卷积神将网络手写数字识别报告

景奕瑞-软2304-20232241467

实验目的

1. 掌握卷积神经网络的基本原理与结构，理解卷积、池化和全连接层在图像特征提取中的作用；
2. 熟悉深度学习框架 MindSpore 的基本使用流程，包括数据加载、模型构建、训练与评估；
3. 实现一个完整的手写数字识别模型，在 MNIST 数据集上完成训练与测试；
4. 掌握模型在实际应用中的推理流程，能够对自定义手写数字图片进行识别；
5. 通过实验深入理解图像分类任务的基本流程与评估方法。

实验背景

手写数字识别是计算机视觉领域的经典任务，广泛应用于邮政编码识别、银行票据处理、表单识别等实际场景。传统方法依赖手工特征提取，而卷积神经网络能够自动从原始像素中学习层次化特征，显著提升识别准确率。

MNIST 数据集是手写数字识别中的标准数据集，包含 0~9 共 10 类手写数字灰度图像，每张图像大小为 28×28 像素。本实验使用 MindSpore 框架构建 CNN 模型，实现对 MNIST 手写数字的自动分类。

实验内容与步骤

数据准备与预处理

使用 MindSpore 内置的 `MnistDataset` 接口加载数据，支持自动下载或从本地路径读取。

包括以下步骤：

- 图像归一化：将像素值从 [0, 255] 缩放到 [0, 1]；
- 标准化：使用 MNIST 数据集的均值 0.1307 和标准差 0.3081 进行标准化；
- 维度转换：将图像从 HWC 格式转换为 CHW 格式；
- 标签类型转换：将标签转换为 int32 类型。

代码如下：

```
def create_dataset(data_path, batch_size=64, repeat_size=1, shuffle=True):
    dataset = ds.MnistDataset(data_path, shuffle=shuffle)

    image_transform = [
        vision.Rescale(1.0 / 255.0, 0.0),
        vision.Normalize(mean=(0.1307,), std=(0.3081,)),
        vision.HWC2CHW()
    ]

    label_transform = transforms.TypeCast(mstype.int32)

    dataset = dataset.map(operations=image_transform, input_columns="image")
    dataset = dataset.map(operations=label_transform, input_columns="label")

    dataset = dataset.batch(batch_size)
```

```
    dataset = dataset.repeat(repeat_size)

    return dataset
```

模型设计

CNN网络设计

本代码构建了一个包含两个卷积层，两个池化层和两个全连接层的CNN模型。具体结构如下

层类型	具体作用
输入层	将数据接收并标准化，同时完成数据图像转换为模型可处理的张量格式
Conv2D	提取图像的初级特征同时实现参数共享与局部连接
ReLU	为网络增加非线性表达能力，同时保持梯度避免梯度消失问题
MaxPool2D	将数据降为找到最显著的特征
Conv2D	将低级特征组合成复杂模式，同时学习数字的部分结构
ReLU	为网络增加非线性表达能力，同时保持梯度避免梯度消失问题
MaxPool2D	进一步将数据将为减少计算的复杂度，同时减少参数数量防止过拟合
Flatten	将多维特征转换为一位向量，为全连接层准备输入格式
FC1	进行全局特征组合，学习特征之间的复杂管理，完成高维特征压缩到低维表示
ReLU	
Dropout	防止过拟合，随机关闭部分神经元，减少神经元之间的相互依赖
FC2	进行分类决策，为Softmax提供输入

实现代码如下：

```
self.conv1 = nn.Conv2d(
            in_channels=1,
            out_channels=32,
            kernel_size=3,
            pad_mode='same',
            weight_init=Normal(0.02)
        )
        self.relu = nn.ReLU()
        self.pool = nn.MaxPool2d(kernel_size=2, stride=2)

        self.conv2 = nn.Conv2d(
            in_channels=32,
            out_channels=64,
            kernel_size=3,
            pad_mode='same',
            weight_init=Normal(0.02)
        )
        self.dropout = nn.Dropout(keep_prob=0.8)
        self.flatten = nn.Flatten()
```

```
        self.fc1 = nn.Dense(7 * 7 * 64, 128)
        self.fc2 = nn.Dense(128, num_classes)
```

```
    def construct(self, x):
        x = self.conv1(x)
        x = self.relu(x)
        x = self.pool(x)

        x = self.conv2(x)
        x = self.relu(x)
        x = self.pool(x)

        x = self.flatten(x)
        x = self.fc1(x)
        x = self.relu(x)
        x = self.dropout(x)
        x = self.fc2(x)
        return x
```

损失函数与优化器

- 损失函数：`SoftmaxCrossEntropyWithLogits`（交叉熵损失）
- 优化器：`Adam`，学习率设为 0.0005

模型应用与测试

模型训练与保存

训练过程中每 epoch 保存一次模型参数到 `./checkpoints` 目录。

自定义手写数字预测

编写 `predict.py` 实现对自定义手写数字图片的预测，预处理流程包括：

- 图像灰度化与缩放至 28×28
- 颜色反转（若背景为白色）
- 归一化与标准化
- 输入模型进行预测

```
def preprocess_image(image_path):
    img = Image.open(image_path).convert("L")  # 转为灰度图

    # 调整大小为 28x28 像素
    img = img.resize((28, 28))

    # 转为numpy数组
    img_array = np.array(img).astype(np.float32)

    # 如果图片是白底黑字（MNIST 是黑底白字），需要反转
    # 检查图片的平均亮度
    if img_array.mean() > 127:  # 如果偏白
        img_array = 255 - img_array  # 反转颜色

    # 归一化到 [0, 1]
```

```
    img_array = img_array / 255.0

    # 应用 MNIST 数据集的标准化参数
    img_array = (img_array - 0.1307) / 0.3081

    # 调整形状为 (batch, channel, height, width)
    img_array = img_array.reshape(1, 1, 28, 28)

    return Tensor(img_array, ms.float32)
```

结果可视化

使用 `read.py` 可从测试集中随机抽取图像保存，便于观察数据分布与模型预测效果。

实验结果与分析

模型性能评估

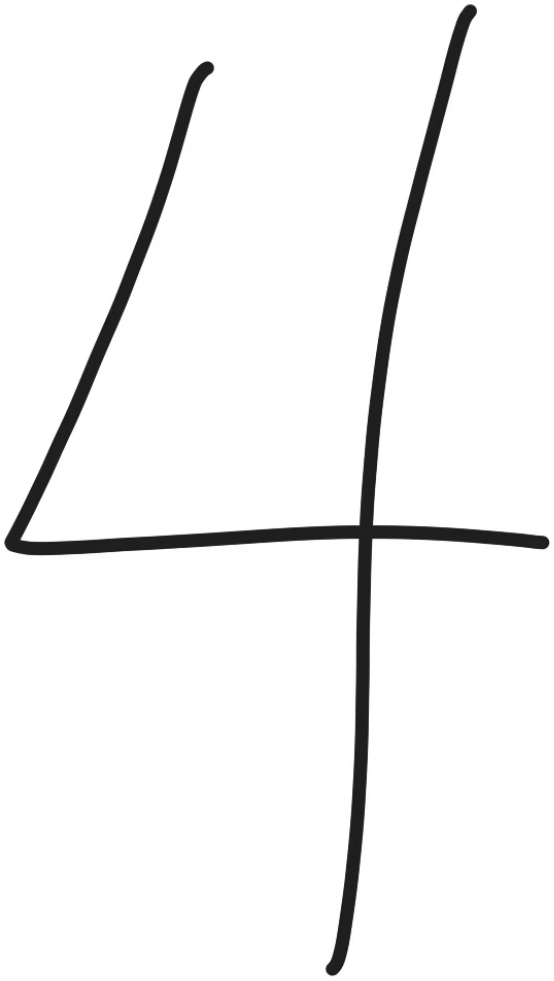
测试集准确率为99.8%,超过目标95%

```
epoch: 10 step: 1093, loss is 0.07058336585760117
epoch: 10 step: 1094, loss is 0.003982314374297857
Train epoch time: 36655.250 ms, per step time: 33.506 ms
========== 开始测试 ==========
[WARNING] ME(7872:20780,MainProcess):2025-12-20-15:12:47.981.000 [mindspore\nn\layer
[WARNING] ME(7872:20780,MainProcess):2025-12-20-15:12:57.621.000 [mindspore\nn\layer
测试集准确率： {'Accuracy': 0.9982857142857143}
```

未出现过拟合现象，Dropout 层起到一定正则化作用

手写数字测试

使用手写数字4，图片如下



最终模型预测正确

```
C:\Users\Jw\.conda\envs\homework1\python.exe "D:\university\3rd\ai\homework1\test 5\
[WARNING] ME(25324:18772,MainProcess):2025-12-20-15:16:28.436.000 [mindspore\contex
[WARNING] ME(25324:18772,MainProcess):2025-12-20-15:16:28.438.000 [mindspore\nn\laye
[WARNING] ME(25324:18772,MainProcess):2025-12-20-15:16:28.558.000 [mindspore\nn\laye
[WARNING] ME(,4954,?):2025-12-20 15:16:28 [mindspore\ccsrc\tools\error_handler\error
[WARNING] ME(,4954,?):2025-12-20 15:16:28 [mindspore\ccsrc\tools\error_handler\error
预测结果： 4
```

实验结果与思考

实验收获

1. 掌握了使用 MindSpore 构建、训练和部署 CNN 模型的完整流程；
2. 理解了卷积神经网络在图像分类任务中的优势与实现细节；
3. 学会了如何对自定义图像进行预处理以适应模型输入要求；
4. 通过实验加深了对数据标准化、模型评估等关键概念的理解。

模型优点

- 结构简单，训练速度快；
- 在 MNIST 数据集上准确率高，达到 99% 以上；
- 代码模块化清晰，易于扩展与修改。

局限性分析

- 模型仅适用于灰度图，且输入尺寸固定为 28×28；
- 对于复杂背景或扭曲数字的识别能力有限；
- 未进行充分的数据增强，泛化能力有待提升。

改进建议

1. 引入更深的网络结构（如 ResNet 变体）提升特征提取能力；
2. 使用数据增强提升模型鲁棒性；
3. 部署为 Web 服务或移动端应用，实现实时识别；
4. 扩展到更大规模数据集（如 Fashion-MNIST、CIFAR-10）。