计算机科学与技术学院神经网络与深度学习课程实验报告

实验题目：华为云 Tensorflow 实现手写数字识别		学号：202020130190
日期：2022.10.31	班级： 人工智能	姓名：刘绪波
Email：2842353032@qq.com		

实验目的：

通过本次实验的进行，熟练掌握如何利用华为云资源来进行训练和检测模型，如何在华为云上部署模型，并了解如何利用 tensorflow 来实现手写数字识别；

实验软件和硬件环境：

华为云

实验原理和方法：

Tensorflow 实现手写数字识别:

对于训练的 main 函数：输入定义，模型定义和运行;

输入定义：input_fn(run_mode, **kwargs) 用户可以根据自己的输入编写。本例中通过迭代的方式从数据集中取数据。

模型定义：def model_fn(inputs, run_mode, **kwargs): 模型结构定义函数，返回 mox.ModelSpec()，用户作业模式定义返回值。 但需要满足如下条件：

For run_mode == ModeKeys.TRAIN: loss is required.

For run_mode == ModeKeys.EVAL: log_info is required.

For run_mode == ModeKeys.PREDICT: output_info is required.

For run_mode == ModeKeys.EXPORT: export_spec is required.、

运行：mox.run()，训练的过程中可指定 optimizer 的一些设置，训练 batch 的大小等，设置内容如下：

输入函数， input_fn: An input_fn defined by user. Allows tfrecord or python data. Returns input tensor list.

模型函数， model_fn: A model_fn defined by user. Returns mox.ModelSpec.

optimizer 定义， optimizer_fn: An optimizer_fn defined by user. Returns an optimizer.

运行模式选择， run_mode: Only takes mox.ModeKeys.TRAIN or mox.ModeKeys.EVAL or mox.ModeKeys.PREDICT

batch 大小设置， batch_size: Mini-batch size.

是否自动化 batch， auto_batch: If True, an extra dimension of batch_size will be expanded to the first dimension of the return value from get_split. Default to True.

日志以及 checkpoint 保存位置， log_dir: The directory to save summaries and checkpoints.

最大数量， max_number_of_steps: Maximum steps for each worker.

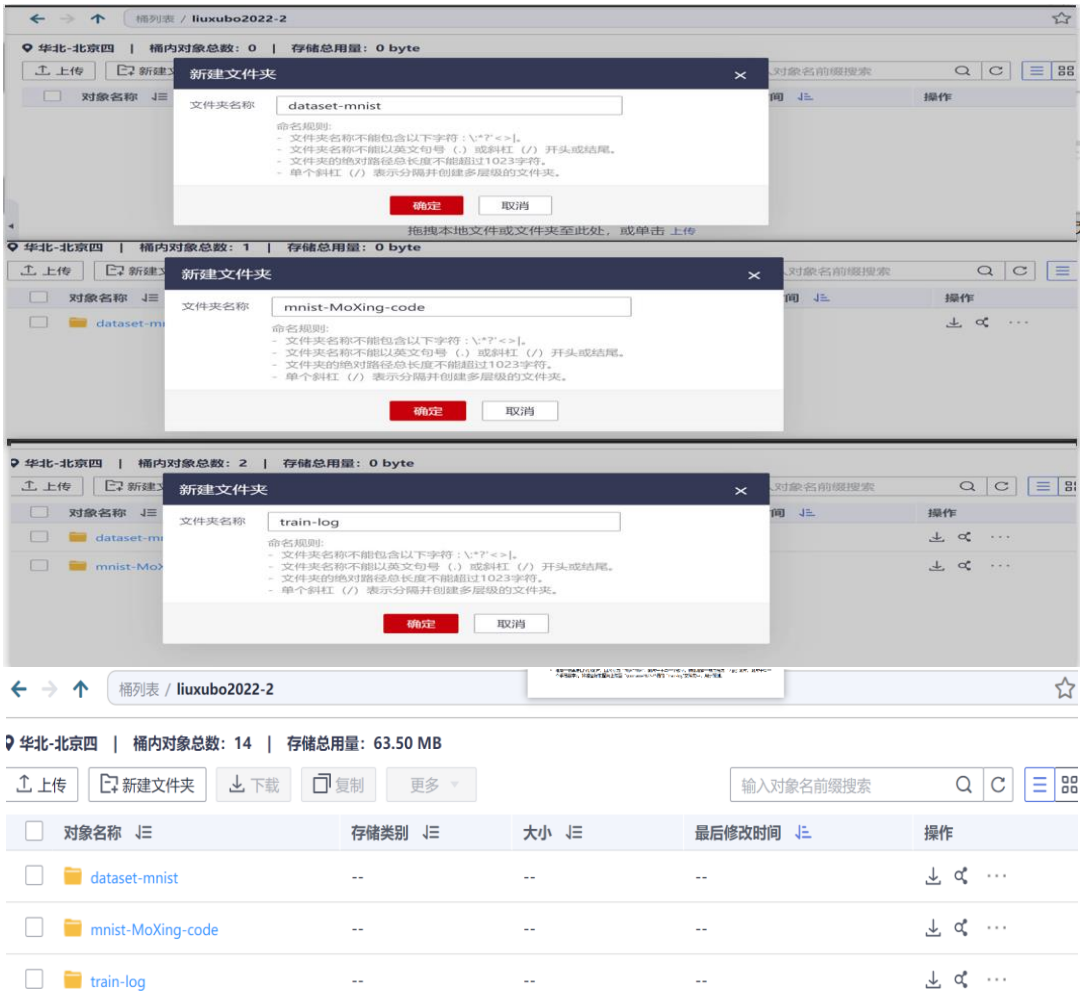
日志打印， log_every_n_steps: Step period to print logs to std I/O.

是否输出模型， export_model: True or False. Where to export model after running the job

实验步骤：（不要求罗列完整源代码）

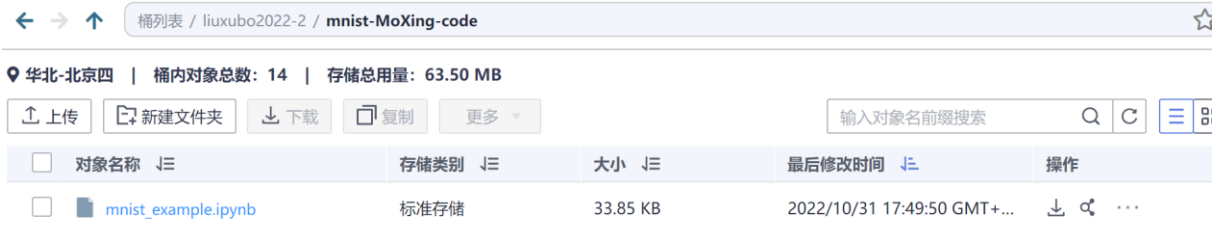
1. 准备工作：

① 创建桶和文件夹：创建过程见下图：



② 上传代码：

若按照正常程序应该从 github 上 clone 到本地然后利用 obs+上传(当然此处用网页版的桶管理工具也可以但是不支持文件夹的传输，因此用软件客户端 obs+上传)；

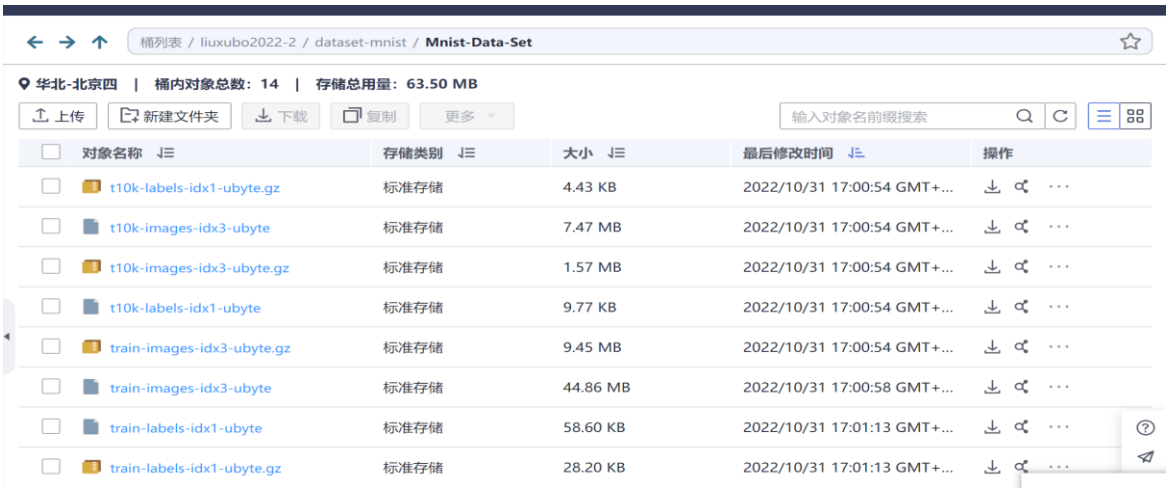


③ 准备好图片：

准备一张黑底白字的图片，且尺寸为“28px*28px”，图片中手写一个数字。例如准备一张命名为“7.jpg”图片，图片中有一 个手写数字 7。将准备好的图片上传至“liuxubo2022-2”桶的“train-log”文件夹中，用于预测。

2. 准备数据：

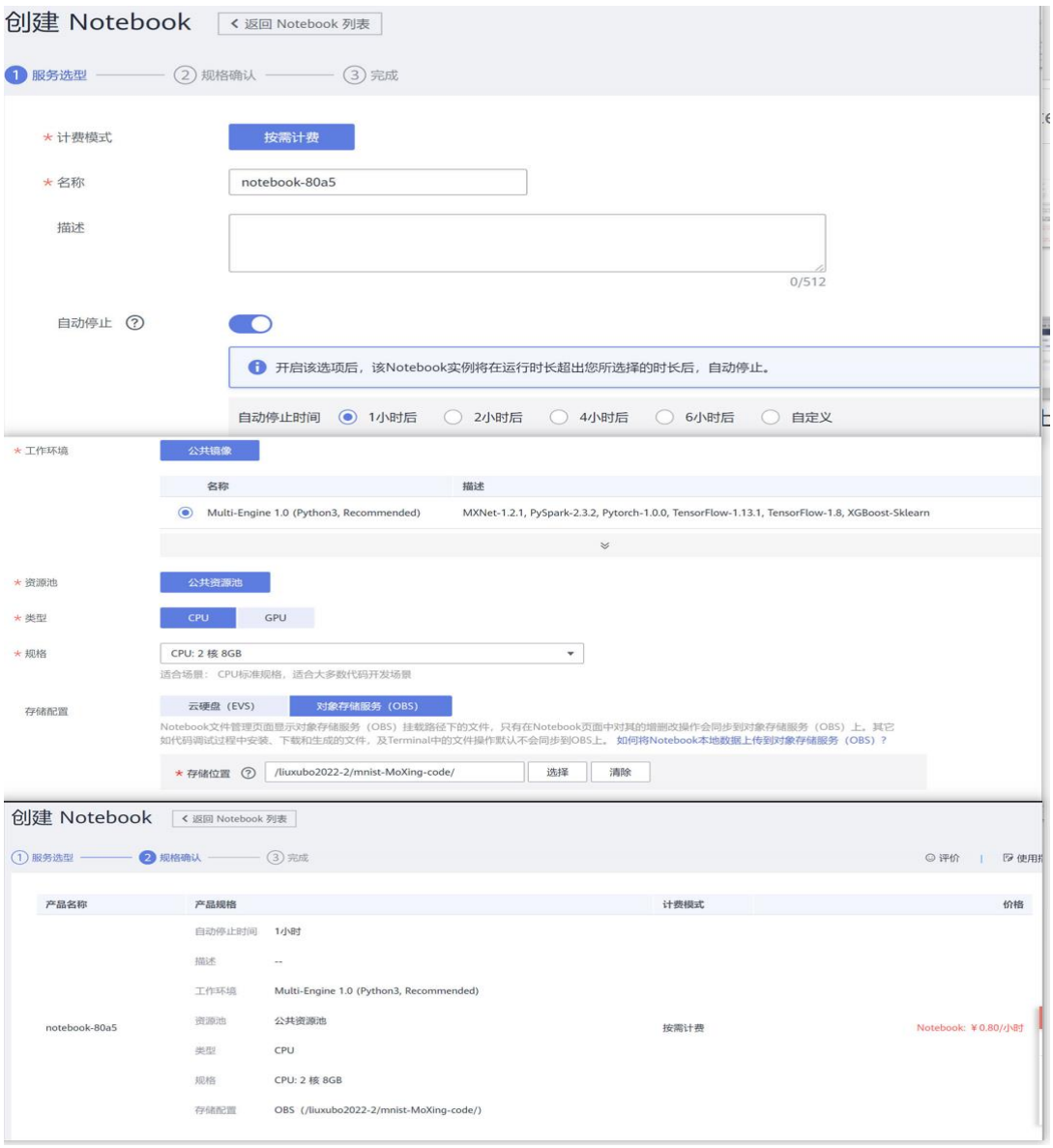
ModelArts 在公共 OBS 桶中提供了 MNIST 数据集，命名为“MnistData-Set”将数据集上传至您的 OBS 目录下，即准备工作中创建的 OBS 目录“liuxubo2022-2 /dataset-mnist 使用 notebook 训练模型并预测：



3. 预测：

① 在 ModelArts 管理控制台，进入“开发环境>Notebook”页面，单击 “返回旧版”；进入创建页面

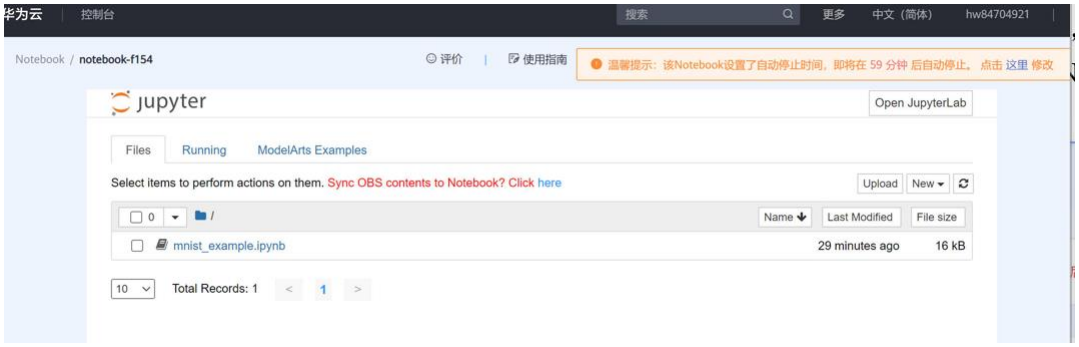
② 在“创建 Notebook”页面，填写相关信息，“存储配置”请选择“对象存储服务“ ， 并在 “存储位置”选择示例文件存储的 OBS 路径，例如 “liuxubo2022-2 /mnistMoXing-code”。



③ 在“规格确认”页面，确认信息无误后，单击“提交”

④ 在“Notebook”管理页面，当新建的 Notebook 状态变为“运行中”时，表示 Notebook 已创建完成。单击操作列的“打开”，进入“Jupyter”页面。

⑤ 在“Jupyter”页面的“Files”页签下，您可以看到步骤 1 上传的示例代码文件。 单击文件名称，进入 Notebook 详情页。在详情页根据页面提示，选择和代码 环境相匹配的 Kernel 环境，本示例使用的 Kernel 为“TensorFlow-1.8”。如果界 面无此提示，可不进行 Kernel 环境设置，直接执行下一步。



⑥ 在 Notebook 详情页，示例代码文件已提供了详细的描述，包含“数据准备”、“训练模型”和“预测”。

a) 数据准备：步骤 1：准备数据已完成数据准备，数据集所在路径为“liuxubo2022-2/dataset-mnist/”。示例代码提供了数据集的介绍说明。

b) 训练模型:

在训练模型区域，将“data_url”修改为步骤 1：准备数据中数据集所在 OBS 路径，您可以从 OBS 管理控制台拷贝 OBS 路径，并将 OBS 路径修改为“s3://”格式。如下图:

```
####### your coding place:  begin  ###########
# 此处必须修改为用户数据桶位置

#数据在OBS的存储位置。
# eg. s3:// : 统一路径输入
#     /uBucket : 桶名，用户的私有桶的名称 eg. bucket
#     /notebook/data/:  文件路径

data_url = 's3://liuxubo2022-2/dataset-mnist/Mnist-Data-Set/'

####### your coding place:  end  ###########
```

```
train_url = './cache/log/'          #训练输出位置。
if not mox.file.exists(data_url):
    raise ValueError('Plese verify your data url!')
if mox.file.exists(train_url):
    mox.file.remove(train_url,recursive=True)
mox.file.make_dirs(train_url)
```

通过mox 能够将数据拷贝到本地，这样能够加快训练。操作如下：

```
# 本地创建数据存储文件夹
local_url = './cache/local_data/'
if mox.file.exists(local_url):
    mox.file.remove(local_url,recursive=True)
os.makedirs(local_url)

#将私有桶中的数据拷贝到本地mox.file.copy_parallel()
"""
```

代码修改完成后，从第一个 Cell 开始，单击运行代码，将训练模型区域下的所有 Cell 运行一遍。在训练模 型区域最后，将显示运行日志，当日志出现如下类似信息时，表示模型训练成功。如下日志信息表示模 型训练成功，且模型文件已成功生成。

```
INFO:tensorflow:global_step/sec: 849.901
INFO:tensorflow:step: 900(global step: 900)     sample/sec: 20188.217   loss: 0.507     accuracy: 0.900
INFO:tensorflow:step: 910(global step: 910)     sample/sec: 53389.817   loss: 0.583     accuracy: 0.860
INFO:tensorflow:step: 920(global step: 920)     sample/sec: 52311.100   loss: 0.691     accuracy: 0.860
INFO:tensorflow:step: 930(global step: 930)     sample/sec: 45849.410   loss: 0.491     accuracy: 0.880
INFO:tensorflow:step: 940(global step: 940)     sample/sec: 52639.357   loss: 0.590     accuracy: 0.860
INFO:tensorflow:step: 950(global step: 950)     sample/sec: 56103.585   loss: 0.486     accuracy: 0.940
INFO:tensorflow:step: 960(global step: 960)     sample/sec: 39009.524   loss: 0.560     accuracy: 0.840
INFO:tensorflow:step: 970(global step: 970)     sample/sec: 62303.981   loss: 0.472     accuracy: 0.960
INFO:tensorflow:step: 980(global step: 980)     sample/sec: 60280.310   loss: 0.655     accuracy: 0.820
INFO:tensorflow:step: 990(global step: 990)     sample/sec: 64488.069   loss: 0.592     accuracy: 0.920
INFO:tensorflow:Saving checkpoints for 1000 into ./cache/log/model.ckpt.
INFO:tensorflow:Ignoring --checkpoint_path because a checkpoint already exists in ./cache/log/
INFO:tensorflow:No assets to save.
INFO:tensorflow:No assets to write.
INFO:tensorflow:Restoring parameters from ./cache/log/model.ckpt-1000
INFO:tensorflow:SavedModel written to: b'./cache/log/model/saved_model.pb'
```

c) 预测:

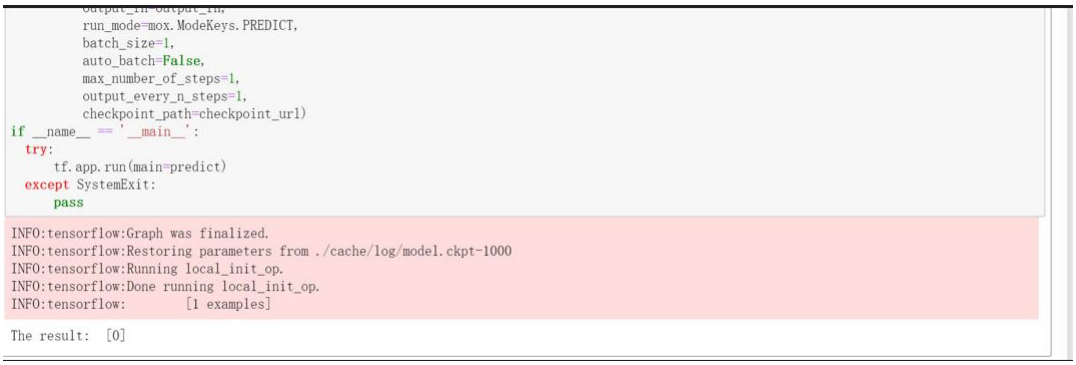
模型训练完成后，可上传一张图片，并使用生成的模型预测。在 Notebook 中，将预测区域的“src_path”修改为图片实际存放的路径和名称。此处请使用 “s3://”格式的 OBS 路径。

```
: ####### your coding place:  begin###########

  #此处必须修改为用户数据存储的OBS位置

  # 预测图片在OBS的存储位置。
  # eg. 图片名称:   image_number.jpg
  #     存储位置为：bucket/test/
  src_path = 's3://liuxubo2022-2/train-log/7.jpg'

  ####### your coding place:  end  ###########
```

```
: # 可以利用moxing 将需要预测的图片从OBS拷贝到本地
  if not mox.file.exists(src_path):
      raise ValueError('Plese verify your src_path!')
  dst_path =  './cache/test.jpg'
  mox.file.copy(src_path,dst_path)
```

d) 代码修改完成后,从第一个 Cell 开始,单击运行代码,将预测区域下的所有 Cell 运行一遍。在预测区域最 后,将显示运行日志,当日志出现如下类似信息时,显示图片预测结果,例如本示例中图片的手写数字 为“7”。请对比图片中的数字和预测结果，判断预测结果是否正确。

呃呃呃。。。。。发现预测错误:



```
            run_mode=mox.ModeKeys.PREDICT,
            batch_size=1,
            auto_batch=False,
            max_number_of_steps=1,
            output_every_n_steps=1,
            checkpoint_path=checkpoint_url)
if __name__ == '__main__':
  try:
      tf.app.run(main=predict)
  except SystemExit:
      pass
```

```
INFO:tensorflow:Graph was finalized.
INFO:tensorflow:Restoring parameters from ./cache/log/model.ckpt-1000
INFO:tensorflow:Running local_init_op.
INFO:tensorflow:Done running local_init_op.
INFO:tensorflow:        [1 examples]

The result:  [0]
```

4. 预测后工作:

为避免产生不必要的费用，在完成试用后，删除相关资源，包含数据和 Notebook。

使用过程中的问题：

1. 出错情况：忘记对 modelarts 进行全局配置；导致无法访问 obs 桶中的文件；
2. 进行新增委托的时候，应该设置为当前用户，即自己的命名，不知道为什么设置为所有用户是并不起效；
3. 注意桶和项目的设置区一致，同为华北-北京四或者其他的；