**计算机视觉工程实验二**

姓名：李智 学号：123106222804

## 实验目标

实现LeNet-5在MNIST数据集上的训练和测试，并进行分析。

## 设计思路

1. 设备环境：

Window11 + mindspore + python3.9

1. 数据准备：

MNIST数据集包含了一系列28x28像素的灰度图像，这些图像代表了手写数字0到9。数据集被分成两个部分：训练集和测试集。训练集包含了60,000个图像样本，而测试集包含了10,000个图像样本。这些样本已经被预处理和标准化，使得每个像素的值介于0到1之间。

下载的数据集文件的目录结构如下：

./mnist/

├── test

│ ├── t10k-images-idx3-ubyte

│ └── t10k-labels-idx1-ubyte

└── train

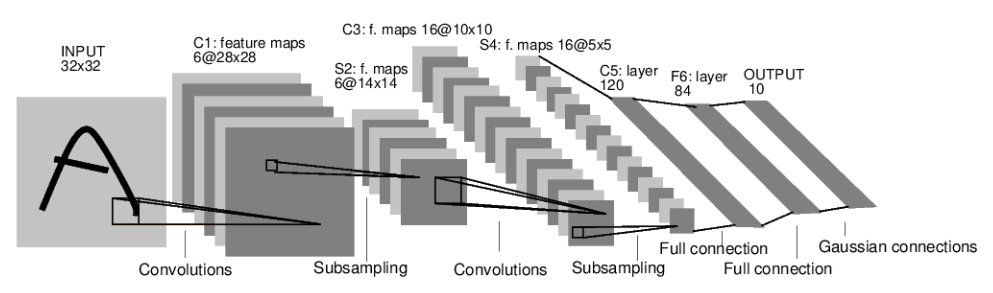
├── train-images-idx3-ubyte

└── train-labels-idx1-ubyte

1. 创建模型

实验采用LeNet-5作为实验模型，LeNet是一个经典的卷积神经网络。也是第一个成功应用于手写数字识别任务的卷积神经网络模型。网络主要由卷积层、池化层和全连接层三个部分构成。其中，卷积层是卷积神经网络的核心部分，它通过对输入图像进行卷积操作来提取图像的特征。卷积层的输入通常是一个多通道的，每个通道代表一个特征，卷积层的输出也是多通道的，其中每个通道表示一个不同的特征。

LeNet-5除去输入层共有7层，其中有2个卷积层，2个子采样层，3个全连接层，网络结构如下图所示。



1. 输入层：接收大小为32\*32的手写数字图像，MNIST图片的输入的32\*32为28\*28填充后的效果。
2. 卷积层C1：包括6个卷积核，每个卷积核的大小为5\*5 ，步长为1，填充为0。每个卷积核会产生一个大小为28\*28的特征图，输出通道数为6。
3. 采样层S2：采用最大池化操作，每个窗口的大小为 2\*2，步长为2。每个池化操作会从4个相邻的特征图中选择最大值，产生一个大小为14\*14的特征图，以减少特征图的大小，提高计算效率。
4. 卷积层C3：包括16个卷积核，每个卷积核的大小为5\*5，步长为1，填充为0。每个卷积核会产生一个大小为10\*10的特征图，输出通道数为16。
5. 采样层S4：采用最大池化操作，每个窗口的大小为 2\*2，步长为2。每个池化操作会从4个相邻的特征图中选择最大值，产生一个大小为 5\*5的特征图，输出通道数为16。
6. 全连接层C5：将每个大小为5\*5的特征图拉成一个长度为400的向量，并通过一个带有120个神经元的全连接层进行连接。
7. 全连接层F6：将120个神经元连接到84个神经元。
8. 输出层：输出层由10个神经元组成，每个神经元对应0-9中的一个数字，并输出最终的分类结果。
9. 模型训练
10. 参数设置

模型训练采用批量随机梯度下降算法进行优化，公式如下：

式中，𝑛是批量大小，𝜂是学习率，为训练轮次t中权重参数，∇𝑙为损失函数的导数，可知道除了梯度本身，这两个因子直接决定了模型的权重更新，从优化本身来看它们是影响模型性能收敛最重要的参数，一般会定义以下超参用于训练：

epochs = 50 #训练50轮

batch\_size = 32 #批次大小32

momentum = 0.9 #动量0.9

learning\_rate = 1e-2 #学习率0.01

1. 损失函数定义

要训练神经网络模型，需要定义损失函数和优化器函数，损失函数使用交叉熵损失函数`SoftmaxCrossEntropyWithLogits`，交叉熵损失函数用于衡量实际输出与期望输出之间的差异。它基于信息论中的概念，通过计算两个概率分布之间的距离来度量模型预测的准确性，特别适用于分类问题，公式如下。

其中， 表示实际概率分布p中事件i的概率，是模型输出的概率分布q中事件i的概率。交叉熵的计算通过对所有事件i的概率乘以log(q(x\_i))的和，然后取负号来得到。交叉熵越小，表示两个概率分布之间的差异越小，模型的预测结果与实际结果越接近。

1. 优化器函数定义

优化器函数采用标准动量优化算法Momentum，主要思想是引入一个积攒历史梯度信息动量来加速随机梯度下降的过程，可以避免陷入局部最优解与梯度振荡。 算法在普通的梯度下降法中引入指数加权移动平均，即定义一个动量，它是梯度的指数加权移动平均值，然后使用该 值代替原来的梯度方向来更新。定义的动量为：

该式中，表述当前动量，是进行指数加权平均移动的超参数，取，为目标函数的当前梯度，使用该动量带入梯度下降公式：

该式和普通梯度下降法迭代公式基本一致，只是方向是定义的动量，为步长，一般也是一个定义的超参数。

## 结果分析

实验一共进行了50 epochs，每一轮的训练模型都保存在./lenet中，共50个模型文件，每轮训练的模型平均损失值记录在AvgLoss.xlsx中，可绘制图表如下：

可以看见，在50 epochs的训练中，avg loss呈下降趋势，模型能够正确收敛。在10轮左右，收敛已经基本完成，在40轮左右，avg loss 值下降为0，模型不再收敛。取模型，在测试集上进行测试，结果如下：

|  |  |
| --- | --- |
| Epoch | Accuracy |
| 5 | 0.988 |
| 10 | 0.990 |
| 25 | 0.992 |
| 45 | 0.993 |

可取第10轮后的训练模型作为实际应用模型，随机选取6张手写数字图片如下：



模型（10epoch）的输出结果为：

Predicted: "[9 9 5 2 7 2]", Actual: "[9 9 5 2 7 2]"

预测值与实际值能够完整对应，模型已能够初步应用到一些简单的手写数字识别的场景中。