

《计算机视觉》实验报告

姓名：汪江豪 学号：22121630

实验十

一. 任务 1

- 1、完成深度学习环境配置，框架自行选择，推荐 pytorch, tensorflow
- 2、学习搭建一个卷积神经网络，画出网络结构示意图
- 3、用 CNN 实现 mnist 手写数字识别，给出实验结果

a) 核心代码：

1.数据集划分：

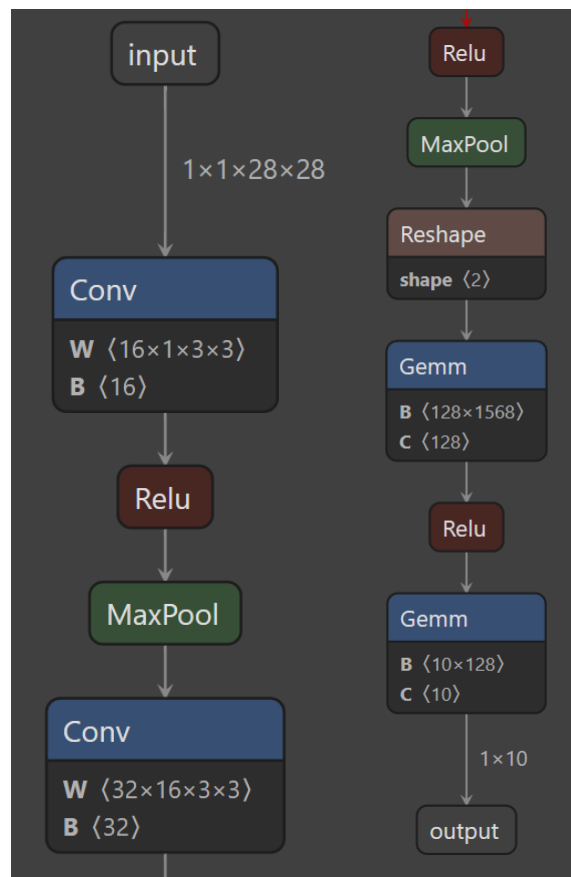
```
train_loader = DataLoader(train_dataset, batch_size=64, shuffle=True)
test_loader = DataLoader(test_dataset, batch_size=1000, shuffle=False)
```

2.构建的 CNN 卷积神经网络：

```
class SimpleCNN(nn.Module):
    def __init__(self):
        super(SimpleCNN, self).__init__()
        self.conv1 = nn.Conv2d(1, 16, kernel_size=3, padding=1) # 输出通道 16
        self.conv2 = nn.Conv2d(16, 32, kernel_size=3, padding=1) # 输出通道 32
        self.fc1 = nn.Linear(32 * 7 * 7, 128)
        self.fc2 = nn.Linear(128, 10)
        self.dropout = nn.Dropout(0.25)

    def forward(self, x):
        x = F.relu(self.conv1(x))
        x = F.max_pool2d(x, 2)
        x = F.relu(self.conv2(x))
        x = F.max_pool2d(x, 2)
        x = x.view(-1, 32 * 7 * 7)
        x = F.relu(self.fc1(x))
        x = self.dropout(x)
        x = self.fc2(x)
        return x
```

其神经网络结构图如下所示



其计算图的构建和执行流程如下图：


```

    for batch_idx, (data, target) in tqdm(enumerate(train_loader),
desc=f"Training Epoch {epoch}", total=len(train_loader)):
        data, target = data.to(device), target.to(device)
        optimizer.zero_grad()
        output = model(data)
        loss = criterion(output, target)
        loss.backward()
        optimizer.step()
        total_loss += loss.item()
    print(f"Epoch {epoch}, Loss: {total_loss / len(train_loader):.4f}")

```

5.测试模型:

```

def test(model, device, test_loader):
    model.eval()
    correct = 0
    total = 0
    with torch.no_grad():
        for data, target in tqdm(test_loader, desc="Testing",
total=len(test_loader)):
            data, target = data.to(device), target.to(device)
            output = model(data)
            pred = output.argmax(dim=1)
            correct += (pred == target).sum().item()
            total += target.size(0)
    acc = correct / total
    print(f"Test Accuracy: {acc * 100:.2f}%")
    return acc

```

b) 实验结果截图

1.模型准确率:

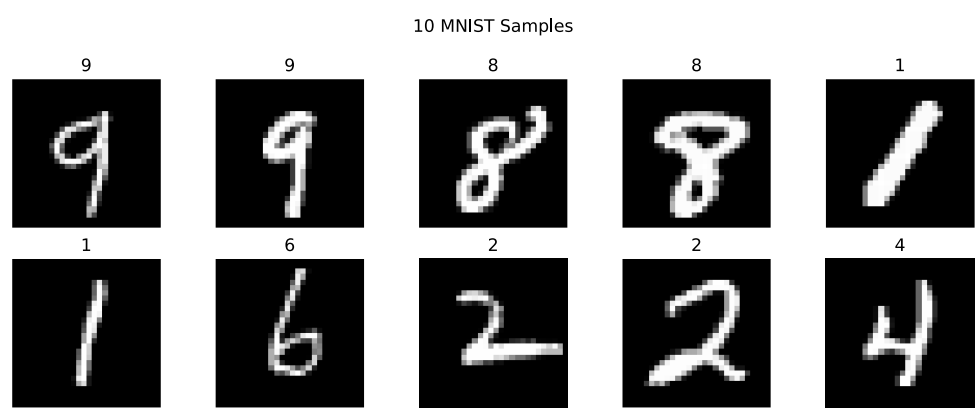
```

D:\Desktop\third_spring\computer_vision\test10>D:/Python/python.exe d:/Desktop/third_spring/computer_vision/test10/main.py
Training Epoch 1: 100% | 938/938 [00:12<00:00, 76.59it/s]
Epoch 1, Loss: 0.2097% | 932/938 [00:12<00:00, 73.34it/s]
Testing: 100% | 10/10 [00:01<00:00, 5.29it/s]
Test Accuracy: 98.23% | 10/10 [00:01<00:00, 6.03it/s]
New best model found with accuracy: 98.23%
Training Epoch 2: 100% | 938/938 [00:13<00:00, 71.53it/s]
Epoch 2, Loss: 0.0659% | 937/938 [00:13<00:00, 68.82it/s]
Testing: 100% | 10/10 [00:01<00:00, 6.02it/s]
Test Accuracy: 98.62% | 10/10 [00:01<00:00, 5.92it/s]
New best model found with accuracy: 98.62%
Training Epoch 3: 100% | 938/938 [00:13<00:00, 71.78it/s]
Epoch 3, Loss: 0.0490% | 931/938 [00:12<00:00, 66.15it/s]
Testing: 100% | 10/10 [00:01<00:00, 6.11it/s]
Test Accuracy: 98.92% | 10/10 [00:01<00:00, 6.22it/s]
New best model found with accuracy: 98.92%
Training Epoch 4: 100% | 938/938 [00:13<00:00, 70.82it/s]
Epoch 4, Loss: 0.0402% | 934/938 [00:13<00:00, 72.47it/s]
Testing: 100% | 10/10 [00:01<00:00, 5.65it/s]
Test Accuracy: 99.05% | 10/10 [00:01<00:00, 6.07it/s]
New best model found with accuracy: 99.05%
Training Epoch 5: 100% | 938/938 [00:13<00:00, 71.50it/s]
Epoch 5, Loss: 0.0320% | 936/938 [00:13<00:00, 71.07it/s]
Testing: 100% | 10/10 [00:01<00:00, 5.76it/s]
Test Accuracy: 99.01% | 10/10 [00:01<00:00, 5.47it/s]
Epochs: 100% | 5/5 [01:13<00:00, 14.70s/it]
Saved model with accuracy: 99.05%

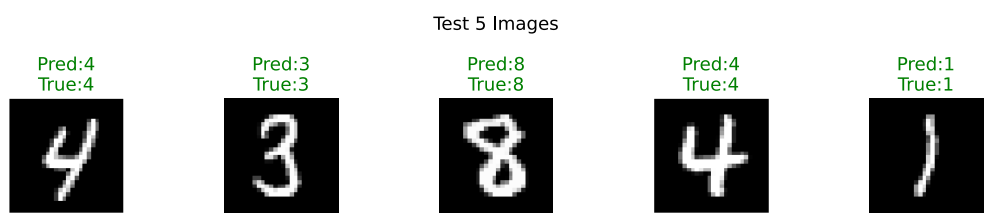
```

在 5 轮训练下，得出最高准确率 99.05%

2.MNIST 数据集展示：



3. 模型测试：



c) 实验小结

本实验基于 PyTorch 框架，搭建了一个简化的卷积神经网络（SimpleCNN），并在 MNIST 手写数字识别数据集上进行了训练和测试。经过 5 轮训练，模型在测试集上表现不错，准确率达到了预期效果。整个实验过程中，我不仅实现了基本的模型训练和测试，还额外做了模型保存、网络结构可视化等拓展内容，对 CNN 的工作原理有了更直观的理解。

通过这次实验，我深刻体会到调参对模型性能的影响，也熟悉了 PyTorch 的基本使用流程。虽然 MNIST 是比较简单的数据集，但这次实践让我对图像分类任务有了更扎实的认识，为后续学习更复杂的网络结构打下了基础。