《计算机视觉》实验报告

姓名：刘远航 学号：22121883

实验 10

一． 任务 1

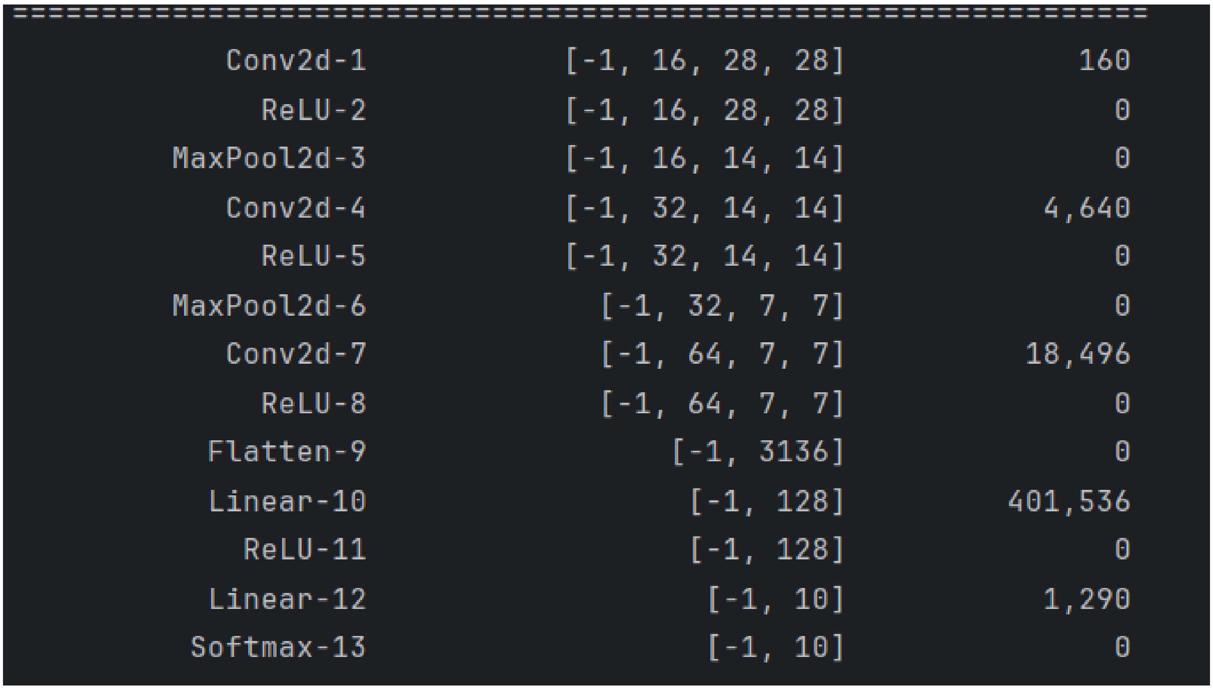
a) 核心代码：

```
# 定义 CNN 模型
class Net(torch.nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.model = torch.nn.Sequential(
            # 图像大小为 28x28
            torch.nn.Conv2d(in_channels=1, out_channels=16, kernel_size=3, stride=1, padding=1),
            torch.nn.ReLU(),
            torch.nn.MaxPool2d(kernel_size=2, stride=2),
            # 图像大小为 14x14
            torch.nn.Conv2d(in_channels=16, out_channels=32, kernel_size=3, stride=1, padding=1),
            torch.nn.ReLU(),
            torch.nn.MaxPool2d(kernel_size=2, stride=2),
            # 图像大小为 7x7
            torch.nn.Conv2d(in_channels=32, out_channels=64, kernel_size=3, stride=1, padding=1),
            torch.nn.ReLU(),
            torch.nn.Flatten(),
            torch.nn.Linear(in_features=7 * 7 * 64, out_features=128),
            torch.nn.ReLU(),
            torch.nn.Linear(in_features=128, out_features=10),
            torch.nn.Softmax(dim=1)
        )
    def forward(self, input):
        output = self.model(input)
        return output

lossF = torch.nn.CrossEntropyLoss()
optimizer = torch.optim.Adam(net.parameters())
history = {'Test Loss': [], 'Test Accuracy': []}
```

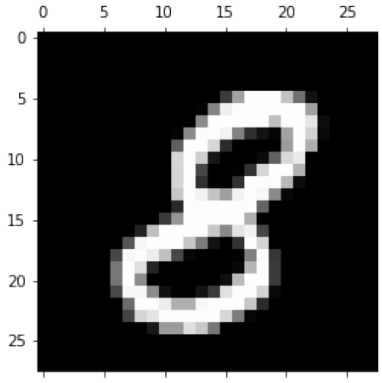
```
for epoch in range(1, EPOCHS + 1):
    processBar = tqdm(trainDataLoader, unit='step')
    net.train(True)
    for step, (trainImgs, labels) in enumerate(processBar):
        trainImgs = trainImgs.to(device)
        labels = labels.to(device)

        net.zero_grad()
        outputs = net(trainImgs)
        loss = lossF(outputs, labels)
        predictions = torch.argmax(outputs, dim=1)
        accuracy = torch.sum(predictions == labels).item() / labels.shape[0]
        loss.backward()
        optimizer.step()
        processBar.set_description(f"[{epoch}/{EPOCHS}] Loss: {loss.item():.4f}, Acc: {accuracy:.4f}")
        if step == len(processBar) - 1:
            correct, totalLoss = 0, 0
            net.train(False)
            with torch.no_grad():
                for testImgs, labels in testDataLoader:
                    testImgs = testImgs.to(device)
                    labels = labels.to(device)
                    outputs = net(testImgs)
                    loss = lossF(outputs, labels)
                    predictions = torch.argmax(outputs, dim=1)
                    totalLos += loss
                    correct += torch.sum(predictions == labels)
                testAccuracy = correct.item() / (BATCH_SIZE * len(testDataLoader))
                testLoss = totalLoss / len(testDataLoader)
                history['Test Loss'].append(testLoss.item())
                history['Test Accuracy'].append(testAccuracy)

            processBar.set_description(f"[{epoch}/{EPOCHS}] Loss: {loss.item():.4f}, Acc: {accuracy:.4f}, Test Loss: {testLoss.item():.4f}, Test Acc: {testAccuracy:.4f}"
torch.save(net.state_dict(), './model.pth')
```

b)　实验结果截图

网络结构示意图

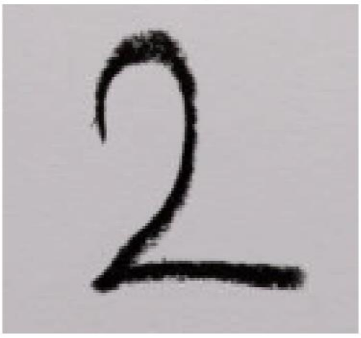


```
================================================================
            Conv2d-1           [-1, 16, 28, 28]             160
              ReLU-2           [-1, 16, 28, 28]               0
         MaxPool2d-3           [-1, 16, 14, 14]               0
            Conv2d-4           [-1, 32, 14, 14]           4,640
              ReLU-5           [-1, 32, 14, 14]               0
         MaxPool2d-6             [-1, 32, 7, 7]               0
            Conv2d-7             [-1, 64, 7, 7]          18,496
              ReLU-8             [-1, 64, 7, 7]               0
           Flatten-9                 [-1, 3136]               0
           Linear-10                  [-1, 128]         401,536
             ReLU-11                  [-1, 128]               0
           Linear-12                   [-1, 10]           1,290
          Softmax-13                   [-1, 10]               0
```

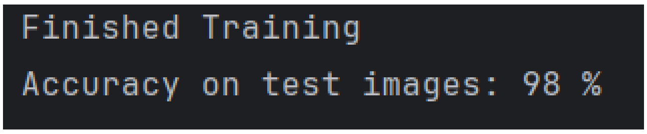


Predicted Digit: 8

Predicted Digit: 9



Predicted Digit: 2



```
Finished Training
Accuracy on test images: 98 %
```

c) 实验小结

通过这次实验，我学习到了卷积神经网络的简单应用，这次实验构建了一个简单的网络进行训练，一进行了 5 轮训练，最后准确率都高达 98%，很好的完成了实验目的，对深度学习有了直观的感受。