，即重写模式，另外还支持a，为追加模式。

7.6.4 解析 Pascal VOC 文件

解析xml文件支持本地和OBS，如果是OBS，需要Session信息。

```
PascalVoc.parse_xml(xml_file_path, session=None)
```

示例代码

指定xml路径，通过调用parse_xml来解析获取xml文件的信息。

```
from modelarts.dataset.format.voc.pascal_voc import PascalVoc
from modelarts.session import Session

path = "obs://your-obs-bucket/voc/test.xml"
session = Session()
pascal_voc = PascalVoc.parse_xml(path, session=session)
print(pascal_voc) # 打印解析结果
```

参数说明

表 7-29 请求参数

参数	是否必选	参数类型	描述
xml_file_path	是	String	xml文件路径。
session	否	Object	会话对象，初始化方法请参见**Session鉴权**。当需要操作OBS时必填。

表 7-30 pascal_voc 返回参数

参数	参数类型	描述
folder	String	文件夹名称。
file_name	String	文件名称。

参数	参数类型	描述
source	Object	数据源信息，详细请见表7-31。
width	Long	图片长度。
height	Long	图片高度。
depth	Long	图片深度。
segmented	String	分割。
mask_source	String	图像分割得到的mask文件的云存储路径，目前只支持PNG格式。
voc_objects	JSON Array	标注对象列表，详细请见表7-32。

表 7-31 source 参数

参数	参数类型	描述
database	String	数据集名称，比如“The VOC2007 Database”。
annotation	String	标注，比如“PASCAL VOC2007”。
image	String	图片信息。

表 7-32 voc_object 参数

参数	参数类型	描述
name	String	文件夹名称。
properties	JSON Array	标注对象属性，为key-value列表格式，其中key和value的值均为String类型。
pose	String	标注内容的拍摄角度。
truncated	String	标注内容是否被截断（0表示完整）。
occluded	String	标注内容是否被遮挡（0表示未遮挡）。
difficult	String	标注目标是否难以识别（0表示容易识别）。
confidence	Double	置信度，数值类型，范围0<=confidence<=1，表示机器标注的置信度。
position	Object	标注对象的位置信息，详细请见表7-33。
parts	Object	子标注对象列表，即嵌套的voc_object列表，详细请见表7-32。

参数	参数类型	描述
mask_color	String	图像分割mask图像的颜色。

表 7-33 Position 说明

type	形状	标注信息
point	点	点的坐标 <x>100<x> <y>100<y>
line	线	各点坐标 <x1>100<x1> <y1>100<y1> <x2>200<x2> <y2>200<y2>
bndbox	矩形框	左下和右上两个点坐标 <x_min>100<x_min> <y_min>100<y_min> <x_max>200<x_max> <y_max>200<y_max>
polygon	多边形	各点坐标 <x1>100<x1> <y1>100<y1> <x2>200<x2> <y2>100<y2> <x3>250<x3> <y3>150<y3> <x4>200<x4> <y4>200<y4> <x5>100<x5> <y5>200<y5> <x6>50<x6> <y6>150<y6>
circle	圆形	圆心坐标和半径 <cx>100<cx> <cy>100<cy> <r>50<r>

7.6.5 创建和保存 Pascal VOC 文件

需要先创建包含Pascal VOC信息的对象，然后保存。Pascal VOC信息请见表7-30。路径支持本地和OBS，如果是OBS，需要Session信息。

```
pascal_voc.save_xml(xml_file_path, save_mode='w', session=None)
```

示例代码

在保存Pascal VOC的XML文件之前需要先创建包含Pascal VOC信息的对象，包括voc object信息等。保存的时候调用save_xml接口，将session信息传入，即可保存到指定路径。

```
from modelarts.dataset.format.voc.pascal_voc import PascalVoc
from modelarts.dataset.format.voc.voc_object import VocObject
from modelarts.session import Session

path = "obs://your-obs-bucket/voc/test2.xml"
size_list = [640, 321, 3]
file_name = "000000089955.jpg"
voc_object_tags = ["trafficlight", "trafficlight"]
voc_object_properties = [{"@modelarts:color": "#FFFFF0", "@modelarts:shortcut": "C",
                          "pose": "0", "truncated": "0", "difficult": "0",
                          "@modelarts:shape": "bndbox", "@modelarts:feature": [[347, 186], [382, 249]]},
                         {"@modelarts:color": "#FFFFE0", "@modelarts:shortcut": "D",
                          "pose": "0", "truncated": "0", "difficult": "0",
                          "@modelarts:shape": "bndbox", "@modelarts:feature": [[544, 50], [591, 149]]}]
voc_objects = []
for i in range(len(voc_object_tags)):
    object_tag = voc_object_tags[i]
    object_properties = voc_object_properties[i]
    voc_objects.append(VocObject(name=object_tag, properties=object_properties))

pascal_voc = PascalVoc(file_name=file_name, width=size_list[0], height=size_list[1], depth=size_list[2],
                       voc_objects=voc_objects)
session = Session()
pascal_voc.save_xml(path, session=session)
```

参数说明

表 7-34 请求参数

参数	是否必选	参数类型	描述
xml_file_path	是	String	Pascal VOC格式的XML文件保存路径。
session	否	Object	会话对象，初始化方法请参见Session鉴权。当需要操作OBS时必填。
save_mode	否	String	保存模式。默认为w，即重写模式，另外还支持a，为追加模式。

7.7 标注任务管理

7.7.1 创建标注任务

基于数据集创建标注任务。

```
dataset.create_label_task(self, task_name=None, task_type=None, **kwargs)
```

示例代码

示例一：基于图像类型的数据集创建物体检测标注任务。

```
from modelarts.session import Session
from modelarts.dataset import Dataset

session = Session()
dataset = Dataset(session, dataset_id="VukxA2FlaTUm7tkDtq0") # 初始化数据集
create_task_resp = dataset.create_label_task(task_name="obj_detection_task", task_type=1,
description="label task")
```

参数说明

表 7-35 请求参数

参数	是否必选	参数类型	描述
task_name	是	String	标注任务的名称。
task_type	是	Integer	标注任务的类型。可选值如下： ● 0：图像分类 ● 1：物体检测 ● 3: 图像分割 ● 100：文本分类 ● 101：命名实体 ● 102：文本三元组 ● 200：声音分类 ● 201：语音内容 ● 202：语音分割 ● 400：表格数据集 ● 600：视频标注 ● 900：自由格式
description	否	String	标注任务的描述信息。

7.7.2 查询数据集的标注任务列表

查询当前数据集的所有标注任务列表。

```
dataset.get_label_tasks(is_workforce_task=False, **kwargs)
```

示例代码

- 示例一：查询数据集下所有的标注任务，根据标注任务创建时间降序排序。

  ```
  from modelarts.session import Session
  from modelarts.dataset import Dataset
  ```

```
session = Session()
dataset = Dataset(session,dataset_id="VukxA2FlaTUm7tkDtq0")
list_label_task_resp = dataset.get_label_tasks(sort_key="create_time", sort_dir="desc")
print(list_label_task_resp)
```

- 示例二：查询数据集下所有的团队标注任务。

```
list_label_task_resp = dataset.get_label_tasks(is_workforce_task=True)
print(list_label_task_resp)
```

参数说明

表 7-36 请求参数

参数	是否必选	参数类型	描述
is_workforce_task	否	Boolean	过滤条件，是否只获取团队标注任务。 ● True：只查询团队标注任务 ● False：默认值，查询所有标注任务
sort_key	否	String	排序字段。可选值如下： ● create_time：根据创建时间排序 ● task_name：根据任务名称进行排序
sort_dir	否	String	排序方式。可选值如下： ● asc：按照升序排序 ● desc：默认值，按照降序排序

7.7.3 查询标注任务详情

查询标注任务的详细信息。

```
datset.get_label_task_info(task_id=None)
```

示例代码

查询标注任务的详情。

```
task_info = dataset.get_label_task_info(task_id="xs9ZKzLluKzccQfsyi2")
print(task_info)
```

参数说明

表 7-37 请求参数

参数	是否必选	参数类型	描述
task_id	是	String	标注任务的ID。

8 训练管理

8.1 训练作业

8.1.1 创建训练作业

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

说明

ModelArts SDK不支持通过在AI Gallery中订阅的算法创建训练作业。

- 示例一：**提交常用框架训练作业**

  Estimator中同时指定framework_type和framework_version，会提交一个常用框架训练作业。

```
from modelarts.session import Session
from modelarts.train_params import TrainingFiles
from modelarts.train_params import OutputData
from modelarts.train_params import InputData
from modelarts.estimatorV2 import Estimator
session = Session()
#训练脚本里接收的参数，请根据实际情况填写

parameters = [{"name": "mod", "value":"gpu"},
              {"name": "epoc_num", "value":2}]
estimator = Estimator(session=session,
                      training_files=TrainingFiles(code_dir= "obs://bucket_name/code_dir/",
boot_file="boot_file.py"),
                      outputs=[OutputData(obs_path="obs://bucket_name/output/", name="output_dir")],
                      parameters=parameters,
                      framework_type='PyTorch',                    # 常用框架类型
                      framework_version='PyTorch-1.4.0-python3.6',   # 常用框架版本
                      train_instance_type="modelarts.p3.large.public",
                      train_instance_count=1,
                      log_url="obs://bucket_name/log/",
                      env_variables={"USER_ENV_VAR": "customize environment variable"},
                      working_dir="/home/ma-user/modelarts/user-job-dir",
                      local_code_dir="/home/ma-user/modelarts/user-job-dir",
                      job_description='This is a image net train job')
job_instance = estimator.fit(inputs=[InputData(obs_path="obs://bucket_name/input/",
```



```
name="data_url")],
                        job_name="job_name_1")
```

- 示例二：**提交自定义镜像训练作业**

  Estimator中同时指定user_image_url和user_command，会提交自定义镜像训练作业，并使用自定义启动命令来启动训练任务。

```
from modelarts.session import Session
from modelarts.train_params import TrainingFiles
from modelarts.train_params import OutputData
from modelarts.train_params import InputData
from modelarts.estimatorV2 import Estimator
session = Session()
#训练脚本里接收的参数，请根据实际情况填写

parameters = [{"name": "mod", "value":"gpu"},
              {"name": "epoc_num", "value":2}]
estimator = Estimator(session=session,
                      training_files=TrainingFiles(code_dir= "obs://bucket_name/code_dir/",
boot_file="boot_file.py"),
                      outputs=[OutputData(obs_path="obs://bucket_name/output/", name="output_dir")],
                      parameters=parameters,
                      user_image_url="sdk-test/pytorch1_4:1.0.1",     # 自定义镜像地址
                      user_command="/home/ma-user/anaconda3/envs/PyTorch-1.4/bin/python /home/ma-
user/modelarts/user-job-dir/train/test-pytorch.py",  # 自定义启动命令
                      train_instance_type="modelarts.p3.large.public",
                      train_instance_count=1,
                      log_url="obs://bucket_name/log/",
                      env_variables={"USER_ENV_VAR": "customize environment variable"},
                      working_dir="/home/ma-user/modelarts/user-job-dir",
                      local_code_dir="/home/ma-user/modelarts/user-job-dir",
                      job_description='This is a image net train job')
job_instance = estimator.fit(inputs=[InputData(obs_path="obs://bucket_name/input/",
name="data_url")],
                        job_name="job_name_2")
```

- 示例三：**向专属资源池提交训练作业**

```
from modelarts.session import Session
from modelarts.train_params import TrainingFiles
from modelarts.train_params import OutputData
from modelarts.train_params import InputData
from modelarts.estimatorV2 import Estimator
session = Session()
#训练脚本里接收的参数，请根据实际情况填写

parameters = [{"name": "mod", "value":"gpu"},
              {"name": "epoc_num", "value":2}]
estimator = Estimator(session=session,
                      training_files=TrainingFiles(code_dir= "obs://bucket_name/code_dir/",
boot_file="boot_file.py"),
                      outputs=[OutputData(obs_path="obs://bucket_name/output/", name="output_dir")],
                      parameters=parameters,
                      framework_type='PyTorch',
                      framework_version='PyTorch-1.4.0-python3.6',
                      pool_id="your pool id",                                # 专属资源池id
                      train_instance_type="modelarts.pool.visual.xlarge",     # 专属池的虚拟子规格
                      train_instance_count=1,
                      log_url="obs://bucket_name/log/",
                      env_variables={"USER_ENV_VAR": "customize environment variable"},
                      working_dir="/home/ma-user/modelarts/user-job-dir",
                      local_code_dir="/home/ma-user/modelarts/user-job-dir",
                      job_description='This is a image net train job')
job_instance = estimator.fit(inputs=[InputData(obs_path="obs://bucket_name/input/",
name="data_url")],
                        job_name="job_name_3")
```

- 示例四：**使用数据集创建训练作业**

```
from modelarts.session import Session
from modelarts.train_params import TrainingFiles
from modelarts.train_params import OutputData
```



```
from modelarts.train_params import InputData
from modelarts.estimatorV2 import Estimator
session = Session()
#训练脚本里接收的参数，请根据实际情况填写
parameters = [{"name": "model_name", "value":"s"},
              {"name": "batch-size", "value": 32},
              {"name": "epochs", "value":100},
              {"name": "img-size", "value":"640,640"} ]
estimator = Estimator(session=session,
                      training_files=TrainingFiles(code_dir= "obs://bucket_name/code_dir/",
boot_file="boot_file.py"),
                      outputs=[OutputData(obs_path="obs://bucket_name/output/", name="output_dir")],
                      parameters=parameters,
                      framework_type='PyTorch',                    # 常用框架类型
                      framework_version='PyTorch-1.4.0-python3.6',    # 常用框架版本
                      train_instance_type="modelarts.p3.large.public",
                      train_instance_count=1,
                      log_url="obs://bucket_name/log/",
                      working_dir="/home/ma-user/modelarts/user-job-dir",
                      local_code_dir="/home/ma-user/modelarts/user-job-dir",
                      job_description='This is a image net train job')
job_instance = estimator.fit(dataset_id="your dataset id",
                             dataset_version_id="your dataset version id",
                             job_name="job_name_5")
```

参数说明

表 8-1 Estimator 请求参数说明

参数	是否必选	类型	描述
session	是	Object	会话对象，初始化方法请参考**Session鉴权**。
training_files	否	**TrainingFiles** Object	训练脚本在OBS中的路径，具体请参考**TrainingFiles初始化**。
outputs	否	Array of **OutputData** objects	训练的输出位置信息，具体请参考**OutputData初始化**。
parameters	否	JSON Array	训练任务的运行参数。格式为 [{"name":"your name", "value": "your value"}]，value的值可以是string，也可以是int。
train_instance_type	是	String	训练作业选择的资源规格，请参考**查询资源规格列表**
train_instance_count	是	int	训练作业计算节点个数。
framework_type	否	String	训练作业选择的引擎规格，请参考**查询引擎规格列表**。
framework_version	否	String	训练作业选择的引擎版本，请参考**查询引擎规格列表**。

参数	是否必选	类型	描述
user_image_url	否	String	自定义镜像训练作业的自定义镜像的SWR-URL。
user_command	否	String	自定义镜像训练作业的启动命令。
log_url	否	String	训练作业日志的保存位置，是一个OBS路径，如"obs://xx/yy/zz/"。
local_code_dir	否	String	算法的代码目录下载到训练容器内的本地路径。规则： ● 必须为/home下的目录。 ● v1兼容模式下，当前字段不生效。 ● 当code_dir以file://为前缀时，当前字段不生效。
working_dir	否	String	运行算法时所在的工作目录。规则：v1兼容模式下，当前字段不生效。
job_description	否	String	训练作业的描述。

volumes	否	JSON Array	训练作业挂载卷信息，格式参考：
			[{
			"nfs": {
			"local_path": "/xx/yy/zz",
			"read_only": False,
			"nfs_server_path": "xxx.xxx.xxx.xxx:/"
			}
			}]

env_variables	否	Dict	训练作业的环境变量。
pool_id	否	String	训练作业选择的资源池ID。可在ModelArts管理控制台，单击左侧“专属资源池”，在专属资源池列表中查看资源池ID。

表 8-2 TrainingFiles 初始化参数说明

参数	是否必选	类型	描述
code_dir	是	String	训练作业的代码目录，是一个OBS路径，需要以"obs:/"开头，如"obs://xx/yy/"。

参数	是否必选	类型	描述
boot_file	是	String	训练作业的代码启动文件，需要在代码目录下，可填写相对路径，如"boot_file.py"，也可填写绝对路径，如"obs://xx/yy/boot_file.py"。

表 8-3 OutputData 初始化参数说明

参数	是否必选	类型	描述
obs_path	是	String	数据实际输出到OBS的路径。
name	是	String	输出数据的关键字参数名称，如"output_dir"。

表 8-4 fit 请求参数说明

参数	是否必选	类型	描述
inputs	否	Array of **InputData** Object	保存在OBS中的训练作业输入数据。inputs和(dataset_id, dataset_version_id)两者不可同时出现。
wait	否	Boolean	是否等待训练作业结束，默认为False。
job_name	否	String	训练作业名称。
show_log	否	Boolean	作业提交成功后，是否输出训练作业的日志，默认为False。
dataset_id	否	String	训练作业的数据集ID。需要与dataset_version_id同时出现，但是不可与inputs同时出现。
dataset_version_id	否	String	训练作业的数据集版本ID。需要与dataset_id同时出现，但是不可与inputs同时出现。

表 8-5 InputData 初始化参数说明

参数	是否必选	类型	描述
obs_path	是	String	训练作业需要的数据集OBS路径，如"obs://xx/yy/"。
name	是	String	输入数据的关键字参数名称，如"data_url"。

表 8-6 训练作业创建成功响应说明

参数	类型	描述
TrainingJob	Object	训练对象，该对象包含job_id等属性，对训练作业的查询、更新、删除等操作时，可通过job_instance.job_id获取训练作业ID。

表 8-7 调用训练接口失败响应参数

参数	类型	描述
error_msg	String	调用失败时的错误信息，调用成功时无此字段。
error_code	String	调用失败时的错误码，具体请参见**错误码**，调用成功时无此字段。
error_solution	String	调用失败时的提示解决信息，调用成功时无此字段。

8.1.2 训练作业调测

8.1.2.1 使用 SDK 调测单机训练作业

代码中涉及到的OBS路径，请用户替换为自己的实际OBS路径。

代码是以PyTorch为例编写的，不同的AI框架之间，整体流程是完全相同的，仅需修改**步骤6**和**步骤10**中的framework_type参数值即可，例如：MindSpore框架，此处framework_type=Ascend-Powered-Engine。

步骤1 Session初始化。

代码如下：

```
from modelarts.session import Session
session = Session()
```

步骤2 准备训练数据，这里支持三种形式，用户可根据自己的情况选择一种。

```
import os
from modelarts.train_params import InputData
base_bucket_path = "obs://modelarts-xxx-a0de02a6/dis-train/cifar10/"
base_local_path = "/home/ma-user/work/cifar10/"
```

```
# 形式1，数据在OBS上，且是一个压缩文件
obs_path = os.path.join(base_bucket_path, "dataset-zip/dataset.zip")
data_local = os.path.join(base_local_path, "dataset/")
input_data = InputData(obs_path=obs_path, local_path=data_local, is_local_source=False)

# 形式2，数据在OBS上，且是一个目录
#obs_path = os.path.join(base_bucket_path, "dataset/")
#data_local = os.path.join(base_local_path, "dataset/")
#input_data = InputData(obs_path=obs_path, local_path=data_local, is_local_source=False)

# 形式3，数据在Notebook中，且是一个目录，一般是使用SFS挂载磁盘的场景
#obs_path = os.path.join(base_bucket_path, "dataset-local/")
#data_local = os.path.join(base_local_path, "dataset/")
#input_data = InputData(obs_path=obs_path, local_path=data_local, is_local_source=True)
```

参数解释：

- is_local_source：可选参数，默认为False，指定训练数据的保存位置。
  - False：训练数据保存在参数obs_path指定的位置中；
  - True：训练数据保存在notebook中，由local_path指定。
- obs_path：obs地址。根据is_local_source值的变化，有不同的含义。
  - is_local_source=False，此时是必选参数，代表训练数据位置，支持文件夹和压缩文件。
  - is_local_source=True，此时是可选参数。如果用户填写了该参数，则开始训练时会将Notebook中的训练数据压缩并上传到该位置，不可重复上传。如果第一次上传后，建议将is_local_source修改为False，obs_path指向刚才上传的压缩数据文件位置；如果用户没有填写，则不会进行压缩上传。
- local_path：必选参数，Notebook中的路径。用户的训练脚本需要从该目录中读取数据，完成训练任务。根据is_local_source值的变化，有不同的含义。
  - is_local_source=True，此时代表训练数据位置，仅支持文件夹。
  - is_local_source=False，训练过程中SDK会帮助用户将数据下载到该位置，如果训练数据是压缩文件，下载完成后会进行解压缩。

步骤3 准备训练脚本。

```
from modelarts.train_params import TrainingFiles
code_dir = os.path.join(base_local_path, "train/")

# 这里提前将训练脚本放在了obs中，实际上训练脚本可以是任何来源，只要能够放到Notebook里边就行

session.obs.download_file(os.path.join(base_bucket_path, "train/test-pytorch.py"), code_dir)
training_file = TrainingFiles(code_dir=code_dir, boot_file="test-pytorch.py", obs_path=base_bucket_path + 'train/')
```

参数解释：

- code_dir：必选参数，训练脚本所在的目录。在训练任务调测的情况下，必须是notebook中的目录，不能是OBS目录。
- boot_file：必选参数，训练启动文件路径，路径格式为基于code_dir目录的相对路径，如实例代码中boot_file的完整路径为/home/ma-user/work/cifar10/train/test-pytorch.py，这里就只需要填写test-pytorch.py。
- obs_path：可选参数，一个OBS目录。仅在本地单机调试时不需要该参数，提交远程训练时必选，会将训练脚本压缩并上传到该路径。

步骤4 准备训练输出，如果用户不需要将训练输出上传到OBS，可以省略这一步。

```
from modelarts.train_params import OutputData
output = OutputData(local_path=os.path.join(base_local_path, "output/"),
obs_path=os.path.join(base_bucket_path, 'output/'))
```


- local_path：必选参数，一个notebook中的路径，训练脚本需要将输出的模型或其他数据保存在该目录下。
- obs_path：必选参数，一个OBS目录。SDK会将local_path中的模型文件自动上传到这里。

步骤5 查看训练支持的AI框架。

```
from modelarts.estimatorV2 import Estimator
Estimator.get_framework_list(session)
```

参数session即是第一步初始化的数据。如果用户知道要使用的AI框架，可以略过这一步。

步骤6 Estimator初始化。

```
from modelarts.estimatorV2 import Estimator
parameters = []
parameters.append({"name": "data_url", "value": data_local})
parameters.append({"name": "output_dir", "value": os.path.join(base_local_path, "output/")})
parameters.append({"name": "epoc_num", "value": 2})
estimator = Estimator(session=session,
                      training_files=training_file,
                      outputs=[output],
                      parameters=parameters,
                      framework_type='PyTorch',
                      train_instance_type='local',
                      train_instance_count=1,
                      script_interpreter="/home/ma-user/anaconda3/envs/PyTorch-1.4/bin/python",
                      log_url=base_bucket_path + 'log/',
                      job_description='This is a image net train job')
```

参数解释：

- session：必选参数，**步骤1**中初始化的参数。
- training_files：必选参数，**步骤3**中初始化的训练文件。
- outputs：可选参数，这里传入的是一个list，每个元素都是**步骤4**中初始化的训练输出。
- parameters：可选参数，一个list，每个元素都是一个字典，包含"name"和"value"两个字段，以"--name=value"的形式传递给训练启动文件。value支持字符串，整数，布尔等类型。对于布尔类型，建议用户在训练脚本中使用action='store_true'的形式来解析。
- framework_type：必选参数，训练作业使用的AI框架类型，可参考步骤5查询的返回结果。
- train_instance_type：必选参数，训练实例类型，这里指定'local'即为在notebook中进行训练。
- train_instance_count：必选参数，训练使用的worker个数，单机训练时为1，训练作业只在当前使用的notebook中运行。
- script_interpreter：可选参数，指定使用哪个python环境来执行训练任务，如果未指定，会默认使用当前的kernel。
- log_url：可选参数，一个OBS地址，训练过程中，SDK会自动将训练的日志上传到该位置。但是如果训练任务运行在Ascend上，则是必选参数。
- job_description：可选参数，训练任务的描述。

步骤7 开始训练。

```
estimator.fit(inputs=[input_data], job_name="cifar10-dis")
```

参数解释：

- inputs：可选参数，一个list，每个元素都是**步骤2**生成的实例。
- job_name：可选参数，训练任务名，便于区分和记忆。

本地单机调试训练任务开始后，SDK会依次帮助用户完成以下流程：

1. 初始化训练作业，如果**步骤2**指定的训练数据在OBS上，这里会将数据下载到local_path中。
2. 执行训练任务，用户的训练代码需要将训练输出保存在**步骤4**中指定的local_path中。
3. 将训练任务得到的输出上传到**步骤4**指定的obs_path中，日志上传到第六步指定的log_url中。

同时，可以在任务名后增加时间后缀，区分不同的任务名称。

```
from datetime import datetime, timedelta
import time
base_name = "cifar10-dis"
job_name = base_name + '-' + (datetime.now() + timedelta(hours=8)).strftime('%Y%m%d-%H%M%S')
estimator.fit(inputs=[input_data], job_name=job_name)
```

步骤8 多次调试。

上一步执行过程中，训练脚本的日志会实时打印到控制台，如果用户的代码或者参数有误的话，可以很方便地看到。在中经过多次调试，得到想要的结果后，可以进行下一步。

步骤9 查询训练支持的计算节点类型和最大个数。

```
from modelarts.estimatorV2 import Estimator
Estimator.get_spec_list(session=session)
```

参数session即是**步骤1**初始化的数据。返回的是一个字典，其中flavors值是一个列表，描述了训练服务支持的所有规格的信息。每个元素中flavor_id是可直接用于远程训练任务的计算规格，max_num是该规格的最大节点数。如果用户知道要使用的计算规格，可以略过这一步。

步骤10 提交远程训练作业。

```
from modelarts.estimatorV2 import Estimator
parameters = []
parameters.append({"name": "data_url", "value": data_local})
parameters.append({"name": "output_dir", "value": os.path.join(base_local_path, "output/")})
parameters.append({"name": "epoch_num", "value": 2})
estimator = Estimator(session=session,
                      training_files=training_file,
                      outputs=[output],
                      parameters=parameters,
                      framework_type='PyTorch',
                      train_instance_type='modelarts.vm.cpu.8u',
                      train_instance_count=1,
                      script_interpreter="/home/ma-user/anaconda3/envs/PyTorch-1.4/bin/python",
                      log_url=base_bucket_path + 'log/',
                      job_description='This is a image net train job')
estimator.fit(inputs=[input_data], job_name="cifar10-dis")
```

在本地调测完成的基础上，只需要Estimator初始化时将参数train_instance_type修改为训练服务支持的规格即可（即第10步查询出来的flavor_id的值）。执行fit函数后，即可提交远程训练任务。

训练任务提交后，SDK会依次帮助用户完成以下流程：

1. 将训练脚本打包成zip文件，上传到**步骤3**中指定的obs_path中。

2. 当训练数据保存在中，则将其打包成zip文件并上传到指定的obs_path中。
3. 向ModelArts训练服务提交自定义镜像训练作业，使用的镜像为当前的镜像，这样保证了远程训练作业和在中的训练作业使用的运行环境一致。
4. 训练任务得到的输出上传到**步骤4**指定的obs_path中，日志上传到这一步log_url指定的位置中。

   在这一步中需要注意的一个问题：

   如果用户在自己的训练脚本中要创建新的目录或文件，请在以下几种目录中创建：

   - /home/ma-user/work；
   - /cache；
   - inputs或者outputs中指定的local_path，如在步骤2中初始化InputData时，填写了local_path="/home/ma-user/work/xx/yy/"，则在该目录下也可以创建新目录或文件。

----结束

8.1.2.2 使用 SDK 调测多机分布式训练作业

代码中涉及到的OBS路径，请用户替换为自己的实际OBS路径。

代码是以PyTorch为例编写的，不同的AI框架之间，整体流程是完全相同的，仅需修改**步骤7**和**步骤11**中的 framework_type参数值即可，例如：MindSpore框架，此处framework_type=Ascend-Powered-Engine。

步骤1 Session初始化，与**使用SDK调测单机训练作业中的1**相同。

步骤2 准备训练数据，与**使用SDK调测单机训练作业中的2**相同，唯一的不同在于obs_path参数是必选的。

步骤3 准备训练脚本。

```
from modelarts.train_params import TrainingFiles
code_dir = os.path.join(base_local_path, "train/")

# 这里提前将训练脚本放在了obs中，实际上训练脚本可以是任何来源，只要能够放到Notebook里边就行

session.obs.download_file(os.path.join(base_bucket_path, "train/test-pytorch.py"), code_dir)
training_file = TrainingFiles(code_dir=code_dir, boot_file="test-pytorch.py", obs_path=base_bucket_path + 'train/')
```

参数解释：

- code_dir：必选参数，训练脚本所在的目录。在本地调试的情况下，必须是notebook目录，不能是OBS目录。
- boot_file：必选参数，训练启动文件，在code_dir目录下。
- obs_path：在多机分布式调测时必选参数，一个OBS目录，SDK会将notebook目录code_dir打包上传到obs_path中。

步骤4 准备训练输出，与单机训练作业调试**步骤4**相同。

步骤5 查看训练支持的AI框架，与单机训练作业调试**步骤5**相同。

步骤6 保存当前Notebook为新镜像，与单机训练作业调试**步骤9**相同。

步骤7 Estimator初始化。

```
from modelarts.estimatorV2 import Estimator
parameters = []
```



```
parameters.append({"name": "data_url", "value": data_local})
parameters.append({"name": "output_dir", "value": os.path.join(base_local_path, "output/")})
parameters.append({"name": "epoc_num", "value": 2})
# 启动脚本以parser.add_argument('--dist', action='store_true')的形式来接收该布尔类型的参数，如果要传入True，则以本行代码的形式传递；
parameters.append({"name": "dist"})
estimator = Estimator(session=session,
                      training_files=training_file,
                      outputs=[output],
                      parameters=parameters,
                      framework_type='PyTorch',
                      train_instance_type='local',
                      train_instance_count=2,
                      script_interpreter="/home/ma-user/anaconda3/envs/PyTorch-1.4/bin/python",
                      log_url=base_bucket_path + 'log/',
                      job_description='This is a image net train job')
```

参数解释：

- session：必选参数，**步骤1**中初始化的参数。
- training_files：必选参数，**步骤3**中初始化的训练文件。
- outputs：可选参数，这里传入的是一个list，每个元素都是步骤4中初始化的训练输出。
- parameters：可选参数，一个list，每个元素都是一个字典，包含"name"和"value"两个字段，以"–name=value"的形式传递给训练启动文件。value支持字符串，整数，布尔等类型。对于布尔类型，建议用户在训练脚本中使用action='store_true'的形式来解析。
- framework_type：必选参数，训练作业使用的AI框架类型，可参考步骤5的返回结果。
- train_instance_type：必选参数，训练实例类型，这里指定’local’即为本地训练。
- train_instance_count：必选参数，训练使用的worker个数，分布式调测时为2，训练开始时SDK还会再创建一个Notebook，与当前的Notebook组成一个2节点的分布式调试环境。
- script_interpreter：可选参数，指定使用哪个python环境来执行训练任务，如果未指定，会默认使用当前的kernel。
- log_url：可选参数，一个OBS地址，本地训练过程中，SDK会自动将训练的日志上传到该位置；但是如果训练任务运行在Ascend上，则是必选参数。
- job_description：可选参数，训练任务的描述。

步骤8 开始训练。

```
estimator.fit(inputs=[input_data], job_name="cifar10-dis")
```

参数解释：

- inputs：可选参数，一个list，每个元素都是步骤2中生成的实例；
- job_name：可选参数，训练任务名，便于区分和记忆。

本地分布式训练任务开始后，SDK会依次帮助用户完成以下流程：

1. 将训练脚本打包成zip文件，上传到**步骤3**中指定的obs_path中。
2. 如果训练数据保存在中，则将其打包成zip文件并上传到指定的obs_path中。
3. 创建一个附属，与当前使用的组成分布式训练的两个worker。
4. 初始化训练作业，将数据下载到local_path中。

5. 执行训练任务，用户的代码需要将训练输出保存在**步骤4**指定的local_path中。
6. 将训练任务得到的输出上传到**步骤4**指定的obs_path中，日志上传到**步骤7**指定的log_url中。

步骤9 多次调试，与单机调测时**步骤8**作用相同。

步骤10 查询训练支持的工作节点类型，与单机调测时**步骤9**相同。

步骤11 提交远程训练作业。

```
from modelarts.estimatorV2 import Estimator
parameters = []
parameters.append({"name": "data_url", "value": data_local})
parameters.append({"name": "output_dir", "value": os.path.join(base_local_path, "output/")})
parameters.append({"name": "epoc_num", "value": 2})
# 启动脚本以parser.add_argument('--dist', action='store_true')的形式来接收该布尔类型的参数，如果要传入True，则以本行代码的形式传递；
parameters.append({"name": "dist"})
estimator = Estimator(session=session,
                      training_files=training_file,
                      outputs=[output],
                      parameters=parameters,
                      framework_type='PyTorch',
                      train_instance_type='modelarts.p3.large.public.distributed',
                      train_instance_count=2,
                      script_interpreter="/home/ma-user/anaconda3/envs/PyTorch-1.4/bin/python",
                      log_url=base_bucket_path + 'log/',
                      job_description='This is a image net train job')
estimator.fit(inputs=[input_data], job_name="cifar10-dis-1")
```

Estimator初始化时与本地训练的区别在于参数train_instance_type，需要从**步骤10**得到的结果中选择一个；参数train_instance_count的值取决于第10步中的max_num。

训练任务提交后，SDK会依次帮助用户完成以下流程：

1. 将训练脚本打包成zip文件，上传到**步骤3**中指定的obs_path中；
2. 如果训练数据保存在中，则将其打包成zip文件并上传到指定的obs_path中；
3. 将训练作业提交到ModelArts训练服务中，训练作业会使用当前的镜像来执行训练作业；
4. 训练任务得到的输出上传到**步骤4**指定的obs_path中，日志上传到log_url指定的位置中。

   在这一步中需要注意的一个问题：

   如果用户在自己的训练脚本中要创建新的目录或文件，请在以下几种目录中创建：

   （1）/home/ma-user/work；

   （2）/cache；

   （3）inputs或者outputs中指定的local_path，如在**步骤2**中初始化InputData时，填写了local_path="/home/ma-user/work/xx/yy/"，则在该目录下也可以创建新目录或文件；

----结束

8.1.3 查询训练作业列表

示例代码

在ModelArts Notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。


```
from modelarts.session import Session
from modelarts.estimatorV2 import Estimator
session = Session()
job_list = Estimator.get_job_list(session=session, offset=10, limit=5, sort_by="create_time", order="asc",
                                  filters=[{"key": "name", "operator": "like", "value": ["trainjob"]}])

print(job_list)
```

参数说明

表 8-8 get_job_list 请求参数说明

参数	是否必选	参数类型	描述
session	是	Object	会话对象，初始化方法请参考**Session鉴权**。
offset	否	Integer	查询作业的偏移量，最小为0。例如设置为1，则表示从第二条开始查。
limit	否	Integer	查询作业的限制量。最小为1，最大为50。
sort_by	否	String	查询作业排列顺序的指标。默认使用create_time排序。
order	否	String	查询作业排列顺序，默认为“desc”，降序排序。也可以选择对应的“asc”，升序排序。 缺省值：**desc** 枚举值： • **asc：升序** • **desc：降序**
group_by	否	String	查询作业要搜索的分组条件。
filters	否	Array of **表2 filters** objects	查询作业要过滤的一系列条件。

表 8-9 filters

参数	是否必选	参数类型	描述
key	否	String	分组条件键值。

参数	是否必选	参数类型	描述
operator	否	String	分组条件键值键关系，支持between、like、in、not、rely_to。 缺省值：**in** 枚举值： • **like：类似** • **in：包含** • **not：非** • **between：范围**
value	否	Array of strings	分组条件键对应值。

表 8-10 get_job_list 返回参数说明

参数	参数类型	描述
total	Integer	查询到当前用户名下的所有作业总数。
count	Integer	查询到当前用户名下的所有符合查询条件的作业总数。
limit	Integer	查询作业的限制量。最小为1，最大为50。
offset	Integer	查询作业的偏移量，最小为0。例如设置为1，则表示从第二条开始查。
sort_by	String	查询作业排列顺序的指标。默认使用create_time排序。
order	String	查询作业排列顺序，默认为“desc”，降序排序。也可以选择对应的“asc”，升序排序。
group_by	String	查询作业要搜索的分组条件。
workspace_id	String	作业所处的工作空间，默认值为“0”。
ai_project	String	作业所属的AI项目，默认值为"default-ai-project"。

参数	参数类型	描述
items	Array of **JobResponse** objects	查询到当前用户名下的所有符合查询条件的作业详情。

表 8-11 JobResponse

参数	参数类型	描述
kind	String	训练作业类型。默认使用job。 枚举值： • **job：训练作业** • **hetero_job：异构作业** • **autosearch_job：自动搜索作业** • **mrs_job：mrs作业** • **edge_job：边缘作业**
metadata	**JobMetadata** object	训练作业元信息。
status	**Status** object	训练作业状态信息。创建作业无需填写。
algorithm	**JobAlgorithmResponse** object	训练作业算法。目前支持三种形式： • id只取算法的id； • subscription_id +item_version_id取算法的订阅id和版本id； • code_dir+boot_file取训练作业的代码目录和启动文件。
tasks	Array of **TaskResponse** objects	异构训练作业的任务列表。
spec	**spec** object	训练作业规格参数。

表 8-12 JobMetadata

参数	参数类型	描述
id	String	训练作业ID，创建成功后由ModelArts生成返回，无需填写。

参数	参数类型	描述
name	String	训练作业名称。限制为1-64位只含数字、字母、下划线和中划线的名称。
workspace_id	String	指定作业所处的工作空间，默认值为“0”。
description	String	对训练作业的描述，默认为“NULL”，字符串的长度限制为[0, 256]。
create_time	Long	训练作业创建时间戳，单位为毫秒，创建成功后由ModelArts生成返回，无需填写。
user_name	String	训练作业创建用户的用户名，创建成功后由ModelArts生成返回，无需填写。
annotations	Map<String,String>	训练作业申明模板，异构作业job_template字段默认为Template RL，其余默认为Template DL。

表 8-13 Status

参数	参数类型	描述
phase	String	训练作业一级状态，状态值稳定不变，可选值如下： “Creating”、“Pending”、“Running”、“Failed”、“Completed”、“Terminating”、“Terminated”、“Abnormal”。

参数	参数类型	描述
secondary_phase	String	训练作业二级状态，状态值不稳定，可选值如下： “Creating”、 “Queuing”、 “Running”、 “Failed”、 “Completed”、 “Terminating”、 “Terminated”、 “CreateFailed”、 “TerminatedFailed”、 “Unknown”、 “Lost”。
duration	Long	训练作业运行时长，单位为毫秒。
node_count_metrics	Array<Array<Integer>>	训练作业运行时节点数变化指标。
tasks	Array of strings	训练作业子任务名称。
start_time	String	训练作业开始时间，格式为时间戳。
task_statuses	Array of objects	训练在子任务状态信息。

表 8-14 task_statuses

参数	参数类型	描述
task	String	训练作业子任务名称。
exit_code	Integer	训练作业子任务退出码。
message	String	训练作业子任务错误消息。

表 8-15 JobAlgorithmResponse

参数	参数类型	描述
id	String	算法管理的算法id。 枚举值： • **id只取算法的id；** • **subscription_id +item_version_id取算法的订阅id和版本id；** • **code_dir+boot_file取训练作业的代码目录和启动文件。**
name	String	算法名称。
subscription_id	String	订阅算法的订阅ID。应与item_version_id一同出现。
item_version_id	String	订阅算法的版本。应与subscription_id一同出现。
code_dir	String	训练作业的代码目录。如：“/usr/app/”。应与boot_file一同出现，如果填入id或subscription_id +item_version_id则无需填写。
boot_file	String	训练作业的代码启动文件，需要在代码目录下。如：“/usr/app/boot.py”。应与code_dir一同出现，如果填入id或subscription_id +item_version_id则无需填写。
autosearch_config_path	String	自动化搜索作业的yaml配置路径，需要提供一个OBS路径。
autosearch_framework_path	String	自动化搜索作业的框架代码目录，需要提供一个OBS路径。
command	String	自定义镜像训练作业的自定义镜像的容器的启动命令。可填code_dir。
parameters	Array of **Parameter** objects	训练作业的运行参数。

参数	参数类型	描述
policies	**policies** object	作业支持的策略。
inputs	Array of **Input** objects	训练作业的数据输入。
outputs	Array of **Output** objects	训练作业的结果输出。
engine	**engine** object	训练作业的引擎。使用算法管理的算法id或订阅算法subscription_id +item_version_id创建作业时，无需填写。
environments	Map<String,String>	训练作业的环境变量。格式："key":"value"，无需填写。

表 8-16 Parameter

参数	参数类型	描述
name	String	参数名称。
value	String	参数值。
description	String	参数描述信息。
constraint	**constraint** object	参数属性。
i18n_description	**i18n_description** object	国际化描述。

表 8-17 constraint

参数	参数类型	描述
type	String	参数种类。
editable	Boolean	是否可编辑。
required	Boolean	是否必须。
sensitive	Boolean	是否敏感。
valid_type	String	有效种类。
valid_range	Array of strings	有效范围。

表 8-18 i18n_description

参数	参数类型	描述
language	String	国际语种。
description	String	对应描述。

表 8-19 policies

参数	参数类型	描述
auto_search	auto_search object	超参搜索配置。

表 8-20 auto_search

参数	参数类型	描述
skip_search_params	String	需要排除的超参组合。
reward_attrs	Array of objects	搜索指标列表。
search_params	Array of objects	搜索参数。
algo_configs	Array of objects	搜索算法配置。

表 8-21 reward_attrs

参数	参数类型	描述
name	String	指标名称。
mode	String	搜索方向。 • max：指定时，表示指标值越大越好； • min：指定时，表示指标值越小越好。
regex	String	指标正则表达式。

表 8-22 search_params

参数	参数类型	描述
name	String	超参名称。

参数	参数类型	描述
param_type	String	参数类型。 • continuous：指定时，表示参数类型为连续值； • discreate：指定时，表示参数类型为离散值。
lower_bound	String	超参下界。
upper_bound	String	超参上界。
discrete_points_num	String	连续型超参离散化取值个数。
discrete_values	Array of strings	离散型超参的取值列表。

表 8-23 algo_configs

参数	参数类型	描述
name	String	搜索算法名称。
params	Array of **AutoSearchAlgoConfigParameter** objects	搜索算法参数。

表 8-24 AutoSearchAlgoConfigParameter

参数	参数类型	描述
key	String	参数键。
value	String	参数值。
type	String	参数种类。

表 8-25 Input

参数	参数类型	描述
name	String	数据输入通道名称。
description	String	数据输入通道描述信息。
local_dir	String	数据输入通道映射的容器本地路径。
remote	**InputDataInfo** object	数据实际输入信息。

参数	参数类型	描述
remote_constraint	Array of objects	数据输入约束。

表 8-26 InputDataInfo

参数	参数类型	描述
dataset	**dataset** object	数据输入信息为数据集。
obs	**obs** object	数据输入输出信息为OBS方式。

表 8-27 dataset

参数	参数类型	描述
id	String	训练作业的数据集ID。
version_id	String	训练作业的数据集版本ID。
obs_url	String	训练作业需要的数据集OBS路径URL，ModelArts会通过数据集ID和数据集版本ID自动解析生成。如：“/usr/data/”。

表 8-28 obs

参数	参数类型	描述
obs_url	String	训练作业需要的数据集OBS路径URL。如：“/usr/data/”。

表 8-29 remote_constraint

参数	参数类型	描述
data_type	String	数据输入类型，包括数据存储位置、数据集两种方式。
attributes	String	数据输入为数据集时的相关属性。 枚举值： • data_format：数据格式。 • data_segmentation：数据切分方式。 • dataset_type：标注类型。

表 8-30 Output

参数	参数类型	描述
name	String	数据输出通道名称。
description	String	数据输出通道描述信息。
local_dir	String	数据输出通道映射的容器本地路径。
remote	**remote** object	数据实际输出信息。

表 8-31 remote

参数	参数类型	描述
obs	**obs** object	数据实际输出到OBS。

表 8-32 obs

参数	参数类型	描述
obs_url	String	数据实际输出到OBS的路径。

表 8-33 engine

参数	参数类型	描述
engine_id	String	训练作业选择的引擎规格ID。engine_id、engine_name + engine_version和image_url方式三选一。
engine_name	String	训练作业选择的引擎名称。如果填入engine_id，则无需填写。
engine_version	String	训练作业选择的引擎版本名称。如果填入engine_id，则无需填写。
image_url	String	训练作业选择的自定义镜像地址。

表 8-34 TaskResponse

参数	参数类型	描述
role	String	异构训练作业的任务角色。 枚举值： • learner（支持GPU\CPU规格） • worker（支持CPU规格）
algorithm	**algorithm** object	算法配置。
task_resource	**FlavorResponse** object	训练作业、算法的规格信息。

表 8-35 algorithm

参数	参数类型	描述
code_dir	String	算法启动文件所在目录绝对路径。
boot_file	String	算法启动文件绝对路径。
inputs	**inputs** object	算法输入通道信息。
outputs	**outputs** object	算法输出通道信息。
engine	**engine** object	异构作业所依赖的引擎。

表 8-36 inputs

参数	参数类型	描述
name	String	数据输入通道名称。
local_dir	String	数据输入输出通道映射的容器本地路径。
remote	**remote** object	数据实际输入信息，异构作业只支持OBS。

表 8-37 remote

参数	参数类型	描述
obs	**obs** object	数据输入输出信息为OBS方式。

表 8-38 obs

参数	参数类型	描述
obs_url	String	训练作业需要的数据集OBS路径URL。如：“/usr/data/”。

表 8-39 outputs

参数	参数类型	描述
name	String	数据输出通道名称。
local_dir	String	数据输出通道映射的容器本地路径。
remote	**remote** object	数据实际输出信息。
mode	String	数据传输模式，默认为“upload_periodically”。
period	String	数据传输周期，默认为30s。

表 8-40 remote

参数	参数类型	描述
obs	**obs** object	数据实际输出到OBS。

表 8-41 obs

参数	参数类型	描述
obs_url	String	数据实际输出到OBS的路径。

表 8-42 engine

参数	参数类型	描述
engine_id	String	异构作业引擎规格的ID。如“caffe-1.0.0-python2.7”。
engine_name	String	异构作业引擎规格的名称。如“Caffe”。
engine_version	String	异构作业引擎规格的版本。
v1_compatible	Boolean	是否为v1兼容模式。

参数	参数类型	描述
run_user	String	引擎默认启动用户uid。

表 8-43 FlavorResponse

参数	参数类型	描述
flavor_id	String	资源规格的ID。
flavor_name	String	资源规格的名称。
max_num	Integer	资源规格的最大节点数。
flavor_type	String	资源规格的类型。可选值如下： • CPU； • GPU； • Ascend。
billing	**billing** object	资源规格计费信息。
flavor_info	**flavor_info** object	资源规格详细信息。
attributes	Map<String,String>	其他规格属性。

表 8-44 billing

参数	参数类型	描述
code	String	计费码。
unit_num	Integer	计费卡数。

表 8-45 flavor_info

参数	参数类型	描述
max_num	Integer	可以选择的最大节点数量（max_num，为1代表不支持分布式）。
cpu	**cpu** object	cpu规格信息。
gpu	**gpu** object	gpu规格信息。
npu	**npu** object	Ascend规格信息。
memory	**memory** object	内存信息。

表 8-46 cpu

参数	参数类型	描述
arch	String	cpu架构。
core_num	Integer	核数。

表 8-47 gpu

参数	参数类型	描述
unit_num	Integer	gpu卡数。
product_name	String	产品名。
memory	String	内存。

表 8-48 npu

参数	参数类型	描述
unit_num	String	npu卡数。
product_name	String	产品名。
memory	String	内存。

表 8-49 memory

参数	参数类型	描述
size	Integer	内存大小。
unit	String	内存单元数。

表 8-50 spec

参数	参数类型	描述
resource	**Resource** object	训练作业资源规格信息。flavor_id和pool_id+[flavor_id]方式二选一。
volumes	Array of objects	训练作业挂载卷信息。
log_export_path	**log_export_path** object	训练作业日志输出信息。

表 8-51 Resource

参数	参数类型	描述
policy	String	训练作业资源规格模式，可选值如下： “regular”、“economic”、“turbo”。
flavor_id	String	训练作业选择的资源规格ID。
flavor_name	String	使用flavor_id时，由ModelArts返回的只读规格名称。
node_count	Integer	训练作业选择的资源副本数。 最小值：**1**
pool_id	String	训练作业选择的资源池ID。
flavor_detail	**flavor_detail** object	训练作业、算法的规格信息。

表 8-52 flavor_detail

参数	参数类型	描述
flavor_type	String	资源规格的类型。可选值如下： • CPU； • GPU； • Ascend。
billing	**billing** object	资源规格计费信息。
flavor_info	**flavor_info** object	资源规格详细信息。

表 8-53 billing

参数	参数类型	描述
code	String	计费码。
unit_num	Integer	计费卡数。

表 8-54 flavor_info

参数	参数类型	描述
max_num	Integer	可以选择的最大节点数量（max_num，为1代表不支持分布式）。
cpu	**cpu** object	cpu规格信息。

参数	参数类型	描述
gpu	**gpu** object	gpu规格信息。
npu	**npu** object	Ascend规格信息。
memory	**memory** object	内存信息。
disk	**disk** object	磁盘信息。

表 8-55 cpu

参数	参数类型	描述
arch	String	cpu架构。
core_num	Integer	核数。

表 8-56 gpu

参数	参数类型	描述
unit_num	Integer	gpu卡数。
product_name	String	产品名。
memory	String	内存。

表 8-57 npu

参数	参数类型	描述
unit_num	String	npu卡数。
product_name	String	产品名。
memory	String	内存。

表 8-58 memory

参数	参数类型	描述
size	Integer	内存大小。
unit	String	内存单元数。

表 8-59 disk

参数	参数类型	描述
size	String	磁盘大小。
unit	String	磁盘大小单位，一般为GB。

表 8-60 volumes

参数	参数类型	描述
nfs	**nfs** object	nfs方式的挂载卷。

表 8-61 nfs

参数	参数类型	描述
nfs_server_path	String	nfs服务端路径。
local_path	String	挂载到训练容器中的路径。
read_only	Boolean	nfs挂载卷在容器中是否只读。

表 8-62 log_export_path

参数	参数类型	描述
obs_url	String	训练作业日志保存的OBS地址。
host_path	String	训练作业日志保存的宿主机的路径。

表 8-63 调用训练接口失败响应参数

参数	类型	描述
error_msg	String	调用失败时的错误信息，调用成功时无此字段。
error_code	String	调用失败时的错误码，具体请参见**错误码**，调用成功时无此字段。
error_solution	String	调用失败时的提示解决信息，调用成功时无此字段。

表 8-64 调用训练接口失败响应参数

参数	类型	描述
error_msg	String	调用失败时的错误信息，调用成功时无此字段。
error_code	String	调用失败时的错误码，具体请参见**错误码**，调用成功时无此字段。
error_solution	String	调用失败时的提示解决信息，调用成功时无此字段。

8.1.4 查询训练作业详情

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

- 方式一：**根据指定的job_id查询。**

```
from modelarts.session import Session
from modelarts.estimatorV2 import Estimator
session = Session()
estimator = Estimator(session=session, job_id="618222c4-dc2f-4cfe-bc49-72b075b7552f")
job_info = estimator.get_job_info()
print(job_info)
```

- 方式二：**根据创建训练作业生成的训练作业对象查询。**

```
job_info = job_instance.get_job_info()
print(job_info)
```

参数说明

表 8-65 Estimator 请求参数说明

参数	是否必选	参数类型	描述
session	是	Object	会话对象，初始化方法请参考**Session鉴权**。
job_id	是	String	训练作业的id，可通过**创建训练作业**生成的训练作业对象查询，如"job_instance.job_id"，或从**查询训练作业列表**的响应中获得。

表 8-66 get_job_info 返回参数说明

参数	参数类型	描述
kind	String	训练作业类型。默认使用job。 枚举值： • **job：训练作业** • **hetero_job：异构作业** • **autosearch_job：自动搜索作业** • **mrs_job：mrs作业** • **edge_job：边缘作业**
metadata	**JobMetadata** object	训练作业元信息。
status	**Status** object	训练作业状态信息。创建作业无需填写。
algorithm	**JobAlgorithmResponse** object	训练作业算法。目前支持三种形式： • id只取算法的id； • subscription_id+item_version_id取算法的订阅id和版本id； • code_dir+boot_file取训练作业的代码目录和启动文件。
tasks	Array of **TaskResponse** objects	异构训练作业的任务列表。
spec	**spec** object	训练作业规格参数。

表 8-67 JobMetadata

参数	参数类型	描述
id	String	训练作业ID，创建成功后由ModelArts生成返回，无需填写。
name	String	训练作业名称。限制为1-64位只含数字、字母、下划线和中划线的名称。
workspace_id	String	指定作业所处的工作空间，默认值为“0”。
description	String	对训练作业的描述，默认为“NULL”，字符串的长度限制为[0, 256]。
create_time	Long	训练作业创建时间戳，单位为毫秒，创建成功后由ModelArts生成返回，无需填写。
user_name	String	训练作业创建用户的用户名，创建成功后由ModelArts生成返回，无需填写。

参数	参数类型	描述
annotations	Map<String,String>	训练作业申明模板，异构作业job_template字段默认为Template RL，其余默认为Template DL。

表 8-68 Status

参数	参数类型	描述
phase	String	训练作业一级状态，状态值稳定不变，可选值如下：“Creating”、“Pending”、“Running”、“Failed”、“Completed”、“Terminating”、“Terminated”、“Abnormal”。
secondary_phase	String	训练作业二级状态，状态值不稳定，可选值如下：“Creating”、“Queuing”、“Running”、“Failed”、“Completed”、“Terminating”、“Terminated”、“CreateFailed”、“TerminatedFailed”、“Unknown”、“Lost”。
duration	Long	训练作业运行时长，单位为毫秒。
node_count_metrics	Array<Array<Integer>>	训练作业运行时节点数变化指标。
tasks	Array of strings	训练作业子任务名称。
start_time	String	训练作业开始时间，格式为时间戳。
task_statuses	Array of objects	训练在子任务状态信息。

表 8-69 task_statuses

参数	参数类型	描述
task	String	训练作业子任务名称。
exit_code	Integer	训练作业子任务退出码。
message	String	训练作业子任务错误消息。

表 8-70 JobAlgorithmResponse

参数	参数类型	描述
id	String	算法管理的算法id。 枚举值： • **id：只取算法的id；** • **subscription_id+item_version_id：取算法的订阅id和版本id；** • **code_dir+boot_file：取训练作业的代码目录和启动文件。**
name	String	算法名称。
subscription_id	String	订阅算法的订阅ID。应与item_version_id一同出现。
item_version_id	String	订阅算法的版本。应与subscription_id一同出现。
code_dir	String	训练作业的代码目录。如：“/usr/app/”。应与boot_file一同出现，如果填入id或subscription_id+item_version_id则无需填写。
boot_file	String	训练作业的代码启动文件，需要在代码目录下。如：“/usr/app/boot.py”。应与code_dir一同出现，如果填入id或subscription_id+item_version_id则无需填写。
autosearch_config_path	String	自动化搜索作业的yaml配置路径，需要提供一个OBS路径。
autosearch_framework_path	String	自动化搜索作业的框架代码目录，需要提供一个OBS路径。
command	String	自定义镜像训练作业的自定义镜像的容器的启动命令。可填code_dir。
parameters	Array of **Parameter** objects	训练作业的运行参数。
policies	**policies** object	作业支持的策略。
inputs	Array of **Input** objects	训练作业的数据输入。
outputs	Array of **Output** objects	训练作业的结果输出。
engine	**engine** object	训练作业的引擎。使用算法管理的算法id或订阅算法subscription_id+item_version_id创建作业时，无需填写。

参数	参数类型	描述
environments	Map<String,String>	训练作业的环境变量。格式："key":"value"，无需填写。

表 8-71 Parameter

参数	参数类型	描述
name	String	参数名称。
value	String	参数值。
description	String	参数描述信息。
constraint	**constraint** object	参数属性。
i18n_description	**i18n_description** object	国际化描述。

表 8-72 constraint

参数	参数类型	描述
type	String	参数种类。
editable	Boolean	是否可编辑。
required	Boolean	是否必须。
sensitive	Boolean	是否敏感。
valid_type	String	有效种类。
valid_range	Array of strings	有效范围。

表 8-73 i18n_description

参数	参数类型	描述
language	String	国际语种。
description	String	对应描述。

表 8-74 policies

参数	参数类型	描述
auto_search	auto_search object	超参搜索配置。

表 8-75 auto_search

参数	参数类型	描述
skip_search_params	String	需要排除的超参组合。
reward_attrs	Array of objects	搜索指标列表。
search_params	Array of objects	搜索参数。
algo_configs	Array of objects	搜索算法配置。

表 8-76 reward_attrs

参数	参数类型	描述
name	String	指标名称。
mode	String	搜索方向。 ● max：指定时，表示指标值越大越好； ● min：指定时，表示指标值越小越好。
regex	String	指标正则表达式。

表 8-77 search_params

参数	参数类型	描述
name	String	超参名称。
param_type	String	参数类型。 ● continuous：指定时，表示参数类型为连续值； ● discreate：指定时，表示参数类型为离散值。
lower_bound	String	超参下界。
upper_bound	String	超参上界。

参数	参数类型	描述
discrete_points_num	String	连续型超参离散化取值个数。
discrete_values	Array of strings	离散型超参的取值列表。

表 8-78 algo_configs

参数	参数类型	描述
name	String	搜索算法名称。
params	Array of **AutoSearchAlgoConfigParameter** objects	搜索算法参数。

表 8-79 AutoSearchAlgoConfigParameter

参数	参数类型	描述
key	String	参数键。
value	String	参数值。
type	String	参数种类。

表 8-80 Input

参数	参数类型	描述
name	String	数据输入通道名称。
description	String	数据输入通道描述信息。
local_dir	String	数据输入通道映射的容器本地路径。
remote	**InputDataInfo** object	数据实际输入信息。
remote_constraint	Array of objects	数据输入约束。

表 8-81 InputDataInfo

参数	参数类型	描述
dataset	**dataset** object	数据输入信息为数据集。
obs	**obs** object	数据输入输出信息为OBS方式。

表 8-82 dataset

参数	参数类型	描述
id	String	训练作业的数据集ID。
version_id	String	训练作业的数据集版本ID。
obs_url	String	训练作业需要的数据集OBS路径URL，modelarts会通过数据集ID和数据集版本ID自动解析生成。如：“/usr/data/”。

表 8-83 obs

参数	参数类型	描述
obs_url	String	训练作业需要的数据集OBS路径URL。如：“/usr/data/”。

表 8-84 remote_constraint

参数	参数类型	描述
data_type	String	数据输入类型，包括数据存储位置、数据集两种方式。
attributes	String	数据输入为数据集时的相关属性。 枚举值： • data_format：数据格式。 • data_segmentation：数据切分方式。 • dataset_type：标注类型。

表 8-85 Output

参数	参数类型	描述
name	String	数据输出通道名称。

参数	参数类型	描述
description	String	数据输出通道描述信息。
local_dir	String	数据输出通道映射的容器本地路径。
remote	**remote** object	数据实际输出信息。

表 8-86 remote

参数	参数类型	描述
obs	**obs** object	数据实际输出到OBS。

表 8-87 obs

参数	参数类型	描述
obs_url	String	数据实际输出到OBS的路径。

表 8-88 engine

参数	参数类型	描述
engine_id	String	训练作业选择的引擎规格ID。engine_id、engine_name + engine_version和image_url方式三选一。
engine_name	String	训练作业选择的引擎名称。如果填入engine_id，则无需填写。
engine_version	String	训练作业选择的引擎版本名称。如果填入engine_id，则无需填写。
image_url	String	训练作业选择的自定义镜像地址。

表 8-89 TaskResponse

参数	参数类型	描述
role	String	异构训练作业的任务角色。 枚举值： • learner（支持GPU\CPU规格） • worker（支持CPU规格）
algorithm	**algorithm** object	算法配置。

参数	参数类型	描述
task_resource	**FlavorResponse** object	训练作业、算法的规格信息。

表 8-90 algorithm

参数	参数类型	描述
code_dir	String	算法启动文件所在目录绝对路径。
boot_file	String	算法启动文件绝对路径。
inputs	**inputs** object	算法输入通道信息。
outputs	**outputs** object	算法输出通道信息。
engine	**engine** object	异构作业所依赖的引擎。

表 8-91 inputs

参数	参数类型	描述
name	String	数据输入通道名称。
local_dir	String	数据输入输出通道映射的容器本地路径。
remote	**remote** object	数据实际输入信息，异构作业只支持OBS。

表 8-92 remote

参数	参数类型	描述
obs	**obs** object	数据输入输出信息为OBS方式。

表 8-93 obs

参数	参数类型	描述
obs_url	String	训练作业需要的数据集OBS路径URL。如：“/usr/data/”。

表 8-94 outputs

参数	参数类型	描述
name	String	数据输出通道名称。
local_dir	String	数据输出通道映射的容器本地路径。
remote	**remote** object	数据实际输出信息。
mode	String	数据传输模式，默认为“upload_periodically”。
period	String	数据传输周期，默认为30s。

表 8-95 remote

参数	参数类型	描述
obs	**obs** object	数据实际输出到OBS。

表 8-96 obs

参数	参数类型	描述
obs_url	String	数据实际输出到OBS的路径。

表 8-97 engine

参数	参数类型	描述
engine_id	String	异构作业引擎规格的ID。如“caffe-1.0.0-python2.7”。
engine_name	String	异构作业引擎规格的名称。如“Caffe”。
engine_version	String	异构作业引擎规格的版本。
v1_compatible	Boolean	是否为v1兼容模式。
run_user	String	引擎默认启动用户uid。

表 8-98 FlavorResponse

参数	参数类型	描述
flavor_id	String	资源规格的ID。

参数	参数类型	描述
flavor_name	String	资源规格的名称。
max_num	Integer	资源规格的最大节点数。
flavor_type	String	资源规格的类型。可选值如下： • CPU； • GPU； • Ascend。
billing	**billing** object	资源规格计费信息。
flavor_info	**flavor_info** object	资源规格详细信息。
attributes	Map<String,String>	其他规格属性。

表 8-99 billing

参数	参数类型	描述
code	String	计费码。
unit_num	Integer	计费卡数。

表 8-100 flavor_info

参数	参数类型	描述
max_num	Integer	可以选择的最大节点数量（max_num，为1代表不支持分布式）。
cpu	**cpu** object	cpu规格信息。
gpu	**gpu** object	gpu规格信息。
npu	**npu** object	Ascend规格信息。
memory	**memory** object	内存信息。

表 8-101 cpu

参数	参数类型	描述
arch	String	cpu架构。
core_num	Integer	核数。

表 8-102 gpu

参数	参数类型	描述
unit_num	Integer	gpu卡数。
product_name	String	产品名。
memory	String	内存。

表 8-103 npu

参数	参数类型	描述
unit_num	String	npu卡数。
product_name	String	产品名。
memory	String	内存。

表 8-104 memory

参数	参数类型	描述
size	Integer	内存大小。
unit	String	内存单元数。

表 8-105 spec

参数	参数类型	描述
resource	**Resource** object	训练作业资源规格信息。flavor_id和pool_id+[flavor_id]方式二选一。
volumes	Array of objects	训练作业挂载卷信息。
log_export_path	**log_export_path** object	训练作业日志输出信息。

表 8-106 Resource

参数	参数类型	描述
policy	String	训练作业资源规格模式，可选值如下： “regular”、“economic”、“turbo”。
flavor_id	String	训练作业选择的资源规格ID。

参数	参数类型	描述
flavor_name	String	使用flavor_id时，由ModelArts返回的只读规格名称。
node_count	Integer	训练作业选择的资源副本数。 最小值：**1**
pool_id	String	训练作业选择的资源池ID。
flavor_detail	**flavor_detail** object	训练作业、算法的规格信息。

表 8-107 flavor_detail

参数	参数类型	描述
flavor_type	String	资源规格的类型。可选值如下： • CPU； • GPU； • Ascend。
billing	**billing** object	资源规格计费信息。
flavor_info	**flavor_info** object	资源规格详细信息。

表 8-108 billing

参数	参数类型	描述
code	String	计费码。
unit_num	Integer	计费卡数。

表 8-109 flavor_info

参数	参数类型	描述
max_num	Integer	可以选择的最大节点数量（max_num，为1代表不支持分布式）。
cpu	**cpu** object	cpu规格信息。
gpu	**gpu** object	gpu规格信息。
npu	**npu** object	Ascend规格信息。

参数	参数类型	描述
memory	**memory** object	内存信息。
disk	**disk** object	磁盘信息。

表 8-110 cpu

参数	参数类型	描述
arch	String	cpu架构。
core_num	Integer	核数。

表 8-111 gpu

参数	参数类型	描述
unit_num	Integer	gpu卡数。
product_name	String	产品名。
memory	String	内存。

表 8-112 npu

参数	参数类型	描述
unit_num	String	npu卡数。
product_name	String	产品名。
memory	String	内存。

表 8-113 memory

参数	参数类型	描述
size	Integer	内存大小。
unit	String	内存单元数。

表 8-114 disk

参数	参数类型	描述
size	String	磁盘大小。
unit	String	磁盘大小单位，一般为GB。

表 8-115 volumes

参数	参数类型	描述
nfs	**nfs** object	nfs方式的挂载卷。

表 8-116 nfs

参数	参数类型	描述
nfs_server_path	String	nfs服务端路径。
local_path	String	挂载到训练容器中的路径。
read_only	Boolean	nfs挂载卷在容器中是否只读。

表 8-117 log_export_path

参数	参数类型	描述
obs_url	String	训练作业日志保存的OBS地址。
host_path	String	训练作业日志保存的宿主机的路径。

表 8-118 调用训练接口失败响应参数

参数	类型	描述
error_msg	String	调用失败时的错误信息，调用成功时无此字段。
error_code	String	调用失败时的错误码，具体请参见**错误码**，调用成功时无此字段。
error_solution	String	调用失败时的提示解决信息，调用成功时无此字段。

8.1.5 更新训练作业描述

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

- 方式一：**根据指定的job_id更新。**

```
from modelarts.session import Session
from modelarts.estimatorV2 import Estimator
session = Session()
estimator = Estimator(session=session, job_id="your job id")
estimator.update_job_configs(description="update job description")
```

- 方式二：**根据创建训练作业生成的训练作业对象更新。**

```
job_instance.update_job_configs(description="update job description fourth")
```

参数说明

表 8-119 Estimator 请求参数说明

参数	是否必选	参数类型	描述
session	是	Object	会话对象，初始化方法请参考**Session鉴权**。
job_id	是	String	训练作业的id，可通过**创建训练作业**生成的训练作业对象查询，如"job_instance.job_id"，或从**查询训练作业列表**的响应中获得。

表 8-120 update_job_configs 请求参数说明

参数	是否必选	参数类型	描述
description	是	String	需要更改的训练作业的描述信息。

无成功响应参数

表 8-121 调用训练接口失败响应参数

参数	类型	描述
error_msg	String	调用失败时的错误信息，调用成功时无此字段。
error_code	String	调用失败时的错误码，具体请参见**错误码**，调用成功时无此字段。
error_solution	String	调用失败时的提示解决信息，调用成功时无此字段。

8.1.6 删除训练作业

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

- 方式一：**根据指定的job_id删除。**

```
from modelarts.session import Session
from modelarts.estimatorV2 import Estimator
session = Session()
Estimator.delete_job_by_id(session=session, job_id="your job id")
```

- 方式二：**根据创建训练作业生成的训练作业对象删除。**

```
job_instance.delete_job()
```

参数说明

表 8-122 delete_job_by_id 请求参数说明

参数	是否必选	参数类型	描述
session	是	Object	会话对象，初始化方法请参考**Session鉴权**。
job_id	是	String	训练作业的id，可通过**创建训练作业**生成的训练作业对象查询，如"job_instance.job_id"，或从**查询训练作业列表**的响应中获得。

无成功响应参数。

表 8-123 调用训练接口失败响应参数

参数	类型	描述
error_msg	String	调用失败时的错误信息，调用成功时无此字段。
error_code	String	调用失败时的错误码，具体请参见**错误码**，调用成功时无此字段。
error_solution	String	调用失败时的提示解决信息，调用成功时无此字段。

8.1.7 终止训练作业

终止训练作业，只可终止创建中、等待中、运行中的作业。

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

- 方式一：**根据指定的job_id终止。**

```
from modelarts.session import Session
from modelarts.estimatorV2 import Estimator
session = Session()
info = Estimator.control_job_by_id(session=session, job_id="your job id")
print(info)
```

- 方式二：**根据创建训练作业生成的训练作业对象终止。**

```
job_instance.control_job()
```

参数说明

表 8-124 control_job_by_id 请求参数说明

参数	是否必选	参数类型	描述
session	是	Object	会话对象，初始化方法请参考**Session鉴权**。
job_id	是	String	训练作业的id，可通过**创建训练作业**生成的训练作业对象查询，如"job_instance.job_id"，或从**查询训练作业列表**的响应中获得。

表 8-125 返回参数说明

参数	参数类型	描述
kind	String	训练作业类型。默认使用job。 枚举值： • **job：训练作业** • **hetero_job：异构作业** • **autosearch_job：自动搜索作业** • **mrs_job：mrs作业** • **edge_job：边缘作业**
metadata	**JobMetadata** object	训练作业元信息。
status	**Status** object	训练作业状态信息。创建作业无需填写。

参数	参数类型	描述
algorithm	JobAlgorithmResponse object	训练作业算法。目前支持三种形式： • id只取算法的id； • subscription_id+item_version_id取算法的订阅id和版本id； • code_dir+boot_file取训练作业的代码目录和启动文件。
tasks	Array of TaskResponse objects	异构训练作业的任务列表。
spec	spec object	训练作业规格参数。

表 8-126 JobMetadata

参数	参数类型	描述
id	String	训练作业ID，创建成功后由ModelArts生成返回，无需填写。
name	String	训练作业名称。限制为1-64位只含数字、字母、下划线和中划线的名称。
workspace_id	String	指定作业所处的工作空间，默认值为“0”。
description	String	对训练作业的描述，默认为“NULL”，字符串的长度限制为[0, 256]。
create_time	Long	训练作业创建时间戳，单位为毫秒，创建成功后由ModelArts生成返回，无需填写。
user_name	String	训练作业创建用户的用户名，创建成功后由ModelArts生成返回，无需填写。
annotations	Map<String,String>	训练作业申明模板，异构作业job_template字段默认为Template RL，其余默认为Template DL。

表 8-127 Status

参数	参数类型	描述
phase	String	训练作业一级状态，状态值稳定不变，可选值如下：“Creating”、“Pending”、“Running”、“Failed”、“Completed”、“Terminating”、“Terminated”、“Abnormal”。

参数	参数类型	描述
secondary_phase	String	训练作业二级状态，状态值不稳定，可选值如下：“Creating”、“Queuing”、“Running”、“Failed”、“Completed”、“Terminating”、“Terminated”、“CreateFailed”、“TerminatedFailed”、“Unknown”、“Lost”。
duration	Long	训练作业运行时长，单位为毫秒。
node_count_metrics	Array<Array<Integer>>	训练作业运行时节点数变化指标。
tasks	Array of strings	训练作业子任务名称。
start_time	String	训练作业开始时间，格式为时间戳。
task_statuses	Array of objects	训练在子任务状态信息。

表 8-128 task_statuses

参数	参数类型	描述
task	String	训练作业子任务名称。
exit_code	Integer	训练作业子任务退出码。
message	String	训练作业子任务错误消息。

表 8-129 JobAlgorithmResponse

参数	参数类型	描述
id	String	算法管理的算法id。 枚举值： • **id：只取算法的id；** • **subscription_id+item_version_id：取算法的订阅id和版本id；** • **code_dir+boot_file：取训练作业的代码目录和启动文件。**
name	String	算法名称。
subscription_id	String	订阅算法的订阅ID。
item_version_id	String	订阅算法的版本。

参数	参数类型	描述
code_dir	String	训练作业的代码目录。如：“/usr/app/”。应与boot_file一同出现，如果填入id或subscription_id+item_version_id则无需填写。
boot_file	String	训练作业的代码启动文件，需要在代码目录下。如：“/usr/app/boot.py”。应与code_dir一同出现，如果填入id或subscription_id+item_version_id则无需填写。
autosearch_config_path	String	自动化搜索作业的yaml配置路径，需要提供一个OBS路径。
autosearch_framework_path	String	自动化搜索作业的框架代码目录，需要提供一个OBS路径。
command	String	自定义镜像训练作业的自定义镜像的容器的启动命令。可填code_dir。
parameters	Array of **Parameter** objects	训练作业的运行参数。
policies	**policies** object	作业支持的策略。
inputs	Array of **Input** objects	训练作业的数据输入。
outputs	Array of **Output** objects	训练作业的结果输出。
engine	**engine** object	训练作业的引擎。使用算法管理的算法id或订阅算法subscription_id+item_version_id创建作业时，无需填写。
environments	Map<String,String>	训练作业的环境变量。格式："key":"value"，无需填写。

表 8-130 Parameter

参数	参数类型	描述
name	String	参数名称。
value	String	参数值。
description	String	参数描述信息。
constraint	**constraint** object	参数属性。

参数	参数类型	描述
i18n_description	**i18n_description** object	国际化描述。

表 8-131 constraint

参数	参数类型	描述
type	String	参数种类。
editable	Boolean	是否可编辑。
required	Boolean	是否必须。
sensitive	Boolean	是否敏感。
valid_type	String	参数有效种类。
valid_range	Array of strings	参数有效范围。

表 8-132 i18n_description

参数	参数类型	描述
language	String	国际语种。
description	String	对应描述。

表 8-133 policies

参数	参数类型	描述
auto_search	**auto_search** object	超参搜索配置。

表 8-134 auto_search

参数	参数类型	描述
skip_search_params	String	需要排除的超参组合。
reward_attrs	Array of objects	搜索指标列表。

参数	参数类型	描述
search_params	Array of objects	搜索参数。
algo_configs	Array of objects	搜索算法配置。

表 8-135 reward_attrs

参数	参数类型	描述
name	String	指标名称。
mode	String	搜索方向。 ● max：指定时，表示指标值越大越好； ● min：指定时，表示指标值越小越好。
regex	String	指标正则表达式。

表 8-136 search_params

参数	参数类型	描述
name	String	超参名称。
param_type	String	参数类型。 ● continuous：指定时，表示参数类型为连续值； ● discreate：指定时，表示参数类型为离散值。
lower_bound	String	超参下界。
upper_bound	String	超参上界。
discrete_points_num	String	连续型超参离散化取值个数。
discrete_values	Array of strings	离散型超参的取值列表。

表 8-137 algo_configs

参数	参数类型	描述
name	String	搜索算法名称。

参数	参数类型	描述
params	Array of **AutoSearchAlgoConfigParameter** objects	搜索算法参数。

表 8-138 AutoSearchAlgoConfigParameter

参数	参数类型	描述
key	String	参数键。
value	String	参数值。
type	String	参数种类。

表 8-139 Input

参数	参数类型	描述
name	String	数据输入通道名称。
description	String	数据输入通道描述信息。
local_dir	String	数据输入通道映射的容器本地路径。
remote	**InputDataInfo** object	数据实际输入信息。
remote_constraint	Array of objects	数据输入约束。

表 8-140 InputDataInfo

参数	参数类型	描述
dataset	**dataset** object	数据输入信息为数据集。
obs	**obs** object	数据输入输出信息为OBS方式。

表 8-141 dataset

参数	参数类型	描述
id	String	训练作业的数据集ID。

参数	参数类型	描述
version_id	String	训练作业的数据集版本ID。
obs_url	String	训练作业需要的数据集OBS路径URL，ModelArts会通过数据集ID和数据集版本ID自动解析生成。如：“/usr/data/”。

表 8-142 obs

参数	参数类型	描述
obs_url	String	训练作业需要的数据集OBS路径URL。如：“/usr/data/”。

表 8-143 remote_constraint

参数	参数类型	描述
data_type	String	数据输入类型，包括数据存储位置、数据集两种方式。
attributes	String	数据输入为数据集时的相关属性。 枚举值： • data_format：数据格式。 • data_segmentation：数据切分方式。 • dataset_type：标注类型。

表 8-144 Output

参数	参数类型	描述
name	String	数据输出通道名称。
description	String	数据输出通道描述信息。
local_dir	String	数据输出通道映射的容器本地路径。
remote	**remote** object	数据实际输出信息。

表 8-145 remote

参数	参数类型	描述
obs	**obs** object	数据实际输出到OBS。

表 8-146 obs

参数	参数类型	描述
obs_url	String	数据实际输出到OBS的路径。

表 8-147 engine

参数	参数类型	描述
engine_id	String	训练作业选择的引擎规格ID。engine_id、engine_name + engine_version和image_url方式三选一。
engine_name	String	训练作业选择的引擎名称。如果填入engine_id，则无需填写。
engine_version	String	训练作业选择的引擎版本名称。如果填入engine_id，则无需填写。
image_url	String	训练作业选择的自定义镜像地址。

表 8-148 TaskResponse

参数	参数类型	描述
role	String	异构训练作业的任务角色。 枚举值： • learner（支持GPU\CPU规格） • worker（支持CPU规格）
algorithm	**algorithm** object	算法配置。
task_resource	**FlavorResponse** object	训练作业、算法的规格信息。

表 8-149 algorithm

参数	参数类型	描述
code_dir	String	算法启动文件所在目录绝对路径。
boot_file	String	算法启动文件绝对路径。
inputs	**inputs** object	算法输入通道信息。
outputs	**outputs** object	算法输出通道信息。

参数	参数类型	描述
engine	**engine** object	异构作业所依赖的引擎。

表 8-150 inputs

参数	参数类型	描述
name	String	数据输入通道名称。
local_dir	String	数据输入输出通道映射的容器本地路径。
remote	**remote** object	数据实际输入信息，异构作业只支持OBS。

表 8-151 remote

参数	参数类型	描述
obs	**obs** object	数据输入输出信息为OBS方式。

表 8-152 obs

参数	参数类型	描述
obs_url	String	训练作业需要的数据集OBS路径URL。如：“/usr/data/”。

表 8-153 outputs

参数	参数类型	描述
name	String	数据输出通道名称。
local_dir	String	数据输出通道映射的容器本地路径。
remote	**remote** object	数据实际输出信息。
mode	String	数据传输模式，默认为“upload_periodically”。
period	String	数据传输周期，默认为30s。

表 8-154 remote

参数	参数类型	描述
obs	**obs** object	数据实际输出到OBS。

表 8-155 obs

参数	参数类型	描述
obs_url	String	数据实际输出到OBS的路径

表 8-156 engine

参数	参数类型	描述
engine_id	String	异构作业引擎规格的ID。如“caffe-1.0.0-python2.7”。
engine_name	String	异构作业引擎规格的名称。如“Caffe”。
engine_version	String	异构作业引擎规格的版本。
v1_compatible	Boolean	是否为v1兼容模式。
run_user	String	引擎默认启动用户uid。

表 8-157 FlavorResponse

参数	参数类型	描述
flavor_id	String	资源规格的ID。
flavor_name	String	资源规格的名称。
max_num	Integer	资源规格的最大节点数。
flavor_type	String	资源规格的类型。可选值如下： • CPU； • GPU； • Ascend。
billing	**billing** object	资源规格计费信息。
flavor_info	**flavor_info** object	资源规格详细信息。

参数	参数类型	描述
attributes	Map<String,String>	其他规格属性。

表 8-158 billing

参数	参数类型	描述
code	String	计费码。
unit_num	Integer	计费卡数。

表 8-159 flavor_info

参数	参数类型	描述
max_num	Integer	可以选择的最大节点数量（max_num，为1代表不支持分布式）。
cpu	**cpu** object	cpu规格信息。
gpu	**gpu** object	gpu规格信息。
npu	**npu** object	Ascend规格信息。
memory	**memory** object	内存信息。

表 8-160 cpu

参数	参数类型	描述
arch	String	cpu架构。
core_num	Integer	核数。

表 8-161 gpu

参数	参数类型	描述
unit_num	Integer	gpu卡数。
product_name	String	产品名。
memory	String	内存。

表 8-162 npu

参数	参数类型	描述
unit_num	String	npu卡数。
product_name	String	产品名。
memory	String	内存。

表 8-163 memory

参数	参数类型	描述
size	Integer	内存大小。
unit	String	内存单元数。

表 8-164 spec

参数	参数类型	描述
resource	**Resource** object	训练作业资源规格信息。flavor_id和pool_id+[flavor_id]方式二选一。
volumes	Array of objects	训练作业挂载卷信息。
log_export_path	**log_export_path** object	训练作业日志输出信息。

表 8-165 Resource

参数	参数类型	描述
policy	String	训练作业资源规格模式，可选值如下： “regular”、“economic”、“turbo”。
flavor_id	String	训练作业选择的资源规格ID。
flavor_name	String	使用flavor_id时，由ModelArts返回的只读规格名称。
node_count	Integer	训练作业选择的资源副本数。 最小值：**1**
pool_id	String	训练作业选择的资源池ID。
flavor_detail	**flavor_detail** object	训练作业、算法的规格信息。

表 8-166 flavor_detail

参数	参数类型	描述
flavor_type	String	资源规格的类型。可选值如下： • CPU； • GPU； • Ascend。
billing	**billing** object	资源规格计费信息。
flavor_info	**flavor_info** object	资源规格详细信息。

表 8-167 billing

参数	参数类型	描述
code	String	计费码。
unit_num	Integer	计费卡数。

表 8-168 flavor_info

参数	参数类型	描述
max_num	Integer	可以选择的最大节点数量（max_num，为1代表不支持分布式）。
cpu	**cpu** object	cpu规格信息。
gpu	**gpu** object	gpu规格信息。
npu	**npu** object	Ascend规格信息。
memory	**memory** object	内存信息。
disk	**disk** object	磁盘信息。

表 8-169 cpu

参数	参数类型	描述
arch	String	cpu架构。
core_num	Integer	核数。

表 8-170 gpu

参数	参数类型	描述
unit_num	Integer	gpu卡数。
product_name	String	产品名。
memory	String	内存。

表 8-171 npu

参数	参数类型	描述
unit_num	String	npu卡数。
product_name	String	产品名。
memory	String	内存。

表 8-172 memory

参数	参数类型	描述
size	Integer	内存大小。
unit	String	内存单元数。

表 8-173 disk

参数	参数类型	描述
size	String	磁盘大小。
unit	String	磁盘大小单位，一般为GB。

表 8-174 volumes

参数	参数类型	描述
nfs	**nfs** object	nfs方式的挂载卷。

表 8-175 nfs

参数	参数类型	描述
nfs_server_path	String	nfs服务端路径。
local_path	String	挂载到训练容器中的路径。
read_only	Boolean	nfs挂载卷在容器中是否只读。

表 8-176 log_export_path

参数	参数类型	描述
obs_url	String	训练作业日志保存的OBS地址。
host_path	String	训练作业日志保存的宿主机的路径。

表 8-177 调用训练接口失败响应参数

参数	类型	描述
error_msg	String	调用失败时的错误信息，调用成功时无此字段。
error_code	String	调用失败时的错误码，具体请参见**错误码**，调用成功时无此字段。
error_solution	String	调用失败时的提示解决信息，调用成功时无此字段。

8.1.8 查询训练日志

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

- 方式一：根据指定的job_id查询。

```
from modelarts.session import Session
from modelarts.estimatorV2 import Estimator
session = Session()
estimator = Estimator(session=session, job_id="your job id")
info = estimator.get_job_log()
print(info)
```

- 方式二：根据**创建训练作业**生成的训练作业对象查询。

```
log = job_instance.get_job_log(task_id="worker-0")
print(log)
```


参数说明

表 8-178 Estimator 初始化参数说明

参数	是否必选	参数类型	描述
session	是	Object	会话对象，初始化方法请参考**Session鉴权**。
job_id	是	String	训练作业的id，可通过**创建训练作业**生成的训练作业对象查询，如"job_instance.job_id"，或从**查询训练作业列表**的响应中获得。

表 8-179 get_job_log 请求参数说明

参数	是否必选	参数类型	描述
task_id	否	String	要查看哪个工作节点的日志，默认值为"worker-0"；如果在**创建训练作业**时参数train_instance_count选择了2，则可选值为"worker-0","worker-1"，依次类推。

表 8-180 返回参数

参数	参数类型	描述
content	String	日志内容。 • 如果日志大小没有超过上限（n兆），则返回全部内容； • 如果日志超过了上限（n兆），则返回最新的n兆的日志。
current_size	Integer	当前返回的日志大小（单位：字节）。最大为5兆。
full_size	Integer	完整的日志大小（单位：字节）。

表 8-181 调用训练接口失败响应参数

参数	类型	描述
error_msg	String	调用失败时的错误信息，调用成功时无此字段。

参数	类型	描述
error_code	String	调用失败时的错误码，具体请参见**错误码**，调用成功时无此字段。
error_solution	String	调用失败时的提示解决信息，调用成功时无此字段。

8.1.9 查询训练作业的运行指标

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

- 方式一：根据指定的job_id查询。

```
from modelarts.session import Session
from modelarts.estimatorV2 import Estimator
session = Session()
estimator = Estimator(session=session, job_id="your job id")
info = estimator.get_job_metrics()
print(info)
```

- 方式二：根据**创建训练作业**生成的训练作业对象查询。

```
info = job_instance.get_job_metrics(task_id="worker-0")
print(info)
```

参数说明

表 8-182 Estimator 初始化参数说明

参数	是否必选	参数类型	描述
session	是	Object	会话对象，初始化方法请参考**Session鉴权**。
job_id	是	String	训练作业的id，可通过**创建训练作业**生成的训练作业对象查询，如"job_instance.job_id"，或从**查询训练作业列表**的响应中获得。

表 8-183 get_job_log 请求参数说明

参数	是否必选	参数类型	描述
task_id	否	String	要查看哪个工作节点的日志，默认值为"worker-0"；如果在**创建训练作业**时参数train_instance_count选择了2，则可选值为"worker-0","worker-1"，依次类推。

表 8-184 返回参数说明

参数	参数类型	描述
metrics	Array of objects	运行指标。

表 8-185 metrics

参数	参数类型	描述
metric	String	运行指标，可选值如下： cpuUsage（CPU使用率）、memUsage（物理内存使用率）、gpuUtil（GPU使用率）、gpuMemUsage（显存使用率）、npuUtil（NPU使用率）、npuMemUsage（NPU显存使用率）。
value	Array of numbers	运行指标对应数值，1min统计一个平均值。

表 8-186 调用训练接口失败响应参数

参数	类型	描述
error_msg	String	调用失败时的错误信息，调用成功时无此字段。
error_code	String	调用失败时的错误码，具体请参见**错误码**，调用成功时无此字段。
error_solution	String	调用失败时的提示解决信息，调用成功时无此字段。

8.2 资源和引擎规格接口

8.2.1 查询资源规格列表

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

```
from modelarts.session import Session
from modelarts.estimatorV2 import Estimator
session = Session()
info = Estimator.get_train_instance_types(session=session)
print(info)
```


参数说明

表 8-187 get_train_instance_types 参数说明

参数	是否必选	参数类型	描述
session	是	Object	会话对象，初始化方法请参考**Session鉴权**。

表 8-188 成功响应参数说明

参数类型	描述
List	资源规格参数列表。

表 8-189 调用训练接口失败响应参数

参数	类型	描述
error_msg	String	调用失败时的错误信息，调用成功时无此字段。
error_code	String	调用失败时的错误码，具体请参见**错误码**，调用成功时无此字段。
error_solution	String	调用失败时的提示解决信息，调用成功时无此字段。

8.2.2 查询引擎规格列表

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

```
from modelarts.session import Session
from modelarts.estimatorV2 import Estimator
session = Session()
info = Estimator.get_framework_list(session=session)
print(info)
```

参数说明

表 8-190 get_train_instance_types 参数说明

参数	是否必选	参数类型	描述
session	是	Object	会话对象，初始化方法请参考**Session鉴权**。

表 8-191 get_framework_list 成功响应参数说明

参数类型	描述
List	引擎规格参数列表，请参见**表3**。

表 8-192 framework_list 参数说明

参数	参数类型	描述
framework_type	String	引擎类型。
framework_version	String	引擎版本。

表 8-193 调用训练接口失败响应参数

参数	类型	描述
error_msg	String	调用失败时的错误信息，调用成功时无此字段。
error_code	String	调用失败时的错误码，具体请参见**错误码**，调用成功时无此字段。
error_solution	String	调用失败时的提示解决信息，调用成功时无此字段。

9 模型管理

9.1 模型调试

训练完成后，可先在开发环境Notebook中创建本地模型，在开发环境Notebook调试完成后再部署到推理服务上。

说明

只支持使用ModelArts Notebook部署本地服务。

示例代码

在ModelArts Notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

步骤1 将自定义的推理文件和模型配置文件保存在训练生成的模型文件目录下。如训练生成的模型保存在“/home/ma-user/work/tensorflow_mlp_mnist_local_mode/train/model/”中，则推理文件“customize_service.py”和模型配置文件“config.json”也保存在该目录中。

步骤2 创建模型运行的conda虚拟环境。

```
from modelarts.environment import Environment
from modelarts.environment.conda_env import CondaDependencies

env = Environment("tensorflow_mlp_mnist")
cd = CondaDependencies.create(pip_packages=["tensorflow==1.13.1", "Pillow>=8.0.1"],
                              conda_packages=["python=3.6.2"])
env.conda = cd
```

步骤3 创建本地模型。

```
from modelarts.session import Session
from modelarts.model import Model

session = Session()
src_local_path = "/home/ma-user/work/tensorflow_mlp_mnist_local_mode/train/"
model = Model(session,
              publish=False,
              source_location_type="LOCAL_SOURCE",         # 模型文件位置类型，
              source_location=src_local_path + 'model',    # 模型文件位置
              environment=env,
              model_version='1.0.1',
              model_type='TensorFlow',                     # 模型使用的AI框架
```



```
        model_algorithm="image_classification",
        model_name="tensorflow_mlp_mnist_local_infer")
```

本地模型创建好后，可部署为本地服务。

步骤4 本地模型创建完，可以调用接口发布模型。

```
model.publish_model(obs_location=obs_location)
```

指定参数“obs_location”后，会将本地的模型文件上传到该目录下。参数可省略，示例如下：

```
model.publish_model()
```

此时模型文件会上传到默认OBS桶以当前时间戳结尾的目录中。该目录会在命令执行后打印出来，示例如下：

```
Successfully upload file /home/ma-user/work/tensorflow_mlp_mnist_local_mode/train/model to OBS modelarts-cn-north-4-08aae033/model-0107-224502
```

----结束

参数说明

表 9-1 创建模型场景参数说明

参数	是否必选	参数类型	描述
session	是	Object	会话对象，初始化方法见**Session鉴权**。
model_name	否	String	模型名称，名称只能字母，中文开头，为字母、数字、下划线、中文或者中划线组成的合法字符，支持1-64个字符。如果未输入该参数，系统会自动生成模型name。
model_version	是	String	模型版本，格式需为“数值.数值.数值”，其中数值为1-2位正整数。版本不可以出现以0开头的版本号形式，如“01.01.01”等。
publish	否	Bool	是否发布模型。可选值： ● True: 发布模型。（默认值） ● False: 不发布模型，创建本地模型，可用来调试相关代码
source_location_type	否	String	模型位置类型。可选值： ● OBS_SOURCE：source_location为OBS路径。（默认值） ● LOCAL_SOURCE：source_location为本地路径。

参数	是否必选	参数类型	描述
source_location	是	String	模型文件所在路径，指定到模型文件的父目录。 ● 当source_location_type为OBS_SOURCE时，模型文件所在路径为OBS路径，格式为“/obs_bucketname/.../model_file_parent_dir/”。 ● 当source_location_type为LOCAL_SOURCE时，模型文件所在路径为本地路径，格式为“/local_path/.../model_file_parent_dir/”。
environment	否	Environment实例	描述模型正常运行需要的环境，如使用的python版本、tensorflow版本等。请参见表9-2
source_job_id	否	String	来源训练作业的ID，模型是从训练作业产生的可填写，用于溯源；如模型是从第三方元模型导入，则为空，默认值为空。
source_job_version	否	String	来源训练作业的版本，模型是从训练作业产生的可填写，用于溯源；如模型是从第三方元模型导入，则为空，默认值为空。
source_type	否	String	模型来源的类型，当前仅可取值auto，用于区分通过自动学习部署过来的模型（不提供模型下载功能）；用户通过训练作业部署的模型不设置此值。默认值为空。
model_type	是	String	模型类型，取值为：TensorFlow/MXNet/Spark_MLlib/Scikit_Learn/XGBoost/MindSpore/Image/PyTorch。
model_algorithm	否	String	模型算法，表示模型的算法实现类型，如果已在模型配置文件中配置，则可不填。如：predict_analysis、object_detection 、image_classification。
description	否	String	模型描述信息，不超过100个字符，且不能包含特殊字符!<>=&’”。
execution_code	否	String	存放执行脚本的OBS路径。推理脚本必须放于模型所在路径（请参见“source_location”参数）的model目录下，名称固定为：“customize_service.py”。
input_params	否	params结构数组	模型推理输入参数列表，默认为空。如果已在模型配置文件中配置apis信息时，则可不填，后台自动从配置文件的apis字段中读取输入参数信息。
output_params	否	params结构数组	模型推理输出参数列表，默认为空。如果已在模型配置文件中配置apis信息时，则可不填，后台自动从配置文件的apis字段中读取输出参数信息。

参数	是否必选	参数类型	描述
dependencies	否	dependency结构数组	运行代码及模型需安装的依赖包，默认为空。如果已在模型配置文件中配置dependencies信息时，则可不填，后台自动从配置文件的dependencies字段中读取需要安装的依赖包。
apis	否	String	模型提供的推理接口列表，默认为空。如果已在模型配置文件中配置apis信息时，则可不填，后台自动从配置文件中的apis字段读取所配置的推理接口信息。

表 9-2 Environment 参数说明

参数	是否必选	类型	说明
name	是	String	环境名称。
conda	否	CondaDependencies	conda环境，具体请参见表9-3。

表 9-3 CondaDependencies 参数说明

参数	是否必选	类型	说明
channels	否	List	python包的下载源。
pip_packages	否	List	conda虚拟环境需要使用的python包，如tensorflow，pillow等。
conda_packages	否	List	conda虚拟环境需要使用的conda包，如指定python版本。

表 9-4 params 结构

参数	是否必选	参数类型	描述
url	是	String	模型推理接口的请求路径。
param_name	是	String	参数名，不超过64个字符。
param_type	是	String	JSON Schema基本参数类型，有string、object、array、boolean、number、integer。

参数	是否必选	参数类型	描述
min	否	Double	当param_type为int或float时，可选填，默认为空。
max	否	Double	当param_type为int或float时，可选填，默认为空。
param_desc	否	String	参数描述，不超过100个字符，默认为空。

表 9-5 dependency 结构

参数	是否必选	参数类型	描述
installer	是	String	安装方式，当前只支持“pip”。
packages	是	package结构数组	依赖包集合。

表 9-6 package 结构

参数	是否必选	参数类型	描述
package_name	是	String	依赖包名称。
package_version	否	String	依赖包版本。
restraint	否	String	版本过滤条件，当且仅当package_version存在时必填。取值为： ● EXACT：等于给定版本 ● ATLEAST：不小于给定版本 ● ATMOST：不大于给定版本

表 9-7 创建模型返回参数说明

参数	参数类型	描述
model	Model对象	模型对象，可以调用本章节模型管理的所有接口。

9.2 导入模型

导入模型功能包括：

- 初始化已存在的模型，根据模型ID生成模型对象。
- 创建模型。模型对象的属性，请参见**查询模型详情**。

示例模型文件

以**PyTorch**为例，编写模型文件。PyTorch模型包结构可参考**模型包规范介绍**。

```
OBS桶/目录名
├── resnet
│   ├── model 必选：固定子目录名称，用于放置模型相关文件
│   │   ├──<<自定义Python包>> 可选：用户自有的Python包，在模型推理代码中可以直接引用
│   │   ├──mnist_mlp.pt 必选，pytorch模型保存文件，保存为“state_dict”，存有权重变量等信息。
│   │   ├──config.json 必选：模型配置文件，文件名称固定为config.json，只允许放置一个
│   │   ├──customize_service.py 必选：模型推理代码，文件名称固定为customize_service.py，只允许放置一个，customize_service.py依赖的文件可以直接放model目录下
```

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

```
from modelarts.session import Session
from modelarts.model import Model
from modelarts.config.model_config import ServiceConfig, Params, Dependencies, Packages

session = Session()
```

- **方式1：初始化已存在模型**

  ```
  model_instance = Model(session, model_id="your_model_id")
  ```

- **方式2：创建模型**
  - 基于预置镜像和OBS路径创建模型

    ```
    model_location = "/your_obs_bucket/model_path"            # 修改为真实的模型文件OBS路径
    execution_code = "/your_obs_bucket/model_path/customize_service.py"
    runtime = "python3.7"

    model_instance = Model(
                        session,
                        model_name="input_model_name",    # （可选）模型名称
                        model_version="1.0.0",            # （可选）模型版本
                        source_location=model_location,   # 模型文件所在的OBS路径，如："/your_obs_bucket/model_path"
                        model_type="PyTorch",             # 模型类型
                        execution_code=execution_code,    # （可选）存放执行脚本的OBS路径，如："/your_obs_bucket/model_path/customize_service.py"
                        runtime=runtime                   # （可选）支持的运行环境
                       )
    ```

    说明

    dependencies会覆盖样例中config.json的相关内容，上述无需使用dependencies，dependencies格式样例可参考下方格式定义。

    - **dependencies参数组的定义格式**

      SDK提供了Dependencies类对其定义，dependencies为list，list中的元组对象是Dependencies。

      定义代码如下：

```
dependencies = []
dependency1 = Dependencies(
    installer="pip",                    # 安装方式，目前支持pip
    packages=packages                   # 依赖包集合, 定义格式参考下文关于packages的定义
)
dependencies.append(dependency1)
```

- **package参数组的定义格式**

  SDK提供了Packages类对其定义，packages为list，list中的元组对象是Packages。

  定义代码如下：

```
packages = []
package1 = Packages(
    package_name="package_name",            # 包名
    package_version="version",              # 包版本号
    restraint="EXACT"
)
packages.append(package1)
```

  说明

  dependencies参数组的创建样例：

```
dependencies = []
packages = [{
    "package_name": "numpy",
    "package_version": "1.15.0",
    "restraint": "EXACT"
    }, {
        "package_name": "h5py",
        "package_version": "2.8.0",
        "restraint": "EXACT"
    }]
dependency = Dependencies(installer="pip", packages=packages)
dependencies.append(dependency)
```

– 基于自定义镜像创建模型

  适用于推理服务的脚本已经内置在自定义镜像中，镜像启动时会自动拉起服务的场景。

```
from modelarts.session import Session
from modelarts.model import Model

session = Session()
image_path = "custom_image_path"       # 自定义镜像的swr路径
model_instance = Model(
                    session,
                    model_name="your_model_name",             # 模型名称
                    model_version="0.1.0",                    # 模型版本
                    source_location="image_path",             # 模型文件路径
                    model_type="Image"                        # 模型类型
                 )
```

参数说明

表 9-8 初始化模型场景参数说明

参数	是否必选	参数类型	描述
session	是	Object	会话对象，初始化方法参考Session鉴权。
model_id	是	String	模型ID。

表 9-9 创建模型场景参数说明

参数	是否必选	参数类型	描述
session	是	Object	会话对象，初始化方法见**Session鉴权**。
model_name	否	String	模型名称，名称只能字母，中文开头，为字母、数字、下划线、中文或者中划线组成的合法字符，支持1-64个字符。如果未输入该参数，系统会自动生成模型name。
model_version	是	String	模型版本，格式需为“数值.数值.数值”，其中数值为1-2位正整数。版本不可以出现以0开头的版本号形式，如“01.01.01”等。
publish	否	Bool	是否发布模型。可选值： ● True: 发布模型。（默认值） ● False: 不发布模型，创建本地模型，可用来调试相关代码。
source_location_type	否	String	模型位置类型。可选值： ● OBS_SOURCE：source_location为OBS路径。（默认值） ● LOCAL_SOURCE：source_location为本地路径。
source_location	是	String	模型文件所在路径，指定到模型文件的父目录。 ● 当source_location_type为OBS_SOURCE时，模型文件所在路径为OBS路径，格式为“/obs_bucketname/.../model_file_parent_dir/”。 ● 当source_location_type为LOCAL_SOURCE时，模型文件所在路径为本地路径，格式为“/local_path/.../model_file_parent_dir/”。
environment	否	Environment实例	描述模型正常运行需要的环境，如使用的python版本、tensorflow版本等。 Environment实例的示例请参见**示例代码**。
source_job_id	否	String	来源训练作业的ID，模型是从训练作业产生的可填写，用于溯源；如模型是从第三方元模型导入，则为空，默认值为空。
source_job_version	否	String	来源训练作业的版本，模型是从训练作业产生的可填写，用于溯源；如模型是从第三方元模型导入，则为空，默认值为空。
source_type	否	String	模型来源的类型，当前仅可取值auto，用于区分通过自动学习部署过来的模型（不提供模型下载功能）；用户通过训练作业部署的模型不设置此值。默认值为空。

参数	是否必选	参数类型	描述
model_type	是	String	模型类型，取值为：TensorFlow/MXNet/Spark_MLlib/Scikit_Learn/XGBoost/MindSpore/Image/PyTorch。
model_algorithm	否	String	模型算法，表示模型的算法实现类型，如果已在模型配置文件中配置，则可不填。如：predict_analysis、object_detection 、image_classification。
description	否	String	模型描述信息，不超过100个字符，且不能包含特殊字符!<>=&’ ”。
execution_code	否	String	存放执行脚本的OBS路径，如果模型没有输出“customize_service.py”则需要通过该参数指定。推理脚本必须放于模型所在路径（请参见“source_location”参数）的model目录下，名称固定为：“customize_service.py”。
runtime	否	String	支持的运行环境。如果使用了model_type参数，则需要填该参数。不同引擎对应的runtime可参见**支持的常用引擎及其Runtime**。
input_params	否	params结构数组	模型推理输入参数列表，默认为空。如果已在模型配置文件中配置apis信息时，则可不填，后台自动从配置文件的apis字段中读取输入参数信息。
output_params	否	params结构数组	模型推理输出参数列表，默认为空。如果已在模型配置文件中配置apis信息时，则可不填，后台自动从配置文件的apis字段中读取输出参数信息。
dependencies	否	dependency结构数组	运行代码及模型需安装的依赖包，默认为空。如果已在模型配置文件中配置dependencies信息时，则可不填，后台自动从配置文件的dependencies字段中读取需要安装的依赖包。
apis	否	String	模型提供的推理接口列表，默认为空。如果已在模型配置文件中配置apis信息时，则可不填，后台自动从配置文件中的apis字段读取所配置的推理接口信息。

表 9-10 params 结构

参数	是否必选	参数类型	描述
url	是	String	模型推理接口的请求路径。
param_name	是	String	参数名，不超过64个字符。

参数	是否必选	参数类型	描述
param_type	是	String	JSON Schema基本参数类型，有string、object、array、boolean、number、integer。
min	否	Double	当param_type为int或float时，可选填，默认为空。
max	否	Double	当param_type为int或float时，可选填，默认为空。
param_desc	否	String	参数描述，不超过100个字符，默认为空。

表 9-11 dependency 结构

参数	是否必选	参数类型	描述
installer	是	String	安装方式，当前只支持“pip”。
packages	是	package结构数组	依赖包集合。

表 9-12 package 结构

参数	是否必选	参数类型	描述
package_name	是	String	依赖包名称。
package_version	否	String	依赖包版本。
restraint	否	String	版本过滤条件，当且仅当package_version存在时必填。取值为： ● EXACT：等于给定版本 ● ATLEAST：不小于给定版本 ● ATMOST：不大于给定版本

表 9-13 create_model 返回参数说明

参数	是否必选	参数类型	描述
model_instance	是	Model对象	模型对象，可以调用本章节模型管理的所有接口。

说明

给出MXNet实现手写数字识别项目中模型创建实例：

```
from modelarts.session import Session
from modelarts.model import Model

session = Session()
model_instance = Model(session,
                       model_name="digit_recognition",
                       model_version="1.0.0",
                       source_location=model_location,
                       model_type="MXNet",
                       model_algorithm="image_classification"
                       )
```

9.3 查询模型列表

示例代码

在ModelArts Notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

- **场景1**：查询当前用户所有模型

```
from modelarts.session import Session
from modelarts.model import Model

session = Session()
model_list = Model.get_model_list(session)
print(model_list)
```

- **场景2**：按照检索条件查询当前用户模型

```
from modelarts.session import Session
from modelarts.model import Model

session = Session()
model_list = Model.get_model_list(session, model_status="published", model_name="digit",
order="desc")
print(model_list)
```

参数说明

表 9-14 查询检索参数说明

参数	是否必选	参数类型	说明
model_name	否	String	模型名称，可支持模糊匹配。
model_version	否	String	模型版本。
model_status	否	String	模型状态，可根据模型的“publishing”、“published”、“failed”三种状态执行查询。
description	否	String	描述信息，可支持模糊匹配。
offset	否	Integer	指定要查询页的索引，默认为“0”。

参数	是否必选	参数类型	说明
limit	否	Integer	指定每一页返回的最大条目数，默认为“280”。
sort_by	否	String	指定排序字段，可选“create_at”、“model_version”、“model_size”，默认是可选“create_at”。
order	否	String	排序方式，可选“asc”或“desc”，代表递增排序及递减排序，默认是“desc”。
workspace_id	否	String	工作空间ID，默认为“0”。

表 9-15 get_model_list 打印参数说明

参数	参数类型	描述
total_count	Integer	不分页的情况下，符合查询条件的总模型数量。
count	Integer	模型数量。
models	model结构数组	模型元数据信息。

表 9-16 model 结构

参数	参数类型	描述
model_id	String	模型ID。
model_name	String	模型名称。
model_version	String	模型版本。
model_type	String	模型类型，取值为：TensorFlow/MXNet/Spark_MLlib/Scikit_Learn/XGBoost/MindSpore/Image/PyTorch。
model_size	Long	模型大小，单位为字节数。
tenant	String	模型归属租户。
project	String	模型归属项目。
owner	String	模型归属用户。
create_at	Long	模型创建时间，距'1970.1.1 0:0:0 UTC'的毫秒数。
description	String	模型描述信息。

参数	参数类型	描述
source_type	String	模型来源的类型，仅当模型为自动学习部署过来时有值，取值为“auto”。

9.4 查询模型对象列表

示例代码

在ModelArts Notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

- **场景1**：查询当前用户所有模型对象

```
from modelarts.session import Session
from modelarts.model import Model

session = Session()
model_object_list = Model.get_model_object_list(session)
print(model_object_list)
```

- **场景2**：按照检索条件查询当前用户模型对象

```
from modelarts.session import Session
from modelarts.model import Model

session = Session()
model_object_list = Model.get_model_object_list(session, model_status="published",
model_name="digit", order="desc")
print(model_object_list)
```

参数说明

- 查询模型列表，返回list，list大小等于当前用户所有已经部署的模型个数， list中每个元素都是Model对象，对象属性和**查询模型详情**相同。查询模型列表返回说明： model_list = [model_instance1, model_instance2, model_instance3 ...]，列表中元素model_instance对象即为本章节描述的模型管理，可调用模型接口。
- 支持按照检索参数查询模型列表，返回满足检索条件的模型list，检索参数如**表9-17**所示。
- 在查询列表时，返回list的同时，会打印模型列表的详细信息，如**表9-18**和**表9-19**所示。
- 当前支持最大获取150个模型对象。

表 9-17 查询检索参数说明

参数	是否必选	参数类型	说明
model_name	否	String	模型名称，可支持模糊匹配。
model_version	否	String	模型版本。

参数	是否必选	参数类型	说明
model_status	否	String	模型状态，可根据模型的“publishing”、“published”、“failed”三种状态执行查询。
description	否	String	描述信息，可支持模糊匹配。
offset	否	Integer	指定要查询页的索引，默认为“0”。
limit	否	Integer	指定每一页返回的最大条目数，默认为“280”。
sort_by	否	String	指定排序字段，可选“create_at”、“model_version”、“model_size”，默认是可选“create_at”。
order	否	String	排序方式，可选“asc”或“desc”，代表递增排序及递减排序，默认是“desc”。
workspace_id	否	String	工作空间ID，默认为“0”。

表 9-18 get_model_list 打印参数说明

参数	参数类型	描述
total_count	Integer	不分页的情况下，符合查询条件的总模型数量。
count	Integer	模型数量。
models	model结构数组	模型元数据信息。

表 9-19 model 结构

参数	参数类型	描述
model_id	String	模型ID。
model_name	String	模型名称。
model_version	String	模型版本。
model_type	String	模型类型，取值为：TensorFlow/MXNet/Spark_MLlib/Scikit_Learn/XGBoost/MindSpore/Image/PyTorch。
model_size	Long	模型大小，单位为字节数。
tenant	String	模型归属租户。

参数	参数类型	描述
project	String	模型归属项目。
owner	String	模型归属用户。
create_at	Long	模型创建时间，距'1970.1.1 0:0:0 UTC'的毫秒数。
description	String	模型描述信息。
source_type	String	模型来源的类型，仅当模型为自动学习部署过来时有值，取值为auto。

9.5 查询模型详情

查询当前模型对象的信息。

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

- **方式1**：根据**导入模型**生成的模型对象进行模型详情查询

```
from modelarts.session import Session
from modelarts.model import Model

session = Session()
model_instance = Model(session, model_id="your_model_id")
model_info = model_instance.get_model_info()
print(model_info)
```

- **方式2**：根据**查询模型对象列表**返回的模型对象进行模型详情查询

```
from modelarts.session import Session
from modelarts.model import Model

session = Session()
model_object_list = Model.get_model_object_list(session)
model_instance = model_object_list[0]
model_info = model_instance.get_model_info()
print(model_info)
```

参数说明

表 9-20 get_model_info 返回参数说明

参数	参数类型	描述
model_id	String	模型ID。
model_name	String	模型名称。
model_version	String	模型版本。
tenant	String	租户。

参数	参数类型	描述
project	String	项目。
owner	String	用户。
create_at	Long	模型创建时间，距'1970.1.1 0:0:0 UTC'的毫秒数。
source_location	String	模型所在的OBS路径。
source_job_id	String	来源训练作业的ID。
source_job_version	String	来源训练作业的版本。
source_type	String	模型来源的类型。 ● 当模型为自动学习部署过来时，取值为“auto”。 ● 当模型是用户通过训练作业或OBS模型文件部署时，此值为空。
model_type	String	模型类型，取值为：TensorFlow/MXNet/Spark_MLlib/Scikit_Learn/XGBoost/MindSpore/Image/PyTorch。
model_size	Long	模型大小，单位为字节数。
model_status	String	模型状态，取值为：publishing/published/failed。
description	String	模型描述信息。
execution_code	String	执行代码存放的OBS地址，名称固定为“customize_service.py”。
schema_doc	String	模型schema文档的下载地址。
image_address	String	模型的执行镜像地址，镜像未构建之前（即当前模型从未发布成服务），显示为空。
input_params	params结构数组	模型的输入参数集，默认为空
output_params	params结构数组	模型的输出参数集，默认为空
dependencies	dependency结构数组	运行代码及模型需安装的包。
model_metrics	String	模型评测参数，仅当source_job_id和source_job_version有值且对应的训练作业有评测结果时会返回该结果。
apis	String	模型所有的apis入参出参信息。

表 9-21 params 结构

参数	参数类型	描述
url	String	api代表的url路径。
param_name	String	参数名，不超过64个字符。
param_type	String	参数类型，取值为：int/string/float/timestamp/date/file。
min	Number	当param_type为int或float时创建模型时，有配置min则返回，默认为空。
max	Number	当param_type为int或float时创建模型时，有配置max则返回，默认为空。
param_desc	String	参数描述，不超过100个字符，默认为空。

表 9-22 dependency 结构

参数	参数类型	描述
installer	String	安装器名称。
packages	package结构数组	依赖包集合。

表 9-23 package 结构

参数	参数类型	描述
package_name	String	依赖包名称。
package_version	String	依赖包版本。
restraint	String	版本过滤条件，取值为： ● EXACT：等于给定版本 ● ATLEAST：不小于给定版本 ● ATMOST：不大于给定版本

表 9-24 metric 参数说明

参数	是否必选	参数类型	描述
f1	是	Double	平均数。

参数	是否必选	参数类型	描述
recall	是	Double	召回率。
precision	是	Double	精确率。
accuracy	是	Double	准确率。

9.6 删除模型

删除模型对象。

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

- **方式1**：根据**导入模型**或**模型调试**生成的模型对象进行模型对象删除

```
from modelarts.session import Session
from modelarts.model import Model

session = Session()
model_instance = Model(session, model_id="your_model_id")
model_instance.delete_model()
```

- **方式2**：根据**查询模型对象列表**返回的模型对象进行模型删除

```
from modelarts.session import Session
from modelarts.model import Model

session = Session()
model_object_list = Model.get_model_object_list(session)
model_instance = model_object_list[0]
model_instance.delete_model()
```

10 服务管理

10.1 服务管理概述

服务管理，包括将已创建成功的模型部署为在线服务或本地服务。可以实现在线预测、本地预测、服务详情查询、查看服务日志等功能。

这里的在线服务包括“predictor”和“transformer”两类，都包括下文描述的功能，本章节以“predictor”服务为例进行说明。

说明

本章节的示例代码都是在ModelArts Notebook中实现的，如果在其它开发环境使用，需要进行Session鉴权，请参见**Session鉴权**。

10.2 在开发环境中部署本地服务进行调试

可以通过部署本地服务来进行调试，即在**导入模型**或**模型调试**后，在开发环境Notebook中部署Predictor进行本地推理。

说明

只支持使用ModelArts Notebook部署本地服务。

- **开发环境本地服务Predictor和在线服务Predictor说明**
  - 部署开发环境本地服务Predictor，即将模型文件部署在开发环境中，其环境规格取决于开发环境资源规格；例如在一个modelarts.vm.cpu.2u的Notebook中，部署本地Predictor，其运行环境就是cpu.2u。
  - **部署在线服务**Predictor，即将存储在OBS中的模型文件部署到线上服务管理模块提供的容器中运行，其环境规格（如CPU规格，GPU规格）由**表3 predictor configs结构**决定。
  - **部署在线服务**Predictor需要线上服务端根据AI引擎创建容器，较耗时；本地Predictor部署较快，最长耗时10s，可用以测试模型，不建议进行模型的工业应用。
- 当前版本支持部署本地服务Predictor的AI引擎为：“XGBoost”、“Scikit_Learn”、“PyTorch”、“TensorFlow”和“Spark_MLlib”。具体版本信息可参考**支持的常用引擎及其Runtime**。

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

TensorFlow1.8本地推理示例代码

需要在环境中配置“tensorflow_model_server”，可调用SDK接口快速配置，请参考如下示例代码。

- CPU环境，调用Model.configure_tf_infer_environ(device_type="CPU")完成配置，环境中只需配置运行一次。
- GPU环境，调用Model.configure_tf_infer_environ(device_type="GPU")完成配置，环境中只需配置运行一次。

```
from modelarts.session import Session
from modelarts.model import Model
from modelarts.config.model_config import ServiceConfig

session = Session()
# GPU环境推理配置
Model.configure_tf_infer_environ(device_type="GPU")
# CPU环境推理配置
#Model.configure_tf_infer_environ(device_type="CPU")

model_instance = Model(
                    session,
                    model_name="input_model_name",              # 模型名称
                    model_version="1.0.0",                    # 模型版本
                    source_location=model_location,             # 模型文件路径
                    model_type="MXNet",                       # 模型类型
                    model_algorithm="image_classification",     # 模型算法
                    execution_code="OBS_PATH",
                    input_params=input_params,                # 参考input_params格式描述
                    output_params=output_params,                # 参考output_params格式描述
                    dependencies=dependencies,                # 参考dependencies格式描述
                    apis=apis)

configs = [ServiceConfig(model_id=model_instance.get_model_id(), weight="100", instance_count=1,
                         specification="local")]
predictor_instance = model_instance.deploy_predictor(configs=configs)
if predictor_instance is not None:
    predict_result = predictor_instance.predict(data="your_raw_data_or_data_path",
data_type="your_data_type")     # 本地推理预测，data支持raw data或者文件路径，data_type支持'json'、'files'或者'images'
    print(predict_result)
```

参数说明

表 10-1 部署本地服务 predictor 参数说明

参数	是否必选	参数类型	描述
service_name	否	String	服务名称，支持1-64位可见字符（含中文），只能以英文大小写字母或者中文字符开头，名称可以包含字母、中文、数字、中划线、下划线。
configs	是	JSON Array	本地服务运行配置。

表 10-2 predictor configs 结构

参数	是否必选	参数类型	描述
model_id	是	String	模型ID。“model_id”可以通过**查询模型列表**或者ModelArts管理控制台获取。
weight	是	Integer	权重百分比，分配到此模型的流量权重，部署本地服务Predictor时，取值100。
specification	是	String	部署本地服务时，取值为“local”。
instance_count	是	Integer	模型部署的实例数，当前限制最大实例数为128，部署本地服务Predictor时，取值为1。
envs	否	Map<String, String>	运行模型需要的环境变量键值对，可选填，默认为空。

表 10-3 部署本地服务 predictor 返回参数说明

参数	是否必选	参数类型	描述
predictor	是	Predictor对象	Predictor对象，其属性只包括**推理服务测试**。

10.3 部署在线服务

部署在线服务包括：

- 已部署为在线服务的初始化。
- 部署在线服务predictor。
- 部署批量服务transformer。

部署服务返回服务对象Predictor，其属性包括服务管理章节下的所有功能。

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

- **方式1：已部署为在线服务predictor的初始化**

```
from modelarts.session import Session
from modelarts.model import Predictor

session = Session()
predictor_instance = Predictor(session, service_id="your_service_id")
```

- **方式2：部署在线服务predictor**

- 部署服务到公共资源池

```
from modelarts.session import Session
from modelarts.model import Model
from modelarts.config.model_config import ServiceConfig, TransformerConfig, Schedule

session = Session()
model_instance = Model(session, model_id='your_model_id')
vpc_id = None                                   # （可选）在线服务实例部署的虚拟私有云ID，默认为空
subnet_network_id = None                        # （可选）子网的网络ID，默认为空
security_group_id = None                        # （可选）安全组，默认为空
configs = [ServiceConfig(model_id=model_instance.model_id,
                         weight="100",
                         instance_count=1,
                         specification="modelarts.vm.cpu.2u")]  # 参考表3中specification字段
predictor_instance = model_instance.deploy_predictor(
            service_name="service_predictor_name",
            infer_type="real-time",
            vpc_id=vpc_id,
            subnet_network_id=subnet_network_id,
            security_group_id=security_group_id,
            configs=configs,                    # predictor配置参数, 参考下文configs参数格式说明
            schedule = [Schedule(op_type='stop', time_unit='HOURS', duration=1)]      # （可选）设置在线服务运行时间
)
```

参数“model_id”代表将部署成在线服务的模型。“model_id”可以通过**查询模型列表**或者ModelArts管理控制台获取。

- 部署服务到专属资源池

```
from modelarts.config.model_config import ServiceConfig

configs = [ServiceConfig(model_id=model_instance.model_id, weight="100", instance_count=1,
                         specification="modelarts.vm.cpu.2u")]
predictor_instance = model_instance.deploy_predictor(
                                            service_name="your_service_name",
                                            infer_type="real-time",
                                            configs=configs,
                                            cluster_id="your dedicated pool id"
                                           )
```

configs参数格式说明：SDK提供了ServiceConfig类对其定义，configs为list，list中的元组对象是ServiceConfig。定义代码如下：

```
configs = []
envs = {"model_name":"mxnet-model-1", "load_epoch":"0"}

service_config1 = ServiceConfig(
      model_id="model_id1",              # model_id1和model_id2必须是同一个模型的不同版本对应的model_id
      weight="70",
      specification="modelarts.vm.cpu.2u",  # 参考表3中specification字段
      instance_count=2,
      envs=envs)                         # （可选）设置环境变量的值，如：envs = {"model_name":"mxnet-model-1", "load_epoch":"0"}
service_config2 = ServiceConfig(
      model_id='model_id2',
      weight="30",
      specification="modelarts.vm.cpu.2u",  # 参考表3中specification字段
      instance_count=2,
      envs=envs)                         # （可选）设置环境变量的值，如：envs = {"model_name":"mxnet-model-1", "load_epoch":"0"}
configs.append(service_config1)
configs.append(service_config2)
```

- **方式3：部署批量服务transformer**

```
from modelarts.session import Session
from modelarts.model import Model
from modelarts.config.model_config import TransformerConfig

session = Session()
model_instance = Model(session, model_id='your_model_id')
```

```
vpc_id = None                                  # （可选）批量服务实例部署的虚拟私有云ID，默认为空
subnet_network_id = None                       # （可选）子网的网络ID，默认为空
security_group_id = None                       # （可选）安全组，默认为空

transformer = model_instance.deploy_transformer(
        service_name="service_transformer_name",
        infer_type="batch",
        vpc_id=vpc_id,
        subnet_network_id=subnet_network_id,
        security_group_id=security_group_id,
        configs=configs                        # transformer配置参数, 参考下文configs参数格式说明
)
```

configs参数格式说明：SDK提供了TransformerConfig类对其定义，configs都是list，list中的元组对象是TransformerConfig。定义代码如下：

```
configs = []
mapping_rule = None                            # （可选）输入参数与csv数据的映射关系
mapping_type= "file"                           # file或者csv
envs = {"model_name":"mxnet-model-1", "load_epoch":"0"}

transformer_config1 = TransformerConfig(
            model_id="model_id",
            specification="modelarts.vm.cpu.2u",   # 参考表3中specification字段
            instance_count=2,
            src_path="/shp-cn4/sdk-demo/",         # 批量任务输入数据的OBS路径，如："/your_obs_bucket/
src_path"
            dest_path="/shp-cn4/data-out/",        # 批量任务输出结果的OBS路径，如："/your_obs_bucket/
dest_path"
            req_uri="/",
            mapping_type=mapping_type,
            mapping_rule=mapping_rule,
            envs=envs)                             # （可选）设置环境变量的值，如：envs =
{"model_name":"mxnet-model-1", "load_epoch":"0"}
configs.append(transformer_config1)
```

参数说明

表 10-4 参数说明

参数	是否必选	参数类型	描述
service_id	是	String	服务ID，可从ModelArts前端在线服务中获取。
session	是	Object	会话对象，初始化方法见**Session鉴权**。

表 10-5 部署在线服务 predictor 和 transformer 参数说明

参数	是否必选	参数类型	描述
service_name	否	String	服务名称，支持1-64位可见字符（含中文），只能以英文大小写字母或者中文字符开头，名称可以包含字母、中文、数字、中划线、下划线。
description	否	String	服务备注，默认为空，不超过100个字符。

参数	是否必选	参数类型	描述
infer_type	否	String	推理方式，取值为real-time/batch。默认为real-time。 ● real-time代表在线服务，将模型部署为一个Web Service，并且提供在线的测试UI与监控能力，服务一直保持运行。 ● batch为批量服务，批量服务可对批量数据进行推理，完成数据处理后自动停止。
vpc_id	否	String	在线服务实例部署的虚拟私有云ID，默认为空，此时ModelArts会为每个用户分配一个专属的VPC，用户之间隔离；如需要在服务实例中访问名下VPC内的其他服务组件，则可配置此参数为对应VPC的ID。 VPC一旦配置，不支持修改。当vpc_id与cluster_id一同配置时，只有专属集群参数生效。
subnet_network_id	否	String	子网的网络ID，默认为空，当配置了vpc_id则此参数必填。需填写虚拟私有云控制台子网详情中显示的“网络ID”。通过子网可提供与其他网络隔离的、可以独享的网络资源。
security_group_id	否	String	安全组，默认为空，当配置了vpc_id则此参数必填。安全组起着虚拟防火墙的作用，为服务实例提供安全的网络访问控制策略。安全组须包含至少一条入方向规则，对协议为TCP、源地址为0.0.0.0/0、端口为8080的请求放行。
configs	是	包括predictor configs结构和transformer configs	模型运行配置。 ● 当推理方式为batch时仅支持配置一个模型。 ● 当推理方式为real-time时，可根据业务需要配置多个模型并分配权重，但多个模型的版本号不能相同
schedule	否	schedule结构数组	服务调度配置，仅在线服务可配置，默认不使用，服务长期运行。请参见**表10-9**。
cluster_id	否	String	旧版专属池id，默认为空，当配置cluster_id时，表示将服务部署到旧版专属资源池中。
pool_name	否	String	新版专属池名称。

表 10-6 predictor configs 结构

参数	是否必选	参数类型	描述
model_id	是	String	模型ID。“model_id”可以通过**查询模型列表**或者ModelArts管理控制台获取。

weight	是	Integer	权重百分比，分配到此模型的流量权重，仅当infer_type为real-time时需要配置，多个权重相加必须等于100；当在一个在线服务中同时配置了多个模型版本且设置不同的流量权重比例时，持续地访问此服务的预测接口，ModelArts会按此权重比例将预测请求转发到对应的模型版本实例。
			{
			"service_name": "mnist",
			"description": "mnist service",
			"infer_type": "real-time",
			"config": [
			{
			"model_id": "xxxmodel-idxxx",
			"weight": "70",
			"specification": "modelarts.vm.cpu.2u",
			"instance_count": 1,
			"envs":
			{
			"model_name": "mxnet-model-1",
			"load_epoch": "0"
			}
			},
			{
			"model_id": "xxxxxx",
			"weight": "30",
			"specification": "modelarts.vm.cpu.2u",
			"instance_count": 1
			}
			]
			}

specification	是	String	资源规格，当前版本可选modelarts.vm.cpu.2u/modelarts.vm.gpu.p4(需申请)/modelarts.vm.ai1.a310(需申请)，需申请权限才能使用的规格请在华为云创建工单，由ModelArts运维工程师添加权限。
instance_count	是	Integer	模型部署的实例数，当前限制最大实例数为128，如需使用更多的实例数，需提交工单申请。
envs	否	Map<String, String>	运行模型需要的环境变量键值对，可选填，默认为空。

表 10-7 transformer configs 结构

参数	是否必选	参数类型	描述
model_id	是	String	模型ID。
specification	是	String	资源规格，当前版本可选modelarts.vm.cpu.2u/modelarts.vm.gpu.p4。
instance_count	是	Integer	模型部署的实例数，邀测阶段取值范围[1, 2]。
envs	否	Map<String, String>	运行模型需要的环境变量键值对，可选填，默认为空。
src_path	是	String	批量任务输入数据的OBS路径。
dest_path	是	String	批量任务输出结果的OBS路径。
req_uri	是	String	批量任务中调用的推理接口，即模型镜像中暴露的REST接口，需要从模型的config.json文件中选取一个api路径用于此次推理；如使用ModelArts提供的预置推理镜像，则此接口为“/”。

参数	是否必选	参数类型	描述

mapping_type	是	String	输入数据的映射类型，可选“file”或“csv”。
			● file指每个推理请求对应到输入数据目录下的一个文件，当使用此方式时，此模型对应req_uri只能有一个输入参数且此参数的类型是file。
			● csv指每个推理请求对应到csv里的一行数据，当使用此方式时，输入数据目录下的文件只能以.csv为后缀，且需配置mapping_rule参数，以表达推理请求体中各个参数对应到csv的索引。
			创建批量服务且输入数据映射方式为file的样例
			{
			"service_name": "batchservicetest",
			"description": "",
			"infer_type": "batch",
			"config": [{
			"model_id": "598b913a-af3e-41ba-a1b5-bf065320f1e2",
			"specification": "modelarts.vm.cpu.2u",
			"instance_count": 1,
			"src_path": "https://infers-data.obs.example.com/xgboosterdata/",
			"dest_path": "https://infers-data.obs.example.com/output/",
			"req_uri": "/",
			"mapping_type": "file"
			}]
			}
			创建批量服务且输入数据映射方式为csv的样例
			{
			"service_name": "batchservicetest",
			"description": "",
			"infer_type": "batch",
			"config": [{
			"model_id": "598b913a-af3e-41ba-a1b5-bf065320f1e2",
			"specification": "modelarts.vm.cpu.2u",
			"instance_count": 1,
			"src_path": "https://infers-data.obs.example.com/xgboosterdata/",
			"dest_path": "https://infers-data.obs.example.com/output/",
			"req_uri": "/",
			"mapping_type": "csv",
			"mapping_rule": {
			"type": "object",
			"properties": {
			"data": {
			"type": "object",
			"properties": {
			"req_data": {
			"type": "array",
			"items": [{
			"type": "object",
			"properties": {
			"input5": {
			"type": "number",
			"index": 0
			},
			"input4": {
			"type": "number",
			"index": 1
			},
			"input3": {
			"type": "number",
			"index": 2
			},
			"input2": {

参数	是否必选	参数类型	描述

			<pre>"type": "number",
			"index": 3
			},
			"input1": {
			"type": "number",
			"index": 4
			}
			}
			}]
			}
			}
			}
			}
			}]
			}</pre>

mapping_rule	否	Map	输入参数与csv数据的映射关系，仅当mapping_type为csv时需要填写。映射规则与模型配置文件config.json中输入参数的定义方式相似，只需要在每一个基本类型（string/number/integer/boolean）的参数下配置index参数，指定使用csv数据中对应索引下标的数据作为此参数的值去发送推理请求，csv数据必须以英文半角逗号分隔，index从0开始计数，特殊地，当index为-1时忽略此参数，具体请参见**部署transformer的示例代码**的样例。
			样例中mapping_rule描述的推理请求体格式为：
			<pre>{
			"data": {
			"req_data": [{
			"input1": 1,
			"input2": 2,
			"input3": 3,
			"input4": 4,
			"input5": 5
			}]
			}
			}</pre>

表 10-8 部署 predictor 和 transformer 返回参数说明

参数	是否必选	参数类型	描述
predictor	是	Predictor对象	Predictor对象，其属性描述包括服务管理章节全部功能。

表 10-9 schedule 结构

参数	是否必选	参数类型	说明
op_type	是	String	调度类型，当前仅支持取值为“stop”。
time_unit	是	String	调度时间单位，可选： • DAYS • HOURS • MINUTES
duration	是	Integer	对应时间单位的数值，比如2小时后停止，则“time_unit”填“HOURS”，“duration”填“2”。

说明

- **给出MXNet实现手写数字识别项目中部署在线predictor实例：**

```
from modelarts.session import Session
from modelarts.model import Model
from modelarts.config.model_config import ServiceConfig, TransformerConfig

model_instance = Model(session, model_id = "you_model_id")
configs = []
config1 = ServiceConfig(model_id="you_model_id",
                        weight="100",
                        instance_count=1,
                        specification="modelarts.vm.cpu.2u",
                        envs={"input_data_name":"images",
                              "input_data_shape":"0,1,28,28",
                              "output_data_shape":"0,10"})
configs.append(config1)
predictor = model_instance.deploy_predictor(service_name="DigitRecognition", configs=configs)
```

- **给出MXNet实现手写数字识别项目中部署transformer实例（批量推理）：**

```
from modelarts.session import Session
from modelarts.model import Model
from modelarts.config.model_config import ServiceConfig, TransformerConfig

model_instance = Model(session, model_id = "your_model_id")
configs = []
config1 = TransformerConfig(model_id="your_model_id",
                            specification="modelarts.vm.cpu.2u",
                            instance_count=1,

envs={"input_data_name":"images","input_data_shape":"0,1,28,28","output_data_shape":"0,10"},
                            src_path="/w0403/testdigitrecognition/inferimages/",
                            dest_path="/w0403/testdigitrecognition/" ,
                            req_uri = "/",
                            mapping_type = "file")
configs.append(config1)
predictor = model_instance.deploy_transformer(service_name="DigitRecognition",
infer_type="batch", configs=configs)
```


10.4 查询服务详情

查询当前服务对象的详细信息。

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

- **方式1**：根据**部署在线服务**生成的服务对象进行服务详情查询

```
from modelarts.session import Session
from modelarts.model import Predictor

session = Session()
predictor_instance = Predictor(session, service_id="your_service_id")
predictor_info = predictor_instance.get_service_info()
print(predictor_info)
```

- **方式2**：根据**查询服务对象列表**返回的服务对象进行服务详情查询

```
from modelarts.session import Session
from modelarts.model import Predictor

session = Session()
predictor_object_list = Predictor.get_service_object_list(session)
predictor_instance = predictor_object_list[0]
predictor_info = predictor_instance.get_service_info()
print(predictor_info)
```

参数说明

表 10-10 get_service_info 返回参数说明

参数	参数类型	描述
service_id	String	服务ID。
service_name	String	服务名称。
description	String	服务描述。
tenant	String	服务归属租户。
project	String	服务归属项目。
owner	String	服务归属用户。
publish_at	Number	服务最新的发布时间，距'1970.1.1 0:0:0 UTC'的毫秒数。
infer_type	String	推理方式，取值为real-time/batch。
vpc_id	String	服务实例所在的虚拟私有云ID，服务自定义网络配置时返回。
subnet_network_id	String	服务实例所在的子网的网络ID，服务自定义网络配置时返回。

参数	参数类型	描述
security_group_id	String	服务实例所在的安全组，服务自定义网络配置时返回。
status	String	服务状态，取值为：running/deploying/concerning/failed/stopped/finished。
error_msg	String	错误信息，当status为failed时，返回注明部署失败原因。
config	不同infer_type的config结构数组	不同infer_type的config结构数组 服务配置（如果是共享过来的服务，只返回model_id ，model_name，model_version ）
access_address	String	推理请求的访问地址，当infer_type为real-time时会返回此值
invocation_times	Number	服务的总调用次数。
failed_times	Number	服务调用失败次数。
is_shared	Boolean	是否是订阅的服务。
shared_count	Number	订阅的服务数。
progress	Integer	部署进度，当状态是deploying时，返回此参数。

表 10-11 real-time config 结构

参数	参数类型	描述
model_id	String	模型ID。“model_id”可以通过**查询模型列表**或者ModelArts管理控制台获取。
model_name	String	模型名称。
model_version	String	模型版本。
source_type	String	模型来源，当模型是由自动学习产生时，返回此字段，取值为：auto。
status	String	模型实例运行状态，取值为： • ready：已就绪（所有实例已启动） • concerning：部分就绪（部分实例已启动、部分实例未启动） • notReady：未就绪（所有实例都没启动）
weight	Integer	权重，分配到此模型的流量权重。

参数	参数类型	描述
specification	String	资源规格，取值为：modelarts.vm.cpu.2u/ modelarts.vm.gpu.p4/ modelarts.vm.ai1.a310。
envs	Map<String, String>	运行模型需要的环境变量键值对。
instance_count	Integer	模型部署的实例数。
scaling	Boolean	是否启用弹性伸缩。

表 10-12 batch config 结构

参数	参数类型	描述
model_id	String	模型ID。“model_id”可以通过**查询模型列表**或者ModelArts管理控制台获取。
model_name	String	模型名称。
model_version	String	模型版本。
specification	String	资源规格，取值为：modelarts.vm.cpu.2u/ modelarts.vm.gpu.p4。
envs	Map<String, String>	运行模型需要的环境变量键值对。
instance_count	Integer	模型部署的实例数。
src_path	String	批量任务输入数据的OBS路径。
dest_path	String	批量任务输出结果的OBS路径。
req_uri	String	批量任务中调用的推理路径。
mapping_type	String	输入数据的映射类型，取值为：file或csv。
mapping_rule	Map	输入参数与csv数据的映射关系，仅当mapping_type为csv时，会返回。

10.5 推理服务测试

推理服务在线测试支持文件、图片、json三种格式。通过部署为在线服务Predictor可以完成在线推理预测。

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

场景：部署在线服务Predictor的推理预测

```
from modelarts.session import Session
from modelarts.model import Predictor

session = Session()
predictor_instance = Predictor(session, service_id="your_service_id")
predict_result = predictor_instance.predict(data=data_path, data_type=data_type)
print(predict_result)
```

参数说明

表 10-13 参数说明

参数	是否必选	参数类型	描述
data_type	是	String	当前支持三种格式：files、images、json， 即文本、图片、json格式。

data	是	String	• 针对files、images类型的数据， 该参数为其本地路径，如：
			`data = "/home/ma-user/work/test.jpg"`
			• 针对json类型的数据， 该参数可以是其本地路径，如：
			`data = "/home/ma-user/work/test.json"`
			同时也可以为“dict”类型的变量 ，如：
			`data = {`
			`"is_training": "False",`
			`"observations": [[1,2,3,4]],`
			`"default_policy/eps:0" : "0.0"`
			`}`

path	否	String	服务内的推理路径，默认为"/"。

表 10-14 predict 返回参数说明

参数	描述
返回消息体	输出的参数和值，平台只做转发，不做识别。

10.6 查询服务列表

获取当前用户服务列表。

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

- **场景1**：查询当前用户所有服务

```
from modelarts.session import Session
from modelarts.model import Predictor

session = Session()
predictor_list = Predictor.get_service_list(session)
print(predictor_list)
```

- **场景2**：按照检索条件查询当前用户服务

```
from modelarts.session import Session
from modelarts.model import Predictor

session = Session()
predictor_list = Predictor.get_service_list(session, service_name="digit", order="asc", offset="0",
infer_type="real-time")
print(predictor_list)
```

参数说明

表 10-15 查询检索参数说明

参数	是否必选	参数类型	描述
session	是	Object	会话对象，初始化方法见Session鉴权。
service_id	否	String	服务ID，默认不过滤服务ID。
service_name	否	String	服务名称，默认不过滤服务名。
infer_type	否	String	推理方式，取值为：real-time/batch，默认不过滤推理方式。
offset	否	Integer	分页列表的起始页，默认为：“0”。
limit	否	Integer	指定每一页返回的最大条目数，默认为：“1000”。
service_status	否	String	服务状态，默认不过滤服务状态。可根据服务状态查询，取值如下。 • running：运行中，服务正常运行，正在计费。 • deploying：部署中，服务正在部署，调度资源部署等。 • concerning：告警，后端实例异常，可能正在计费。例如多实例的情况下，有的实例正常，有的实例异常。正常的实例会产生费用，此时服务状态是concerning。 • failed：失败，服务部署失败，失败原因可以查看事件和日志标签。 • stopped：停止。 • finished：只有批量服务会有这个状态，表示运行完成。

参数	是否必选	参数类型	描述
sort_by	否	String	指定排序字段，可选“publish_at”、“service_name”，默认可选“publish_at”。
order	否	String	排序方式，可选“asc”或“desc”，代表递增排序及递减排序，默认为：“desc”。
model_id	否	String	模型ID，默认不过滤模型ID。

表 10-16 get_service_list 返回参数说明

参数	参数类型	描述
total_count	Integer	不分页的情况下，符合查询条件的总服务数量。
count	Integer	当前查询结果的服务数量，不设置offset、limit查询参数时，count与total相同。
services	service结构数组	查询到的服务集合。

表 10-17 service 结构

参数	参数类型	描述
service_id	String	服务ID。
service_name	String	服务名称。
description	String	服务描述。
tenant	String	服务归属租户。
project	String	服务归属项目。
owner	String	服务归属用户。
publish_at	Number	服务最新的发布时间，距'1970.1.1 0:0:0 UTC'的毫秒数。
infer_type	String	推理方式，取值为：real-time/batch。
status	String	服务状态，取值为：running/deploying/concerning/failed/stopped/finished。
progress	Integer	部署进度，当状态是deploying时，返回。
invocation_times	Number	服务的总调用次数。
failed_times	Number	服务调用失败次数。

参数	参数类型	描述
is_shared	Boolean	是否是订阅的服务。
shared_count	Number	订阅的服务数。

10.7 查询服务对象列表

获取当前用户服务对象列表。

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

- **场景1**：查询当前用户所有服务对象

```
from modelarts.session import Session
from modelarts.model import Predictor

session = Session()
predictor_list_object_resp = Predictor.get_service_object_list(session)
print(predictor_list_object_resp)
```

- **场景2**：按照检索条件查询当前用户服务对象

```
from modelarts.session import Session
from modelarts.model import Predictor

session = Session()
predictor_object_list = Predictor.get_service_object_list(session, service_name="digit", order="asc",
offset="0", infer_type="real-time")
print(predictor_object_list)
```

参数说明

- 查询服务列表，返回list，list大小等于当前用户所有已经部署的服务个数，list中每个元素都是Predictor对象，对象属性同本章初始化服务。

  查询服务列表返回说明：service_list_resp = [service_instance1, service_instance2, service_instance3 ...]，列表中元素“service_instance”对象即为服务管理章节描述的可调用服务接口。
- 支持按照检索参数查询服务列表，返回满足检索条件的服务list，检索参数如**表10-18**所示。
- 在查询列表时，返回list的同时，默认会打印模型列表的详细信息，如**表10-19**和**表10-20**所示。

表 10-18 查询检索参数说明

参数	是否必选	参数类型	描述
session	是	Object	会话对象，初始化方法见**Session鉴权**。
is_show	否	Boolean	是否打印出服务对象信息，默认为“True”。

参数	是否必选	参数类型	描述
service_id	否	String	服务ID，默认不过滤服务ID。
service_name	否	String	服务名称，默认不过滤服务名。
infer_type	否	String	推理方式，取值为：real-time/batch，默认不过滤推理方式。
offset	否	Integer	分页列表的起始页，默认为“0”。
limit	否	Integer	指定每一页返回的最大条目数，默认为“1000”。
sort_by	否	String	指定排序字段，可选“publish_at”、“service_name”，默认可选“publish_at”。
order	否	String	排序方式，可选“asc”或“desc”，代表递增排序及递减排序，默认为：“desc”。
model_id	否	String	模型ID，默认不过滤模型ID。

表 10-19 get_service_list 返回参数说明

参数	参数类型	描述
total_count	Integer	不分页的情况下，符合查询条件的总服务数量。
count	Integer	当前查询结果的服务数量，不设置offset、limit查询参数时，count与total相同。
services	service结构数组	查询到的服务集合。

表 10-20 service 结构

参数	参数类型	描述
service_id	String	服务ID。
service_name	String	服务名称。
description	String	服务描述。
tenant	String	服务归属租户。
project	String	服务归属项目。
owner	String	服务归属用户。
publish_at	Number	服务最新的发布时间，距'1970.1.1 0:0:0 UTC'的毫秒数。

参数	参数类型	描述
infer_type	String	推理方式，取值为：real-time/batch。
status	String	服务状态，取值为：running/deploying/concerning/failed/stopped/finished。
progress	Integer	部署进度，当状态是deploying时，返回。
invocation_times	Number	服务的总调用次数。
failed_times	Number	服务调用失败次数。
is_shared	Boolean	是否是订阅的服务。
shared_count	Number	订阅的服务数。

10.8 更新服务配置

更新当前服务对象配置。

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数；其它平台的Session鉴权请参考**Session鉴权**。

- **方式1**：根据**部署在线服务**生成的服务对象进行更新服务配置

```
from modelarts.session import Session
from modelarts.model import Predictor
from modelarts.config.model_config import ServiceConfig

session = Session()
predictor_instance = Predictor(session, service_id="your_service_id")
configs = [ServiceConfig(weight="100", instance_count=1,
specification="modelarts.vm.cpu.2u",model_id="your_model_id")]
service_config = predictor_instance.update_service_config(description="description",
                                                          status="running",
                                                          configs=configs)
```

- **方式2**：根据**查询服务对象列表**返回的服务对象进行更新服务配置

```
from modelarts.session import Session
from modelarts.model import Predictor
from modelarts.config.model_config import ServiceConfig

session = Session()
predictor_object_list = Predictor.get_service_object_list(session)
predictor_instance = predictor_object_list[0]
configs = [ServiceConfig(weight="100", instance_count=1,
specification="modelarts.vm.cpu.2u",model_id="your_model_id")]
predictor_config = predictor_instance.update_service_config(description="description",
                                                            status="running",
                                                            configs=configs)
```

参数说明

表 10-21 部署 predictor 参数说明

参数	是否必选	参数类型	描述
description	否	String	服务描述，不超过100个字符，不设置此参数时，表示不更新。
status	否	String	服务状态，可设置状态为running或stopped来启动、停止服务，不设置此参数则不修改状态。status不可跟configs同时修改，同时存在则只修改status。
configs	否	包括predictor configs结构和transformer configs	服务配置，不设置此参数时，表示不更新。关于configs如何生成，请参见**部署在线服务**。

说明

更新服务配置时，存在以下约束：

- 参数status指定的目标状态不允许和当前服务状态相同。
- 当前服务状态是deploying（部署中）、stopping（停止中）、deleting（删除中）时，不允许参数status设置成running（启动）或设置参数configs（服务配置）。
- 当前服务状态是waiting（排队中）时，不允许参数status设置成running（启动）。
- 当前服务状态是concerning（告警）时，不允许参数status设置成running（启动）。

表 10-22 predictor configs 结构

参数	是否必选	参数类型	描述
model_id	是	String	模型ID。“model_id”可以通过**查询模型列表**或者ModelArts管理控制台获取。
weight	是	Integer	权重百分比，分配到此模型的流量权重，仅当infer_type为real-time时需要配置，多个权重相加必须等于100；当在一个在线服务中同时配置了多个模型版本且设置不同的流量权重比例时，持续地访问此服务的预测接口，ModelArts会按此权重比例将预测请求转发到对应的模型版本实例。
specification	是	String	资源规格，当前版本可选modelarts.vm.cpu.2u/modelarts.vm.gpu.p4(需申请)/modelarts.vm.ai1.a310(需申请)，需申请权限才能使用的规格请在华为云创建工单，由ModelArts运维工程师添加权限。

参数	是否必选	参数类型	描述
instance_count	是	Integer	模型部署的实例数，当前限制最大实例数为128，如需使用更多的实例数，需提交工单申请。
envs	否	Map<String, String>	运行模型需要的环境变量键值对，可选填，默认为空。

表 10-23 transformer configs 结构

参数	是否必选	参数类型	描述
model_id	是	String	模型ID。“model_id”可以通过**查询模型列表**或者ModelArts管理控制台获取。
specification	是	String	资源规格，当前版本可选modelarts.vm.cpu.2u/modelarts.vm.gpu.p4。
instance_count	是	Integer	模型部署的实例数，当前限制最大实例数为128，如需使用更多的实例数，需提交工单申请。
envs	否	Map<String, String>	运行模型需要的环境变量键值对，可选填，默认为空。
src_path	是	String	批量任务输入数据的OBS路径。
dest_path	是	String	批量任务输出结果的OBS路径。
req_uri	是	String	批量任务中调用的推理接口，需要从模型的config.json文件中选取一个api路径用于此次推理
mapping_type	是	String	输入数据的映射类型，可选“file”或“csv”。 • file指每个推理请求对应到输入数据目录下的一个文件，当使用此方式时，此模型对应req_uri只能有一个输入参数且此参数的类型是file。 • csv指每个推理请求对应到csv里的一行数据，当使用此方式时，输入数据目录下的文件只能以.csv为后缀，且需配置mapping_rule参数，以表达推理请求体中各个参数对应到csv的索引。

参数	是否必选	参数类型	描述
mapping_rule	否	Map	输入参数与csv数据的映射关系，仅当mapping_type为csv时需要填写。映射规则与模型配置文件config.json中输入参数的定义方式相似，只需要在每一个基本类型（string/number/integer/boolean）的参数下配置index参数，指定使用csv数据中对应索引下标的数据作为此参数的值去发送推理请求，csv数据必须以英文半角逗号分隔，index从0开始计数，特殊地，当index为-1时忽略此参数。

表 10-24 update_service_config 返回参数说明

参数	是否必选	参数类型	描述
error_code	是	String	调用失败时，的错误码。 调用成功时，无此字段。
error_msg	是	String	调用失败时，错误信息。 调用成功时，无此字段。

10.9 查询服务监控信息

查询当前服务对象监控信息。

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

- **方式1**：根据**部署在线服务**生成的服务对象进行查询服务监控

```
from modelarts.session import Session
from modelarts.model import Predictor

session = Session()
predictor_instance = Predictor(session, service_id="your_service_id")
predictor_monitor = predictor_instance.get_service_monitor()
print(predictor_monitor)
```

- **方式2**：根据**查询服务对象列表**返回的服务对象进行查询服务监控

```
from modelarts.session import Session
from modelarts.model import Predictor

session = Session()
predictor_object_list = Predictor.get_service_object_list(session)
predictor_instance = predictor_object_list[0]
predictor_monitor = predictor_instance.get_service_monitor()
print(predictor_monitor)
```

参数说明

表 10-25 get_service_monitor 返回参数说明

参数	参数类型	描述
service_id	String	服务ID。
service_name	String	服务名称。
monitors	服务的infer_type对应的monitor结构数组	监控信息详情。

表 10-26 real-time monitor 结构

参数	参数类型	描述
model_id	String	模型ID。
model_name	String	模型名称。
model_version	String	模型版本。
invocation_times	Number	模型实例的总调用次数。
failed_times	Number	模型实例调用失败次数。
cpu_core_usage	Float	已使用CPU核数。
cpu_core_total	Float	总CPU核数。
cpu_memory_usage	Integer	已使用内存，单位MB。
cpu_memory_total	Integer	总内存，单位MB。
gpu_usage	Float	已使用GPU个数。
gpu_total	Float	总GPU个数。

10.10 查询服务日志

查询当前服务对象的日志信息。

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

- **方式1**：根据**部署在线服务**生成的服务对象进行查询服务日志

```
from modelarts.session import Session
from modelarts.model import Predictor

session = Session()
predictor_instance = Predictor(session, service_id="your_service_id")
predictor_log = predictor_instance.get_service_logs()
print(predictor_log)
```

- **方式2**：根据**查询服务对象列表**返回的服务对象进行查询服务日志

```
from modelarts.session import Session
from modelarts.model import Predictor

session = Session()
predictor_object_list = Predictor.get_service_object_list(session)
predictor_instance = predictor_object_list[0]
predictor_log = predictor_instance.get_service_logs()
print(predictor_log)
```

参数说明

表 10-27 get_service_logs 返回参数说明

参数	参数类型	描述
service_id	String	服务ID。
service_name	String	服务名称。
logs	log结构数组	服务的更新记录。

表 10-28 log 结构

参数	参数类型	描述
update_time	Long	更新时间，距'1970.1.1 0:0:0 UTC'的毫秒数。
result	String	更新结果，取值为：SUCCESS/FAIL/RUNNING。
config	Config结构数组	更新后的服务配置，当infer_type为real-time时，会返回此值。

表 10-29 config 结构

参数	参数类型	描述
model_id	String	模型ID。
model_name	String	模型名称。
model_version	String	模型版本。
weight	Integer	权重，分配到此模型的流量权重。

参数	参数类型	描述
specification	String	资源规格。
instance_count	Integer	模型部署的实例数。
envs	Map<String, String>	运行模型需要的环境变量键值对。

表 10-30 result 结构

参数	参数类型	描述
node_name	String	边缘节点名称。
operation	String	操作类型，取值deploy/delete。
result	Boolean	操作结果，true代表成功，false表示操作失败。

10.11 删除服务

删除服务存在如下两种删除方式。

- 根据**部署在线服务**生成的服务对象删除服务。
- 根据**查询服务对象列表**返回的服务对象删除服务。

示例代码

在ModelArts notebook平台，Session鉴权无需输入鉴权参数。其它平台的Session鉴权请参见**Session鉴权**。

- **方式1**：根据**部署在线服务**生成的服务对象删除服务

```
from modelarts.session import Session
from modelarts.model import Predictor

session = Session()
predictor_instance = Predictor(session, service_id="your_service_id")
predictor_instance.delete_service()
```

- **方式2**：根据**查询服务对象列表**返回的服务对象删除服务

```
from modelarts.session import Session
from modelarts.model import Predictor

session = Session()
predictor_object_list = Predictor.get_service_object_list(session)
predictor_instance = predictor_object_list[0]
predictor_instance.delete_service()
```