



# 《华为云基本使用方法》





# 目录

**01**

**准备数据**

**02**

**代码调试**

**03**

**训练模型**

**04**

**后续任务**

# 01

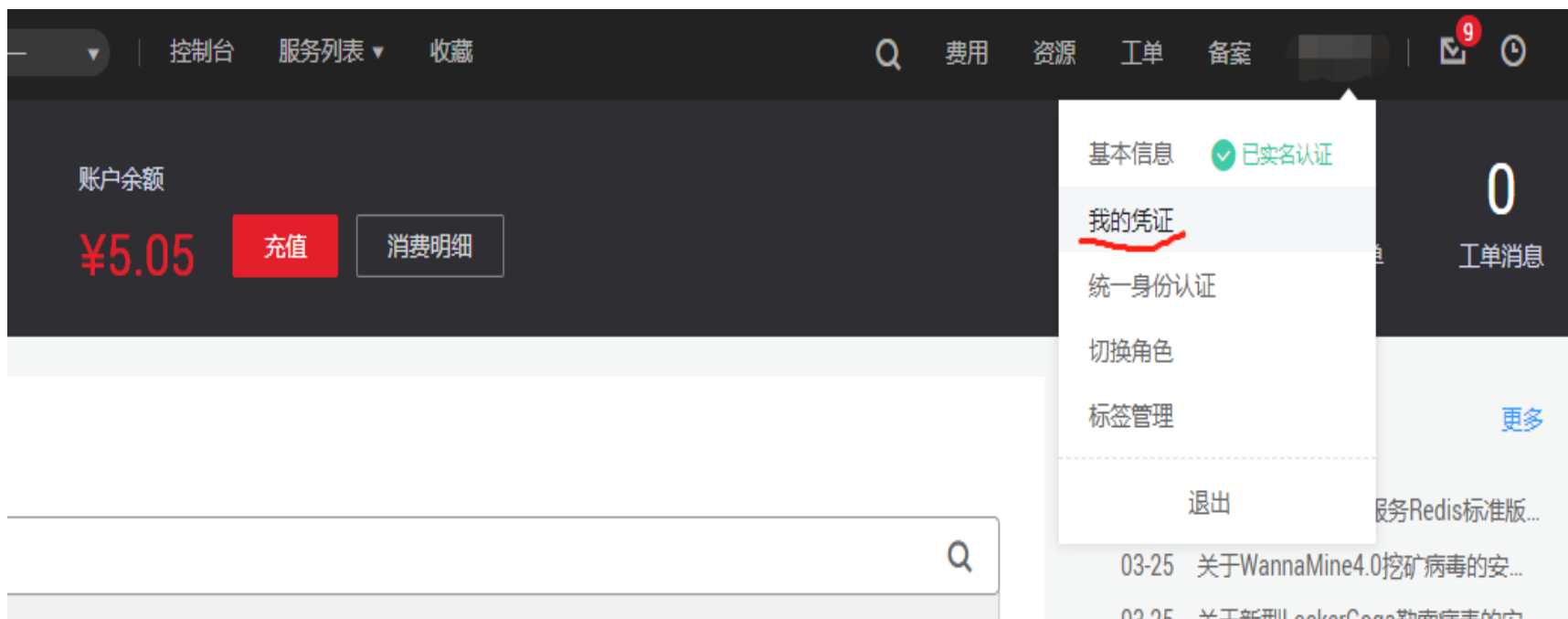
## 准备数据

- 建立凭证
- 开启OBS服务
- 上传数据



# 建立凭证

凭证：访问对象存储服务所需的KEY，相当于账号和密码，通过凭证管理页面，由系统自动下发给用户



华为云地址：<https://console.huaweicloud.com/modelarts/?region=cn-north-1#/manage/dashboard>



# 建立凭证

请妥善储存备份自己的凭证

我的凭证



更换

用户名

[Redacted]

用户ID

[Redacted]

账号名

[Redacted]

账号ID

[Redacted]

已验证邮箱

[Redacted]

修改

已验证手机

[Redacted]

修改

密码

[Redacted] 安全程度强

修改

登录验证方式

关闭

修改

[Redacted]

项目列表

管理访问密钥

访问密钥对账号具有完全的访问权限，如果访问密钥泄露，会带来数据泄露风险，为了账号安全性，建议您定期更换并妥善保存访问密钥。

新增访问密钥

您还可以添加0个访问密钥。

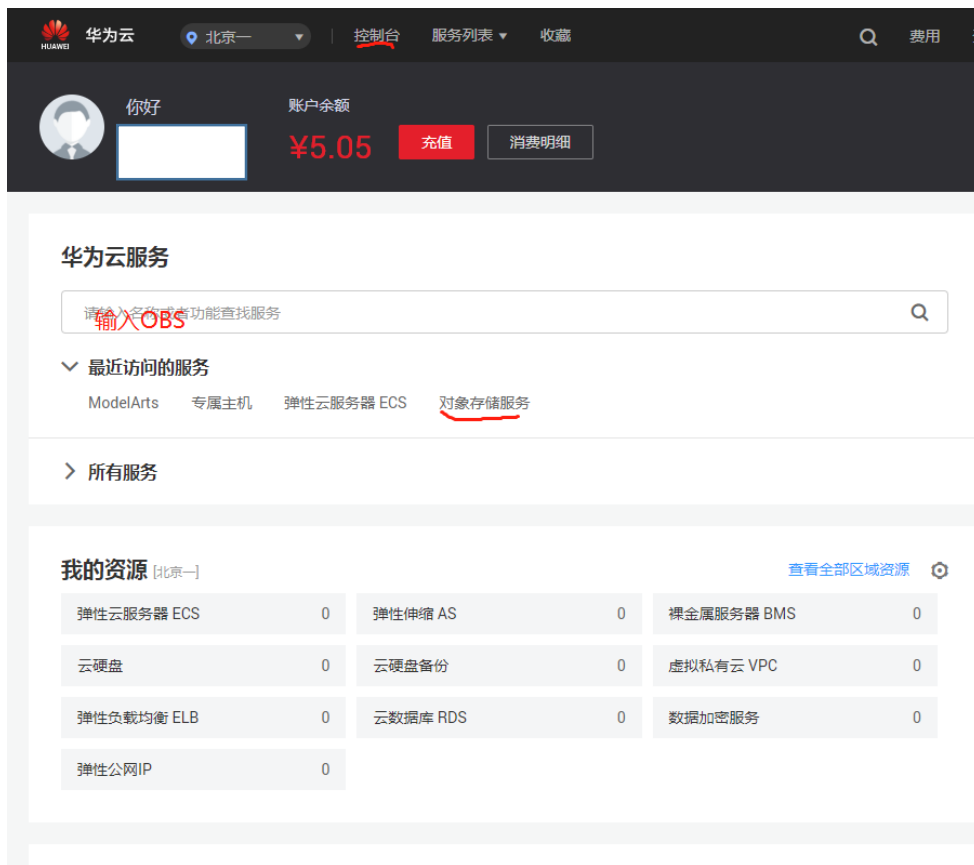
请输入访问密钥

访问密钥ID	创建时间	状态	操作
	2019/03/11 18:51:24 GMT+08:00	启用	删除
	2019/03/26 19:43:23 GMT+08:00	启用	删除



# 开启OBS服务

1. 先选择控制台，然后搜索OBS，选择对象存储服务。







# 开启OBS服务

2. 创建桶，并选择一个客户端下载  
(使用客户端时可以上传文件夹到数据桶,而在网页时只可以上传文件)

任务 ②

+ 创建桶

购买资源包

存储服务客户端，可以非常方便的让您在个人电脑上完成对象存储管理。[Windows版本下载\(32位\)](#) [Windows版本下载\(64位\)](#) [Mac版本下载](#)  
提供的命令行工具，可以并发处理上传下载任务，实现快速传输数据能力。  
发包，您可以轻松构建基于对象存储的互联网应用。 [获取SDK](#)  
，请先[获取访问密钥\(AK和SK\)](#)。  
[更多更完整的帮助信息。](#)

查看资源包

请输入桶名称



类别	区域	桶权限	存储用量	对象数量	创建时间	操作
字储	华北-北京一	私有	0 byte	1	2019/03/26 17:02:41 GMT+08:00	<a href="#">修改存储类别</a> <a href="#">删除</a>
字储	华北-北京一	私有	157.70 MB	489	2019/03/26 17:01:32 GMT+08:00	<a href="#">修改存储类别</a> <a href="#">删除</a>



# 开启OBS服务

注:在创建桶时请选择华北·北京一

## 创建桶

[返回桶列表](#)

区域

华北·北京一

不同区域的资源之间内网互不相通，请选择靠近您业务的区域，可以降低网络时延，提高访问速度。桶创建成功后不支持变更区域，请谨慎选择。

桶名称

obs-1446

命名规则:

- 需全局唯一，不能与已有的任何桶名称重复。
- 长度范围为3到63个字符，支持小写字母、数字、中划线(-)、英文句号(.)。
- 禁止两个英文句号(.)或英文句号(.)和中划线(-)相邻，禁止以英文句号(.)和中划线(-)开头或结尾。
- 禁止使用IP地址。
- 如果名称中包含英文句号(.)，访问桶或对象时可能会进行安全证书校验。

存储类别

标准存储

低频访问存储

归档存储

适用于有大量热点文件或小文件，且需要频繁访问（平均一个月多次）并快速获取数据的业务场景。

上传对象时，对象默认与桶的存储类别相同，也可以根据适用场景修改。[了解更多](#)

桶策略

私有

公共读

公共读写

桶的拥有者拥有完全控制权，其他用户在未经授权的情况下均无访问权限。

多AZ

当前仅【华东·上海二,华北·北京四,西南·贵阳一】区域支持配置多AZ。

标签

如果您需要使用同一标签标识多种云资源，即所有服务均可在标签输入框下拉选择同一标签，建议在TMS中创建预定义标签。[查看预定义标签](#)

标签键

标签值

您还可以添加10个标签。

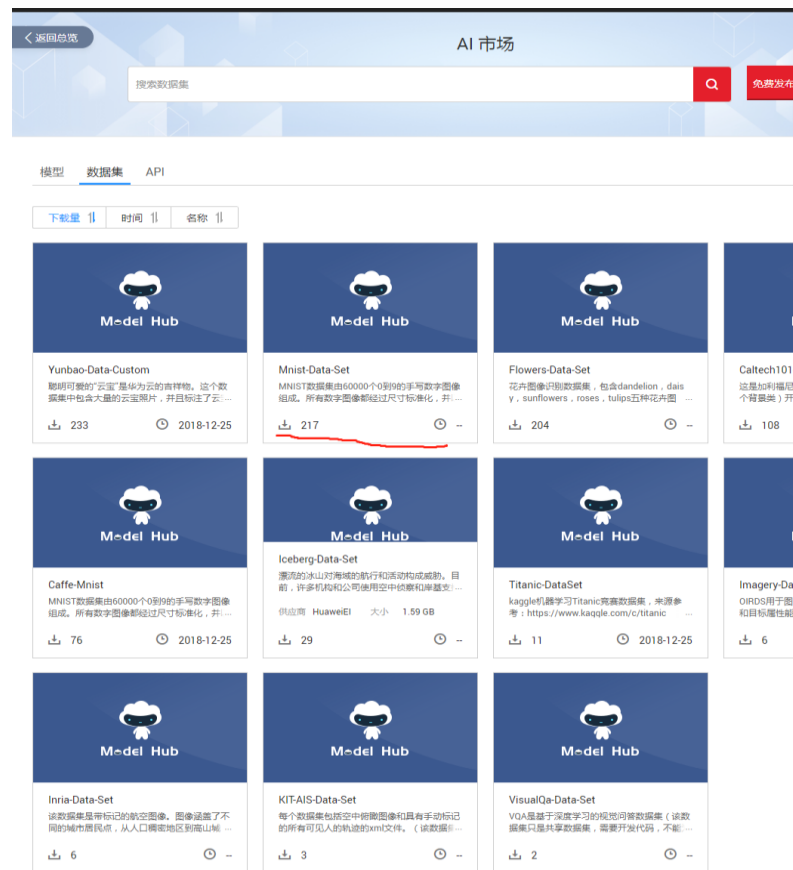




# 上传或导入数据

登录客户端上传数据，或者通过ModelArts的市场导入数据：

在“ModelArts”管理控制台的“数据管理>数据集”页面查看找到mnist数据集(Mnist-Data-Set)创建完成本例中为桶名为“mnist-example”，将数据集保存到/mnist-example/mnist/路径下



注：在客户端显示的桶中看到转入或上传的数据表明操作成功



# 02

## 调试代码

- 建立调试环境
- 修改调试代码
- 查看运行情况



# 打开ModelArts服务

- 打开控制台，搜索ModelArts服务

华为云

北京一

控制台

服务列表

收藏

Q

费用

你好

账户余额

¥5.05

充值

消费明细

华为云服务

请输入名称或者功能查找服务

最近访问的服务

ModelArts

对象存储服务

专属主机

弹性云服务器 ECS

所有服务

我的资源 北京一

查看全部区域资源

弹性云服务器 ECS	0	弹性伸缩 AS	0	裸金属服务器 BMS	0
云硬盘	0	云硬盘备份	0	虚拟私有云 VPC	0
弹性负载均衡 ELB	0	云数据库 RDS	0	数据加密服务	0
弹性公网IP	0				

云监控 北京一

ECS资源监控

告警中 0

CPU使用率 0.00%



# 建立调试环境

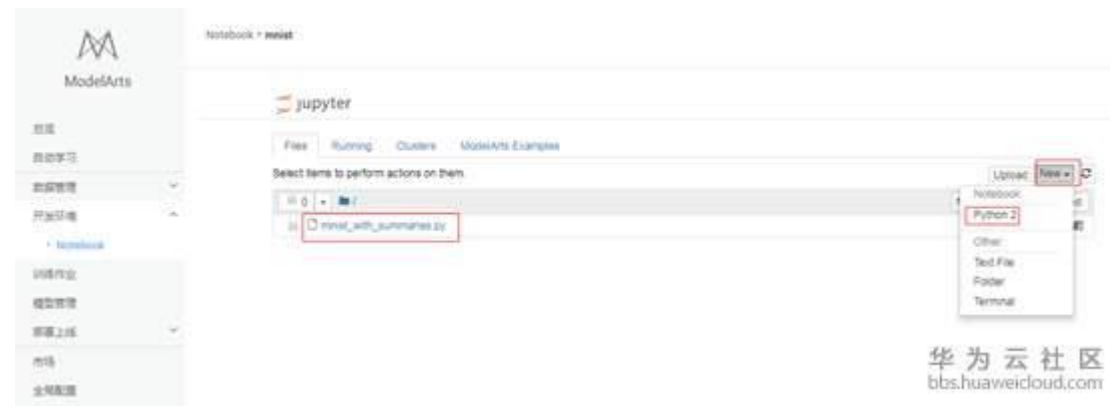
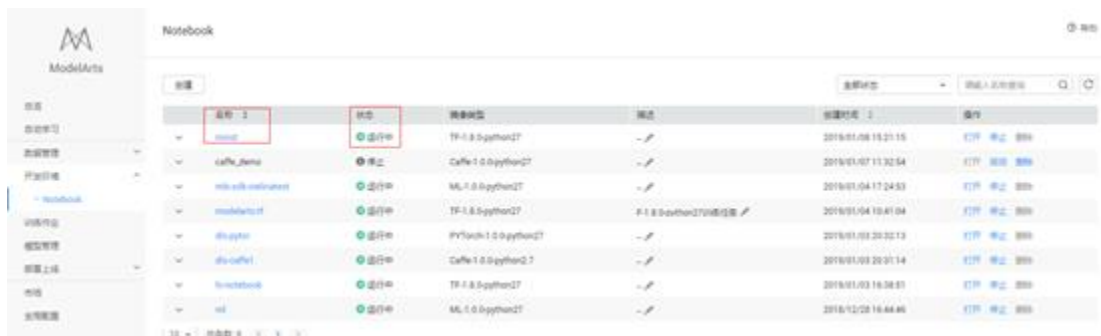
1. 下载模型训练脚本文件 [https://github.com/tensorflow/tensorflow/blob/master/tensorflow/examples/tutorials/mnist/mnist\\_with\\_summaries.py](https://github.com/tensorflow/tensorflow/blob/master/tensorflow/examples/tutorials/mnist/mnist_with_summaries.py)。将脚本文件上传至华为云OBS桶（假设OBS桶路径为：/mnist-example/codes/）。注：也可以在建立调试环境后，复制代码粘贴到环境中的ipynb中。
2. 在“开发环境”界面，单击左上角的“创建”，“名称”和“描述”可以随意填写，本例中名称为“mnist”；“镜像类型”选择TF-1.8.0-python27或TF-1.8.0-python36，“存储位置”选择/mnist-example/codes/，任务提交成功后返回Notebook列表。





# 建立调试环境

- 状态显示为“运行中”时表示创建成功，点击名称“mnist”进入Notebook界面，点击右上角的“new” -> “Python”，并打开mnist\_with\_summaries.py文件将代码复制到cell框中。





# 修改调试代码

数据存储在OBS中，需要在原生tensorflow代码中配置AK，SK，才可以访问。在代码中添加一下内容

```
from __future__ import absolute_import
from __future__ import division
from __future__ import print_function

import argparse
import os
import sys

import tensorflow as tf

from tensorflow.examples.tutorials.mnist import input_data

FLAGS = None

import moxing as mox
S3_SECRET_ACCESS_KEY = (os.environ.get('SECRET_ACCESS_KEY', None)
                        or os.environ.get('S3_SECRET_ACCESS_KEY', None)
                        or os.environ.get('AWS_SECRET_ACCESS_KEY', None))
S3_ACCESS_KEY_ID = (os.environ.get('ACCESS_KEY_ID', None)
                    or os.environ.get('S3_ACCESS_KEY_ID', None)
                    or os.environ.get('AWS_ACCESS_KEY_ID', None))
mox.file.set_auth(ak=_S3_ACCESS_KEY_ID, sk=_S3_SECRET_ACCESS_KEY)

def train():
    # Import data
    mnist = input_data.read_data_sets(FLAGS.data_dir,
                                       fake_data=FLAGS.fake_data)
```





# 修改调试代码

创建训练作业时，训练作业界面的训练数据集路径是以train\_url参数传入，训练输出路径以train\_url传入，脚本的参数改动部分如图：

需要注意此时修改的parser.add\_argument下的两处default后的地址需要根据自己先前创建的数据桶的名称做对应修改，mnist-example改为自己创建的桶的名称。

```
if __name__ == '__main__':
    parser = argparse.ArgumentParser()
    parser.add_argument('--fake_data', nargs='?', const=True, type=bool,
                        default=False,
                        help='If true, uses fake data for unit testing.')
    parser.add_argument('--max_steps', type=int, default=1000,
                        help='Number of steps to run trainer.')
    parser.add_argument('--learning_rate', type=float, default=0.001,
                        help='Initial learning rate')
    parser.add_argument('--dropout', type=float, default=0.9,
                        help='Keep probability for training dropout.')
    parser.add_argument(
        '--data_dir',
        type=str,
        default=os.path.join(os.getenv('TEST_TMPDIR', '/tmp'),
                              'tensorflow/mnist/input_data'),
        help='Directory for storing input data')
    parser.add_argument(
        '--log_dir',
        type=str,
        default=os.path.join(os.getenv('TEST_TMPDIR', '/tmp'),
                              'tensorflow/mnist/logs/mnist with summaries'),
        help='Summaries log directory')
    parser.add_argument(
        '--data_url',
        type=str,
        default='s3://mnist-example/mnist',
        help='Directory for storing input data')
    parser.add_argument(
        '--train_url',
        type=str,
        default='s3://mnist-example/log',
        help='Summaries log directory')
    FLAGS, unparsed = parser.parse_known_args()
    FLAGS.log_dir = FLAGS.train_url
    FLAGS.data_dir = FLAGS.data_url
    tf.app.run(main=main, argv=[sys.argv[0]] + unparsed)
```





# 运行代码查看结果

点击run运行代码，如果代码修改成功，则会显示正确率和LOSS

```
tensorflow/mnist/input_data',  
    help='Directory for storing input data')  
parser.add_argument(  
    '--log_dir',  
    type=str,  
    default=os.path.join(os.getenv('TEST_TMPDIR', '/tmp'),  
                          'tensorflow/mnist/logs/mnist_with_summaries'),  
    help='Summaries log directory')  
parser.add_argument('--data_url',  
                    type=str,  
                    default='s3://obs-lian/MNIST/DATA/',  
                    help='Directory for storing input data')  
parser.add_argument('--train_url',  
                    type=str,  
                    default='s3://obs-lian/MNIST/log/',  
                    help='Summaries log directory')  
FLAGS, unparsed = parser.parse_known_args()  
FLAGS.log_dir = FLAGS.train_url  
FLAGS.data_dir = FLAGS.data_url  
  
tf.app.run(main=main, argv=[sys.argv[0]] + unparsed)
```

```
Accuracy at step 70: 0.9015  
Accuracy at step 80: 0.9095  
Accuracy at step 90: 0.9095  
Adding run metadata for 99  
Accuracy at step 100: 0.9075  
Accuracy at step 110: 0.911  
Accuracy at step 120: 0.9123  
Accuracy at step 130: 0.9225  
Accuracy at step 140: 0.9187  
Accuracy at step 150: 0.9245  
Accuracy at step 160: 0.9274  
Accuracy at step 170: 0.9306  
Accuracy at step 180: 0.9327  
Accuracy at step 190: 0.9353  
Adding run metadata for 199  
Accuracy at step 200: 0.9353  
Accuracy at step 210: 0.9333  
Accuracy at step 220: 0.9362  
Accuracy at step 230: 0.9375
```



# 03

## 训练模型

- 建立训练作业
- 建立TensorBoard



# 建立训练作业

训练作业中，原生tensorflow代码中无需配置ak，sk，可去除配置ak，sk代码（保留亦无影响）。

在“训练作业”界面，单击左上角的“创建”，“名称”和“描述”可以随意填写；

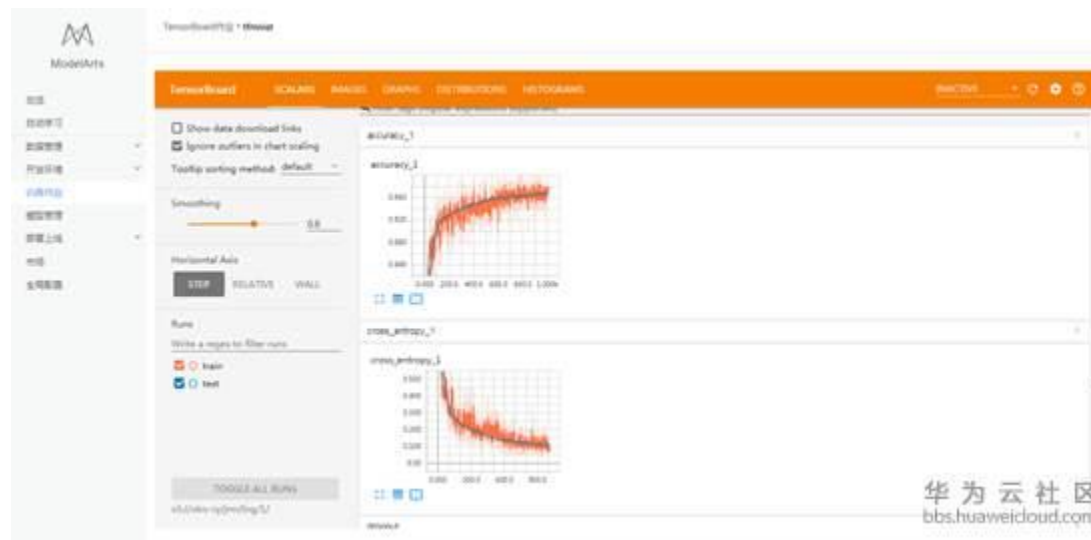
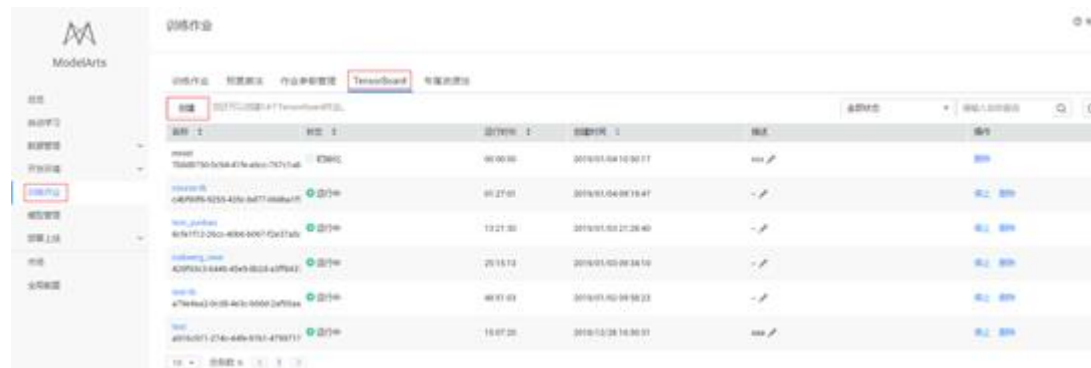
“数据来源”请选择“数据集”Mnist-Data-Set{或者“数据的存储位置”（本例中为mnist-example/mnist)}；“算法来源”请选择“常用框架”，“AI引擎”选择“TensorFlow”；“代码目录”请选择型训练脚本文件mnist\_with\_summaries.py所在的OBS父目录（/mnist-example/codes/）；“启动文件”请选择“train\_mnist.py”；“训练输出位置”请选择一个路径（例如/mnist-example/log/）用于保存输出模型和预测文件，参考下图填写训练作业参数。



# 建立训练作业

在模型训练的过程中或者完成后，通过创建TensorBoard作业查看一些参数的统计信息，如loss， accuracy等。在“训练作业”界面，点击TensorBoard，再点击“创建”按钮，参数“名称”可随意填写，“日志路径”请选择步骤3中“训练输出位置”参数中的路径（/mnist-example/log/），或者直接进入训练作业界面点击作业名称，点击右上角的“创建TensorBoard”

训练作业完成后，即完成了模型训练过程。如有问题，可点击作业名称，进入作业详情界面查看训练作业日志信息。





# 注意事项

- 1.给出的实验示例是Tensorflow框架，运行环境是python3.6版。
- 2.用完开发环境调试代码后记得停止开发环境运行，实测会计时。



# 04

## 后续任务

- 尝试使用 pytorch 框架在 ModelArts上完成MNIST
- 尝试自己使用自定义框架完成MNIST
- 尝试学习模型管理和部署上线功能



# 作业提交

- 提交内容：实验报告（关于本次华为云的使用）
- Deadline：2019.4.8