

CNN_pytorch_ch05

October 30, 2025

```
[ ]: import os
import torch
import torch.nn as nn
from torch.autograd import Variable
import torch.utils.data as Data
import torchvision
import torch.nn.functional as F
import numpy as np
#
learning_rate = 1e-4
# Dropout      PyTorch  Dropout      p      (1 - )
keep_prob_rate = 0.7
#
max_epoch = 3
#
BATCH_SIZE = 50

DOWNLOAD_MNIST = False
#      ./mnist/
if not(os.path.exists('./mnist/')) or not os.listdir('./mnist/'):
    # not mnist dir or mnist is empty dir
    DOWNLOAD_MNIST = True

#  ToTensor()  [0,255]  [0,1]  (C,H,W)
train_data = torchvision.datasets.MNIST(root='./mnist/', train=True, ↴
    transform=torchvision.transforms.ToTensor(), download=DOWNLOAD_MNIST,)
# DataLoader
train_loader = Data.DataLoader(dataset=train_data, batch_size=BATCH_SIZE, ↴
    shuffle=True)

#      transform
test_data = torchvision.datasets.MNIST(root='./mnist/', train=False)
#  500      [0,1]  [batch, 1, 28, 28]
test_x = test_data.test_data.unsqueeze(1).float()[:500] / 255.  #  [500, 1, ↴
    28, 28]
#      torchvision      data/targets
```

```

test_y = test_data.test_labels[:500].numpy()

class CNN(nn.Module):
    def __init__(self):
        super(CNN, self).__init__()
        self.conv1 = nn.Sequential(#nn.Sequential      "
            #nn.Conv2d PyTorch      "      "      "      "
            # [B, 1, 28, 28] -> Conv7x7  28x28 -> MaxPool2d(2) ->
            ↵ [B, 32, 14, 14]      = 32      32
            nn.Conv2d(
                # 7x7      =1      =32      =1
                # padding = (kernel_size - 1) // 2 = 3      28x28
                in_channels=1,#      "      "      MNIST      → 1
                out_channels=32,#      =      32      7x7
                kernel_size=7,#      "      "
                stride=1,#      "
                padding=3#      0      0
            ),
            nn.ReLU(),      #
            nn.MaxPool2d(2)      # 2x2      28x28 -> 14x14
        )
        self.conv2 = nn.Sequential(
            # [B, 32, 14, 14] -> Conv5x5  14x14 -> MaxPool2d(2) -> [B, 64, 7, 7]
            nn.Conv2d(
                # 5x5      =32      =64      =1 padding=2      14x14
                in_channels=32,
                out_channels=64,
                kernel_size=5,
                stride=1,
                padding=2
            ),
            nn.ReLU(),      #
            nn.MaxPool2d(2)      # 2x2      14x14 -> 7x7
        )
        # Flatten      64*7*7=3136
        self.out1 = nn.Linear(7*7*64, 1024, bias=True)      #      1      64*7*7      1024
        self.dropout = nn.Dropout(p=1 - keep_prob_rate)      # PyTorch      p      ↵
    ↵model.train()
        self.out2 = nn.Linear(1024,10,bias=True)

#forward      "      "      PyTorch      "      "
def forward(self, x):# 28x28      → 10      0~9
    x = self.conv1(x)      # -> [B, 32, 14, 14]
    x = self.conv2(x)      # -> [B, 64, 7, 7]
    x = x.view(x.size(0), -1)      #      [batch, 64, 7, 7] -> [batch, 64*7*7]
    out1 = self.out1(x)      #out1      [B, 1024]

```

```

        out1 = F.relu(out1) #   out1      ReLU max(0, x)    0
        out1 = self.dropout(out1) #
        out2 = self.out2(out1)          #   logits
        output = F.softmax(out2, dim=1) #   softmax      logits □
        ↪CrossEntropyLoss
        return output

def test(cnn):
    #   Dropout
    cnn.eval()
    with torch.no_grad():#
        y_pre = cnn(test_x) #   test_x
        #
        _, pre_index = torch.max(y_pre, 1)#
        prediction = pre_index.view(-1).cpu().numpy()
    correct = np.sum(prediction == test_y)#
    return correct / 500.0

def train(cnn):
    # Adam
    optimizer = torch.optim.Adam(cnn.parameters(), lr=learning_rate)
    #   logits softmax
    loss_func = nn.CrossEntropyLoss()
    for epoch in range(max_epoch):#   max_epoch      max_epoch
        cnn.train() #   Dropout
        for step, (x, y) in enumerate(train_loader):#   train_loader
            output = cnn(x) #   softmax      logits
            loss = loss_func(output, y) #
            optimizer.zero_grad() #
            loss.backward() #
            optimizer.step() #

            if step != 0 and step % 20 == 0:
                print("=" * 10, step, "=" * 5, "=" * 5, "test accuracy is ", ↪
        ↪test(cnn), "=" * 10)

if __name__ == '__main__':
    cnn = CNN()
    train(cnn)

```

```

100.0%
100.0%
100.0%
100.0%
d:\code\python\deeplearning\.venv\Lib\site-

```

```
packages\torchvision\datasets\mnist.py:81: UserWarning: test_data has been
renamed data
    warnings.warn("test_data has been renamed data")
d:\code\python\deeplearning\.venv\Lib\site-
packages\torchvision\datasets\mnist.py:71: UserWarning: test_labels has been
renamed targets
    warnings.warn("test_labels has been renamed targets")

===== 20 ===== test accuracy is 0.25 ======
===== 40 ===== test accuracy is 0.442 ======
===== 60 ===== test accuracy is 0.604 ======
===== 80 ===== test accuracy is 0.614 ======
===== 100 ===== test accuracy is 0.646 ======
===== 120 ===== test accuracy is 0.73 ======
===== 140 ===== test accuracy is 0.772 ======
===== 160 ===== test accuracy is 0.852 ======
===== 180 ===== test accuracy is 0.878 ======
===== 200 ===== test accuracy is 0.872 ======
===== 220 ===== test accuracy is 0.884 ======
===== 240 ===== test accuracy is 0.892 ======
===== 260 ===== test accuracy is 0.898 ======
===== 280 ===== test accuracy is 0.902 ======
===== 300 ===== test accuracy is 0.898 ======
===== 320 ===== test accuracy is 0.908 ======
===== 340 ===== test accuracy is 0.906 ======
===== 360 ===== test accuracy is 0.918 ======
===== 380 ===== test accuracy is 0.91 ======
===== 400 ===== test accuracy is 0.912 ======
===== 420 ===== test accuracy is 0.902 ======
===== 440 ===== test accuracy is 0.924 ======
===== 460 ===== test accuracy is 0.932 ======
===== 480 ===== test accuracy is 0.942 ======
===== 500 ===== test accuracy is 0.932 ======
===== 520 ===== test accuracy is 0.926 ======
===== 540 ===== test accuracy is 0.928 ======
===== 560 ===== test accuracy is 0.932 ======
===== 580 ===== test accuracy is 0.94 ======
===== 600 ===== test accuracy is 0.932 ======
===== 620 ===== test accuracy is 0.944 ======
===== 640 ===== test accuracy is 0.944 ======
===== 660 ===== test accuracy is 0.952 ======
===== 680 ===== test accuracy is 0.95 ======
===== 700 ===== test accuracy is 0.954 ======
===== 720 ===== test accuracy is 0.958 ======
===== 740 ===== test accuracy is 0.958 ======
===== 760 ===== test accuracy is 0.948 ======
===== 780 ===== test accuracy is 0.954 ======
===== 800 ===== test accuracy is 0.956 ======
```

```
===== 820 ===== test accuracy is 0.964 =====
===== 840 ===== test accuracy is 0.964 =====
===== 860 ===== test accuracy is 0.952 =====
===== 880 ===== test accuracy is 0.96 =====
===== 900 ===== test accuracy is 0.954 =====
===== 920 ===== test accuracy is 0.95 =====
===== 940 ===== test accuracy is 0.954 =====
===== 960 ===== test accuracy is 0.96 =====
===== 980 ===== test accuracy is 0.96 =====
===== 1000 ===== test accuracy is 0.95 =====
===== 1020 ===== test accuracy is 0.952 =====
===== 1040 ===== test accuracy is 0.962 =====
===== 1060 ===== test accuracy is 0.958 =====
===== 1080 ===== test accuracy is 0.956 =====
===== 1100 ===== test accuracy is 0.968 =====
===== 1120 ===== test accuracy is 0.962 =====
===== 1140 ===== test accuracy is 0.964 =====
===== 1160 ===== test accuracy is 0.966 =====
===== 1180 ===== test accuracy is 0.958 =====
===== 20 ===== test accuracy is 0.962 =====
===== 40 ===== test accuracy is 0.968 =====
===== 60 ===== test accuracy is 0.96 =====
===== 80 ===== test accuracy is 0.964 =====
===== 100 ===== test accuracy is 0.964 =====
===== 120 ===== test accuracy is 0.958 =====
===== 140 ===== test accuracy is 0.962 =====
===== 160 ===== test accuracy is 0.964 =====
===== 180 ===== test accuracy is 0.964 =====
===== 200 ===== test accuracy is 0.956 =====
===== 220 ===== test accuracy is 0.964 =====
===== 240 ===== test accuracy is 0.964 =====
===== 260 ===== test accuracy is 0.962 =====
===== 280 ===== test accuracy is 0.97 =====
===== 300 ===== test accuracy is 0.974 =====
===== 320 ===== test accuracy is 0.97 =====
===== 340 ===== test accuracy is 0.978 =====
===== 360 ===== test accuracy is 0.968 =====
===== 380 ===== test accuracy is 0.974 =====
===== 400 ===== test accuracy is 0.968 =====
===== 420 ===== test accuracy is 0.974 =====
===== 440 ===== test accuracy is 0.978 =====
===== 460 ===== test accuracy is 0.966 =====
===== 480 ===== test accuracy is 0.976 =====
===== 500 ===== test accuracy is 0.972 =====
===== 520 ===== test accuracy is 0.974 =====
===== 540 ===== test accuracy is 0.974 =====
===== 560 ===== test accuracy is 0.974 =====
===== 580 ===== test accuracy is 0.978 =====
```

```
===== 600 ===== test accuracy is 0.97 =====
===== 620 ===== test accuracy is 0.98 =====
===== 640 ===== test accuracy is 0.978 =====
===== 660 ===== test accuracy is 0.97 =====
===== 680 ===== test accuracy is 0.976 =====
===== 700 ===== test accuracy is 0.978 =====
===== 720 ===== test accuracy is 0.976 =====
===== 740 ===== test accuracy is 0.982 =====
===== 760 ===== test accuracy is 0.978 =====
===== 780 ===== test accuracy is 0.978 =====
===== 800 ===== test accuracy is 0.97 =====
===== 820 ===== test accuracy is 0.97 =====
===== 840 ===== test accuracy is 0.98 =====
===== 860 ===== test accuracy is 0.976 =====
===== 880 ===== test accuracy is 0.976 =====
===== 900 ===== test accuracy is 0.978 =====
===== 920 ===== test accuracy is 0.974 =====
===== 940 ===== test accuracy is 0.976 =====
===== 960 ===== test accuracy is 0.978 =====
===== 980 ===== test accuracy is 0.972 =====
===== 1000 ===== test accuracy is 0.978 =====
===== 1020 ===== test accuracy is 0.976 =====
===== 1040 ===== test accuracy is 0.976 =====
===== 1060 ===== test accuracy is 0.974 =====
===== 1080 ===== test accuracy is 0.978 =====
===== 1100 ===== test accuracy is 0.978 =====
===== 1120 ===== test accuracy is 0.972 =====
===== 1140 ===== test accuracy is 0.978 =====
===== 1160 ===== test accuracy is 0.976 =====
===== 1180 ===== test accuracy is 0.99 =====
===== 20 ===== test accuracy is 0.984 =====
===== 40 ===== test accuracy is 0.986 =====
===== 60 ===== test accuracy is 0.978 =====
===== 80 ===== test accuracy is 0.98 =====
===== 100 ===== test accuracy is 0.988 =====
===== 120 ===== test accuracy is 0.982 =====
===== 140 ===== test accuracy is 0.984 =====
===== 160 ===== test accuracy is 0.974 =====
===== 180 ===== test accuracy is 0.98 =====
===== 200 ===== test accuracy is 0.982 =====
===== 220 ===== test accuracy is 0.984 =====
===== 240 ===== test accuracy is 0.978 =====
===== 260 ===== test accuracy is 0.982 =====
===== 280 ===== test accuracy is 0.978 =====
===== 300 ===== test accuracy is 0.982 =====
===== 320 ===== test accuracy is 0.962 =====
===== 340 ===== test accuracy is 0.988 =====
===== 360 ===== test accuracy is 0.984 =====
```

```
===== 380 ===== test accuracy is 0.98 =====
===== 400 ===== test accuracy is 0.98 =====
===== 420 ===== test accuracy is 0.988 =====
===== 440 ===== test accuracy is 0.98 =====
===== 460 ===== test accuracy is 0.98 =====
===== 480 ===== test accuracy is 0.978 =====
===== 500 ===== test accuracy is 0.974 =====
===== 520 ===== test accuracy is 0.978 =====
===== 540 ===== test accuracy is 0.986 =====
===== 560 ===== test accuracy is 0.978 =====
===== 580 ===== test accuracy is 0.982 =====
===== 600 ===== test accuracy is 0.984 =====
===== 620 ===== test accuracy is 0.988 =====
===== 640 ===== test accuracy is 0.982 =====
===== 660 ===== test accuracy is 0.982 =====
===== 680 ===== test accuracy is 0.978 =====
===== 700 ===== test accuracy is 0.976 =====
===== 720 ===== test accuracy is 0.978 =====
===== 740 ===== test accuracy is 0.982 =====
===== 760 ===== test accuracy is 0.98 =====
===== 780 ===== test accuracy is 0.984 =====
===== 800 ===== test accuracy is 0.978 =====
===== 820 ===== test accuracy is 0.982 =====
===== 840 ===== test accuracy is 0.984 =====
===== 860 ===== test accuracy is 0.978 =====
===== 880 ===== test accuracy is 0.984 =====
===== 900 ===== test accuracy is 0.992 =====
===== 920 ===== test accuracy is 0.982 =====
===== 940 ===== test accuracy is 0.992 =====
===== 960 ===== test accuracy is 0.982 =====
===== 980 ===== test accuracy is 0.98 =====
===== 1000 ===== test accuracy is 0.974 =====
===== 1020 ===== test accuracy is 0.98 =====
===== 1040 ===== test accuracy is 0.982 =====
===== 1060 ===== test accuracy is 0.98 =====
===== 1080 ===== test accuracy is 0.986 =====
===== 1100 ===== test accuracy is 0.986 =====
===== 1120 ===== test accuracy is 0.98 =====
===== 1140 ===== test accuracy is 0.982 =====
===== 1160 ===== test accuracy is 0.992 =====
===== 1180 ===== test accuracy is 0.988 =====
```

[]:

[]:

[]: