

哈爾濱工業大學

人工智能实验报告

题目 基于 Mindspore 框架与 ModelArts

平台的 MNIST 手写体识别实验

专业 人工智能

学号 2021112845

姓名 张智雄

一. 问题描述

该实验包含两个部分，基于 Mindspore 框架的模型本地训练及预测和基于 Modelarts 平台和 Tensorflow 框架的模型训练及部署，针对 MNIST 数据集实现一个简单的图片分类的功能。

二. 算法介绍

2.1 基于 Mindspore 框架的模型本地训练

本实验基于 Mindspore 框架，通过模型的本地训练，实现一个简单的图片分类的功能，整体流程如下：

- 1) 处理需要的 MNIST 数据集。
- 2) 定义一个 LeNet 网络和损失函数、优化器
- 3) 加载数据集并进行训练，训练完成后，查看结果及保存模型文件。
- 4) 加载保存的模型，进行推理。
- 5) 验证模型，加载测试数据集和训练后的模型，验证结果精度。

上述使用到的 LeNet 网络是一种经典的卷积神经网络(Convolutional Neural Network, CNN)，由 Yann LeCun 等人于 1998 年提出，LeNet 的设计初衷是用于识别手写数字的邮政编码和地址。LeNet 网络结构相对简单，不包括输入层的情况下，共有 7 层：2 个卷积层、2 个下采样层（池化层）、3 个全连接层，各层次具体功能如下：

a) 卷积层(Convolutional Layer)：每个卷积层包含一个卷积核(filter)，卷积核通过滑动窗口的方式在输入图像上进行卷积运算，提取图像的特征，从而产生一系列的特征图(feature maps)。

b) 池化层(Pooling Layer)：用于降低特征图的空间尺寸，减少参数数量。

c) 全连接层(Fully Connected Layer)：将池化层的输出连接到全连接层，用于进行分类。

在 LeNet 中，最终使用 softmax 激活函数将网络的输出映射为类别概率，选择其中概率最大的选项作为分类结果。

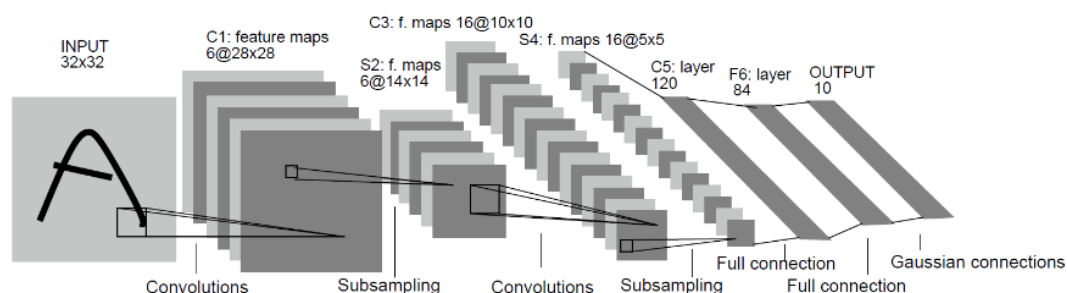


图 1 LeNet 网络结构

2.2 基于 Modelarts 平台和 Tensorflow 框架的模型训练及部署

本实验基于 Modelarts 平台和 Tensorflow 框架，通过模型的在华为云服务器上的训练，实现手写数字图像识别，了解如何在 ModelArts 平台上训练作业、部署推理模型并预测的完整流程。操作流程如下：

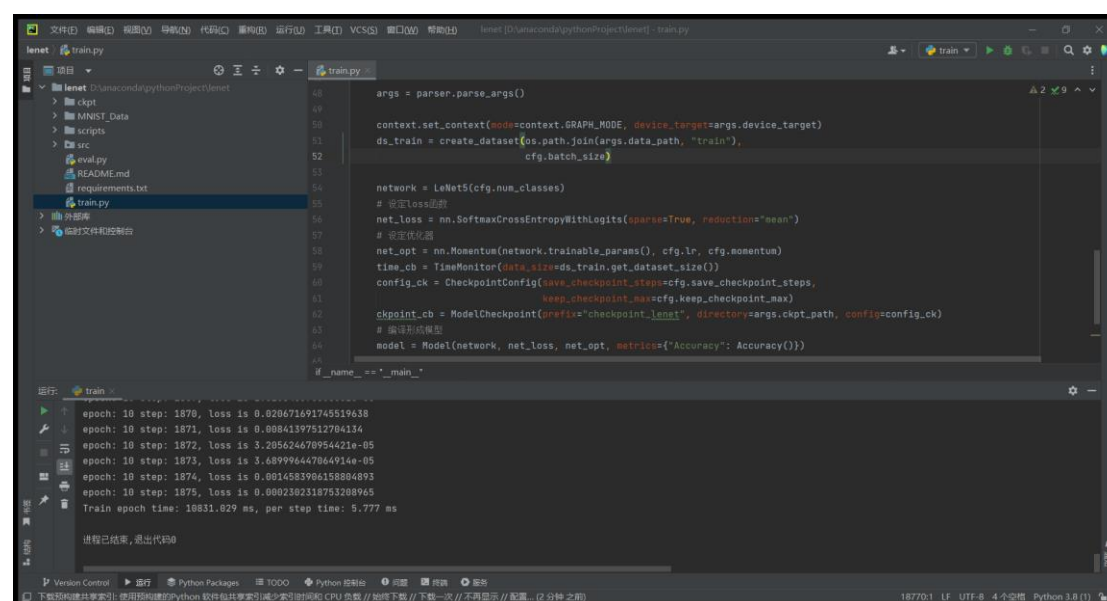
- 1) 准备训练数据：下载 MNIST 数据集。
- 2) 准备训练文件和推理文件：编写训练与推理代码。
- 3) 创建 OBS 桶并上传文件：创建 OBS 桶和文件夹，并将数据集和训练脚本、推理脚本、推理配置文件上传到 OBS 中。
- 4) 创建训练作业：进行模型训练。
- 5) 推理部署：训练结束后，将生成的模型导入 ModelArts 用于创建 AI 应用，并将 AI 应用部署为在线服务。
- 6) 预测结果：上传一张手写数字图片，发起预测请求获取预测结果。

TensorFlow 是由 Google Brain 团队开发的开源的深度学习框架，用于构建和训练各种机器学习模型，尤其是神经网络模型。TensorFlow 能够实现大规模分布式计算的高性能，并且支持在各种硬件平台上进行部署，如 CPU、GPU 和 TPU(Tensor Processing Unit)等。

三. 算法实现

3.1 基于 Mindspore 框架的模型本地训练

首先是模型的训练，可以发现 loss 值在波动中，逐步减小，精度逐步提高。同时 loss 值有一定随机性，每一次运行结果都不完全相同。



The screenshot shows a code editor with a Python script for training a LeNet5 model. The script includes imports for argparse, context, dataset creation, and model building. It sets up a training context, creates a dataset, and defines a LeNet5 network with a loss function and optimizer. The training process is monitored using a TimeMonitor and a CheckpointConfig. The output shows the training progress over 10 epochs, with the loss decreasing from approximately 0.82 to 0.00023.

```
args = parser.parse_args()

context.set_context(mode=context.GRAPH_MODE, device_target=args.device_target)
ds_train = create_dataset(os.path.join(args.data_path, "train"),
                           cfg.batch_size)

network = LeNet5(cfg.num_classes)
# 设置loss函数
net_loss = nn.SoftmaxCrossEntropyWithLogits(sparse=True, reduction="mean")
# 设置优化器
net_opt = nn.Momentum(network.trainable_params(), cfg.lr, cfg.momentum)
time_cb = TimeMonitor(data_size=ds_train.get_dataset_size())
config_ck = CheckpointConfig(save_checkpoint_steps=cfg.save_checkpoint_steps,
                              keep_checkpoint_max=cfg.keep_checkpoint_max)
ckpt_cb = ModelCheckpoint(prefix="checkpoint-leet", directory=args.ckpt_path, config=config_ck)
# 编译和训练模型
model = Model(network, net_loss, net_opt, metrics={"Accuracy": Accuracy()})

if __name__ == '__main__':
```

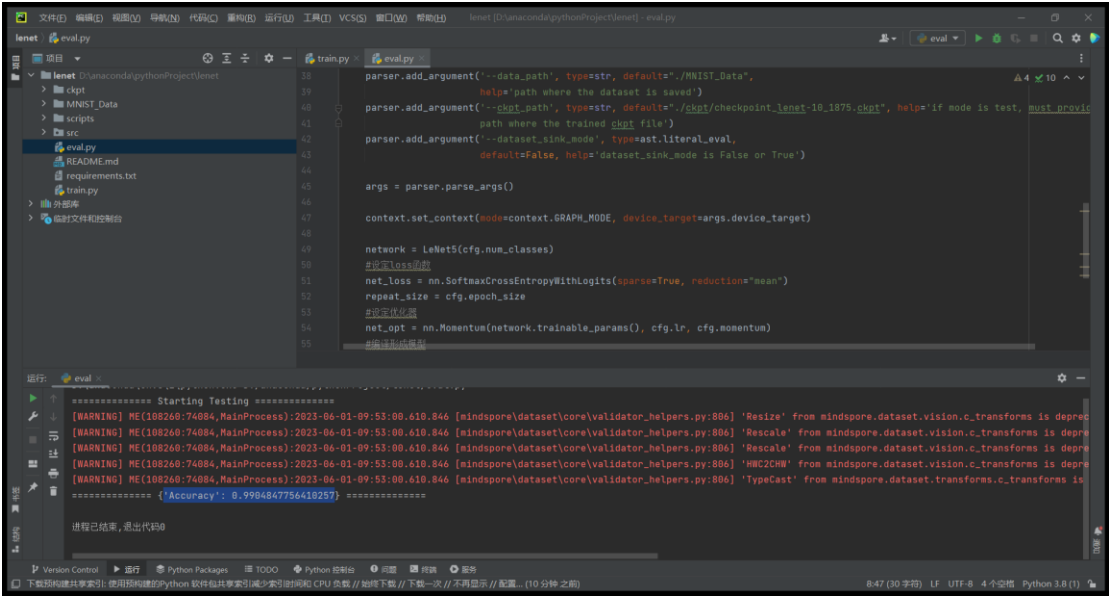
训练输出:

```
epoch: 10 step: 1870, loss is 0.820671691745519638
epoch: 10 step: 1871, loss is 0.80841397512784134
epoch: 10 step: 1872, loss is 3.285624678954421e-05
epoch: 10 step: 1873, loss is 3.689996447864914e-05
epoch: 10 step: 1874, loss is 0.8014583986158884893
epoch: 10 step: 1875, loss is 0.0002302318753208965
Train epoch time: 10831.029 ms, per step time: 5.777 ms

训练已结束,退出代码0
```

图 2 训练过程

而后进行模型的验证，模型在测试集上的分类正确率为 99.05%，说明模型能较好的完成手写数字的分类识别。



```
eval.py
18 parser.add_argument('--data_path', type=str, default='./MNIST_Data',
19                     help='path where the dataset is saved')
20 parser.add_argument('--ckpt_path', type=str, default='./ckpt/checkpoint_lenet_10_1075.ckpt', help='if mode is test, must provide
21                     path where the trained ckpt file')
22 parser.add_argument('--dataset_sink_mode', type=ast.literal_eval,
23                     default=False, help='dataset_sink_mode is False or True')
24
25 args = parser.parse_args()
26
27 context.set_context(mode=context.GRAPH_MODE, device_target=args.device_target)
28
29 network = LeNet5(cfg.num_classes)
30 #设置Loss函数
31 net_loss = nn.SoftmaxCrossEntropyWithLogits(sparse=True, reduction='mean')
32 repeat_size = cfg.epoch_size
33 #设置优化器
34 net_opt = nn.Momentum(network.trainable_params(), cfg.lr, cfg.momentum)
35 #编译生成模型
```

图 3 模型验证

3.2 基于 Modelarts 平台和 Tensorflow 框架的模型训练及部署

按照 2.2 节中描述的过程在 Modelarts 平台上进行操作，而后上传一张手写数字图片，查看预测结果如下。

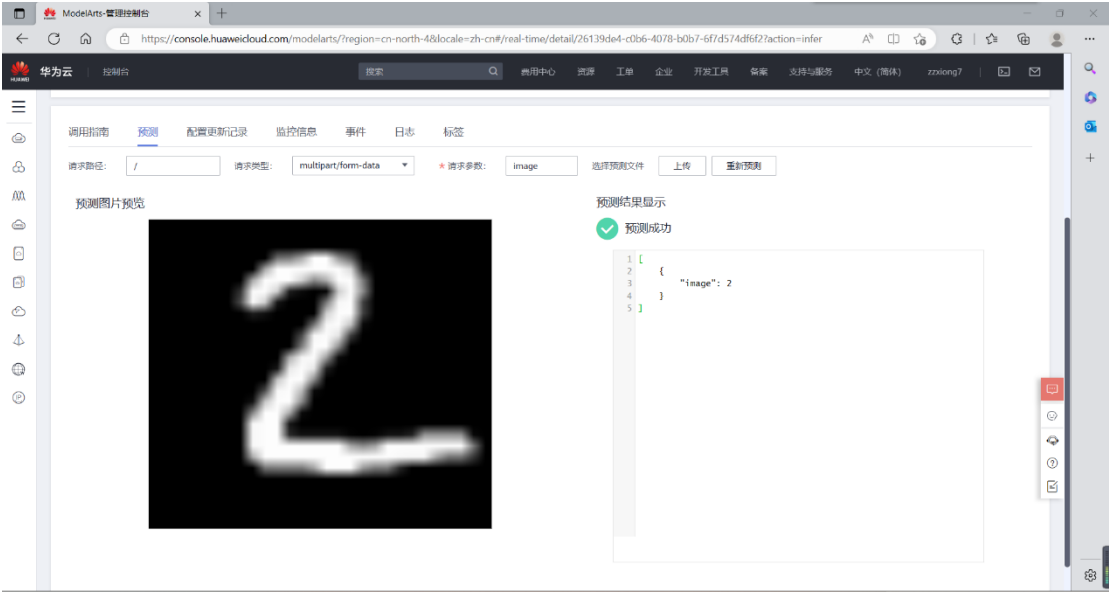


图 4 模型预测

四. 讨论及结论

本实验熟悉了如何在本地基于Mindspore框架训练LeNet网络识别手写数字；了解了如何在ModelArts平台上训练作业、部署推理模型并预测的完整流程。对机器学习的简单模型如LeNet网络结构有了简单的认识，对模型的构建过程有了基本的复现，同时熟悉了机器学习的训练及验证过程。

参考文献

- [1] 使用自定义算法构建模型（手写数字识别）https://support.huaweicloud.com/bestpractice-modelarts/modelarts_10_0080.html
- [2] LeCun Y. LeNet-5, convolutional neural networks[J]. URL: <http://yann.lecun.com/exdb/lenet>, 2015, 20(5): 14.