浙江大學

浙江大学实验报告

课　　程	:	计算机视觉
实验名称	:	HONOR
姓　　名	:	吕皓明
专　　业	:	计算机科学与技术
学　　号	:	3190103303
指导老师	:	宋明黎
日　　期	:	2021/12/26

实验目的和要求

- 利用CNN进行手写数字识别

实验内容和原理

卷积神经网络

实验步骤和分析

导入库

```
import torch
from torch import nn
from torch.nn import functional as F
from torch.autograd import Variable
from torch import optim
import torchvision
```

数据导入`DataLoader`

```
batch_size = 100
train_loader = torch.utils.data.DataLoader(
    torchvision.datasets.MNIST('mnist_data',train=True,download=True,
                               transform=torchvision.transforms.Compose([
                                   torchvision.transforms.ToTensor(),
                                   torchvision.transforms.Normalize(
                                       (0.137,),(0.3081,))
                               ])),
    batch_size=batch_size,shuffle=True)
test_loader = torch.utils.data.DataLoader(
    torchvision.datasets.MNIST('mnist_data/',train=False,download=True,
                               transform=torchvision.transforms.Compose([
                                   torchvision.transforms.ToTensor(),
                                   torchvision.transforms.Normalize(
                                       (0.1307,),(0.3081,))
                               ])),
    batch_size=batch_size,shuffle=False)
```

定义CNN

```
class CNN(nn.Module):
    def __init__(self):
        super().__init__()
```

```
        self.cnn1 = nn.Conv2d(in_channels=1, out_channels=16, kernel_size=5, stride=1,
padding=0)
        self.relu1 = nn.ReLU()
        self.maxpool1 = nn.MaxPool2d(kernel_size=2)

        self.cnn2 = nn.Conv2d(in_channels=16, out_channels=32, kernel_size=5, stride=1,
padding=0)
        self.relu2 = nn.ReLU()
        self.maxpool2 = nn.MaxPool2d(kernel_size=2)

        self.fc = nn.Linear(32 * 4 * 4, 10)

    def forward(self, x):
        out = self.cnn1(x)
        out = self.relu1(out)
        out = self.maxpool1(out)

        out = self.cnn2(out)
        out = self.relu2(out)
        out = self.maxpool2(out)

        out = out.view(out.size(0), -1)

        out = self.fc(out)

        return out
```

定义参数

```
n_iters = 2500
num_epochs = n_iters / (60000 / batch_size)
num_epochs = int(num_epochs)

model = CNN()

error = nn.CrossEntropyLoss()

learning_rate = 0.1
optimizer = torch.optim.SGD(model.parameters(), lr=learning_rate)
```

训练并测试

```
count = 0
for epoch in range(num_epochs):
    for i, (images, labels) in enumerate(train_loader):
        train = Variable(images.view(100,1,28,28)).float()
        labels = Variable(labels)
```

```
        # Clear gradients
        optimizer.zero_grad()

        # Forward propagation
        outputs = model(train)

        # Calculate softmax and ross entropy loss
        loss = error(outputs, labels)

        # Calculating gradients
        loss.backward()

        # Update parameters
        optimizer.step()

        count += 1

        if count % 50 == 0:
            # Calculate Accuracy
            correct = 0
            total = 0
            # Iterate through test dataset
            for images, labels in test_loader:

                test = Variable(images.view(100,1,28,28)).float()

                # Forward propagation
                outputs = model(test)

                # Get predictions from the maximum value
                predicted = torch.max(outputs.data, 1)[1]

                # Total number of labels
                total += len(labels)

                correct += (predicted == labels).sum()

            accuracy = 100 * correct / float(total)

        if count % 500 == 0:
            # Print Loss
            print('Iteration: {}  Loss: {}  Accuracy: {} %'.format(count, loss.data,
accuracy))
```

实验环境及运行方法

- 实验环境：Mac OS

- 运行方法

  ipynb运行，可以直接RUN ALL

实验结果展示

```
Iteration: 500  Loss: 0.09648547321557999  Accuracy: 98.12999725341797 %
Iteration: 1000  Loss: 0.009674507193267345  Accuracy: 98.72000122070312 %
Iteration: 1500  Loss: 0.031248390674591064  Accuracy: 98.9000015258789 %
Iteration: 2000  Loss: 0.002966133877635002  Accuracy: 98.87999725341797 %
```

心得体会

因为AI已经小有入门，所以这个经典的手写数字识别的题目肯定是做过了。这次作业直接拿了以前参加kaggle上面竞赛的自己提交的代码，修改了一下数据读取就好了。