**1 任务要求**

任务要求：设计一个卷积神经网络，并在其中使用ResNet模块，在MNIST数据集上实现10分类手写体数字识别。

**2 任务设计**

**2.1 数据集准备**

使用torchvision模块提供的方法可以很简单地获取数据。

1. transform = transforms.Compose([
2. transforms.ToTensor(),
3. transforms.Normalize((0.1307,), (0.3081,))
4. ])
5. train\_dataset = datasets.MNIST(root='./data', train=True,
6. download=True, transform=transform)
7. test\_dataset = datasets.MNIST(root='./data', train=False,
8. download=True, transform=transform)
9. *# 下面是将张量分组，每一组大小为batch\_size，然后可以用for in 循环遍历*
10. train\_loader = DataLoader(train\_dataset, train\_batch\_size,
11. shuffle=True)
12. test\_loader = DataLoader(test\_dataset, test\_batch\_size,
13. shuffle=False)

**2.2 网络结构**

根据任务要求，构建一个简单的resnet模块，使用该模块构建网络模型。一个简单的resnet模块如下图所示。

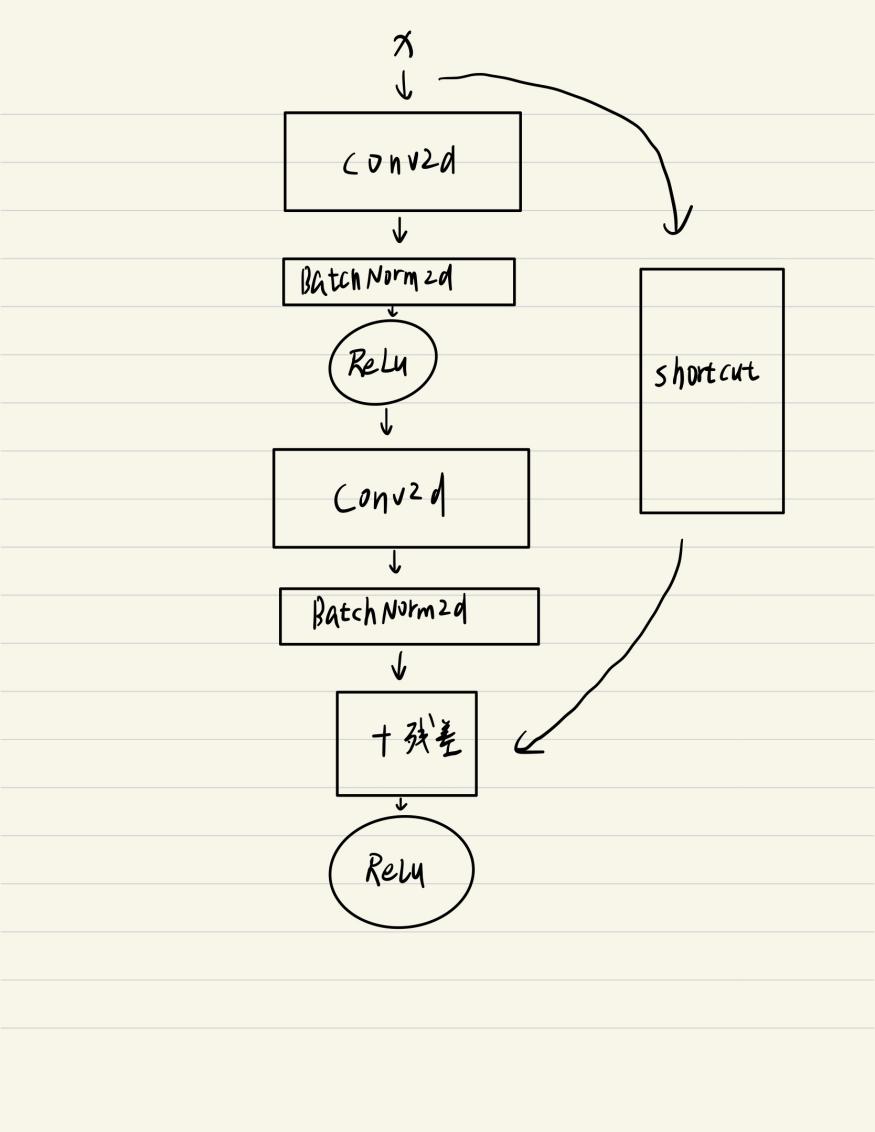


图 1 resnet模块

由于残差相加要求张量维度相同，所以如果卷积层改变了x的维度，shortcut中需要使用1\*1的卷积核将维度修正。

本次实验叠加三块resnet模块：

1. class ResNet(nn.Module):
2. def \_\_init\_\_(self):
3. super(ResNet, self).\_\_init\_\_()
4. self.block1 = My\_Block(1, 16, 1)
5. *# 28*
6. self.block2 = My\_Block(16, 32, 2)
7. *# 14*
8. self.block3 = My\_Block(32, 64, 2)
9. *# 7*
10. self.fc = nn.Linear(64\*7\*7, 10)
11. def forward(self, x):
12. x = self.block1.forward(x)
13. x = self.block2.forward(x)
14. x = self.block3.forward(x)
15. x = x.view(x.size(0), -1)
16. x = self.fc(x)
17. return x

**2.3 架构**

采用pytorch架构。

**3 实验**

本实验尝试使用不同的激活函数、改变实验超参数等，观察网络性能。

**3.1 激活函数**

控制其他参数不变，train\_batch\_size = 64，test\_batch\_size = 1000，学习率=0.001。只改变激活函数。此处对比三个激活函数，分别是ReLu,Sigmoid,Tanh对应model1，model2，model3。loss为每一次batch上的总损失。运行后可以得到以下图像。

训练集上：

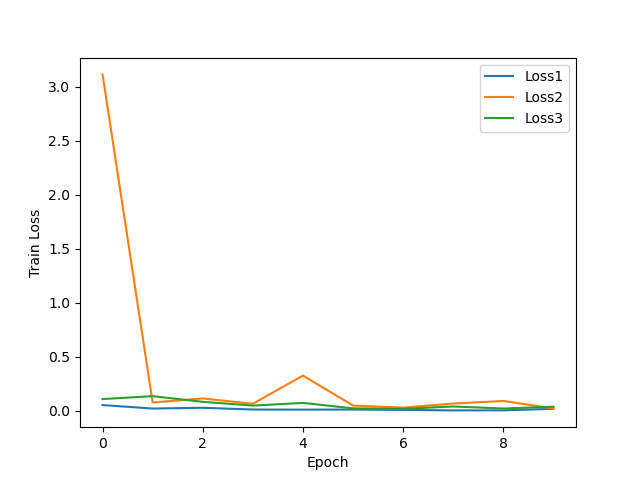


图 2 训练集loss

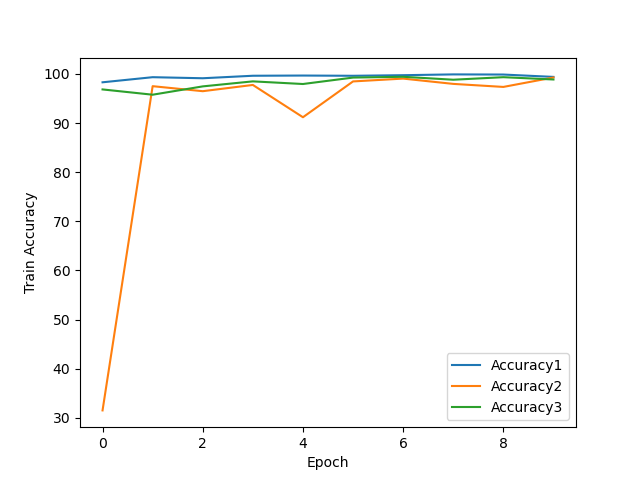


图 3 训练集accuracy

测试集上：

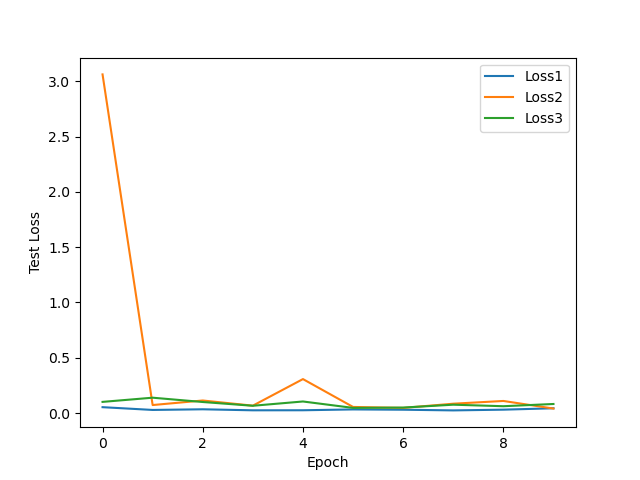


图 4 测试集loss

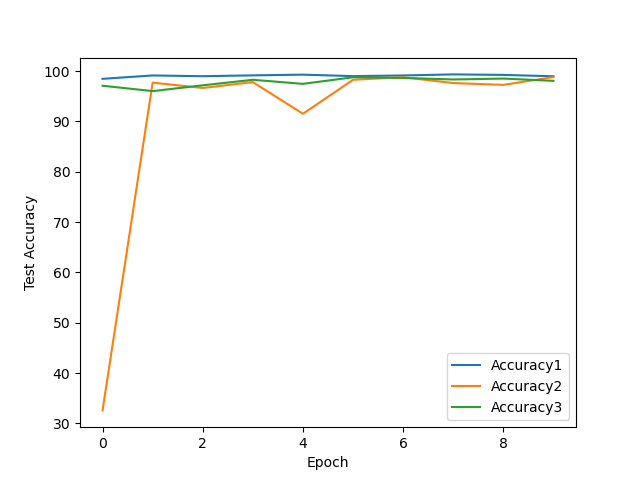


图 5 测试集accuracy

最终测试loss和accuracy如下表：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 训练集 | | 测试集 | |
| loss | accuracy | loss | accuracy |
| ReLu | 0.0177 | 0.9940 | 0.0431 | 0.9896 |
| Sigmoid | 0.0210 | 0.9928 | 0.0405 | 0.9880 |
| Tanh | 0.0374 | 0.9887 | 0.0818 | 0.9805 |

最终每个数字预测率如下所示。

ReLu：

训练集：

the number 0 accuracy is 0.9993

the number 1 accuracy is 0.9953

the number 2 accuracy is 0.9961

the number 3 accuracy is 1.0000

the number 4 accuracy is 0.9993

the number 5 accuracy is 0.9991

the number 6 accuracy is 0.9973

the number 7 accuracy is 0.9909

the number 8 accuracy is 0.9773

the number 9 accuracy is 0.9854

测试集：

the number 0 accuracy is 0.9980

the number 1 accuracy is 0.9930

the number 2 accuracy is 0.9981

the number 3 accuracy is 0.9980

the number 4 accuracy is 0.9980

the number 5 accuracy is 0.9933

the number 6 accuracy is 0.9833

the number 7 accuracy is 0.9883

the number 8 accuracy is 0.9754

the number 9 accuracy is 0.9703

Sigmoid：

训练集：

the number 0 accuracy is 1.0000

the number 1 accuracy is 0.9988

the number 2 accuracy is 0.9804

the number 3 accuracy is 0.9966

the number 4 accuracy is 0.9983

the number 5 accuracy is 0.9945

the number 6 accuracy is 0.9986

the number 7 accuracy is 0.9864

the number 8 accuracy is 0.9810

the number 9 accuracy is 0.9928

测试集：

the number 0 accuracy is 0.9990

the number 1 accuracy is 0.9982

the number 2 accuracy is 0.9719

the number 3 accuracy is 0.9931

the number 4 accuracy is 0.9959

the number 5 accuracy is 0.9899

the number 6 accuracy is 0.9875

the number 7 accuracy is 0.9825

the number 8 accuracy is 0.9754

the number 9 accuracy is 0.9861

Tanh：

训练集：

the number 0 accuracy is 0.9985

the number 1 accuracy is 0.9941

the number 2 accuracy is 0.9968

the number 3 accuracy is 0.9953

the number 4 accuracy is 0.9855

the number 5 accuracy is 0.9799

the number 6 accuracy is 0.9981

the number 7 accuracy is 0.9427

the number 8 accuracy is 1.0000

the number 9 accuracy is 0.9973

测试集：

the number 0 accuracy is 0.9969

the number 1 accuracy is 0.9885

the number 2 accuracy is 0.9893

the number 3 accuracy is 0.9901

the number 4 accuracy is 0.9796

the number 5 accuracy is 0.9742

the number 6 accuracy is 0.9916

the number 7 accuracy is 0.9212

the number 8 accuracy is 0.9928

the number 9 accuracy is 0.9812

每一轮训练过后的训练集和测试集上的loss、accuracy和每个数字的准确率详见同目录下的“不同激活函数.txt”文件。画图程序见画图代码目录下的对应py文件。

由上述图片和数据可知，三个不同的激活函数最终都收敛到差不多的水平。ReLu收敛最快，Tanh次之，Sigmoid效果最差。Sigmoid不仅收敛较差，而且中间出现了较大的波动，可能是由于Sigmoid激活函数的本质是判断输入是否属于哪个类，即二分类问题，对于多分类性能较差。

**3.2 超参数**

采用ReLu作为激活函数，改变学习率观察网络性能。其他参数同3.1所述。

依然观察三个不同的学习率分别是0.001, 0.1 , 1 ，对应model1，model2和model3。结果如下。

训练集：

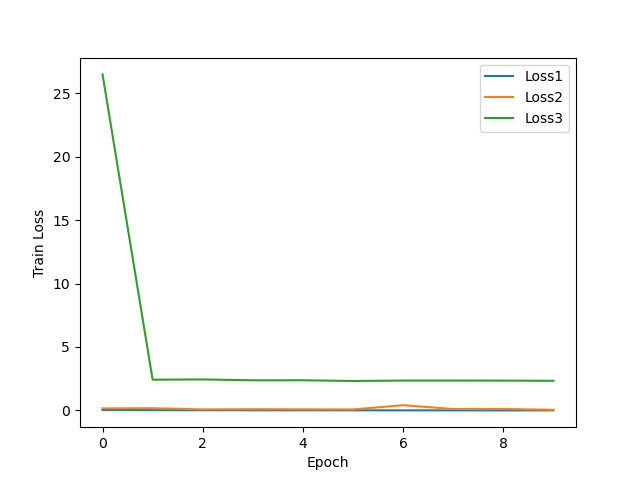


图 6 训练集损失

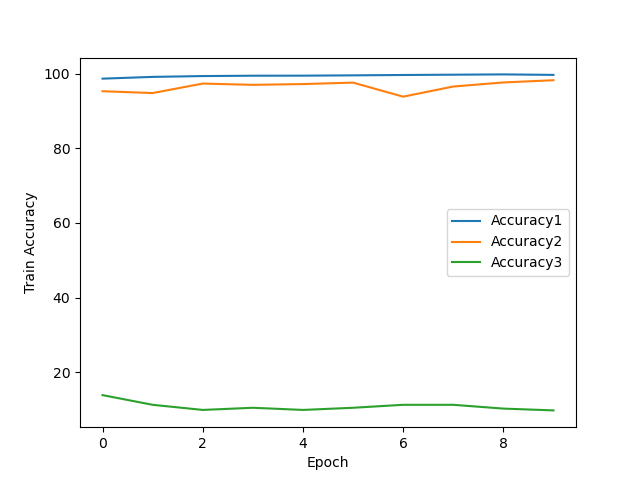


图 7 训练集准确率

测试集上：

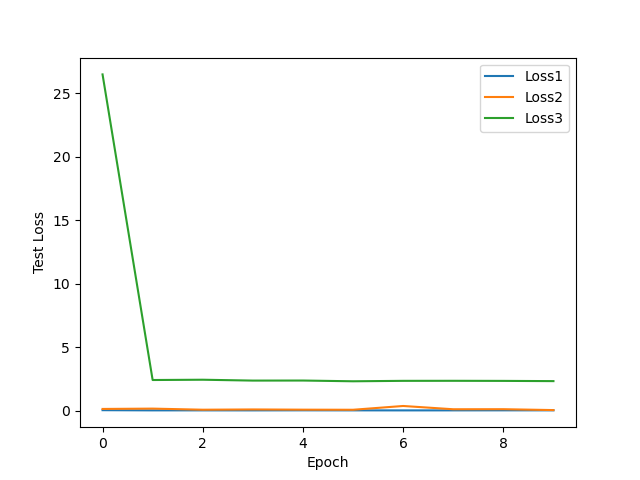


图 8 测试集损失

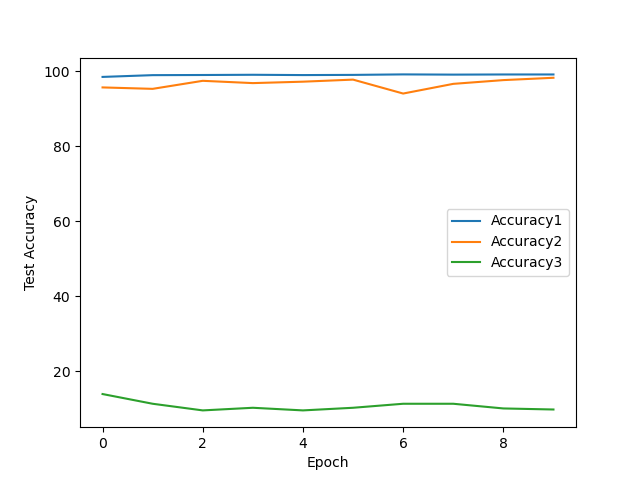


图 9 测试集准确率

最终各模型在训练集和测试集上的loss和accuracy如下：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 学习率 | 训练集 | | 测试集 | |
| loss | accuracy | loss | accuracy |
| 0.001 | 0.0086 | 0.9971 | 0.0355 | 0.9909 |
| 0.1 | 0.0536 | 0.9828 | 0.0555 | 0.9821 |
| 1 | 2.3358 | 0.0974 | 2.3356 | 0.0982 |

最终各模型在各个数字上的准确率：

学习率=0.001：

训练集：

the number 0 accuracy is 0.9998

the number 1 accuracy is 0.9956

the number 2 accuracy is 0.9936

the number 3 accuracy is 0.9984

the number 4 accuracy is 0.9997

the number 5 accuracy is 0.9952

the number 6 accuracy is 0.9998

the number 7 accuracy is 0.9995

the number 8 accuracy is 0.9998

the number 9 accuracy is 0.9894

测试集：

the number 0 accuracy is 0.9980

the number 1 accuracy is 0.9947

the number 2 accuracy is 0.9884

the number 3 accuracy is 0.9911

the number 4 accuracy is 0.9949

the number 5 accuracy is 0.9865

the number 6 accuracy is 0.9885

the number 7 accuracy is 0.9951

the number 8 accuracy is 0.9959

the number 9 accuracy is 0.9752

学习率=0.1：

训练集：

the number 0 accuracy is 0.9941

the number 1 accuracy is 0.9878

the number 2 accuracy is 0.9688

the number 3 accuracy is 0.9858

the number 4 accuracy is 0.9884

the number 5 accuracy is 0.9812

the number 6 accuracy is 0.9829

the number 7 accuracy is 0.9936

the number 8 accuracy is 0.9754

the number 9 accuracy is 0.9682

测试集：

the number 0 accuracy is 0.9918

the number 1 accuracy is 0.9903

the number 2 accuracy is 0.9709

the number 3 accuracy is 0.9911

the number 4 accuracy is 0.9888

the number 5 accuracy is 0.9843

the number 6 accuracy is 0.9739

the number 7 accuracy is 0.9883

the number 8 accuracy is 0.9661

the number 9 accuracy is 0.9742

学习率=1：

训练集：

the number 0 accuracy is 0.0000

the number 1 accuracy is 0.0000

the number 2 accuracy is 0.0000

the number 3 accuracy is 0.0000

the number 4 accuracy is 1.0000

the number 5 accuracy is 0.0000

the number 6 accuracy is 0.0000

the number 7 accuracy is 0.0000

the number 8 accuracy is 0.0000

the number 9 accuracy is 0.0000

测试集：

the number 0 accuracy is 0.0000

the number 1 accuracy is 0.0000

the number 2 accuracy is 0.0000

the number 3 accuracy is 0.0000

the number 4 accuracy is 1.0000

the number 5 accuracy is 0.0000

the number 6 accuracy is 0.0000

the number 7 accuracy is 0.0000

the number 8 accuracy is 0.0000

the number 9 accuracy is 0.0000

具体的每一轮训练之后的数据见同目录下的“不同学习率.txt”。

从图像和数据来看，学习率为0.001时学习最为快速平稳。学习率为0.1时模型性能也比较好，但是波动较大，收敛效果不如0.001。当学习率为1时，有趣的事情发生了，由于每一次梯度下降更新的权重值范围过大，造成模型的极度震荡，体现为被困在局部解中。可以去观察测试数据，模型到最后面只有一个数字的识别率较高，而且这个数字还在变动，一会是6，一会是4。

**4 结论**

本次实验构建了简单的resnet模块，并用其搭建了一个简单的神经网络模型。但是resnet中的加残差好像并没有发挥多大效果。我尝试将加残差这一步去掉，神经网络的性能并没有受到多大影响。经过查阅资料，应该是自己搭的神经网络层数过少，没有产生退化问题，而残差是[何恺明](http://www.baidu.com/link?url=9lkTCqi8DiqIiG6dn7ZesSxP5m49ue29ajFOcKm98PKXNcBa4Vo6pRgRbhrVeUw8A3JzvGzWv17o0W7Q85qtTdNZ3ELESjeLARZfHG8_Fxq" \t "https://www.baidu.com/_blank)为了解决此问题引入的，发生在深层网络结构中。通过不同超参数的尝试和不同激活函数的尝试，明白设计一个好的模型需要了解各种参数的大概意义和使用方式。比如学习率不能设太大，不然会使模型过于震荡。