

## 202001555 지은미

```
In [1]: import torch
import torch.nn as nn
import torch.optim as optim
import torch.nn.functional as F
from torchvision import transforms, datasets
```

```
In [2]: USE_CUDA = torch.cuda.is_available()
DEVICE = torch.device("cuda" if USE_CUDA else "cpu")
```

```
In [3]: EPOCHS = 40
BATCH_SIZE = 64
```

```
In [4]: loader = torch.utils.data.DataLoader(
    datasets.FashionMNIST('../data',
                           train=True,
                           download=True,
                           transform=transforms.Compose([
                               transforms.RandomHorizontalFlip(),
                               transforms.ToTensor()]),
    batch_size=32,
    num_workers=0,
    shuffle=False)

mean = 0.
mean_square = 0.
samples = 0
for images, _ in loader:
    batch_samples = images.size(0)
    images = images.view(batch_samples, images.size(1), -1)
    mean += images.mean(2).sum(0)
    mean_square += (images**2).mean(2).sum(0)
    samples += images.size(2) * images.size(0)

mean /= len(loader.dataset)
mean_square /= len(loader.dataset)

# extra scale factor for unbiased std estimate (it's effectively 1.0)
scale = samples / (samples - 1)
std = torch.sqrt((mean_square - mean**2) * scale)

print("The mean is ", mean)
print("The standard deviation is ", std)
```

The mean is tensor([0.2860])  
The standard deviation is tensor([0.3530])

```
In [5]: train_loader = torch.utils.data.DataLoader(
    datasets.FashionMNIST('../data',
                           train=True,
                           download=True,
                           transform=transforms.Compose([
                               transforms.ToTensor(),
                               transforms.Normalize((mean,), (std,))
                           ])),
    batch_size=BATCH_SIZE, shuffle=True)
test_loader = torch.utils.data.DataLoader(
    datasets.FashionMNIST('../data',
```

```

        train=False,
        transform=transforms.Compose([
            transforms.ToTensor(),
            transforms.Normalize((mean,), (std,))
        ]),
        batch_size=BATCH_SIZE, shuffle=True)

```

In [6]:

```

class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(1, 10, kernel_size=5)
        self.conv2 = nn.Conv2d(10, 20, kernel_size=5)
        self.conv2_drop = nn.Dropout2d()
        self.fc1 = nn.Linear(320, 50)
        self.fc2 = nn.Linear(50, 10)

    def forward(self, x):
        x = F.relu(F.max_pool2d(self.conv1(x), 2))
        x = F.relu(F.max_pool2d(self.conv2_drop(self.conv2(x)), 2))
        x = x.view(-1, 320)
        x = F.relu(self.fc1(x))
        x = F.dropout(x, training=self.training)
        x = self.fc2(x)
        return x

```

In [7]:

```

model = Net().to(DEVICE)
optimizer = optim.SGD(model.parameters(), lr=0.01, momentum=0.5)

```

In [8]:

```

def train(model, train_loader, optimizer, epoch):
    model.train()
    for batch_idx, (data, target) in enumerate(train_loader):
        data, target = data.to(DEVICE), target.to(DEVICE)
        optimizer.zero_grad()
        output = model(data)
        loss = F.cross_entropy(output, target)
        loss.backward()
        optimizer.step()

        if batch_idx % 200 == 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()))

```

In [9]:

```

def evaluate(model, test_loader):
    model.eval()
    test_loss = 0
    correct = 0
    with torch.no_grad():
        for data, target in test_loader:
            data, target = data.to(DEVICE), target.to(DEVICE)
            output = model(data)

            # 배치 오차를 합산
            test_loss += F.cross_entropy(output, target,
                                         reduction='sum').item()

            # 가장 높은 값을 가진 인덱스가 바로 예측값
            pred = output.max(1, keepdim=True)[1]
            correct += pred.eq(target.view_as(pred)).sum().item()

```

```

test_loss /= len(test_loader.dataset)
test_accuracy = 100. * correct / len(test_loader.dataset)
return test_loss, test_accuracy

```

In [10]:

```

for epoch in range(1, EPOCHS + 1):
    train(model, train_loader, optimizer, epoch)
    test_loss, test_accuracy = evaluate(model, test_loader)

    print('[{}] Test Loss: {:.4f}, Accuracy: {:.2f}%'.format(
        epoch, test_loss, test_accuracy))

```

```

Train Epoch: 1 [0/60000 (0%)]    Loss: 2.319977
Train Epoch: 1 [12800/60000 (21%)]    Loss: 1.588379
Train Epoch: 1 [25600/60000 (43%)]    Loss: 0.911626
Train Epoch: 1 [38400/60000 (64%)]    Loss: 0.955980
Train Epoch: 1 [51200/60000 (85%)]    Loss: 1.017538
[1] Test Loss: 0.7271, Accuracy: 72.17%
Train Epoch: 2 [0/60000 (0%)]    Loss: 0.750698
Train Epoch: 2 [12800/60000 (21%)]    Loss: 0.907721
Train Epoch: 2 [25600/60000 (43%)]    Loss: 0.674810
Train Epoch: 2 [38400/60000 (64%)]    Loss: 0.791311
Train Epoch: 2 [51200/60000 (85%)]    Loss: 0.587482
[2] Test Loss: 0.6132, Accuracy: 76.81%
Train Epoch: 3 [0/60000 (0%)]    Loss: 0.838469
Train Epoch: 3 [12800/60000 (21%)]    Loss: 0.679400
Train Epoch: 3 [25600/60000 (43%)]    Loss: 0.943937
Train Epoch: 3 [38400/60000 (64%)]    Loss: 0.678740
Train Epoch: 3 [51200/60000 (85%)]    Loss: 0.548743
[3] Test Loss: 0.5587, Accuracy: 77.84%
Train Epoch: 4 [0/60000 (0%)]    Loss: 0.711359
Train Epoch: 4 [12800/60000 (21%)]    Loss: 0.694363
Train Epoch: 4 [25600/60000 (43%)]    Loss: 0.560752
Train Epoch: 4 [38400/60000 (64%)]    Loss: 0.441485
Train Epoch: 4 [51200/60000 (85%)]    Loss: 0.726723
[4] Test Loss: 0.5250, Accuracy: 78.76%
Train Epoch: 5 [0/60000 (0%)]    Loss: 0.671974
Train Epoch: 5 [12800/60000 (21%)]    Loss: 0.585856
Train Epoch: 5 [25600/60000 (43%)]    Loss: 0.551943
Train Epoch: 5 [38400/60000 (64%)]    Loss: 0.507813
Train Epoch: 5 [51200/60000 (85%)]    Loss: 0.584773
[5] Test Loss: 0.5051, Accuracy: 80.10%
Train Epoch: 6 [0/60000 (0%)]    Loss: 0.889461
Train Epoch: 6 [12800/60000 (21%)]    Loss: 0.586654
Train Epoch: 6 [25600/60000 (43%)]    Loss: 0.485817
Train Epoch: 6 [38400/60000 (64%)]    Loss: 0.633044
Train Epoch: 6 [51200/60000 (85%)]    Loss: 0.573032
[6] Test Loss: 0.4862, Accuracy: 81.98%
Train Epoch: 7 [0/60000 (0%)]    Loss: 0.667470
Train Epoch: 7 [12800/60000 (21%)]    Loss: 0.809613
Train Epoch: 7 [25600/60000 (43%)]    Loss: 0.734363
Train Epoch: 7 [38400/60000 (64%)]    Loss: 0.631630
Train Epoch: 7 [51200/60000 (85%)]    Loss: 0.454155
[7] Test Loss: 0.4717, Accuracy: 82.66%
Train Epoch: 8 [0/60000 (0%)]    Loss: 0.719791
Train Epoch: 8 [12800/60000 (21%)]    Loss: 0.480072
Train Epoch: 8 [25600/60000 (43%)]    Loss: 0.736078
Train Epoch: 8 [38400/60000 (64%)]    Loss: 0.692172
Train Epoch: 8 [51200/60000 (85%)]    Loss: 0.606108
[8] Test Loss: 0.4554, Accuracy: 82.83%
Train Epoch: 9 [0/60000 (0%)]    Loss: 0.626469
Train Epoch: 9 [12800/60000 (21%)]    Loss: 0.620480
Train Epoch: 9 [25600/60000 (43%)]    Loss: 0.489601
Train Epoch: 9 [38400/60000 (64%)]    Loss: 0.550617
Train Epoch: 9 [51200/60000 (85%)]    Loss: 0.550356
[9] Test Loss: 0.4373, Accuracy: 83.72%
Train Epoch: 10 [0/60000 (0%)]    Loss: 0.459419

```

```

Train Epoch: 10 [12800/60000 (21%)] Loss: 0.503605
Train Epoch: 10 [25600/60000 (43%)] Loss: 0.505842
Train Epoch: 10 [38400/60000 (64%)] Loss: 0.409993
Train Epoch: 10 [51200/60000 (85%)] Loss: 0.492366
[10] Test Loss: 0.4262, Accuracy: 84.30%
Train Epoch: 11 [0/60000 (0%)] Loss: 0.673038
Train Epoch: 11 [12800/60000 (21%)] Loss: 0.477576
Train Epoch: 11 [25600/60000 (43%)] Loss: 0.508039
Train Epoch: 11 [38400/60000 (64%)] Loss: 0.524533
Train Epoch: 11 [51200/60000 (85%)] Loss: 0.560572
[11] Test Loss: 0.4123, Accuracy: 84.88%
Train Epoch: 12 [0/60000 (0%)] Loss: 0.481200
Train Epoch: 12 [12800/60000 (21%)] Loss: 0.503749
Train Epoch: 12 [25600/60000 (43%)] Loss: 0.338469
Train Epoch: 12 [38400/60000 (64%)] Loss: 0.465317
Train Epoch: 12 [51200/60000 (85%)] Loss: 0.570132
[12] Test Loss: 0.4030, Accuracy: 85.20%
Train Epoch: 13 [0/60000 (0%)] Loss: 0.623183
Train Epoch: 13 [12800/60000 (21%)] Loss: 0.494937
Train Epoch: 13 [25600/60000 (43%)] Loss: 0.443710
Train Epoch: 13 [38400/60000 (64%)] Loss: 0.401626
Train Epoch: 13 [51200/60000 (85%)] Loss: 0.600605
[13] Test Loss: 0.3914, Accuracy: 85.62%
Train Epoch: 14 [0/60000 (0%)] Loss: 0.432251
Train Epoch: 14 [12800/60000 (21%)] Loss: 0.540863
Train Epoch: 14 [25600/60000 (43%)] Loss: 0.563019
Train Epoch: 14 [38400/60000 (64%)] Loss: 0.359335
Train Epoch: 14 [51200/60000 (85%)] Loss: 0.707939
[14] Test Loss: 0.3902, Accuracy: 85.58%
Train Epoch: 15 [0/60000 (0%)] Loss: 0.528391
Train Epoch: 15 [12800/60000 (21%)] Loss: 0.390835
Train Epoch: 15 [25600/60000 (43%)] Loss: 0.488475
Train Epoch: 15 [38400/60000 (64%)] Loss: 0.576490
Train Epoch: 15 [51200/60000 (85%)] Loss: 0.443603
[15] Test Loss: 0.3820, Accuracy: 85.75%
Train Epoch: 16 [0/60000 (0%)] Loss: 0.345115
Train Epoch: 16 [12800/60000 (21%)] Loss: 0.473607
Train Epoch: 16 [25600/60000 (43%)] Loss: 0.393219
Train Epoch: 16 [38400/60000 (64%)] Loss: 0.567626
Train Epoch: 16 [51200/60000 (85%)] Loss: 0.452330
[16] Test Loss: 0.3769, Accuracy: 86.08%
Train Epoch: 17 [0/60000 (0%)] Loss: 0.290755
Train Epoch: 17 [12800/60000 (21%)] Loss: 0.572357
Train Epoch: 17 [25600/60000 (43%)] Loss: 0.548765
Train Epoch: 17 [38400/60000 (64%)] Loss: 0.381994
Train Epoch: 17 [51200/60000 (85%)] Loss: 0.616695
[17] Test Loss: 0.3685, Accuracy: 86.35%
Train Epoch: 18 [0/60000 (0%)] Loss: 0.494723
Train Epoch: 18 [12800/60000 (21%)] Loss: 0.446014
Train Epoch: 18 [25600/60000 (43%)] Loss: 0.638823
Train Epoch: 18 [38400/60000 (64%)] Loss: 0.437493
Train Epoch: 18 [51200/60000 (85%)] Loss: 0.581493
[18] Test Loss: 0.3691, Accuracy: 86.46%
Train Epoch: 19 [0/60000 (0%)] Loss: 0.398640
Train Epoch: 19 [12800/60000 (21%)] Loss: 0.403329
Train Epoch: 19 [25600/60000 (43%)] Loss: 0.463190
Train Epoch: 19 [38400/60000 (64%)] Loss: 0.492799
Train Epoch: 19 [51200/60000 (85%)] Loss: 0.370772
[19] Test Loss: 0.3621, Accuracy: 86.59%
Train Epoch: 20 [0/60000 (0%)] Loss: 0.373118
Train Epoch: 20 [12800/60000 (21%)] Loss: 0.484145
Train Epoch: 20 [25600/60000 (43%)] Loss: 0.567737
Train Epoch: 20 [38400/60000 (64%)] Loss: 0.454376
Train Epoch: 20 [51200/60000 (85%)] Loss: 0.424503
[20] Test Loss: 0.3627, Accuracy: 86.94%
Train Epoch: 21 [0/60000 (0%)] Loss: 0.337388
Train Epoch: 21 [12800/60000 (21%)] Loss: 0.468200
Train Epoch: 21 [25600/60000 (43%)] Loss: 0.270211
Train Epoch: 21 [38400/60000 (64%)] Loss: 0.311423

```

```

Train Epoch: 21 [51200/60000 (85%)] Loss: 0.239775
[21] Test Loss: 0.3506, Accuracy: 87.11%
Train Epoch: 22 [0/60000 (0%)] Loss: 0.452401
Train Epoch: 22 [12800/60000 (21%)] Loss: 0.425303
Train Epoch: 22 [25600/60000 (43%)] Loss: 0.378685
Train Epoch: 22 [38400/60000 (64%)] Loss: 0.346386
Train Epoch: 22 [51200/60000 (85%)] Loss: 0.462733
[22] Test Loss: 0.3497, Accuracy: 87.44%
Train Epoch: 23 [0/60000 (0%)] Loss: 0.512982
Train Epoch: 23 [12800/60000 (21%)] Loss: 0.437987
Train Epoch: 23 [25600/60000 (43%)] Loss: 0.560622
Train Epoch: 23 [38400/60000 (64%)] Loss: 0.628212
Train Epoch: 23 [51200/60000 (85%)] Loss: 0.296091
[23] Test Loss: 0.3481, Accuracy: 87.24%
Train Epoch: 24 [0/60000 (0%)] Loss: 0.433579
Train Epoch: 24 [12800/60000 (21%)] Loss: 0.329710
Train Epoch: 24 [25600/60000 (43%)] Loss: 0.392465
Train Epoch: 24 [38400/60000 (64%)] Loss: 0.569929
Train Epoch: 24 [51200/60000 (85%)] Loss: 0.485383
[24] Test Loss: 0.3446, Accuracy: 87.41%
Train Epoch: 25 [0/60000 (0%)] Loss: 0.500091
Train Epoch: 25 [12800/60000 (21%)] Loss: 0.393415
Train Epoch: 25 [25600/60000 (43%)] Loss: 0.470903
Train Epoch: 25 [38400/60000 (64%)] Loss: 0.287376
Train Epoch: 25 [51200/60000 (85%)] Loss: 0.456667
[25] Test Loss: 0.3479, Accuracy: 87.59%
Train Epoch: 26 [0/60000 (0%)] Loss: 0.494848
Train Epoch: 26 [12800/60000 (21%)] Loss: 0.446524
Train Epoch: 26 [25600/60000 (43%)] Loss: 0.368122
Train Epoch: 26 [38400/60000 (64%)] Loss: 0.570919
Train Epoch: 26 [51200/60000 (85%)] Loss: 0.345025
[26] Test Loss: 0.3427, Accuracy: 87.44%
Train Epoch: 27 [0/60000 (0%)] Loss: 0.336383
Train Epoch: 27 [12800/60000 (21%)] Loss: 0.270558
Train Epoch: 27 [25600/60000 (43%)] Loss: 0.488943
Train Epoch: 27 [38400/60000 (64%)] Loss: 0.631811
Train Epoch: 27 [51200/60000 (85%)] Loss: 0.432441
[27] Test Loss: 0.3481, Accuracy: 87.53%
Train Epoch: 28 [0/60000 (0%)] Loss: 0.743866
Train Epoch: 28 [12800/60000 (21%)] Loss: 0.504986
Train Epoch: 28 [25600/60000 (43%)] Loss: 0.376035
Train Epoch: 28 [38400/60000 (64%)] Loss: 0.326081
Train Epoch: 28 [51200/60000 (85%)] Loss: 0.530602
[28] Test Loss: 0.3378, Accuracy: 87.80%
Train Epoch: 29 [0/60000 (0%)] Loss: 0.351070
Train Epoch: 29 [12800/60000 (21%)] Loss: 0.562236
Train Epoch: 29 [25600/60000 (43%)] Loss: 0.613472
Train Epoch: 29 [38400/60000 (64%)] Loss: 0.559397
Train Epoch: 29 [51200/60000 (85%)] Loss: 0.390347
[29] Test Loss: 0.3332, Accuracy: 88.04%
Train Epoch: 30 [0/60000 (0%)] Loss: 0.507118
Train Epoch: 30 [12800/60000 (21%)] Loss: 0.358356
Train Epoch: 30 [25600/60000 (43%)] Loss: 0.458844
Train Epoch: 30 [38400/60000 (64%)] Loss: 0.485223
Train Epoch: 30 [51200/60000 (85%)] Loss: 0.313737
[30] Test Loss: 0.3352, Accuracy: 87.93%
Train Epoch: 31 [0/60000 (0%)] Loss: 0.526629
Train Epoch: 31 [12800/60000 (21%)] Loss: 0.216413
Train Epoch: 31 [25600/60000 (43%)] Loss: 0.288969
Train Epoch: 31 [38400/60000 (64%)] Loss: 0.380071
Train Epoch: 31 [51200/60000 (85%)] Loss: 0.335202
[31] Test Loss: 0.3340, Accuracy: 88.09%
Train Epoch: 32 [0/60000 (0%)] Loss: 0.344233
Train Epoch: 32 [12800/60000 (21%)] Loss: 0.419559
Train Epoch: 32 [25600/60000 (43%)] Loss: 0.379910
Train Epoch: 32 [38400/60000 (64%)] Loss: 0.502644
Train Epoch: 32 [51200/60000 (85%)] Loss: 0.663314
[32] Test Loss: 0.3290, Accuracy: 88.20%
Train Epoch: 33 [0/60000 (0%)] Loss: 0.424478

```

```

Train Epoch: 33 [12800/60000 (21%)] Loss: 0.361744
Train Epoch: 33 [25600/60000 (43%)] Loss: 0.614522
Train Epoch: 33 [38400/60000 (64%)] Loss: 0.352329
Train Epoch: 33 [51200/60000 (85%)] Loss: 0.476837
[33] Test Loss: 0.3312, Accuracy: 87.88%
Train Epoch: 34 [0/60000 (0%)] Loss: 0.515763
Train Epoch: 34 [12800/60000 (21%)] Loss: 0.296344
Train Epoch: 34 [25600/60000 (43%)] Loss: 0.546789
Train Epoch: 34 [38400/60000 (64%)] Loss: 0.425861
Train Epoch: 34 [51200/60000 (85%)] Loss: 0.333236
[34] Test Loss: 0.3291, Accuracy: 88.10%
Train Epoch: 35 [0/60000 (0%)] Loss: 0.501049
Train Epoch: 35 [12800/60000 (21%)] Loss: 0.290423
Train Epoch: 35 [25600/60000 (43%)] Loss: 0.407065
Train Epoch: 35 [38400/60000 (64%)] Loss: 0.326234
Train Epoch: 35 [51200/60000 (85%)] Loss: 0.534543
[35] Test Loss: 0.3322, Accuracy: 87.61%
Train Epoch: 36 [0/60000 (0%)] Loss: 0.445086
Train Epoch: 36 [12800/60000 (21%)] Loss: 0.371631
Train Epoch: 36 [25600/60000 (43%)] Loss: 0.380554
Train Epoch: 36 [38400/60000 (64%)] Loss: 0.552887
Train Epoch: 36 [51200/60000 (85%)] Loss: 0.359830
[36] Test Loss: 0.3251, Accuracy: 88.45%
Train Epoch: 37 [0/60000 (0%)] Loss: 0.472712
Train Epoch: 37 [12800/60000 (21%)] Loss: 0.321023
Train Epoch: 37 [25600/60000 (43%)] Loss: 0.391652
Train Epoch: 37 [38400/60000 (64%)] Loss: 0.465972
Train Epoch: 37 [51200/60000 (85%)] Loss: 0.390703
[37] Test Loss: 0.3367, Accuracy: 87.61%
Train Epoch: 38 [0/60000 (0%)] Loss: 0.328333
Train Epoch: 38 [12800/60000 (21%)] Loss: 0.387482
Train Epoch: 38 [25600/60000 (43%)] Loss: 0.377134
Train Epoch: 38 [38400/60000 (64%)] Loss: 0.368343
Train Epoch: 38 [51200/60000 (85%)] Loss: 0.432579
[38] Test Loss: 0.3247, Accuracy: 88.30%
Train Epoch: 39 [0/60000 (0%)] Loss: 0.621666
Train Epoch: 39 [12800/60000 (21%)] Loss: 0.283878
Train Epoch: 39 [25600/60000 (43%)] Loss: 0.241283
Train Epoch: 39 [38400/60000 (64%)] Loss: 0.407759
Train Epoch: 39 [51200/60000 (85%)] Loss: 0.311443
[39] Test Loss: 0.3209, Accuracy: 88.47%
Train Epoch: 40 [0/60000 (0%)] Loss: 0.195035
Train Epoch: 40 [12800/60000 (21%)] Loss: 0.362013
Train Epoch: 40 [25600/60000 (43%)] Loss: 0.402263
Train Epoch: 40 [38400/60000 (64%)] Loss: 0.556012
Train Epoch: 40 [51200/60000 (85%)] Loss: 0.330151
[40] Test Loss: 0.3180, Accuracy: 88.71%

```

202001555 지은미