智能算法与应用实验一

MNIST图像分类

葉珺明 19335253

一、实验目标

- 掌握pytorch等深度学习框架的环境搭建
- 掌握图像分类任务的训练和测试流程

二、实验内容

- 完成MNIST数据集图像分类
- 学习率自适应调整

三、实验步骤与实验代码

3.1 环境准备

- 操作系统：Window 10
- 编译器：Vscode + pytorch 1.11.0
- 语言：python 3.7

3.2 数据加载及预处理

- 设置一个batch_size为100；迭代次数为10；
- 通过pytorch获取MNIST的训练数据集和测试数据集：

  参数解释： *train* 为True代表训练集， *transform* 代表对数据预处理为Tensor

```
train_data = mnist.MNIST('./optimization/',
                         train=True,
                         transform=transforms.ToTensor(),
                         download=False)
test_data = mnist.MNIST('./optimization/',
                        train=False,
                        transform=transforms.ToTensor(),
                        download=False)
```

- 通过DataLoader将数据集以batch的大小存储到迭代器中，并且打乱数据的顺序

  参数解释： *shuffle* 为True表示对数据进行重排

```
train_loader = DataLoader(train_data,
                          batch_size=batch_size,
                          shuffle=True)
test_loader = DataLoader(test_data,
                         batch_size=1,
                         shuffle=False)
```

- 数据增强：随机选取数据进行上下翻转、左右翻转和旋转操作，使得搭建的卷积神经网络变得健壮

  代码解释：以一定概率对图片进行翻转和旋转，再载入训练

```
if random.random() < 0.5:
            imtemp1 = np.flipud(temp)
        if random.random() < 0.5:
            temp = np.fliplr(temp)
        angle = random.choice([0, 1, 2, 3])
        temp = np.rot90(temp, angle)
```

3.3 超参数及优化

- $epoch = 50;\ lr = 1e-3;$
- 优化：
  - 梯度下降采用Adam优化算法：结合$Adagrad$算法和$RMSPro$算法

```
optimizer = optim.Adam(net.parameters(), lr=lr)
```

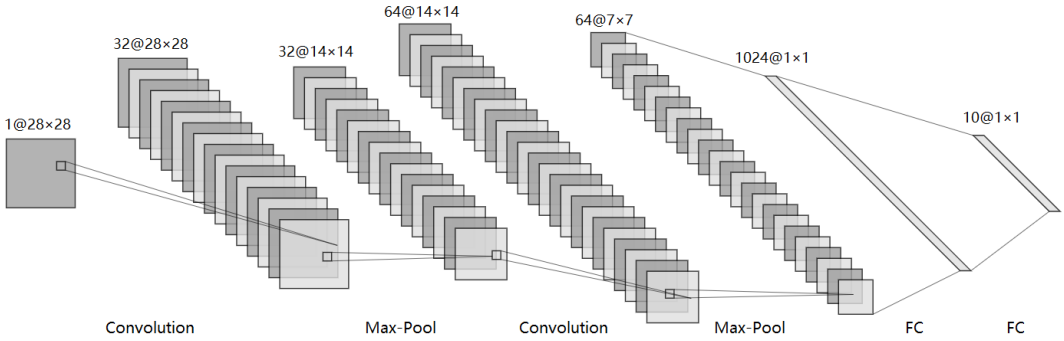
  - 学习率调整：在训练的过程中，随着epoch的增大逐渐减小学习率，有助于模型的收敛，借进最优解。实验采用余弦变换动态调整：

```
scheduler = lr_scheduler.CosineAnnealingLR(optimizer, T_max=T_max,
eta_min=eta_min)
```

3.4 模型搭建

- 采用卷积神经网络（CNN）：

  参考LeNet-5模型的搭建，搭建6层网络，具体示意图：

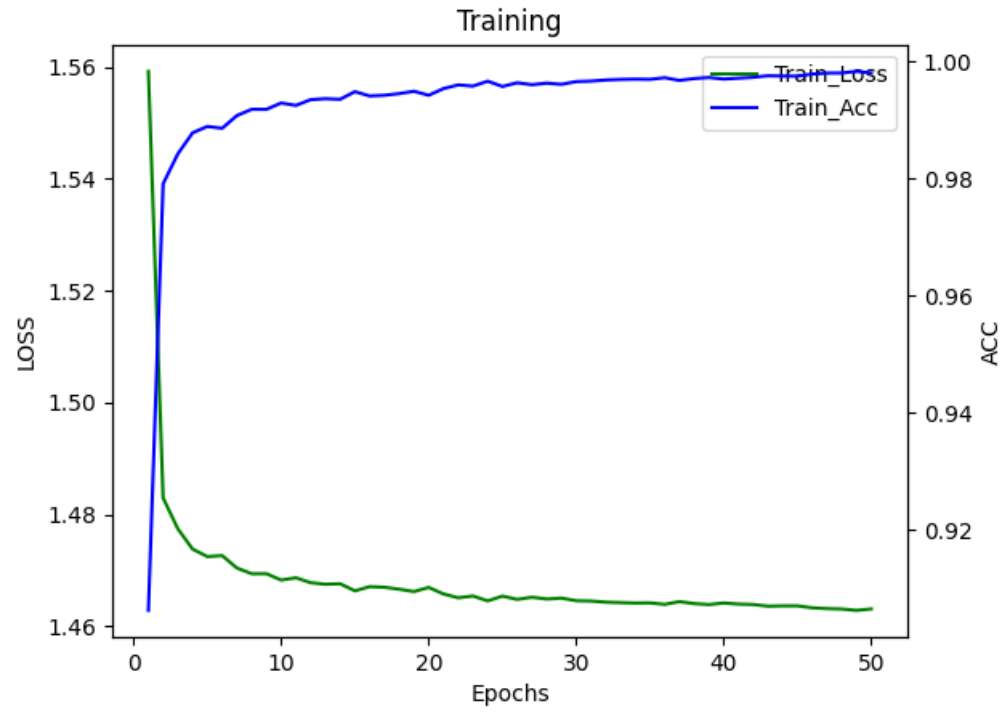


- 核心代码：

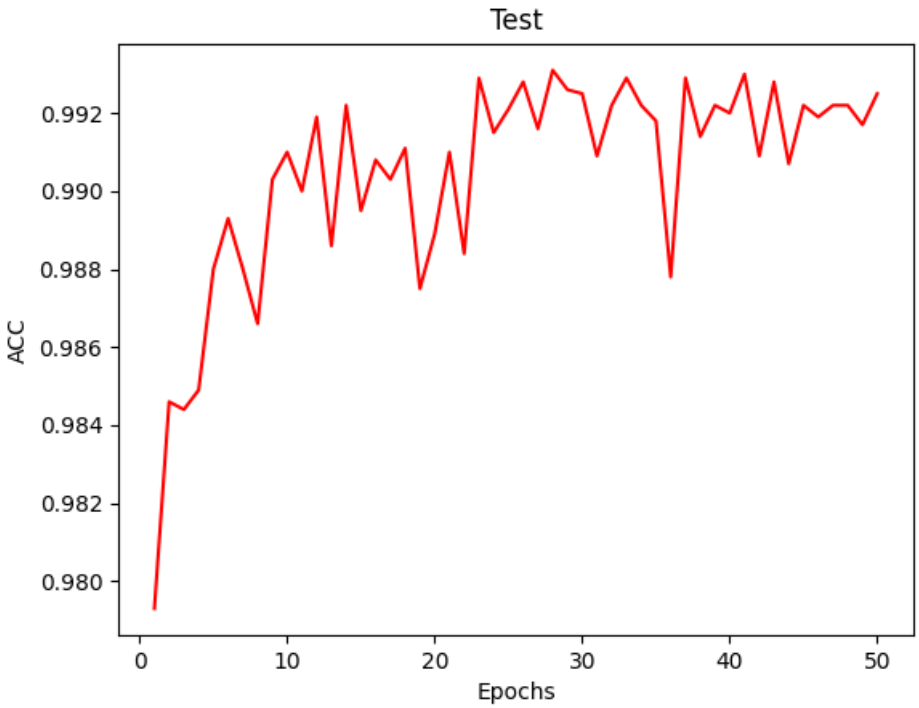
```
def __init__(self):
    super(CNN, self).__init__()

    self.conv1 = nn.Conv2d(1, 32, kernel_size=3, padding=1)
    self.relu1 = nn.ReLU()
    self.pool1 = nn.MaxPool2d(kernel_size=2, stride=2)

    self.conv2 = nn.Conv2d(32, 64, kernel_size=3, padding=1)
    self.relu2 = nn.ReLU()
    self.pool2 = nn.MaxPool2d(kernel_size=2, stride=2)

    self.fc3 = nn.Linear(64*7*7, 1024)
    self.relu3 = nn.ReLU()
    self.fc4 = nn.Linear(1024, 10)
    self.softmax4 = nn.Softmax(dim=1)
```

四、实验结果

- 50迭代的训练结果：



- 历经每次迭代后进行训练：



随着迭代次数的增加，训练集和测试集的准确度上升，但是迭代次数一定时，测试集准确度下降，说明模型出现过拟合，此时，模型不用再进行迭代训练了。

五、实验心得

通过本次实验，实现了MNIST数据集分类，利用pytorch搭建了卷积神经网络模型，本次搭建的卷积神经神经网络模型不深，只有六层，与LeNet5相比，不同的地方是采用3×3的卷积核、最大池化和ReLU激活函数，在3×3的卷积核带来的计算量耕地，通过最大池化突出主要特征，以及ReLU激活函数增加非线性。总体实验难度不高，借助pytorch有效学习了卷积神经网络的搭建，以及一些对数据集的处理如数据增强，训练的神经网络时的梯度算法选择和学习率的调整。