```
## uncomment these if you upload this on google drive and mount the
drive
# from google.colab import drive
import torch
import torchvision
import torchvision.transforms as transforms
import matplotlib.pyplot as plt
import numpy as np
import torch.nn as nn
import torch.nn.functional as F
import torch.optim as optim
# drive.mount('/content/gdrive', force_remount=True)
```

Q1. Loading Data

Run the below cell to load CIFAR-10 train and test data. Answer the corresponding questions in the overleaf document

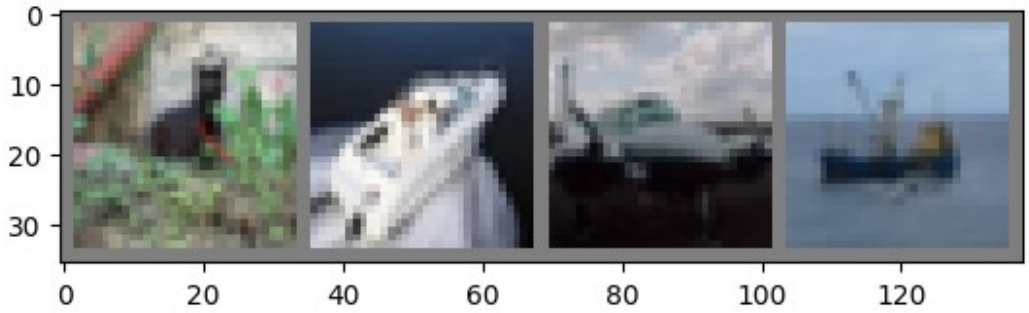
```
## Define transforms to apply on images
transform = transforms.Compose(
    [transforms.ToTensor(),
    transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5))
    ])

## defining training and test data
train_data = torchvision.datasets.CIFAR10(root='./data', train=True,
download=True, transform=transform)
test_data = torchvision.datasets.CIFAR10(root='./data', train=False,
download=True, transform=transform)

\
## creating data loaders
batch_size = 4 ## set the batch size value
train_loader = torch.utils.data.DataLoader(train_data,
batch_size=batch_size, shuffle=True, num_workers=2)
test_loader = torch.utils.data.DataLoader(test_data,
batch_size=batch_size, shuffle=False, num_workers=2)

## image labels in cifar 10
class_labels = classes = ('plane', 'car', 'bird', 'cat', 'deer',
'dog', 'frog', 'horse', 'ship', 'truck')
```

```
Downloading https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz to
./data/cifar-10-python.tar.gz

100%|██████████| 170498071/170498071 [00:03<00:00, 44684559.42it/s]

Extracting ./data/cifar-10-python.tar.gz to ./data
Files already downloaded and verified
```

Helper function

To display images in the training set

```
# function to display images in the training set
def display(img):
    img = img / 2 + 0.5     # unnormalize
    npimg = img.numpy()
    plt.imshow(np.transpose(npimg, (1, 2, 0)))
    plt.show()


## displaying images in 1 batch of the training set

# get 1 batch of training images
dataiter = iter(train_loader)
images, labels = next(dataiter)

# show images
display(torchvision.utils.make_grid(images[0:4]))
# print labels
print(' '.join('%5s' % class_labels[labels[j]] for j in range(4)))
```




```
  cat  ship plane  ship
```

Q2. Classifier Architecture

```
## Defining Classifier architecture

class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(in_channels=3, out_channels=6,
kernel_size=5, stride=1, padding=0)
        self.pool = nn.MaxPool2d(2, 2)
        self.conv2 = nn.Conv2d(in_channels=6, out_channels=16,
kernel_size=5, stride=1, padding=0)
        self.fc1 = nn.Linear(16 * 5 * 5, 120)
        self.fc2 = nn.Linear(120, 84)
```

```
        self.fc3 = nn.Linear(84, 10)

    def forward(self, x):
        x = self.pool(F.relu(self.conv1(x)))
        x = self.pool(F.relu(self.conv2(x)))
        x = x.view(-1, 16 * 5 * 5)
        x = F.relu(self.fc1(x))
        x = F.relu(self.fc2(x))
        x = self.fc3(x)
        return x
```

Q3. Training the network

(i) Training on CPU

```
### Complete the code in the training box

## for reproducibility
torch.manual_seed(7)
np.random.seed(7)

net = Net().cuda()
criterion = nn.CrossEntropyLoss()
optimizer = optim.SGD(net.parameters(), lr=0.001, momentum=0.9)



num_epochs = 3
running_loss_list = [] # list to store running loss in the code below
for epoch in range(num_epochs):  # loop over the dataset multiple
times
    running_loss = 0.0
    for i, data in enumerate(train_loader, 0):
        # get the inputs; data is a list of [inputs, labels]
        inputs, labels = data
        #================================================#
        # Fill in the training loop here.
        #================================================#

        optimizer.zero_grad()
        outputs = net(inputs.cuda())
        loss = criterion(outputs, labels.cuda())
        loss.backward()
        optimizer.step()
        running_loss += loss.cpu().item()
        if i % 250 == 249:    # print every 250 mini-batches
            print('[{}, {}] loss: {:.3f}'.format(epoch + 1, i + 1,
running_loss / 250))
            running_loss_list.append(running_loss)
```

```
            running_loss = 0.0

print('Training Complete')
PATH = './net.pth'
torch.save(net.state_dict(), PATH)

## complete the code to plot the running loss per 250 mini batches
curve

def plot_loss_curve(running_loss_list):
    ## complete code
    plt.plot(running_loss_list)
    plt.xlabel('Mini Batch')
    plt.ylabel('Loss')
    plt.title('Loss Curve')
    plt.show()
plot_loss_curve
```

```
[1, 250] loss: 2.304
[1, 500] loss: 2.300
[1, 750] loss: 2.298
[1, 1000] loss: 2.290
[1, 1250] loss: 2.268
[1, 1500] loss: 2.203
[1, 1750] loss: 2.095
[1, 2000] loss: 2.037
[1, 2250] loss: 1.984
[1, 2500] loss: 1.948
[1, 2750] loss: 1.951
[1, 3000] loss: 1.860
[1, 3250] loss: 1.842
[1, 3500] loss: 1.788
[1, 3750] loss: 1.749
[1, 4000] loss: 1.759
[1, 4250] loss: 1.712
[1, 4500] loss: 1.718
[1, 4750] loss: 1.691
[1, 5000] loss: 1.681
[1, 5250] loss: 1.634
[1, 5500] loss: 1.668
[1, 5750] loss: 1.697
[1, 6000] loss: 1.648
[1, 6250] loss: 1.573
[1, 6500] loss: 1.604
[1, 6750] loss: 1.550
[1, 7000] loss: 1.561
[1, 7250] loss: 1.521
[1, 7500] loss: 1.527
[1, 7750] loss: 1.534
[1, 8000] loss: 1.512
```

```
[1, 8250] loss: 1.538
[1, 8500] loss: 1.490
[1, 8750] loss: 1.493
[1, 9000] loss: 1.532
[1, 9250] loss: 1.506
[1, 9500] loss: 1.474
[1, 9750] loss: 1.477
[1, 10000] loss: 1.409
[1, 10250] loss: 1.471
[1, 10500] loss: 1.461
[1, 10750] loss: 1.433
[1, 11000] loss: 1.422
[1, 11250] loss: 1.465
[1, 11500] loss: 1.385
[1, 11750] loss: 1.448
[1, 12000] loss: 1.397
[1, 12250] loss: 1.422
[1, 12500] loss: 1.478
[2, 250] loss: 1.379
[2, 500] loss: 1.420
[2, 750] loss: 1.449
[2, 1000] loss: 1.333
[2, 1250] loss: 1.411
[2, 1500] loss: 1.364
[2, 1750] loss: 1.382
[2, 2000] loss: 1.420
[2, 2250] loss: 1.383
[2, 2500] loss: 1.422
[2, 2750] loss: 1.376
[2, 3000] loss: 1.398
[2, 3250] loss: 1.359
[2, 3500] loss: 1.320
[2, 3750] loss: 1.345
[2, 4000] loss: 1.341
[2, 4250] loss: 1.433
[2, 4500] loss: 1.300
[2, 4750] loss: 1.276
[2, 5000] loss: 1.275
[2, 5250] loss: 1.314
[2, 5500] loss: 1.325
[2, 5750] loss: 1.311
[2, 6000] loss: 1.311
[2, 6250] loss: 1.294
[2, 6500] loss: 1.388
[2, 6750] loss: 1.317
[2, 7000] loss: 1.289
[2, 7250] loss: 1.295
[2, 7500] loss: 1.340
[2, 7750] loss: 1.335
```

```
[2, 8000] loss: 1.310
[2, 8250] loss: 1.320
[2, 8500] loss: 1.308
[2, 8750] loss: 1.411
[2, 9000] loss: 1.342
[2, 9250] loss: 1.299
[2, 9500] loss: 1.255
[2, 9750] loss: 1.311
[2, 10000] loss: 1.275
[2, 10250] loss: 1.257
[2, 10500] loss: 1.249
[2, 10750] loss: 1.283
[2, 11000] loss: 1.241
[2, 11250] loss: 1.291
[2, 11500] loss: 1.262
[2, 11750] loss: 1.255
[2, 12000] loss: 1.298
[2, 12250] loss: 1.305
[2, 12500] loss: 1.264
[3, 250] loss: 1.227
[3, 500] loss: 1.215
[3, 750] loss: 1.240
[3, 1000] loss: 1.181
[3, 1250] loss: 1.241
[3, 1500] loss: 1.225
[3, 1750] loss: 1.163
[3, 2000] loss: 1.226
[3, 2250] loss: 1.221
[3, 2500] loss: 1.127
[3, 2750] loss: 1.250
[3, 3000] loss: 1.267
[3, 3250] loss: 1.216
[3, 3500] loss: 1.182
[3, 3750] loss: 1.166
[3, 4000] loss: 1.173
[3, 4250] loss: 1.195
[3, 4500] loss: 1.209
[3, 4750] loss: 1.169
[3, 5000] loss: 1.202
[3, 5250] loss: 1.115
[3, 5500] loss: 1.235
[3, 5750] loss: 1.232
[3, 6000] loss: 1.155
[3, 6250] loss: 1.211
[3, 6500] loss: 1.119
[3, 6750] loss: 1.228
[3, 7000] loss: 1.179
[3, 7250] loss: 1.210
[3, 7500] loss: 1.232
```

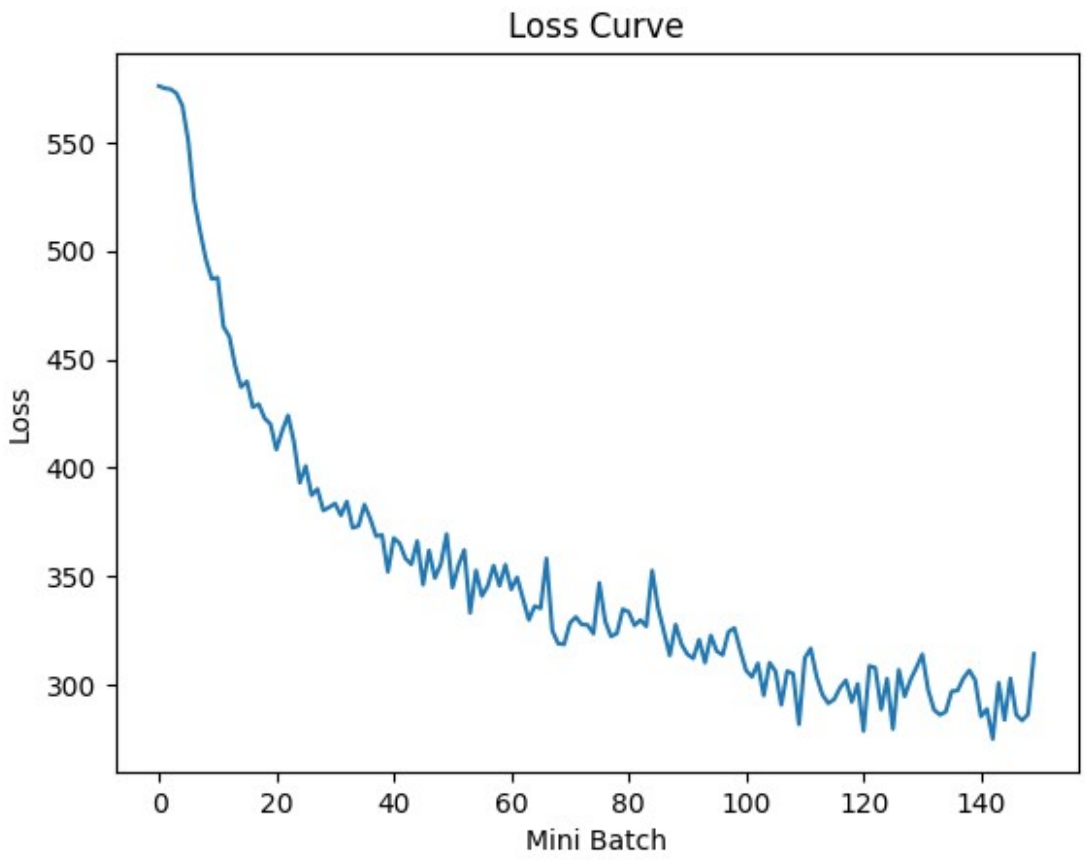
```
[3, 7750] loss: 1.256
[3, 8000] loss: 1.191
[3, 8250] loss: 1.154
[3, 8500] loss: 1.145
[3, 8750] loss: 1.150
[3, 9000] loss: 1.188
[3, 9250] loss: 1.190
[3, 9500] loss: 1.212
[3, 9750] loss: 1.227
[3, 10000] loss: 1.209
[3, 10250] loss: 1.142
[3, 10500] loss: 1.156
[3, 10750] loss: 1.101
[3, 11000] loss: 1.204
[3, 11250] loss: 1.136
[3, 11500] loss: 1.212
[3, 11750] loss: 1.145
[3, 12000] loss: 1.134
[3, 12250] loss: 1.146
[3, 12500] loss: 1.257
Training Complete
```

```
<function __main__.plot_loss_curve(running_loss_list)>
```

```
print(running_loss_list)

plot_loss_curve(running_loss_list)
```

```
[575.9426784515381, 574.9145486354828, 574.477303981781,
572.5861060619354, 566.8765780925751, 550.7392921447754,
523.676367521286, 509.2552273273468, 496.07709288597107,
487.0847477912903, 487.6734608411789, 465.06269657611847,
460.5263193845749, 446.92022383213043, 437.27167642116547,
439.8478503227234, 427.9621824026108, 429.3879623413086,
422.8744004368782, 420.20217168331146, 408.45006960630417,
417.10811507701874, 424.21419137716293, 411.99886453151703,
393.19971761107445, 400.88916015625, 387.4152858555317,
390.30257219076157, 380.3558091968298, 381.83937072753906,
383.597684442997, 378.0983758568764, 384.4962408840656,
372.4111567735672, 373.3185179233551, 383.0832875967026,
376.53929087519646, 368.5308955311775, 369.12563982605934,
352.1565980911255, 367.6573314666748, 365.18990433216095,
358.2076831459999, 355.57538209855556, 366.3064443767071,
346.2289213836193, 361.97862726449966, 349.25835117697716,
355.57682536542416, 369.5158163905144, 344.8566963970661,
354.9369367957115, 362.18217143416405, 333.2006102800369,
352.7920006811619, 340.9748383462429, 345.55875664949417,
354.97292268276215, 345.7056069970131, 355.38515585660934,
344.0593041777611, 349.5685260295868, 339.7939098775387,
329.99112175405025, 336.31762850284576, 335.267880320549,
```

358.25916332006454, 325.0104281306267, 319.0476578325033,
318.70647893846035, 328.52448999881744, 331.3108091801405,
327.8543696850538, 327.6834681481123, 323.6137529462576,
347.032578766346, 329.35814568400383, 322.35706701874733,
323.7905030846596, 334.90039750933647, 333.6609027683735,
327.4285028874874, 329.91883742809296, 327.03217455744743,
352.661311596632, 335.5636506676674, 324.73115703463554,
313.6446585059166, 327.8257195651531, 318.70660960674286,
314.1358596086502, 312.255415096879, 320.8519122414291,
310.2666431069374, 322.78620406985283, 315.57023787498474,
313.80675783753395, 324.4807640314102, 326.2023895084858,
316.0271311700344, 306.6758610457182, 303.7258261665702,
309.8938230276108, 295.1457948386669, 310.2210495173931,
306.32151260226965, 290.8275679945946, 306.3840356916189,
305.22077448666096, 281.8749389052391, 312.4429285675287,
316.7967427968979, 303.9574109762907, 295.486321516335,
291.51427268981934, 293.36662462353706, 298.72821497917175,
302.1418512314558, 292.22017355263233, 300.48421926796436,
278.81348472833633, 308.7488471567631, 307.88165947794914,
288.7134289741516, 302.86087638139725, 279.7093053907156,
306.9665107652545, 294.7287862151861, 302.38229209184647,
308.0148822814226, 314.1127460002899, 297.6950342208147,
288.55130533128977, 286.23604914546013, 287.6189216077328,
297.0562717318535, 297.4061709344387, 303.05449518561363,
306.7074561417103, 302.26868687570095, 285.50780195742846,
288.9041797965765, 275.1623956412077, 300.9789221212268,
283.8876731842756, 302.9830614924431, 286.28185512684286,
283.5910819172859, 286.4790594615042, 314.22460575401783]



(ii) Paste the above code in the code block below and modify it to use GPUs for training

```
### Copy the code from (i), and modify it to run on GPUs for 20 epochs

net = Net()
device = torch.device("cuda:0" if torch.cuda.is_available() else
"cpu")
net.to(device)
optimizer = optim.SGD(net.parameters(), lr=0.001, momentum=0.9)

num_epochs = 20
running_loss_list = []

for epoch in range(num_epochs):
    running_loss = 0.0
    for i, data in enumerate(train_loader, 0):
        inputs, labels = data
        inputs, labels = inputs.to(device), labels.to(device)
        optimizer.zero_grad()
        outputs = net(inputs)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()
```

```
        running_loss += loss.item()
        if i % 250 == 249:
            print('[{}, {}] loss: {:.3f}'.format(epoch + 1, i + 1,
running_loss / 250))
            running_loss_list.append(running_loss / 250)
            running_loss = 0.0

print('Training Complete')

PATH = './net.pth'
torch.save(net.state_dict(), PATH)

# Plotting the loss curve
import matplotlib.pyplot as plt

def plot_loss_curve(running_loss_list):
    plt.plot(running_loss_list)
    plt.xlabel('Mini Batch')
    plt.ylabel('Loss')
    plt.title('Loss Curve')
    plt.show()

plot_loss_curve(running_loss_list)
```

```
[1, 250] loss: 2.303
[1, 500] loss: 2.303
[1, 750] loss: 2.298
[1, 1000] loss: 2.294
[1, 1250] loss: 2.269
[1, 1500] loss: 2.204
[1, 1750] loss: 2.132
[1, 2000] loss: 2.079
[1, 2250] loss: 2.029
[1, 2500] loss: 2.016
[1, 2750] loss: 2.007
[1, 3000] loss: 1.939
[1, 3250] loss: 1.867
[1, 3500] loss: 1.932
[1, 3750] loss: 1.898
[1, 4000] loss: 1.898
[1, 4250] loss: 1.857
[1, 4500] loss: 1.795
[1, 4750] loss: 1.798
[1, 5000] loss: 1.741
[1, 5250] loss: 1.731
[1, 5500] loss: 1.693
[1, 5750] loss: 1.662
[1, 6000] loss: 1.673
[1, 6250] loss: 1.639
[1, 6500] loss: 1.669
```

```
[1, 6750] loss: 1.637
[1, 7000] loss: 1.612
[1, 7250] loss: 1.592
[1, 7500] loss: 1.609
[1, 7750] loss: 1.576
[1, 8000] loss: 1.554
[1, 8250] loss: 1.626
[1, 8500] loss: 1.561
[1, 8750] loss: 1.538
[1, 9000] loss: 1.551
[1, 9250] loss: 1.540
[1, 9500] loss: 1.568
[1, 9750] loss: 1.498
[1, 10000] loss: 1.496
[1, 10250] loss: 1.495
[1, 10500] loss: 1.503
[1, 10750] loss: 1.489
[1, 11000] loss: 1.489
[1, 11250] loss: 1.451
[1, 11500] loss: 1.460
[1, 11750] loss: 1.506
[1, 12000] loss: 1.472
[1, 12250] loss: 1.454
[1, 12500] loss: 1.479
[2, 250] loss: 1.436
[2, 500] loss: 1.462
[2, 750] loss: 1.414
[2, 1000] loss: 1.386
[2, 1250] loss: 1.453
[2, 1500] loss: 1.418
[2, 1750] loss: 1.469
[2, 2000] loss: 1.446
[2, 2250] loss: 1.389
[2, 2500] loss: 1.400
[2, 2750] loss: 1.402
[2, 3000] loss: 1.424
[2, 3250] loss: 1.402
[2, 3500] loss: 1.391
[2, 3750] loss: 1.397
[2, 4000] loss: 1.391
[2, 4250] loss: 1.393
[2, 4500] loss: 1.363
[2, 4750] loss: 1.350
[2, 5000] loss: 1.392
[2, 5250] loss: 1.393
[2, 5500] loss: 1.334
[2, 5750] loss: 1.356
[2, 6000] loss: 1.381
[2, 6250] loss: 1.315
```

```
[2, 6500] loss: 1.357
[2, 6750] loss: 1.297
[2, 7000] loss: 1.325
[2, 7250] loss: 1.366
[2, 7500] loss: 1.406
[2, 7750] loss: 1.322
[2, 8000] loss: 1.368
[2, 8250] loss: 1.355
[2, 8500] loss: 1.307
[2, 8750] loss: 1.310
[2, 9000] loss: 1.282
[2, 9250] loss: 1.341
[2, 9500] loss: 1.246
[2, 9750] loss: 1.336
[2, 10000] loss: 1.295
[2, 10250] loss: 1.404
[2, 10500] loss: 1.312
[2, 10750] loss: 1.345
[2, 11000] loss: 1.306
[2, 11250] loss: 1.304
[2, 11500] loss: 1.301
[2, 11750] loss: 1.380
[2, 12000] loss: 1.293
[2, 12250] loss: 1.291
[2, 12500] loss: 1.297
[3, 250] loss: 1.235
[3, 500] loss: 1.260
[3, 750] loss: 1.318
[3, 1000] loss: 1.253
[3, 1250] loss: 1.273
[3, 1500] loss: 1.211
[3, 1750] loss: 1.215
[3, 2000] loss: 1.273
[3, 2250] loss: 1.269
[3, 2500] loss: 1.246
[3, 2750] loss: 1.252
[3, 3000] loss: 1.268
[3, 3250] loss: 1.212
[3, 3500] loss: 1.208
[3, 3750] loss: 1.253
[3, 4000] loss: 1.190
[3, 4250] loss: 1.265
[3, 4500] loss: 1.230
[3, 4750] loss: 1.207
[3, 5000] loss: 1.242
[3, 5250] loss: 1.252
[3, 5500] loss: 1.303
[3, 5750] loss: 1.234
[3, 6000] loss: 1.268
```

```
[3, 6250] loss: 1.208
[3, 6500] loss: 1.210
[3, 6750] loss: 1.237
[3, 7000] loss: 1.245
[3, 7250] loss: 1.229
[3, 7500] loss: 1.202
[3, 7750] loss: 1.240
[3, 8000] loss: 1.190
[3, 8250] loss: 1.269
[3, 8500] loss: 1.212
[3, 8750] loss: 1.197
[3, 9000] loss: 1.222
[3, 9250] loss: 1.225
[3, 9500] loss: 1.165
[3, 9750] loss: 1.221
[3, 10000] loss: 1.176
[3, 10250] loss: 1.194
[3, 10500] loss: 1.222
[3, 10750] loss: 1.229
[3, 11000] loss: 1.283
[3, 11250] loss: 1.204
[3, 11500] loss: 1.237
[3, 11750] loss: 1.151
[3, 12000] loss: 1.198
[3, 12250] loss: 1.200
[3, 12500] loss: 1.135
[4, 250] loss: 1.135
[4, 500] loss: 1.138
[4, 750] loss: 1.182
[4, 1000] loss: 1.144
[4, 1250] loss: 1.148
[4, 1500] loss: 1.131
[4, 1750] loss: 1.153
[4, 2000] loss: 1.134
[4, 2250] loss: 1.166
[4, 2500] loss: 1.117
[4, 2750] loss: 1.111
[4, 3000] loss: 1.160
[4, 3250] loss: 1.233
[4, 3500] loss: 1.128
[4, 3750] loss: 1.140
[4, 4000] loss: 1.066
[4, 4250] loss: 1.171
[4, 4500] loss: 1.067
[4, 4750] loss: 1.131
[4, 5000] loss: 1.171
[4, 5250] loss: 1.124
[4, 5500] loss: 1.170
[4, 5750] loss: 1.128
```

```
[4, 6000] loss: 1.226
[4, 6250] loss: 1.132
[4, 6500] loss: 1.187
[4, 6750] loss: 1.155
[4, 7000] loss: 1.170
[4, 7250] loss: 1.132
[4, 7500] loss: 1.178
[4, 7750] loss: 1.168
[4, 8000] loss: 1.182
[4, 8250] loss: 1.158
[4, 8500] loss: 1.102
[4, 8750] loss: 1.158
[4, 9000] loss: 1.105
[4, 9250] loss: 1.154
[4, 9500] loss: 1.133
[4, 9750] loss: 1.113
[4, 10000] loss: 1.098
[4, 10250] loss: 1.112
[4, 10500] loss: 1.120
[4, 10750] loss: 1.131
[4, 11000] loss: 1.140
[4, 11250] loss: 1.152
[4, 11500] loss: 1.110
[4, 11750] loss: 1.129
[4, 12000] loss: 1.078
[4, 12250] loss: 1.129
[4, 12500] loss: 1.148
[5, 250] loss: 1.067
[5, 500] loss: 1.050
[5, 750] loss: 1.069
[5, 1000] loss: 1.082
[5, 1250] loss: 1.059
[5, 1500] loss: 1.000
[5, 1750] loss: 1.049
[5, 2000] loss: 1.043
[5, 2250] loss: 1.007
[5, 2500] loss: 1.099
[5, 2750] loss: 1.116
[5, 3000] loss: 1.038
[5, 3250] loss: 1.077
[5, 3500] loss: 1.068
[5, 3750] loss: 1.063
[5, 4000] loss: 1.094
[5, 4250] loss: 1.081
[5, 4500] loss: 1.104
[5, 4750] loss: 0.980
[5, 5000] loss: 1.071
[5, 5250] loss: 1.058
[5, 5500] loss: 1.071
```

```
[5, 5750] loss: 1.046
[5, 6000] loss: 1.069
[5, 6250] loss: 1.178
[5, 6500] loss: 1.144
[5, 6750] loss: 1.020
[5, 7000] loss: 1.118
[5, 7250] loss: 1.086
[5, 7500] loss: 1.017
[5, 7750] loss: 1.107
[5, 8000] loss: 1.134
[5, 8250] loss: 1.057
[5, 8500] loss: 1.045
[5, 8750] loss: 1.079
[5, 9000] loss: 1.100
[5, 9250] loss: 1.090
[5, 9500] loss: 1.039
[5, 9750] loss: 1.066
[5, 10000] loss: 1.063
[5, 10250] loss: 1.136
[5, 10500] loss: 1.048
[5, 10750] loss: 1.100
[5, 11000] loss: 1.127
[5, 11250] loss: 1.036
[5, 11500] loss: 1.099
[5, 11750] loss: 1.104
[5, 12000] loss: 1.069
[5, 12250] loss: 1.042
[5, 12500] loss: 1.085
[6, 250] loss: 0.982
[6, 500] loss: 0.987
[6, 750] loss: 1.005
[6, 1000] loss: 1.036
[6, 1250] loss: 0.966
[6, 1500] loss: 1.018
[6, 1750] loss: 0.975
[6, 2000] loss: 1.030
[6, 2250] loss: 0.961
[6, 2500] loss: 0.980
[6, 2750] loss: 0.961
[6, 3000] loss: 1.007
[6, 3250] loss: 1.065
[6, 3500] loss: 1.002
[6, 3750] loss: 0.994
[6, 4000] loss: 0.997
[6, 4250] loss: 0.988
[6, 4500] loss: 0.979
[6, 4750] loss: 1.059
[6, 5000] loss: 0.995
[6, 5250] loss: 1.036
```

```
[6, 5500] loss: 0.926
[6, 5750] loss: 1.079
[6, 6000] loss: 1.033
[6, 6250] loss: 1.049
[6, 6500] loss: 1.029
[6, 6750] loss: 1.050
[6, 7000] loss: 1.008
[6, 7250] loss: 1.061
[6, 7500] loss: 1.012
[6, 7750] loss: 1.043
[6, 8000] loss: 1.036
[6, 8250] loss: 0.978
[6, 8500] loss: 1.019
[6, 8750] loss: 1.061
[6, 9000] loss: 1.054
[6, 9250] loss: 1.008
[6, 9500] loss: 1.096
[6, 9750] loss: 1.075
[6, 10000] loss: 1.055
[6, 10250] loss: 1.036
[6, 10500] loss: 1.074
[6, 10750] loss: 1.010
[6, 11000] loss: 1.041
[6, 11250] loss: 1.015
[6, 11500] loss: 1.011
[6, 11750] loss: 1.053
[6, 12000] loss: 1.012
[6, 12250] loss: 1.103
[6, 12500] loss: 0.998
[7, 250] loss: 0.909
[7, 500] loss: 0.882
[7, 750] loss: 0.938
[7, 1000] loss: 0.929
[7, 1250] loss: 0.941
[7, 1500] loss: 0.927
[7, 1750] loss: 0.893
[7, 2000] loss: 0.936
[7, 2250] loss: 0.937
[7, 2500] loss: 0.937
[7, 2750] loss: 0.923
[7, 3000] loss: 0.933
[7, 3250] loss: 0.976
[7, 3500] loss: 0.971
[7, 3750] loss: 0.951
[7, 4000] loss: 0.996
[7, 4250] loss: 0.927
[7, 4500] loss: 0.991
[7, 4750] loss: 0.977
[7, 5000] loss: 0.936
```

```
[7, 5250] loss: 0.948
[7, 5500] loss: 0.979
[7, 5750] loss: 0.977
[7, 6000] loss: 0.999
[7, 6250] loss: 0.992
[7, 6500] loss: 0.920
[7, 6750] loss: 0.903
[7, 7000] loss: 0.997
[7, 7250] loss: 1.016
[7, 7500] loss: 0.988
[7, 7750] loss: 1.052
[7, 8000] loss: 0.995
[7, 8250] loss: 1.032
[7, 8500] loss: 1.024
[7, 8750] loss: 0.986
[7, 9000] loss: 1.008
[7, 9250] loss: 0.995
[7, 9500] loss: 1.016
[7, 9750] loss: 0.970
[7, 10000] loss: 1.028
[7, 10250] loss: 0.971
[7, 10500] loss: 0.969
[7, 10750] loss: 0.989
[7, 11000] loss: 1.034
[7, 11250] loss: 0.949
[7, 11500] loss: 1.051
[7, 11750] loss: 1.000
[7, 12000] loss: 1.019
[7, 12250] loss: 1.008
[7, 12500] loss: 1.056
[8, 250] loss: 0.826
[8, 500] loss: 0.865
[8, 750] loss: 0.898
[8, 1000] loss: 0.895
[8, 1250] loss: 0.929
[8, 1500] loss: 0.937
[8, 1750] loss: 0.902
[8, 2000] loss: 0.957
[8, 2250] loss: 0.863
[8, 2500] loss: 0.867
[8, 2750] loss: 0.898
[8, 3000] loss: 0.881
[8, 3250] loss: 0.983
[8, 3500] loss: 0.931
[8, 3750] loss: 0.894
[8, 4000] loss: 0.939
[8, 4250] loss: 0.917
[8, 4500] loss: 0.979
[8, 4750] loss: 0.941
```

```
[8, 5000] loss: 0.853
[8, 5250] loss: 0.935
[8, 5500] loss: 0.927
[8, 5750] loss: 0.935
[8, 6000] loss: 0.959
[8, 6250] loss: 0.951
[8, 6500] loss: 0.854
[8, 6750] loss: 1.024
[8, 7000] loss: 0.931
[8, 7250] loss: 0.881
[8, 7500] loss: 0.901
[8, 7750] loss: 0.981
[8, 8000] loss: 0.973
[8, 8250] loss: 0.986
[8, 8500] loss: 0.970
[8, 8750] loss: 0.975
[8, 9000] loss: 0.962
[8, 9250] loss: 0.920
[8, 9500] loss: 0.965
[8, 9750] loss: 0.971
[8, 10000] loss: 0.954
[8, 10250] loss: 0.951
[8, 10500] loss: 0.976
[8, 10750] loss: 0.984
[8, 11000] loss: 0.953
[8, 11250] loss: 0.964
[8, 11500] loss: 0.951
[8, 11750] loss: 0.960
[8, 12000] loss: 0.913
[8, 12250] loss: 0.993
[8, 12500] loss: 0.952
[9, 250] loss: 0.869
[9, 500] loss: 0.894
[9, 750] loss: 0.903
[9, 1000] loss: 0.874
[9, 1250] loss: 0.901
[9, 1500] loss: 0.865
[9, 1750] loss: 0.926
[9, 2000] loss: 0.850
[9, 2250] loss: 0.872
[9, 2500] loss: 0.872
[9, 2750] loss: 0.887
[9, 3000] loss: 0.860
[9, 3250] loss: 0.903
[9, 3500] loss: 0.852
[9, 3750] loss: 0.920
[9, 4000] loss: 0.869
[9, 4250] loss: 0.898
[9, 4500] loss: 0.889
```

```
[9, 4750] loss: 0.933
[9, 5000] loss: 0.854
[9, 5250] loss: 0.875
[9, 5500] loss: 0.872
[9, 5750] loss: 0.910
[9, 6000] loss: 0.957
[9, 6250] loss: 0.858
[9, 6500] loss: 0.930
[9, 6750] loss: 0.861
[9, 7000] loss: 0.932
[9, 7250] loss: 0.941
[9, 7500] loss: 0.923
[9, 7750] loss: 0.926
[9, 8000] loss: 0.912
[9, 8250] loss: 0.805
[9, 8500] loss: 0.903
[9, 8750] loss: 0.906
[9, 9000] loss: 0.928
[9, 9250] loss: 0.928
[9, 9500] loss: 0.926
[9, 9750] loss: 0.905
[9, 10000] loss: 0.865
[9, 10250] loss: 0.927
[9, 10500] loss: 0.997
[9, 10750] loss: 0.945
[9, 11000] loss: 0.899
[9, 11250] loss: 0.948
[9, 11500] loss: 0.922
[9, 11750] loss: 0.860
[9, 12000] loss: 0.899
[9, 12250] loss: 0.943
[9, 12500] loss: 0.926
[10, 250] loss: 0.822
[10, 500] loss: 0.829
[10, 750] loss: 0.872
[10, 1000] loss: 0.783
[10, 1250] loss: 0.859
[10, 1500] loss: 0.814
[10, 1750] loss: 0.847
[10, 2000] loss: 0.763
[10, 2250] loss: 0.891
[10, 2500] loss: 0.895
[10, 2750] loss: 0.833
[10, 3000] loss: 0.909
[10, 3250] loss: 0.895
[10, 3500] loss: 0.873
[10, 3750] loss: 0.811
[10, 4000] loss: 0.845
[10, 4250] loss: 0.913
```

```
[10, 4500] loss: 0.849
[10, 4750] loss: 0.882
[10, 5000] loss: 0.846
[10, 5250] loss: 0.892
[10, 5500] loss: 0.834
[10, 5750] loss: 0.875
[10, 6000] loss: 0.924
[10, 6250] loss: 0.803
[10, 6500] loss: 0.806
[10, 6750] loss: 0.850
[10, 7000] loss: 0.896
[10, 7250] loss: 0.804
[10, 7500] loss: 0.868
[10, 7750] loss: 0.903
[10, 8000] loss: 0.915
[10, 8250] loss: 0.899
[10, 8500] loss: 0.937
[10, 8750] loss: 0.826
[10, 9000] loss: 0.913
[10, 9250] loss: 0.871
[10, 9500] loss: 0.875
[10, 9750] loss: 0.842
[10, 10000] loss: 0.875
[10, 10250] loss: 0.872
[10, 10500] loss: 0.909
[10, 10750] loss: 0.961
[10, 11000] loss: 0.874
[10, 11250] loss: 0.969
[10, 11500] loss: 0.936
[10, 11750] loss: 0.830
[10, 12000] loss: 0.959
[10, 12250] loss: 0.940
[10, 12500] loss: 0.914
[11, 250] loss: 0.754
[11, 500] loss: 0.711
[11, 750] loss: 0.759
[11, 1000] loss: 0.784
[11, 1250] loss: 0.767
[11, 1500] loss: 0.809
[11, 1750] loss: 0.778
[11, 2000] loss: 0.788
[11, 2250] loss: 0.768
[11, 2500] loss: 0.933
[11, 2750] loss: 0.807
[11, 3000] loss: 0.834
[11, 3250] loss: 0.860
[11, 3500] loss: 0.841
[11, 3750] loss: 0.898
[11, 4000] loss: 0.798
[11, 4250] loss: 0.798
```

```
[11, 4500] loss: 0.782
[11, 4750] loss: 0.793
[11, 5000] loss: 0.823
[11, 5250] loss: 0.864
[11, 5500] loss: 0.803
[11, 5750] loss: 0.775
[11, 6000] loss: 0.883
[11, 6250] loss: 0.841
[11, 6500] loss: 0.815
[11, 6750] loss: 0.822
[11, 7000] loss: 0.842
[11, 7250] loss: 0.876
[11, 7500] loss: 0.870
[11, 7750] loss: 0.814
[11, 8000] loss: 0.865
[11, 8250] loss: 0.901
[11, 8500] loss: 0.904
[11, 8750] loss: 0.851
[11, 9000] loss: 0.879
[11, 9250] loss: 0.849
[11, 9500] loss: 0.866
[11, 9750] loss: 0.909
[11, 10000] loss: 0.939
[11, 10250] loss: 0.930
[11, 10500] loss: 0.885
[11, 10750] loss: 0.815
[11, 11000] loss: 0.881
[11, 11250] loss: 0.862
[11, 11500] loss: 0.938
[11, 11750] loss: 0.854
[11, 12000] loss: 0.875
[11, 12250] loss: 0.913
[11, 12500] loss: 0.840
[12, 250] loss: 0.783
[12, 500] loss: 0.727
[12, 750] loss: 0.773
[12, 1000] loss: 0.759
[12, 1250] loss: 0.826
[12, 1500] loss: 0.733
[12, 1750] loss: 0.744
[12, 2000] loss: 0.800
[12, 2250] loss: 0.780
[12, 2500] loss: 0.837
[12, 2750] loss: 0.805
[12, 3000] loss: 0.765
[12, 3250] loss: 0.826
[12, 3500] loss: 0.762
[12, 3750] loss: 0.895
[12, 4000] loss: 0.807
```

```
[12, 4250] loss: 0.825
[12, 4500] loss: 0.800
[12, 4750] loss: 0.751
[12, 5000] loss: 0.863
[12, 5250] loss: 0.814
[12, 5500] loss: 0.794
[12, 5750] loss: 0.804
[12, 6000] loss: 0.856
[12, 6250] loss: 0.886
[12, 6500] loss: 0.851
[12, 6750] loss: 0.798
[12, 7000] loss: 0.779
[12, 7250] loss: 0.857
[12, 7500] loss: 0.865
[12, 7750] loss: 0.843
[12, 8000] loss: 0.870
[12, 8250] loss: 0.830
[12, 8500] loss: 0.815
[12, 8750] loss: 0.857
[12, 9000] loss: 0.858
[12, 9250] loss: 0.781
[12, 9500] loss: 0.862
[12, 9750] loss: 0.867
[12, 10000] loss: 0.846
[12, 10250] loss: 0.860
[12, 10500] loss: 0.767
[12, 10750] loss: 0.821
[12, 11000] loss: 0.910
[12, 11250] loss: 0.858
[12, 11500] loss: 0.821
[12, 11750] loss: 0.858
[12, 12000] loss: 0.909
[12, 12250] loss: 0.781
[12, 12500] loss: 0.835
[13, 250] loss: 0.721
[13, 500] loss: 0.717
[13, 750] loss: 0.742
[13, 1000] loss: 0.741
[13, 1250] loss: 0.690
[13, 1500] loss: 0.757
[13, 1750] loss: 0.758
[13, 2000] loss: 0.759
[13, 2250] loss: 0.824
[13, 2500] loss: 0.726
[13, 2750] loss: 0.814
[13, 3000] loss: 0.716
[13, 3250] loss: 0.738
[13, 3500] loss: 0.825
[13, 3750] loss: 0.800
```

```
[13, 4000] loss: 0.769
[13, 4250] loss: 0.892
[13, 4500] loss: 0.786
[13, 4750] loss: 0.848
[13, 5000] loss: 0.798
[13, 5250] loss: 0.766
[13, 5500] loss: 0.790
[13, 5750] loss: 0.827
[13, 6000] loss: 0.828
[13, 6250] loss: 0.773
[13, 6500] loss: 0.767
[13, 6750] loss: 0.873
[13, 7000] loss: 0.819
[13, 7250] loss: 0.822
[13, 7500] loss: 0.827
[13, 7750] loss: 0.823
[13, 8000] loss: 0.826
[13, 8250] loss: 0.819
[13, 8500] loss: 0.806
[13, 8750] loss: 0.809
[13, 9000] loss: 0.898
[13, 9250] loss: 0.869
[13, 9500] loss: 0.747
[13, 9750] loss: 0.815
[13, 10000] loss: 0.776
[13, 10250] loss: 0.908
[13, 10500] loss: 0.765
[13, 10750] loss: 0.894
[13, 11000] loss: 0.807
[13, 11250] loss: 0.827
[13, 11500] loss: 0.829
[13, 11750] loss: 0.865
[13, 12000] loss: 0.812
[13, 12250] loss: 0.787
[13, 12500] loss: 0.850
[14, 250] loss: 0.742
[14, 500] loss: 0.677
[14, 750] loss: 0.684
[14, 1000] loss: 0.750
[14, 1250] loss: 0.730
[14, 1500] loss: 0.762
[14, 1750] loss: 0.686
[14, 2000] loss: 0.681
[14, 2250] loss: 0.742
[14, 2500] loss: 0.702
[14, 2750] loss: 0.767
[14, 3000] loss: 0.716
[14, 3250] loss: 0.771
[14, 3500] loss: 0.763
```

```
[14, 3750] loss: 0.801
[14, 4000] loss: 0.724
[14, 4250] loss: 0.789
[14, 4500] loss: 0.820
[14, 4750] loss: 0.761
[14, 5000] loss: 0.702
[14, 5250] loss: 0.729
[14, 5500] loss: 0.811
[14, 5750] loss: 0.806
[14, 6000] loss: 0.735
[14, 6250] loss: 0.841
[14, 6500] loss: 0.744
[14, 6750] loss: 0.825
[14, 7000] loss: 0.868
[14, 7250] loss: 0.753
[14, 7500] loss: 0.795
[14, 7750] loss: 0.833
[14, 8000] loss: 0.808
[14, 8250] loss: 0.860
[14, 8500] loss: 0.795
[14, 8750] loss: 0.758
[14, 9000] loss: 0.774
[14, 9250] loss: 0.817
[14, 9500] loss: 0.775
[14, 9750] loss: 0.871
[14, 10000] loss: 0.762
[14, 10250] loss: 0.818
[14, 10500] loss: 0.816
[14, 10750] loss: 0.797
[14, 11000] loss: 0.846
[14, 11250] loss: 0.797
[14, 11500] loss: 0.801
[14, 11750] loss: 0.833
[14, 12000] loss: 0.778
[14, 12250] loss: 0.808
[14, 12500] loss: 0.900
[15, 250] loss: 0.674
[15, 500] loss: 0.657
[15, 750] loss: 0.664
[15, 1000] loss: 0.682
[15, 1250] loss: 0.653
[15, 1500] loss: 0.704
[15, 1750] loss: 0.694
[15, 2000] loss: 0.780
[15, 2250] loss: 0.734
[15, 2500] loss: 0.772
[15, 2750] loss: 0.731
[15, 3000] loss: 0.734
[15, 3250] loss: 0.691
```

```
[15, 3500] loss: 0.753
[15, 3750] loss: 0.753
[15, 4000] loss: 0.683
[15, 4250] loss: 0.778
[15, 4500] loss: 0.756
[15, 4750] loss: 0.733
[15, 5000] loss: 0.687
[15, 5250] loss: 0.801
[15, 5500] loss: 0.808
[15, 5750] loss: 0.757
[15, 6000] loss: 0.720
[15, 6250] loss: 0.781
[15, 6500] loss: 0.771
[15, 6750] loss: 0.837
[15, 7000] loss: 0.850
[15, 7250] loss: 0.763
[15, 7500] loss: 0.770
[15, 7750] loss: 0.722
[15, 8000] loss: 0.750
[15, 8250] loss: 0.764
[15, 8500] loss: 0.795
[15, 8750] loss: 0.848
[15, 9000] loss: 0.800
[15, 9250] loss: 0.719
[15, 9500] loss: 0.762
[15, 9750] loss: 0.853
[15, 10000] loss: 0.796
[15, 10250] loss: 0.862
[15, 10500] loss: 0.743
[15, 10750] loss: 0.820
[15, 11000] loss: 0.774
[15, 11250] loss: 0.813
[15, 11500] loss: 0.835
[15, 11750] loss: 0.821
[15, 12000] loss: 0.807
[15, 12250] loss: 0.778
[15, 12500] loss: 0.843
[16, 250] loss: 0.676
[16, 500] loss: 0.668
[16, 750] loss: 0.716
[16, 1000] loss: 0.675
[16, 1250] loss: 0.628
[16, 1500] loss: 0.656
[16, 1750] loss: 0.682
[16, 2000] loss: 0.672
[16, 2250] loss: 0.717
[16, 2500] loss: 0.723
[16, 2750] loss: 0.689
[16, 3000] loss: 0.731
```

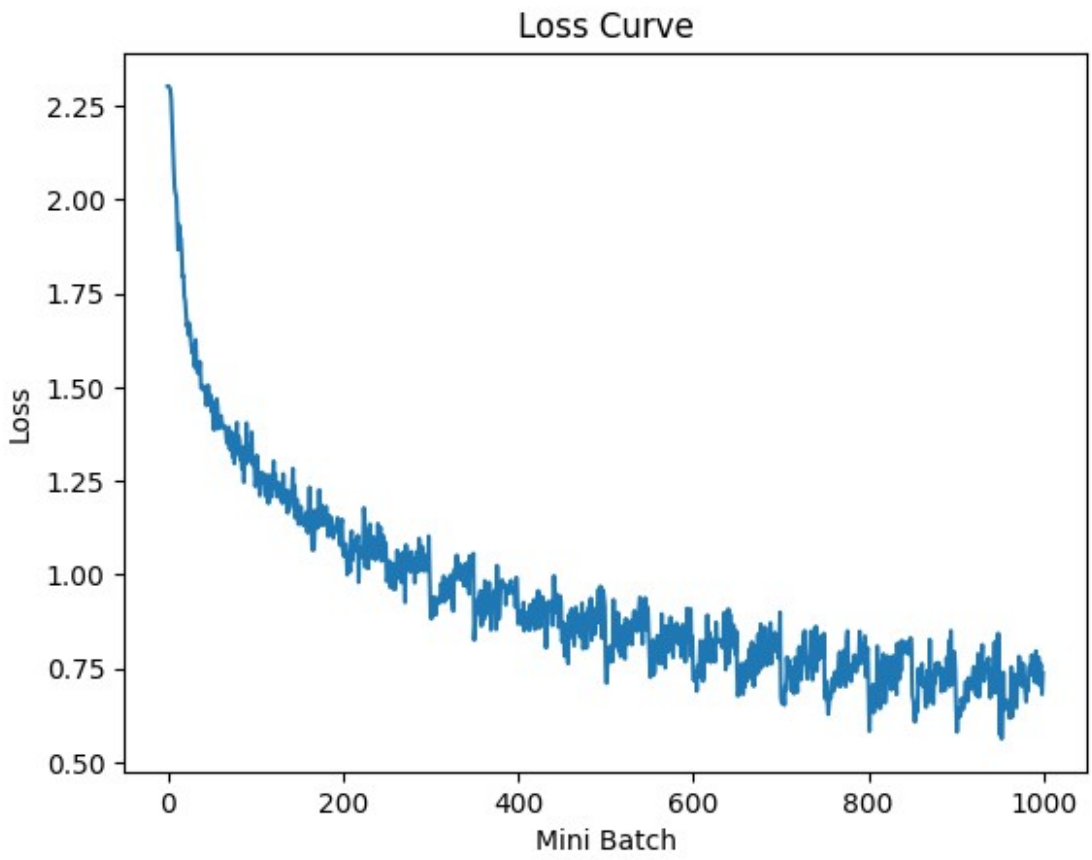
```
[16, 3250] loss: 0.741
[16, 3500] loss: 0.710
[16, 3750] loss: 0.719
[16, 4000] loss: 0.704
[16, 4250] loss: 0.761
[16, 4500] loss: 0.794
[16, 4750] loss: 0.702
[16, 5000] loss: 0.722
[16, 5250] loss: 0.747
[16, 5500] loss: 0.786
[16, 5750] loss: 0.798
[16, 6000] loss: 0.757
[16, 6250] loss: 0.702
[16, 6500] loss: 0.823
[16, 6750] loss: 0.850
[16, 7000] loss: 0.750
[16, 7250] loss: 0.752
[16, 7500] loss: 0.724
[16, 7750] loss: 0.715
[16, 8000] loss: 0.739
[16, 8250] loss: 0.764
[16, 8500] loss: 0.751
[16, 8750] loss: 0.786
[16, 9000] loss: 0.772
[16, 9250] loss: 0.740
[16, 9500] loss: 0.770
[16, 9750] loss: 0.762
[16, 10000] loss: 0.781
[16, 10250] loss: 0.849
[16, 10500] loss: 0.762
[16, 10750] loss: 0.795
[16, 11000] loss: 0.801
[16, 11250] loss: 0.816
[16, 11500] loss: 0.788
[16, 11750] loss: 0.836
[16, 12000] loss: 0.728
[16, 12250] loss: 0.769
[16, 12500] loss: 0.805
[17, 250] loss: 0.696
[17, 500] loss: 0.583
[17, 750] loss: 0.662
[17, 1000] loss: 0.652
[17, 1250] loss: 0.661
[17, 1500] loss: 0.707
[17, 1750] loss: 0.632
[17, 2000] loss: 0.684
[17, 2250] loss: 0.668
[17, 2500] loss: 0.652
[17, 2750] loss: 0.809
```

```
[17, 3000] loss: 0.760
[17, 3250] loss: 0.685
[17, 3500] loss: 0.643
[17, 3750] loss: 0.733
[17, 4000] loss: 0.766
[17, 4250] loss: 0.770
[17, 4500] loss: 0.789
[17, 4750] loss: 0.708
[17, 5000] loss: 0.658
[17, 5250] loss: 0.683
[17, 5500] loss: 0.741
[17, 5750] loss: 0.782
[17, 6000] loss: 0.768
[17, 6250] loss: 0.730
[17, 6500] loss: 0.713
[17, 6750] loss: 0.693
[17, 7000] loss: 0.713
[17, 7250] loss: 0.784
[17, 7500] loss: 0.759
[17, 7750] loss: 0.695
[17, 8000] loss: 0.741
[17, 8250] loss: 0.681
[17, 8500] loss: 0.817
[17, 8750] loss: 0.751
[17, 9000] loss: 0.721
[17, 9250] loss: 0.818
[17, 9500] loss: 0.743
[17, 9750] loss: 0.824
[17, 10000] loss: 0.808
[17, 10250] loss: 0.790
[17, 10500] loss: 0.806
[17, 10750] loss: 0.815
[17, 11000] loss: 0.821
[17, 11250] loss: 0.801
[17, 11500] loss: 0.792
[17, 11750] loss: 0.788
[17, 12000] loss: 0.816
[17, 12250] loss: 0.832
[17, 12500] loss: 0.815
[18, 250] loss: 0.676
[18, 500] loss: 0.668
[18, 750] loss: 0.646
[18, 1000] loss: 0.607
[18, 1250] loss: 0.656
[18, 1500] loss: 0.653
[18, 1750] loss: 0.634
[18, 2000] loss: 0.750
[18, 2250] loss: 0.708
[18, 2500] loss: 0.722
```

```
[18, 2750] loss: 0.712
[18, 3000] loss: 0.763
[18, 3250] loss: 0.704
[18, 3500] loss: 0.721
[18, 3750] loss: 0.723
[18, 4000] loss: 0.648
[18, 4250] loss: 0.647
[18, 4500] loss: 0.745
[18, 4750] loss: 0.686
[18, 5000] loss: 0.660
[18, 5250] loss: 0.827
[18, 5500] loss: 0.699
[18, 5750] loss: 0.694
[18, 6000] loss: 0.739
[18, 6250] loss: 0.656
[18, 6500] loss: 0.736
[18, 6750] loss: 0.710
[18, 7000] loss: 0.764
[18, 7250] loss: 0.745
[18, 7500] loss: 0.750
[18, 7750] loss: 0.765
[18, 8000] loss: 0.733
[18, 8250] loss: 0.753
[18, 8500] loss: 0.688
[18, 8750] loss: 0.749
[18, 9000] loss: 0.693
[18, 9250] loss: 0.731
[18, 9500] loss: 0.692
[18, 9750] loss: 0.693
[18, 10000] loss: 0.778
[18, 10250] loss: 0.746
[18, 10500] loss: 0.770
[18, 10750] loss: 0.824
[18, 11000] loss: 0.703
[18, 11250] loss: 0.851
[18, 11500] loss: 0.742
[18, 11750] loss: 0.709
[18, 12000] loss: 0.771
[18, 12250] loss: 0.794
[18, 12500] loss: 0.746
[19, 250] loss: 0.644
[19, 500] loss: 0.581
[19, 750] loss: 0.635
[19, 1000] loss: 0.645
[19, 1250] loss: 0.622
[19, 1500] loss: 0.625
[19, 1750] loss: 0.670
[19, 2000] loss: 0.657
[19, 2250] loss: 0.641
```

```
[19, 2500] loss: 0.659
[19, 2750] loss: 0.683
[19, 3000] loss: 0.715
[19, 3250] loss: 0.670
[19, 3500] loss: 0.662
[19, 3750] loss: 0.725
[19, 4000] loss: 0.693
[19, 4250] loss: 0.694
[19, 4500] loss: 0.697
[19, 4750] loss: 0.703
[19, 5000] loss: 0.733
[19, 5250] loss: 0.754
[19, 5500] loss: 0.744
[19, 5750] loss: 0.684
[19, 6000] loss: 0.750
[19, 6250] loss: 0.678
[19, 6500] loss: 0.789
[19, 6750] loss: 0.674
[19, 7000] loss: 0.687
[19, 7250] loss: 0.768
[19, 7500] loss: 0.775
[19, 7750] loss: 0.813
[19, 8000] loss: 0.730
[19, 8250] loss: 0.716
[19, 8500] loss: 0.725
[19, 8750] loss: 0.752
[19, 9000] loss: 0.767
[19, 9250] loss: 0.769
[19, 9500] loss: 0.778
[19, 9750] loss: 0.749
[19, 10000] loss: 0.672
[19, 10250] loss: 0.746
[19, 10500] loss: 0.687
[19, 10750] loss: 0.732
[19, 11000] loss: 0.744
[19, 11250] loss: 0.820
[19, 11500] loss: 0.685
[19, 11750] loss: 0.764
[19, 12000] loss: 0.713
[19, 12250] loss: 0.843
[19, 12500] loss: 0.693
[20, 250] loss: 0.573
[20, 500] loss: 0.666
[20, 750] loss: 0.562
[20, 1000] loss: 0.631
[20, 1250] loss: 0.740
[20, 1500] loss: 0.639
[20, 1750] loss: 0.648
[20, 2000] loss: 0.664
```

```
[20, 2250] loss: 0.646
[20, 2500] loss: 0.661
[20, 2750] loss: 0.662
[20, 3000] loss: 0.617
[20, 3250] loss: 0.631
[20, 3500] loss: 0.754
[20, 3750] loss: 0.621
[20, 4000] loss: 0.734
[20, 4250] loss: 0.655
[20, 4500] loss: 0.653
[20, 4750] loss: 0.702
[20, 5000] loss: 0.645
[20, 5250] loss: 0.736
[20, 5500] loss: 0.717
[20, 5750] loss: 0.782
[20, 6000] loss: 0.747
[20, 6250] loss: 0.734
[20, 6500] loss: 0.703
[20, 6750] loss: 0.763
[20, 7000] loss: 0.761
[20, 7250] loss: 0.723
[20, 7500] loss: 0.685
[20, 7750] loss: 0.661
[20, 8000] loss: 0.716
[20, 8250] loss: 0.691
[20, 8500] loss: 0.741
[20, 8750] loss: 0.757
[20, 9000] loss: 0.732
[20, 9250] loss: 0.785
[20, 9500] loss: 0.780
[20, 9750] loss: 0.751
[20, 10000] loss: 0.729
[20, 10250] loss: 0.713
[20, 10500] loss: 0.796
[20, 10750] loss: 0.717
[20, 11000] loss: 0.716
[20, 11250] loss: 0.780
[20, 11500] loss: 0.705
[20, 11750] loss: 0.759
[20, 12000] loss: 0.761
[20, 12250] loss: 0.681
[20, 12500] loss: 0.737
Training Complete
```



Testing the network

```
import matplotlib.pyplot as plt
import numpy as np

# Function to display images
def imshow(img):
    img = img / 2 + 0.5  # Unnormalize the image
    npimg = img.numpy()
    plt.imshow(np.transpose(npimg, (1, 2, 0)))
    plt.show()

# Display images
imshow(torchvision.utils.make_grid(images[0:4]))

# Print ground truth and predicted labels
print('GroundTruth: ', ' '.join('%5s' % class_labels[labels[j]] for j
in range(4)))
print('Predicted: ', ' '.join('%5s' % class_labels[predicted[j]] for j
in range(4)))
```

```
tensor([[[ 0.0000,  0.0000,  0.0000,  ...,  0.0000,  0.0000,  0.0000],
         [ 0.0000,  0.0000,  0.0000,  ...,  0.0000,  0.0000,  0.0000],
         [ 0.0000,  0.0000,  1.0000,  ...,  0.1451,  0.0000,  0.0000],
```

```
          ...,
          [ 0.0000,  0.0000,  0.1294,  ...,  0.3412,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000,  ...,  0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000,  ...,  0.0000,  0.0000,
0.0000]],

         [[ 0.0000,  0.0000,  0.0000,  ...,  0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000,  ...,  0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  1.0000,  ...,  0.4353,  0.0000,  0.0000],
          ...,
          [ 0.0000,  0.0000,  0.1451,  ...,  0.0431,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000,  ...,  0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000,  ...,  0.0000,  0.0000,
0.0000]],

         [[ 0.0000,  0.0000,  0.0000,  ...,  0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000,  ...,  0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  1.0000,  ..., -0.6314,  0.0000,  0.0000],
          ...,
          [ 0.0000,  0.0000, -0.0275,  ..., -0.4980,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000,  ...,  0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000,  ...,  0.0000,  0.0000,
0.0000]]])

GroundTruth:  horse truck horse   cat
Predicted:  truck   cat horse  deer
```

(iv) Complete the code below to test the network on the entire testing set.

```
### Accuracy on whole data set
correct = 0
total = 0
with torch.no_grad():
    for data in test_loader:
        images, labels = data
        outputs = net(images)
        _, predicted = torch.max(outputs.data, 1)
        total += labels.size(0)
        correct += (predicted == labels).sum().item()
acc  =  100 * correct / total ## stores the accuracy computed in the
above loop
print('Accuracy of the network on the 10000 test images: %d %%' %
(acc))

Accuracy of the network on the 10000 test images: 61 %
```

(v) Convert the training code in part (iii) and testing code in part (iv) to define functions `train` and `test` with function definitions as shown below. Train the network with different batch size and number of epochs. Use the `plot_loss_curve` function you defined in (i) above to plot the

loss curves. Use the defined `train` and `test` functions to train the network for various configurations asked in (v) in the problem set.

```
from torchvision import datasets, transforms
from torch.utils.data import DataLoader
import torch.nn as nn
import torch.optim as optim
import torch
import matplotlib.pyplot as plt

# Assuming you have defined your dataset (train_data and test_data)
transform = transforms.Compose([
    transforms.ToTensor(),
    transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5)),
])
train_data = datasets.CIFAR10(root='./data', train=True,
download=True, transform=transform)
test_data = datasets.CIFAR10(root='./data', train=False,
download=True, transform=transform)

def get_data_loader(batch_size, train=True):
    dataset = train_data if train else test_data
    return DataLoader(dataset, batch_size=batch_size, shuffle=True)


def train(train_loader, net, criterion, optimizer, num_epochs=5,
use_gpu=False, lr=0.001, momentum=0.9, model_save_path='./net.pth'):
    """
    The `batch_size` parameter is removed from this function since the
batch size is already defined by the DataLoader.
    """
    if use_gpu and torch.cuda.is_available():
        net = net.cuda()

    running_loss_list = []
    for epoch in range(num_epochs):
        running_loss = 0.0
        for i, data in enumerate(train_loader, 0):
            inputs, labels = data
            if use_gpu and torch.cuda.is_available():
                inputs, labels = inputs.cuda(), labels.cuda()

            optimizer.zero_grad()

            outputs = net(inputs)
            loss = criterion(outputs, labels)
            loss.backward()
            optimizer.step()

            running_loss += loss.item()
```

```
            if i % 500 == 499:  # Print every 2000 mini-batches
                print('[{}, {}] loss: {:.3f}'.format(epoch + 1, i + 1,
running_loss / 500))
                running_loss_list.append(running_loss / 500)
                running_loss = 0.0

    torch.save(net.state_dict(), model_save_path)

    return running_loss_list

def test(test_loader, net, model_path='./net.pth', use_gpu=False):
    net.load_state_dict(torch.load(model_path))
    if use_gpu and torch.cuda.is_available():
        net = net.cuda()

    correct = 0
    total = 0
    with torch.no_grad():
        for data in test_loader:
            images, labels = data
            if use_gpu and torch.cuda.is_available():
                images, labels = images.cuda(), labels.cuda()
            outputs = net(images)
            _, predicted = torch.max(outputs.data, 1)
            total += labels.size(0)
            correct += (predicted == labels).sum().item()

    print('Accuracy of the network on the 10000 test images: %d %%' %
(100 * correct / total))

def plot_loss_curve(running_loss_list):
    plt.plot(running_loss_list)
    plt.xlabel('Epochs')
    plt.ylabel('Loss')
    plt.title('Loss Curve')
    plt.show()

net = Net()
criterion = nn.CrossEntropyLoss()
optimizer = optim.SGD(net.parameters(), lr=0.001, momentum=0.9)

batch_size_4_loader = get_data_loader(batch_size=4, train=True)
batch_size_16_loader = get_data_loader(batch_size=16, train=True)
test_loader = get_data_loader(batch_size=4, train=False)


#training Batch Size 4, 20 training epochs

running_loss_list = train(batch_size_4_loader, net, criterion,
optimizer, num_epochs=20, use_gpu=True, model_save_path='./net.pth')
```

```
plot_loss_curve(running_loss_list)
test(test_loader, net, model_path='./net.pth', use_gpu=True)



#Training Batch Size 4, 5 epochs
net = Net()

running_loss_list = train(batch_size_4_loader, net, criterion,
optimizer, num_epochs=5, use_gpu=True, model_save_path='./net.pth')
plot_loss_curve(running_loss_list)
test(test_loader, net, model_path='./net.pth', use_gpu=True)



#Training Batch Size 16, 5 epochs
net = Net()
running_loss_list = train(batch_size_16_loader, net, criterion,
optimizer, num_epochs=5, use_gpu=True, model_save_path='./net.pth',
batch_size=16)
plot_loss_curve(running_loss_list)
test(test_loader, net, model_path='./net.pth', use_gpu=True)



#Training Batch Size 16, 20 epoch
net = Net()
running_loss_list = train(batch_size_16_loader, net, criterion,
optimizer, num_epochs=20, use_gpu=True, model_save_path='./net.pth',
batch_size=16)
plot_loss_curve(running_loss_list)
test(test_loader, net, model_path='./net.pth', use_gpu=True)



Files already downloaded and verified
Files already downloaded and verified
[1, 500] loss: 2.302
[1, 1000] loss: 2.280
[1, 1500] loss: 2.152
[1, 2000] loss: 2.048
[1, 2500] loss: 1.962
[1, 3000] loss: 1.883
[1, 3500] loss: 1.840
[1, 4000] loss: 1.755
[1, 4500] loss: 1.693
[1, 5000] loss: 1.687
[1, 5500] loss: 1.628
```

```
[1, 6000] loss: 1.642
[1, 6500] loss: 1.619
[1, 7000] loss: 1.549
[1, 7500] loss: 1.586
[1, 8000] loss: 1.532
[1, 8500] loss: 1.583
[1, 9000] loss: 1.544
[1, 9500] loss: 1.505
[1, 10000] loss: 1.512
[1, 10500] loss: 1.521
[1, 11000] loss: 1.439
[1, 11500] loss: 1.457
[1, 12000] loss: 1.482
[1, 12500] loss: 1.464
[2, 500] loss: 1.388
[2, 1000] loss: 1.423
[2, 1500] loss: 1.387
[2, 2000] loss: 1.430
[2, 2500] loss: 1.428
[2, 3000] loss: 1.416
[2, 3500] loss: 1.405
[2, 4000] loss: 1.359
[2, 4500] loss: 1.333
[2, 5000] loss: 1.412
[2, 5500] loss: 1.365
[2, 6000] loss: 1.374
[2, 6500] loss: 1.323
[2, 7000] loss: 1.340
[2, 7500] loss: 1.364
[2, 8000] loss: 1.343
[2, 8500] loss: 1.321
[2, 9000] loss: 1.334
[2, 9500] loss: 1.317
[2, 10000] loss: 1.306
[2, 10500] loss: 1.287
[2, 11000] loss: 1.301
[2, 11500] loss: 1.280
[2, 12000] loss: 1.320
[2, 12500] loss: 1.315
[3, 500] loss: 1.262
[3, 1000] loss: 1.292
[3, 1500] loss: 1.201
[3, 2000] loss: 1.211
[3, 2500] loss: 1.241
[3, 3000] loss: 1.273
[3, 3500] loss: 1.221
[3, 4000] loss: 1.223
[3, 4500] loss: 1.208
[3, 5000] loss: 1.202
```

```
[3, 5500] loss: 1.240
[3, 6000] loss: 1.234
[3, 6500] loss: 1.219
[3, 7000] loss: 1.238
[3, 7500] loss: 1.214
[3, 8000] loss: 1.203
[3, 8500] loss: 1.239
[3, 9000] loss: 1.227
[3, 9500] loss: 1.188
[3, 10000] loss: 1.199
[3, 10500] loss: 1.229
[3, 11000] loss: 1.213
[3, 11500] loss: 1.216
[3, 12000] loss: 1.174
[3, 12500] loss: 1.221
[4, 500] loss: 1.093
[4, 1000] loss: 1.107
[4, 1500] loss: 1.148
[4, 2000] loss: 1.144
[4, 2500] loss: 1.107
[4, 3000] loss: 1.148
[4, 3500] loss: 1.125
[4, 4000] loss: 1.219
[4, 4500] loss: 1.148
[4, 5000] loss: 1.153
[4, 5500] loss: 1.117
[4, 6000] loss: 1.158
[4, 6500] loss: 1.156
[4, 7000] loss: 1.141
[4, 7500] loss: 1.104
[4, 8000] loss: 1.151
[4, 8500] loss: 1.224
[4, 9000] loss: 1.108
[4, 9500] loss: 1.178
[4, 10000] loss: 1.105
[4, 10500] loss: 1.117
[4, 11000] loss: 1.098
[4, 11500] loss: 1.152
[4, 12000] loss: 1.102
[4, 12500] loss: 1.153
[5, 500] loss: 1.043
[5, 1000] loss: 1.031
[5, 1500] loss: 1.024
[5, 2000] loss: 1.072
[5, 2500] loss: 1.033
[5, 3000] loss: 1.013
[5, 3500] loss: 1.069
[5, 4000] loss: 1.058
[5, 4500] loss: 1.045
```

```
[5, 5000] loss: 1.069
[5, 5500] loss: 1.077
[5, 6000] loss: 1.050
[5, 6500] loss: 1.120
[5, 7000] loss: 1.063
[5, 7500] loss: 1.083
[5, 8000] loss: 1.052
[5, 8500] loss: 1.054
[5, 9000] loss: 1.065
[5, 9500] loss: 1.066
[5, 10000] loss: 1.116
[5, 10500] loss: 1.089
[5, 11000] loss: 1.040
[5, 11500] loss: 1.042
[5, 12000] loss: 1.080
[5, 12500] loss: 1.086
[6, 500] loss: 0.974
[6, 1000] loss: 0.994
[6, 1500] loss: 0.967
[6, 2000] loss: 1.033
[6, 2500] loss: 0.955
[6, 3000] loss: 1.008
[6, 3500] loss: 1.010
[6, 4000] loss: 0.958
[6, 4500] loss: 0.970
[6, 5000] loss: 1.009
[6, 5500] loss: 0.998
[6, 6000] loss: 0.998
[6, 6500] loss: 0.995
[6, 7000] loss: 1.019
[6, 7500] loss: 1.004
[6, 8000] loss: 1.006
[6, 8500] loss: 1.016
[6, 9000] loss: 1.053
[6, 9500] loss: 0.996
[6, 10000] loss: 1.018
[6, 10500] loss: 1.017
[6, 11000] loss: 1.056
[6, 11500] loss: 1.022
[6, 12000] loss: 0.991
[6, 12500] loss: 1.032
[7, 500] loss: 0.910
[7, 1000] loss: 0.880
[7, 1500] loss: 0.963
[7, 2000] loss: 0.947
[7, 2500] loss: 0.930
[7, 3000] loss: 0.890
[7, 3500] loss: 0.964
[7, 4000] loss: 0.936
```

```
[7, 4500] loss: 0.968
[7, 5000] loss: 0.924
[7, 5500] loss: 0.996
[7, 6000] loss: 0.983
[7, 6500] loss: 0.945
[7, 7000] loss: 1.010
[7, 7500] loss: 0.896
[7, 8000] loss: 0.947
[7, 8500] loss: 0.986
[7, 9000] loss: 1.009
[7, 9500] loss: 0.940
[7, 10000] loss: 0.998
[7, 10500] loss: 0.935
[7, 11000] loss: 0.995
[7, 11500] loss: 1.025
[7, 12000] loss: 0.951
[7, 12500] loss: 0.983
[8, 500] loss: 0.819
[8, 1000] loss: 0.805
[8, 1500] loss: 0.837
[8, 2000] loss: 0.861
[8, 2500] loss: 0.914
[8, 3000] loss: 0.879
[8, 3500] loss: 0.900
[8, 4000] loss: 0.873
[8, 4500] loss: 0.895
[8, 5000] loss: 0.883
[8, 5500] loss: 0.934
[8, 6000] loss: 0.947
[8, 6500] loss: 0.936
[8, 7000] loss: 0.911
[8, 7500] loss: 0.930
[8, 8000] loss: 0.928
[8, 8500] loss: 0.915
[8, 9000] loss: 0.937
[8, 9500] loss: 0.950
[8, 10000] loss: 0.945
[8, 10500] loss: 0.963
[8, 11000] loss: 0.926
[8, 11500] loss: 0.974
[8, 12000] loss: 0.896
[8, 12500] loss: 0.949
[9, 500] loss: 0.803
[9, 1000] loss: 0.855
[9, 1500] loss: 0.825
[9, 2000] loss: 0.861
[9, 2500] loss: 0.829
[9, 3000] loss: 0.809
[9, 3500] loss: 0.827
```

```
[9, 4000] loss: 0.867
[9, 4500] loss: 0.880
[9, 5000] loss: 0.845
[9, 5500] loss: 0.898
[9, 6000] loss: 0.858
[9, 6500] loss: 0.904
[9, 7000] loss: 0.864
[9, 7500] loss: 0.878
[9, 8000] loss: 0.890
[9, 8500] loss: 0.897
[9, 9000] loss: 0.886
[9, 9500] loss: 0.906
[9, 10000] loss: 0.874
[9, 10500] loss: 0.920
[9, 11000] loss: 0.882
[9, 11500] loss: 0.918
[9, 12000] loss: 0.927
[9, 12500] loss: 0.923
[10, 500] loss: 0.791
[10, 1000] loss: 0.802
[10, 1500] loss: 0.775
[10, 2000] loss: 0.812
[10, 2500] loss: 0.821
[10, 3000] loss: 0.768
[10, 3500] loss: 0.846
[10, 4000] loss: 0.822
[10, 4500] loss: 0.816
[10, 5000] loss: 0.834
[10, 5500] loss: 0.846
[10, 6000] loss: 0.871
[10, 6500] loss: 0.872
[10, 7000] loss: 0.878
[10, 7500] loss: 0.850
[10, 8000] loss: 0.849
[10, 8500] loss: 0.847
[10, 9000] loss: 0.899
[10, 9500] loss: 0.889
[10, 10000] loss: 0.858
[10, 10500] loss: 0.874
[10, 11000] loss: 0.852
[10, 11500] loss: 0.903
[10, 12000] loss: 0.874
[10, 12500] loss: 0.872
[11, 500] loss: 0.710
[11, 1000] loss: 0.713
[11, 1500] loss: 0.721
[11, 2000] loss: 0.781
[11, 2500] loss: 0.793
[11, 3000] loss: 0.762
```

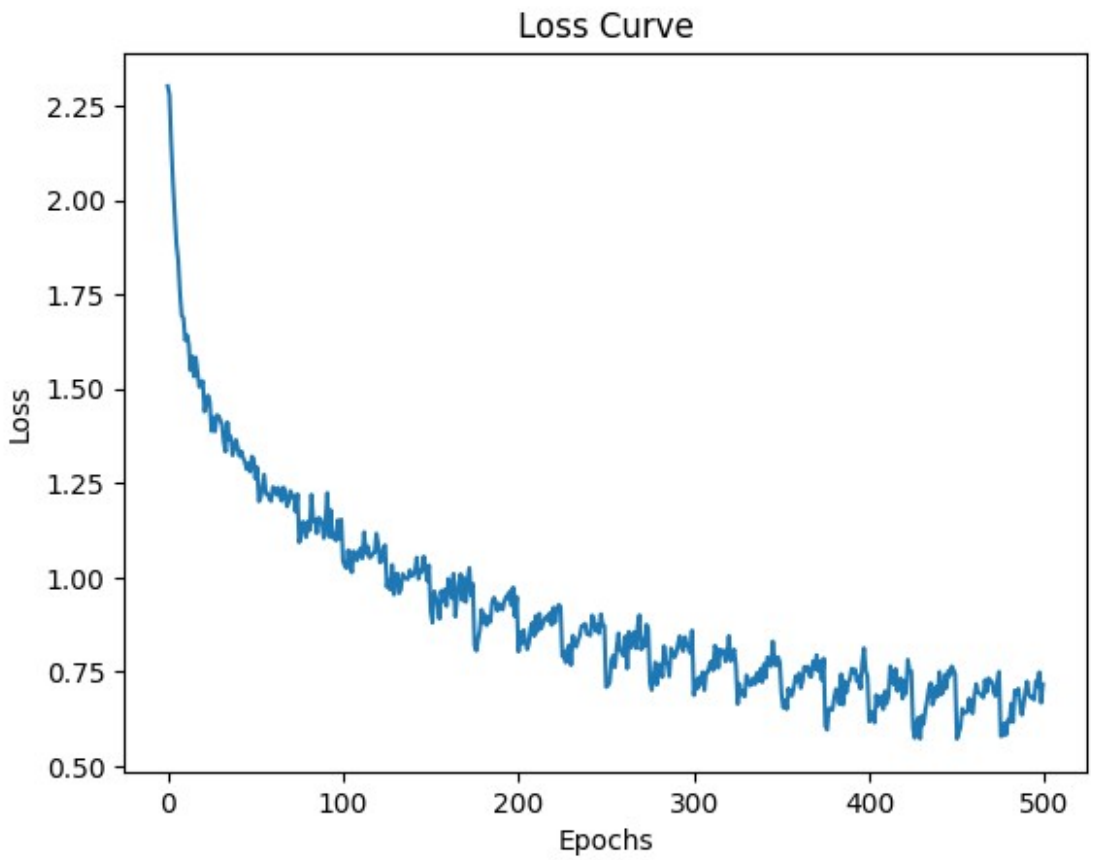
```
[11, 3500] loss: 0.814
[11, 4000] loss: 0.852
[11, 4500] loss: 0.809
[11, 5000] loss: 0.800
[11, 5500] loss: 0.792
[11, 6000] loss: 0.844
[11, 6500] loss: 0.759
[11, 7000] loss: 0.856
[11, 7500] loss: 0.813
[11, 8000] loss: 0.839
[11, 8500] loss: 0.855
[11, 9000] loss: 0.812
[11, 9500] loss: 0.872
[11, 10000] loss: 0.900
[11, 10500] loss: 0.809
[11, 11000] loss: 0.836
[11, 11500] loss: 0.817
[11, 12000] loss: 0.875
[11, 12500] loss: 0.863
[12, 500] loss: 0.721
[12, 1000] loss: 0.701
[12, 1500] loss: 0.737
[12, 2000] loss: 0.775
[12, 2500] loss: 0.716
[12, 3000] loss: 0.769
[12, 3500] loss: 0.769
[12, 4000] loss: 0.736
[12, 4500] loss: 0.818
[12, 5000] loss: 0.778
[12, 5500] loss: 0.769
[12, 6000] loss: 0.738
[12, 6500] loss: 0.812
[12, 7000] loss: 0.803
[12, 7500] loss: 0.788
[12, 8000] loss: 0.799
[12, 8500] loss: 0.825
[12, 9000] loss: 0.845
[12, 9500] loss: 0.813
[12, 10000] loss: 0.804
[12, 10500] loss: 0.842
[12, 11000] loss: 0.824
[12, 11500] loss: 0.815
[12, 12000] loss: 0.799
[12, 12500] loss: 0.860
[13, 500] loss: 0.688
[13, 1000] loss: 0.728
[13, 1500] loss: 0.704
[13, 2000] loss: 0.735
[13, 2500] loss: 0.718
```

```
[13, 3000] loss: 0.749
[13, 3500] loss: 0.701
[13, 4000] loss: 0.749
[13, 4500] loss: 0.748
[13, 5000] loss: 0.769
[13, 5500] loss: 0.778
[13, 6000] loss: 0.754
[13, 6500] loss: 0.820
[13, 7000] loss: 0.810
[13, 7500] loss: 0.760
[13, 8000] loss: 0.809
[13, 8500] loss: 0.797
[13, 9000] loss: 0.807
[13, 9500] loss: 0.777
[13, 10000] loss: 0.797
[13, 10500] loss: 0.845
[13, 11000] loss: 0.778
[13, 11500] loss: 0.797
[13, 12000] loss: 0.809
[13, 12500] loss: 0.767
[14, 500] loss: 0.665
[14, 1000] loss: 0.718
[14, 1500] loss: 0.689
[14, 2000] loss: 0.700
[14, 2500] loss: 0.684
[14, 3000] loss: 0.741
[14, 3500] loss: 0.731
[14, 4000] loss: 0.714
[14, 4500] loss: 0.716
[14, 5000] loss: 0.734
[14, 5500] loss: 0.717
[14, 6000] loss: 0.747
[14, 6500] loss: 0.725
[14, 7000] loss: 0.758
[14, 7500] loss: 0.722
[14, 8000] loss: 0.771
[14, 8500] loss: 0.734
[14, 9000] loss: 0.788
[14, 9500] loss: 0.783
[14, 10000] loss: 0.765
[14, 10500] loss: 0.831
[14, 11000] loss: 0.779
[14, 11500] loss: 0.766
[14, 12000] loss: 0.789
[14, 12500] loss: 0.774
[15, 500] loss: 0.691
[15, 1000] loss: 0.655
[15, 1500] loss: 0.674
[15, 2000] loss: 0.651
```

```
[15, 2500] loss: 0.706
[15, 3000] loss: 0.688
[15, 3500] loss: 0.687
[15, 4000] loss: 0.708
[15, 4500] loss: 0.738
[15, 5000] loss: 0.724
[15, 5500] loss: 0.746
[15, 6000] loss: 0.707
[15, 6500] loss: 0.725
[15, 7000] loss: 0.764
[15, 7500] loss: 0.755
[15, 8000] loss: 0.764
[15, 8500] loss: 0.738
[15, 9000] loss: 0.736
[15, 9500] loss: 0.778
[15, 10000] loss: 0.756
[15, 10500] loss: 0.794
[15, 11000] loss: 0.752
[15, 11500] loss: 0.778
[15, 12000] loss: 0.728
[15, 12500] loss: 0.785
[16, 500] loss: 0.606
[16, 1000] loss: 0.596
[16, 1500] loss: 0.653
[16, 2000] loss: 0.649
[16, 2500] loss: 0.649
[16, 3000] loss: 0.680
[16, 3500] loss: 0.704
[16, 4000] loss: 0.702
[16, 4500] loss: 0.664
[16, 5000] loss: 0.744
[16, 5500] loss: 0.663
[16, 6000] loss: 0.725
[16, 6500] loss: 0.694
[16, 7000] loss: 0.721
[16, 7500] loss: 0.731
[16, 8000] loss: 0.759
[16, 8500] loss: 0.748
[16, 9000] loss: 0.756
[16, 9500] loss: 0.724
[16, 10000] loss: 0.739
[16, 10500] loss: 0.705
[16, 11000] loss: 0.771
[16, 11500] loss: 0.813
[16, 12000] loss: 0.753
[16, 12500] loss: 0.738
[17, 500] loss: 0.618
[17, 1000] loss: 0.631
[17, 1500] loss: 0.645
```

```
[17, 2000] loss: 0.616
[17, 2500] loss: 0.689
[17, 3000] loss: 0.660
[17, 3500] loss: 0.667
[17, 4000] loss: 0.686
[17, 4500] loss: 0.651
[17, 5000] loss: 0.699
[17, 5500] loss: 0.666
[17, 6000] loss: 0.685
[17, 6500] loss: 0.765
[17, 7000] loss: 0.729
[17, 7500] loss: 0.716
[17, 8000] loss: 0.758
[17, 8500] loss: 0.699
[17, 9000] loss: 0.723
[17, 9500] loss: 0.733
[17, 10000] loss: 0.679
[17, 10500] loss: 0.727
[17, 11000] loss: 0.692
[17, 11500] loss: 0.782
[17, 12000] loss: 0.751
[17, 12500] loss: 0.752
[18, 500] loss: 0.601
[18, 1000] loss: 0.575
[18, 1500] loss: 0.615
[18, 2000] loss: 0.629
[18, 2500] loss: 0.573
[18, 3000] loss: 0.633
[18, 3500] loss: 0.611
[18, 4000] loss: 0.649
[18, 4500] loss: 0.673
[18, 5000] loss: 0.705
[18, 5500] loss: 0.720
[18, 6000] loss: 0.662
[18, 6500] loss: 0.675
[18, 7000] loss: 0.711
[18, 7500] loss: 0.685
[18, 8000] loss: 0.749
[18, 8500] loss: 0.690
[18, 9000] loss: 0.695
[18, 9500] loss: 0.741
[18, 10000] loss: 0.707
[18, 10500] loss: 0.752
[18, 11000] loss: 0.739
[18, 11500] loss: 0.765
[18, 12000] loss: 0.756
[18, 12500] loss: 0.743
[19, 500] loss: 0.572
[19, 1000] loss: 0.587
```

```
[19, 1500] loss: 0.602
[19, 2000] loss: 0.652
[19, 2500] loss: 0.642
[19, 3000] loss: 0.641
[19, 3500] loss: 0.645
[19, 4000] loss: 0.679
[19, 4500] loss: 0.648
[19, 5000] loss: 0.640
[19, 5500] loss: 0.687
[19, 6000] loss: 0.718
[19, 6500] loss: 0.696
[19, 7000] loss: 0.693
[19, 7500] loss: 0.688
[19, 8000] loss: 0.682
[19, 8500] loss: 0.728
[19, 9000] loss: 0.696
[19, 9500] loss: 0.733
[19, 10000] loss: 0.725
[19, 10500] loss: 0.724
[19, 11000] loss: 0.696
[19, 11500] loss: 0.686
[19, 12000] loss: 0.723
[19, 12500] loss: 0.750
[20, 500] loss: 0.580
[20, 1000] loss: 0.580
[20, 1500] loss: 0.613
[20, 2000] loss: 0.583
[20, 2500] loss: 0.622
[20, 3000] loss: 0.617
[20, 3500] loss: 0.666
[20, 4000] loss: 0.617
[20, 4500] loss: 0.701
[20, 5000] loss: 0.691
[20, 5500] loss: 0.705
[20, 6000] loss: 0.664
[20, 6500] loss: 0.636
[20, 7000] loss: 0.663
[20, 7500] loss: 0.697
[20, 8000] loss: 0.724
[20, 8500] loss: 0.690
[20, 9000] loss: 0.682
[20, 9500] loss: 0.685
[20, 10000] loss: 0.676
[20, 10500] loss: 0.725
[20, 11000] loss: 0.710
[20, 11500] loss: 0.749
[20, 12000] loss: 0.668
[20, 12500] loss: 0.717
```




```
Accuracy of the network on the 10000 test images: 60 %
[1, 500] loss: 2.307
[1, 1000] loss: 2.304
[1, 1500] loss: 2.304
[1, 2000] loss: 2.305
[1, 2500] loss: 2.307
[1, 3000] loss: 2.306
[1, 3500] loss: 2.305
[1, 4000] loss: 2.307
[1, 4500] loss: 2.309
[1, 5000] loss: 2.304
[1, 5500] loss: 2.305
[1, 6000] loss: 2.303
[1, 6500] loss: 2.304
[1, 7000] loss: 2.305
[1, 7500] loss: 2.305
[1, 8000] loss: 2.306
[1, 8500] loss: 2.306
[1, 9000] loss: 2.305
[1, 9500] loss: 2.306
[1, 10000] loss: 2.303
[1, 10500] loss: 2.307
[1, 11000] loss: 2.304
```

```
[1, 11500] loss: 2.306
[1, 12000] loss: 2.304
[1, 12500] loss: 2.305
[2, 500] loss: 2.306
[2, 1000] loss: 2.307
[2, 1500] loss: 2.306
[2, 2000] loss: 2.303
[2, 2500] loss: 2.303
[2, 3000] loss: 2.303
[2, 3500] loss: 2.306
[2, 4000] loss: 2.308
[2, 4500] loss: 2.304
[2, 5000] loss: 2.306
[2, 5500] loss: 2.305
[2, 6000] loss: 2.306
[2, 6500] loss: 2.304
[2, 7000] loss: 2.306
[2, 7500] loss: 2.303
[2, 8000] loss: 2.307
[2, 8500] loss: 2.307
[2, 9000] loss: 2.305
[2, 9500] loss: 2.307
[2, 10000] loss: 2.305
[2, 10500] loss: 2.305
[2, 11000] loss: 2.305
[2, 11500] loss: 2.304
[2, 12000] loss: 2.305
[2, 12500] loss: 2.304
[3, 500] loss: 2.306
[3, 1000] loss: 2.308
[3, 1500] loss: 2.306
[3, 2000] loss: 2.306
[3, 2500] loss: 2.304
[3, 3000] loss: 2.306
[3, 3500] loss: 2.307
[3, 4000] loss: 2.307
[3, 4500] loss: 2.305
[3, 5000] loss: 2.304
[3, 5500] loss: 2.304
[3, 6000] loss: 2.305
[3, 6500] loss: 2.306
[3, 7000] loss: 2.305
[3, 7500] loss: 2.304
[3, 8000] loss: 2.304
[3, 8500] loss: 2.305
[3, 9000] loss: 2.307
[3, 9500] loss: 2.304
[3, 10000] loss: 2.304
[3, 10500] loss: 2.306
```

```
[3, 11000] loss: 2.307
[3, 11500] loss: 2.303
[3, 12000] loss: 2.302
[3, 12500] loss: 2.305
[4, 500] loss: 2.306
[4, 1000] loss: 2.305
[4, 1500] loss: 2.303
[4, 2000] loss: 2.304
[4, 2500] loss: 2.306
[4, 3000] loss: 2.303
[4, 3500] loss: 2.305
[4, 4000] loss: 2.303
[4, 4500] loss: 2.308
[4, 5000] loss: 2.305
[4, 5500] loss: 2.306
[4, 6000] loss: 2.308
[4, 6500] loss: 2.307
[4, 7000] loss: 2.305
[4, 7500] loss: 2.308
[4, 8000] loss: 2.306
[4, 8500] loss: 2.305
[4, 9000] loss: 2.304
[4, 9500] loss: 2.305
[4, 10000] loss: 2.307
[4, 10500] loss: 2.303
[4, 11000] loss: 2.303
[4, 11500] loss: 2.304
[4, 12000] loss: 2.305
[4, 12500] loss: 2.307
[5, 500] loss: 2.304
[5, 1000] loss: 2.304
[5, 1500] loss: 2.304
[5, 2000] loss: 2.308
[5, 2500] loss: 2.301
[5, 3000] loss: 2.306
[5, 3500] loss: 2.306
[5, 4000] loss: 2.304
[5, 4500] loss: 2.302
[5, 5000] loss: 2.307
[5, 5500] loss: 2.307
[5, 6000] loss: 2.305
[5, 6500] loss: 2.303
[5, 7000] loss: 2.306
[5, 7500] loss: 2.307

---------------------------------------------------------------------------
-----
KeyboardInterrupt                         Traceback (most recent call
last)
<ipython-input-26-fb1f66f5f45c> in <cell line: 99>()
```

```
     97 net = Net()
     98
---> 99 running_loss_list = train(batch_size_4_loader, net, criterion,
optimizer, num_epochs=5, use_gpu=True, model_save_path='./net.pth')
    100 plot_loss_curve(running_loss_list)
    101 test(test_loader, net, model_path='./net.pth', use_gpu=True)

<ipython-input-26-fb1f66f5f45c> in train(train_loader, net, criterion,
optimizer, num_epochs, use_gpu, lr, momentum, model_save_path)
     29     for epoch in range(num_epochs):
     30         running_loss = 0.0
---> 31         for i, data in enumerate(train_loader, 0):
     32             inputs, labels = data
     33             if use_gpu and torch.cuda.is_available():

/usr/local/lib/python3.10/dist-packages/torch/utils/data/dataloader.py
in __next__(self)
    629                 #
TODO(https://github.com/pytorch/pytorch/issues/76750)
    630                 self._reset()  # type: ignore[call-arg]
--> 631             data = self._next_data()
    632             self._num_yielded += 1
    633             if self._dataset_kind == _DatasetKind.Iterable and
\

/usr/local/lib/python3.10/dist-packages/torch/utils/data/dataloader.py
in _next_data(self)
    673     def _next_data(self):
    674         index = self._next_index()  # may raise StopIteration
--> 675         data = self._dataset_fetcher.fetch(index)  # may raise
StopIteration
    676         if self._pin_memory:
    677             data = _utils.pin_memory.pin_memory(data,
self._pin_memory_device)

/usr/local/lib/python3.10/dist-packages/torch/utils/data/_utils/fetch.
py in fetch(self, possibly_batched_index)
     49                 data =
self.dataset.__getitems__(possibly_batched_index)
     50             else:
---> 51                 data = [self.dataset[idx] for idx in
possibly_batched_index]
     52         else:
     53             data = self.dataset[possibly_batched_index]

/usr/local/lib/python3.10/dist-packages/torch/utils/data/_utils/fetch.
py in <listcomp>(.0)
     49                 data =
self.dataset.__getitems__(possibly_batched_index)
     50             else:
```

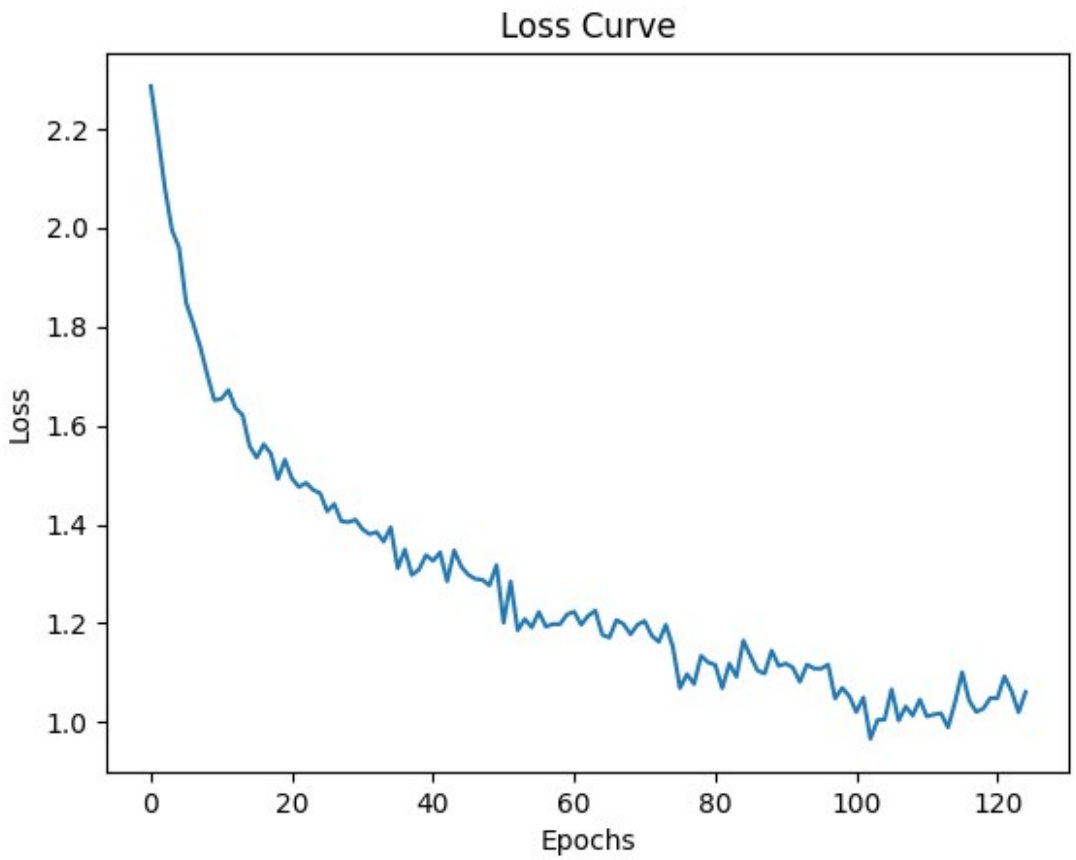
```
---> 51                 data = [self.dataset[idx] for idx in
possibly_batched_index]
     52         else:
     53             data = self.dataset[possibly_batched_index]

/usr/local/lib/python3.10/dist-packages/torchvision/datasets/cifar.py
in __getitem__(self, index)
    116
    117         if self.transform is not None:
--> 118             img = self.transform(img)
    119
    120         if self.target_transform is not None:

/usr/local/lib/python3.10/dist-packages/torchvision/transforms/transfo
rms.py in __call__(self, img)
     93     def __call__(self, img):
     94         for t in self.transforms:
---> 95             img = t(img)
     96         return img
     97

/usr/local/lib/python3.10/dist-packages/torchvision/transforms/transfo
rms.py in __call__(self, pic)
    135             Tensor: Converted image.
    136         """
--> 137         return F.to_tensor(pic)
    138
    139     def __repr__(self) -> str:

/usr/local/lib/python3.10/dist-packages/torchvision/transforms/functio
nal.py in to_tensor(pic)
    165     # handle PIL Image
    166     mode_to_nptype = {"I": np.int32, "I;16" if sys.byteorder
== "little" else "I;16B": np.int16, "F": np.float32}
--> 167     img = torch.from_numpy(np.array(pic,
mode_to_nptype.get(pic.mode, np.uint8), copy=True))
    168
    169     if pic.mode == "1":

/usr/local/lib/python3.10/dist-packages/PIL/Image.py in
__getattr__(self, name)
    523         self._exif = None
    524
--> 525     def __getattr__(self, name):
    526         if name == "category":
    527             deprecate("Image categories", 10, "is_animated",
plural=True)

KeyboardInterrupt:
```

```
net = Net()
criterion = nn.CrossEntropyLoss()
optimizer = optim.SGD(net.parameters(), lr=0.001, momentum=0.9)

batch_size_4_loader = get_data_loader(batch_size=4, train=True)
batch_size_16_loader = get_data_loader(batch_size=16, train=True)
test_loader = get_data_loader(batch_size=4, train=False)


#training Batch Size 4, 5 training epochs

running_loss_list = train(batch_size_4_loader, net, criterion,
optimizer, num_epochs=5, use_gpu=True, model_save_path='./net.pth')
plot_loss_curve(running_loss_list)
test(test_loader, net, model_path='./net.pth', use_gpu=True)
```

```
[1, 500] loss: 2.286
[1, 1000] loss: 2.186
[1, 1500] loss: 2.079
[1, 2000] loss: 1.995
[1, 2500] loss: 1.959
[1, 3000] loss: 1.848
[1, 3500] loss: 1.806
[1, 4000] loss: 1.759
[1, 4500] loss: 1.702
[1, 5000] loss: 1.651
[1, 5500] loss: 1.654
[1, 6000] loss: 1.672
[1, 6500] loss: 1.635
[1, 7000] loss: 1.621
[1, 7500] loss: 1.558
[1, 8000] loss: 1.535
[1, 8500] loss: 1.562
[1, 9000] loss: 1.543
[1, 9500] loss: 1.492
[1, 10000] loss: 1.531
[1, 10500] loss: 1.493
[1, 11000] loss: 1.476
[1, 11500] loss: 1.484
[1, 12000] loss: 1.470
[1, 12500] loss: 1.463
[2, 500] loss: 1.426
[2, 1000] loss: 1.441
[2, 1500] loss: 1.406
[2, 2000] loss: 1.404
[2, 2500] loss: 1.409
[2, 3000] loss: 1.390
[2, 3500] loss: 1.380
[2, 4000] loss: 1.385
[2, 4500] loss: 1.365
```

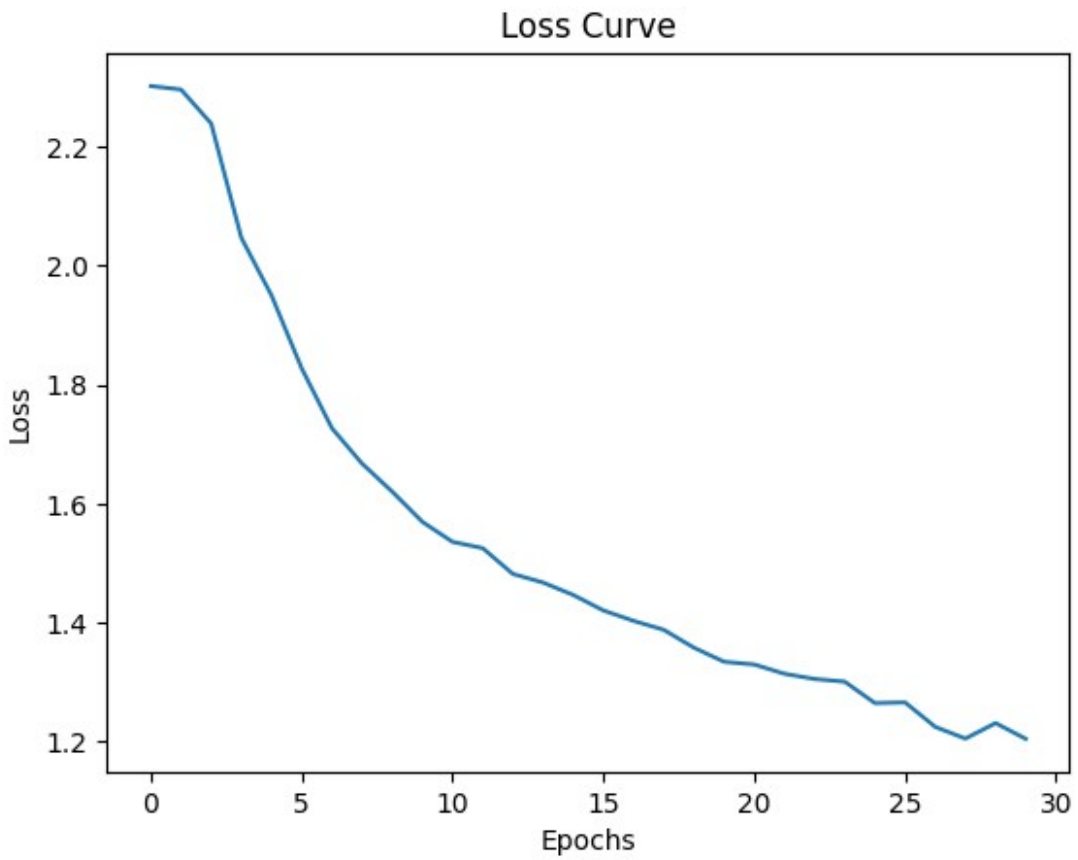
```
[2, 5000] loss: 1.394
[2, 5500] loss: 1.311
[2, 6000] loss: 1.350
[2, 6500] loss: 1.298
[2, 7000] loss: 1.308
[2, 7500] loss: 1.338
[2, 8000] loss: 1.326
[2, 8500] loss: 1.343
[2, 9000] loss: 1.285
[2, 9500] loss: 1.347
[2, 10000] loss: 1.315
[2, 10500] loss: 1.298
[2, 11000] loss: 1.290
[2, 11500] loss: 1.288
[2, 12000] loss: 1.277
[2, 12500] loss: 1.318
[3, 500] loss: 1.201
[3, 1000] loss: 1.284
[3, 1500] loss: 1.186
[3, 2000] loss: 1.209
[3, 2500] loss: 1.192
[3, 3000] loss: 1.223
[3, 3500] loss: 1.193
[3, 4000] loss: 1.198
[3, 4500] loss: 1.198
[3, 5000] loss: 1.218
[3, 5500] loss: 1.223
[3, 6000] loss: 1.197
[3, 6500] loss: 1.215
[3, 7000] loss: 1.226
[3, 7500] loss: 1.177
[3, 8000] loss: 1.171
[3, 8500] loss: 1.206
[3, 9000] loss: 1.198
[3, 9500] loss: 1.178
[3, 10000] loss: 1.197
[3, 10500] loss: 1.205
[3, 11000] loss: 1.175
[3, 11500] loss: 1.163
[3, 12000] loss: 1.197
[3, 12500] loss: 1.152
[4, 500] loss: 1.069
[4, 1000] loss: 1.097
[4, 1500] loss: 1.077
[4, 2000] loss: 1.134
[4, 2500] loss: 1.121
[4, 3000] loss: 1.116
[4, 3500] loss: 1.069
[4, 4000] loss: 1.119
```

```
[4, 4500] loss: 1.092
[4, 5000] loss: 1.165
[4, 5500] loss: 1.133
[4, 6000] loss: 1.105
[4, 6500] loss: 1.098
[4, 7000] loss: 1.145
[4, 7500] loss: 1.114
[4, 8000] loss: 1.119
[4, 8500] loss: 1.111
[4, 9000] loss: 1.082
[4, 9500] loss: 1.117
[4, 10000] loss: 1.109
[4, 10500] loss: 1.108
[4, 11000] loss: 1.117
[4, 11500] loss: 1.048
[4, 12000] loss: 1.069
[4, 12500] loss: 1.053
[5, 500] loss: 1.021
[5, 1000] loss: 1.050
[5, 1500] loss: 0.966
[5, 2000] loss: 1.004
[5, 2500] loss: 1.005
[5, 3000] loss: 1.066
[5, 3500] loss: 1.003
[5, 4000] loss: 1.032
[5, 4500] loss: 1.014
[5, 5000] loss: 1.046
[5, 5500] loss: 1.012
[5, 6000] loss: 1.016
[5, 6500] loss: 1.018
[5, 7000] loss: 0.990
[5, 7500] loss: 1.041
[5, 8000] loss: 1.101
[5, 8500] loss: 1.044
[5, 9000] loss: 1.021
[5, 9500] loss: 1.028
[5, 10000] loss: 1.049
[5, 10500] loss: 1.049
[5, 11000] loss: 1.093
[5, 11500] loss: 1.063
[5, 12000] loss: 1.020
[5, 12500] loss: 1.061
```




```
Accuracy of the network on the 10000 test images: 61 %

net = Net()
criterion = nn.CrossEntropyLoss()
optimizer = optim.SGD(net.parameters(), lr=0.001, momentum=0.9)

batch_size_4_loader = get_data_loader(batch_size=4, train=True)
batch_size_16_loader = get_data_loader(batch_size=16, train=True)
test_loader = get_data_loader(batch_size=4, train=False)


#training Batch Size 16, 5 training epochs

running_loss_list = train(batch_size_16_loader, net, criterion,
optimizer, num_epochs=5, use_gpu=True, model_save_path='./net.pth')
plot_loss_curve(running_loss_list)
test(test_loader, net, model_path='./net.pth', use_gpu=True)

[1, 500] loss: 2.302
[1, 1000] loss: 2.296
[1, 1500] loss: 2.239
[1, 2000] loss: 2.047
[1, 2500] loss: 1.950
[1, 3000] loss: 1.827
```

```
[2, 500] loss: 1.727
[2, 1000] loss: 1.667
[2, 1500] loss: 1.620
[2, 2000] loss: 1.569
[2, 2500] loss: 1.536
[2, 3000] loss: 1.525
[3, 500] loss: 1.482
[3, 1000] loss: 1.467
[3, 1500] loss: 1.446
[3, 2000] loss: 1.420
[3, 2500] loss: 1.403
[3, 3000] loss: 1.388
[4, 500] loss: 1.358
[4, 1000] loss: 1.334
[4, 1500] loss: 1.329
[4, 2000] loss: 1.314
[4, 2500] loss: 1.305
[4, 3000] loss: 1.301
[5, 500] loss: 1.265
[5, 1000] loss: 1.266
[5, 1500] loss: 1.225
[5, 2000] loss: 1.205
[5, 2500] loss: 1.231
[5, 3000] loss: 1.204
```

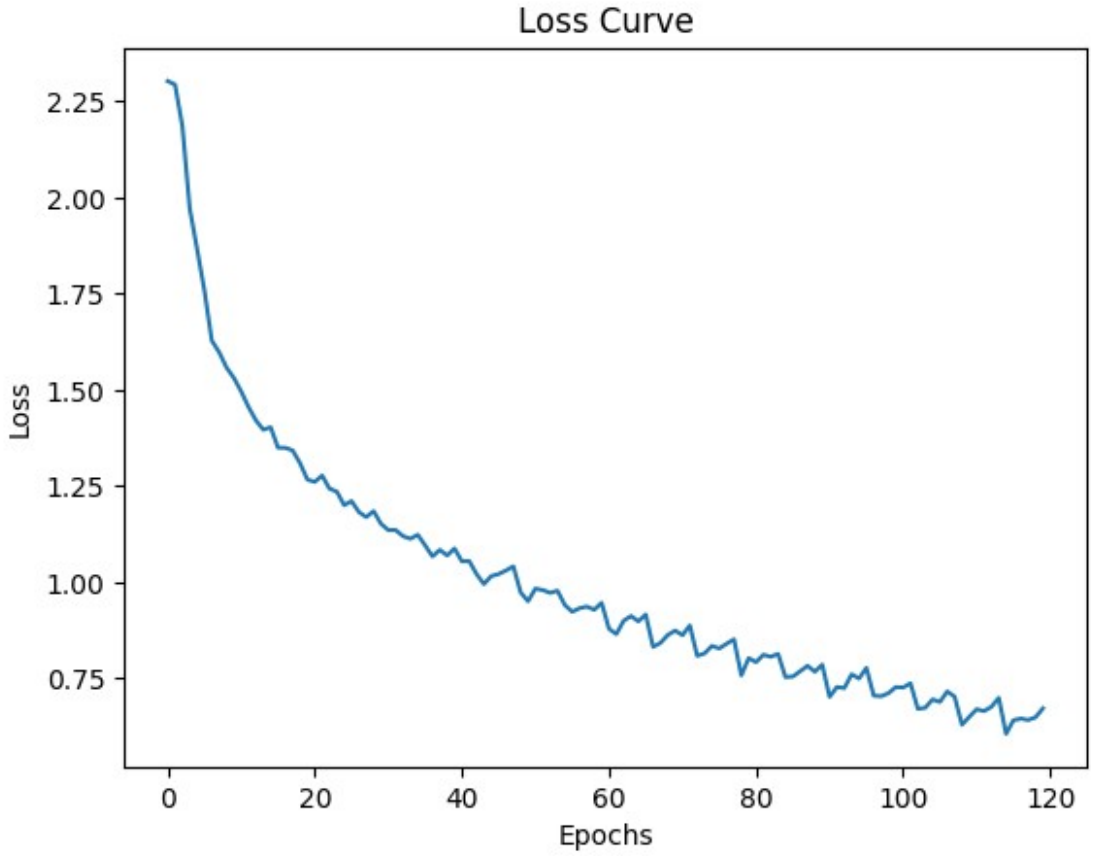



```
Accuracy of the network on the 10000 test images: 56 %

net = Net()

criterion = nn.CrossEntropyLoss()
optimizer = optim.SGD(net.parameters(), lr=0.001, momentum=0.9)

batch_size_4_loader = get_data_loader(batch_size=4, train=True)
batch_size_16_loader = get_data_loader(batch_size=16, train=True)
test_loader = get_data_loader(batch_size=4, train=False)


#training Batch Size 16, 20 training epochs

running_loss_list = train(batch_size_16_loader, net, criterion,
optimizer, num_epochs=20, use_gpu=True, model_save_path='./net.pth')
plot_loss_curve(running_loss_list)
test(test_loader, net, model_path='./net.pth', use_gpu=True)

[1, 500] loss: 2.303
[1, 1000] loss: 2.294
[1, 1500] loss: 2.186
[1, 2000] loss: 1.972
[1, 2500] loss: 1.866
```

```
[1, 3000] loss: 1.758
[2, 500] loss: 1.628
[2, 1000] loss: 1.598
[2, 1500] loss: 1.558
[2, 2000] loss: 1.531
[2, 2500] loss: 1.496
[2, 3000] loss: 1.456
[3, 500] loss: 1.421
[3, 1000] loss: 1.397
[3, 1500] loss: 1.403
[3, 2000] loss: 1.350
[3, 2500] loss: 1.350
[3, 3000] loss: 1.343
[4, 500] loss: 1.310
[4, 1000] loss: 1.267
[4, 1500] loss: 1.261
[4, 2000] loss: 1.278
[4, 2500] loss: 1.244
[4, 3000] loss: 1.236
[5, 500] loss: 1.201
[5, 1000] loss: 1.211
[5, 1500] loss: 1.183
[5, 2000] loss: 1.170
[5, 2500] loss: 1.185
[5, 3000] loss: 1.152
[6, 500] loss: 1.136
[6, 1000] loss: 1.136
[6, 1500] loss: 1.120
[6, 2000] loss: 1.113
[6, 2500] loss: 1.124
[6, 3000] loss: 1.096
[7, 500] loss: 1.067
[7, 1000] loss: 1.084
[7, 1500] loss: 1.070
[7, 2000] loss: 1.088
[7, 2500] loss: 1.055
[7, 3000] loss: 1.055
[8, 500] loss: 1.021
[8, 1000] loss: 0.996
[8, 1500] loss: 1.016
[8, 2000] loss: 1.022
[8, 2500] loss: 1.031
[8, 3000] loss: 1.042
[9, 500] loss: 0.974
[9, 1000] loss: 0.951
[9, 1500] loss: 0.984
[9, 2000] loss: 0.980
[9, 2500] loss: 0.973
[9, 3000] loss: 0.978
```

```
[10, 500] loss: 0.941
[10, 1000] loss: 0.924
[10, 1500] loss: 0.933
[10, 2000] loss: 0.937
[10, 2500] loss: 0.929
[10, 3000] loss: 0.947
[11, 500] loss: 0.880
[11, 1000] loss: 0.865
[11, 1500] loss: 0.900
[11, 2000] loss: 0.912
[11, 2500] loss: 0.899
[11, 3000] loss: 0.916
[12, 500] loss: 0.832
[12, 1000] loss: 0.842
[12, 1500] loss: 0.863
[12, 2000] loss: 0.875
[12, 2500] loss: 0.863
[12, 3000] loss: 0.888
[13, 500] loss: 0.809
[13, 1000] loss: 0.815
[13, 1500] loss: 0.835
[13, 2000] loss: 0.828
[13, 2500] loss: 0.840
[13, 3000] loss: 0.852
[14, 500] loss: 0.758
[14, 1000] loss: 0.803
[14, 1500] loss: 0.793
[14, 2000] loss: 0.812
[14, 2500] loss: 0.807
[14, 3000] loss: 0.814
[15, 500] loss: 0.754
[15, 1000] loss: 0.755
[15, 1500] loss: 0.769
[15, 2000] loss: 0.783
[15, 2500] loss: 0.768
[15, 3000] loss: 0.786
[16, 500] loss: 0.702
[16, 1000] loss: 0.728
[16, 1500] loss: 0.725
[16, 2000] loss: 0.761
[16, 2500] loss: 0.750
[16, 3000] loss: 0.778
[17, 500] loss: 0.706
[17, 1000] loss: 0.704
[17, 1500] loss: 0.712
[17, 2000] loss: 0.728
[17, 2500] loss: 0.727
[17, 3000] loss: 0.738
[18, 500] loss: 0.671
```

```
[18, 1000] loss: 0.674
[18, 1500] loss: 0.696
[18, 2000] loss: 0.689
[18, 2500] loss: 0.717
[18, 3000] loss: 0.703
[19, 500] loss: 0.630
[19, 1000] loss: 0.651
[19, 1500] loss: 0.670
[19, 2000] loss: 0.665
[19, 2500] loss: 0.676
[19, 3000] loss: 0.700
[20, 500] loss: 0.606
[20, 1000] loss: 0.642
[20, 1500] loss: 0.646
[20, 2000] loss: 0.642
[20, 2500] loss: 0.649
[20, 3000] loss: 0.672
```




```
Accuracy of the network on the 10000 test images: 65 %
```