Homework Assignment 4

Name: Linqi Xiao NetID: lx130

Model

```python
class LeNet(nn.Module):
    def __init__(self):
        super(LeNet, self).__init__()
        ############################
        #### Put your code here ####
        ############################
        self.model = nn.Sequential(
            nn.Conv2d(3, 6, kernel_size=5),
            nn.Tanh(),
            nn.MaxPool2d(kernel_size=2, stride=2),
            nn.Conv2d(6, 16, kernel_size=5),
            nn.Tanh(),
            nn.MaxPool2d(kernel_size=2, stride=2),
            nn.Conv2d(16, 120, kernel_size=5),
            nn.Tanh(),
            nn.Flatten(),
            nn.Linear(in_features=120, out_features=84),
            nn.Tanh(),
            nn.Linear(in_features=84, out_features=10)
        )
        ############################
        #### End of your codes ####
        ############################

    def forward(self, x):
        ############################
        #### Put your code here ####
        ############################
        x = self.model(x)
        ############################
        #### End of your codes ####
        ############################
        return x
```

```python
class LeNetWithDropout(nn.Module):
    def __init__(self):
        super(LeNetWithDropout, self).__init__()
        ############################
        #### Put your code here ####
        ############################
        self.model = nn.Sequential(
            nn.Conv2d(3, 6, kernel_size=5),
            nn.Tanh(),
            nn.MaxPool2d(kernel_size=2, stride=2),
            nn.Conv2d(6, 16, kernel_size=5),
            nn.Tanh(),
            nn.MaxPool2d(kernel_size=2, stride=2),
            nn.Conv2d(16, 120, kernel_size=5),
            nn.Tanh(),
            nn.Flatten(),
            nn.Linear(in_features=120, out_features=84),
            nn.Tanh(),
            nn.Dropout(),
            nn.Linear(in_features=84, out_features=10)
        )
        ############################
        #### End of your codes ####
        ############################

    def forward(self, x):
        ############################
        #### Put your code here ####
        ############################
        x = self.model(x)
        ############################
        #### End of your codes ####
        ############################
        return x
```

```python
class LeNetWithBN(nn.Module):
    def __init__(self):
        super(LeNetWithBN, self).__init__()
        ############################
        #### Put your code here ####
        ############################
        self.model = nn.Sequential(
            nn.Conv2d(3, 6, kernel_size=5),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2),
            nn.Conv2d(6, 16, kernel_size=5),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2),
            nn.Conv2d(16, 120, kernel_size=5),
            nn.BatchNorm2d(120),
            nn.ReLU(),
            nn.Flatten(),
            nn.Linear(in_features=120, out_features=84),
            nn.ReLU(),
            nn.Linear(in_features=84, out_features=10)
        )
        ############################
        #### End of your codes ####
        ############################

    def forward(self, x):
        ############################
        #### Put your code here ####
        ############################
        x = self.model(x)
        ############################
        #### End of your codes ####
        ############################
        return x
```

Train

```python
def train(model, device, train_loader, optimizer, epoch, criterion, writer):
    model.train()
    total_loss = 0.
    for batch_idx, (data, target) in enumerate(train_loader):
        data, target = data.to(device), target.to(device)
        ############################
        #### Put your code here ####
        ############################
        optimizer.zero_grad()
        output = model(data)
        loss = criterion(output, target)
        loss.backward()
        optimizer.step()
        total_loss += loss
    train_loss = total_loss/len(train_loader.dataset)
    print("Epoch :{}\tLoss :{:.6f}".format(epoch, train_loss))
    writer.add_scalar("Train Loss", train_loss, epoch)
    writer.flush()
    # return train_loss
        ############################
        #### End of your codes ####
        ############################
        # if batch_idx % 10 == 0:
        #     print('Train Epoch: {} [{}/{} ({:.0f}%)]\tLoss: {:.6f}'.format(
        #         epoch, batch_idx * len(data), len(train_loader.dataset),
        #                100. * batch_idx / len(train_loader), loss.item()/len(train_loader.dataset)))
```

Test

```python
def test(model, device, test_loader, criterion, epoch, writer):
    model.eval()
    test_loss = 0
    correct = 0
    total = 0
    with torch.no_grad():
        for data, target in test_loader:
            data, target = data.to(device), target.to(device)
            output = model(data)
            test_loss += criterion(output, target)
            pred = output.argmax(dim=1, keepdim=True)  # get the index of the max log-probability
            correct += pred.eq(target.view_as(pred)).sum().item()
            total += target.size(0)
        correct /= total
        test_loss /= len(test_loader.dataset)
        print("Average Test Loss :{:.4f}, Accuracy :{:.2f}%\n".format(test_loss, correct * 100.0))
        writer.add_scalar("Test Loss", test_loss, epoch)
        writer.add_scalar("Accuracy", correct, epoch)
        writer.flush()
        # return test_loss, correct
    # print('\nTest set: Average loss: {:.4f}, Accuracy: {}/{} ({:.0f}%)\n'.format(
    #     test_loss, correct, len(test_loader.dataset),
    #     100. * correct / len(test_loader.dataset)))
```

Main

```python
time0 = time.time()
# Training settings
batch_size = 64
epochs = 50
lr = 0.001
save_model = False
torch.manual_seed(100)
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

trainset = torchvision.datasets.CIFAR10(root='./data', train=True, download=True, transform=transform_train)
train_loader = torch.utils.data.DataLoader(trainset, batch_size=128, shuffle=True)
testset = torchvision.datasets.CIFAR10(root='./data', train=False, download=True, transform=transform_test)
test_loader = torch.utils.data.DataLoader(testset, batch_size=100, shuffle=False)

models = {"LeNet": LeNet(), "LeNet With Dropout":LeNetWithDropout(), "LeNet With BN": LeNetWithBN()}
criterion = nn.CrossEntropyLoss()

for model_name in models:
    log_writer = tb_writer(model_name)
    model = models[model_name].to(device=device)
    optimizer = optim.SGD(model.parameters(), lr=lr, momentum=0.9, weight_decay=5e-4)
    for epoch in range(1, epochs + 1):
        train(model, device, train_loader, optimizer, epoch, criterion, log_writer)
        test(model, device, test_loader, criterion, epoch, log_writer)
        log_writer.close()
# if (save_model):
#     torch.save(model.state_dict(), "cifar_lenet.pt")
time1 = time.time()
print('Traning and Testing total excution time is: %s seconds ' % (time1 - time0))
```

Total Time

```
Traning and Testing total excution time is: 1318.7285418510437 seconds
```

Comparison


Show data download links
Ignore outliers in chart scaling
Tooltip sorting method: default
Smoothing
0.6
Horizontal Axis
STEP
RELATIVE
WALL
Runs
Write a regex to filter runs
LeNet
LeNet With Dropout
LeNet With BN
TOGGLE ALL RUNS
Filter tags (regular expressions supported)
Accuracy
Accuracy
Name Smoothed Value Step Time Relative
LeNet With BN 0.6805 0.6806 50 Sat Apr 16, 01:21:31 7m 11s
LeNet With Dropout 0.6073 0.6143 50 Sat Apr 16, 01:14:11 7m 9s
LeNet 0.6415 0.6506 50 Sat Apr 16, 01:06:53 7m 7s
Test Loss
Test Loss
Name Smoothed Value Step Time Relative
LeNet With BN 9.0803e-3 9.0581e-3 50 Sat Apr 16, 01:21:31 7m 11s
LeNet With Dropout 0.01108 0.01093 50 Sat Apr 16, 01:14:11 7m 9s
LeNet 0.01024 0.01001 50 Sat Apr 16, 01:06:53 7m 7s
F:\PythonProject\ECE579_Intro_to_DL\HW4\log_dir
Train Loss
Name Smoothed Value Step Time Relative
LeNet With BN 7.5202e-3 7.4836e-3 50 Sat Apr 16, 01:21:30 7m 11s
LeNet With Dropout 9.9314e-3 9.8372e-3 50 Sat Apr 16, 01:14:09 7m 9s
LeNet 8.493e-3 8.434e-3 50 Sat Apr 16, 01:06:51 7m 7s