Pattern Recognition and Machine Learning: Homework 11

Qingru Hu 2020012996

2023 年 5 月 16 日

1 Screenshots of each part of the code that you have finished

```python
# Prepare the CIFAR-100 dataset and the dataloader
transform = transforms.Compose(
    [transforms.ToTensor(),
     transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5))])

batch_size = 128
trainset = torchvision.datasets.CIFAR100(root='./data', train=True,
                                         download=True, transform=transform)
trainloader = torch.utils.data.DataLoader(trainset, batch_size=batch_size,
                                          shuffle=True, num_workers=4)

testset = torchvision.datasets.CIFAR100(root='./data', train=False,
                                        download=True, transform=transform)
testloader = torch.utils.data.DataLoader(testset, batch_size=batch_size,
                                         shuffle=False, num_workers=4)
```

```python
class DownSamplingBlock(nn.Module):
    def __init__(self, dim_in, dim):
        super().__init__()
        dim = int(2 * dim_in)
        self.conv1 = nn.Conv2d(dim_in, dim, kernel_size=3, padding=1, stride=2)
        self.bn1 = nn.BatchNorm2d(dim)
        self.conv2 = nn.Conv2d(dim, dim, kernel_size=3, padding=1)
        self.bn2 = nn.BatchNorm2d(dim)
        self.shortcut = nn.Sequential(
            nn.Conv2d(dim_in, dim, kernel_size=1, stride=2, bias=False),
            nn.BatchNorm2d(dim)
        )

```

```python
# Define the optimizer and loss function.
lossfunc = nn.CrossEntropyLoss()
optimizer = optim.Adam(net.parameters(), lr=0.001)
```

```python
# Write the training loop.
for epoch in range(10):
    epoch_loss = 0.0
    for i, data in enumerate(trainloader, 0):
        images, labels = data[0].cuda(), data[1].cuda()
        optimizer.zero_grad()
        outputs = net(images)
        loss = lossfunc(outputs, labels)
        loss.backward()
        optimizer.step()
        epoch_loss += loss.item()
    # print epoch loss
    print(f'Epoch: {epoch + 1}, Loss: {epoch_loss / 200:.3f}')
```

```python
# Evaluate the model on the test dataset
correct = 0
total = 0
with torch.no_grad():
    for data in testloader:
        images, labels = data[0].cuda(), data[1].cuda()
        outputs = net(images)
        _, predicted = torch.max(outputs.data, 1)
        total += labels.size(0)
        correct += (predicted == labels).sum().item()
print(f'Accuracy of the ResNet on 10000 test images is {100*correct/total:.2f} %.')

```

```
Files already downloaded and verified
Files already downloaded and verified
Epoch: 1, Loss: 7.528
Epoch: 2, Loss: 6.253
Epoch: 3, Loss: 5.459
Epoch: 4, Loss: 4.902
Epoch: 5, Loss: 4.488
Epoch: 6, Loss: 4.183
Epoch: 7, Loss: 3.927
Epoch: 8, Loss: 3.731
Epoch: 9, Loss: 3.544
Epoch: 10, Loss: 3.374
Accuracy of the ResNet on 10000 test images is 46.61 %.
```

2 The training loss and the test accuracy

3 The complete code of `main.py`

The complete code is in `main.py` under the same fold of the report.

Acknowledgement: Thank Yihan Zhou (周亦涵) for the discussion about this homework.