

Data Science(Machine Learning, Deep Learning)

- 소스 깃 참고

16장 표현학습, 17장 확률론적 관점 패스

CNN(합성곱신경망) - Convolutional Neural Network

패턴 추출

pytorch, torchvision 설치가 안 되어 에러가 뜨는 경우
conda install pytorch torchvision install 해줘야함.

<cnn.py> - 모델 클래스 구현

<train.py 수정>

(base) C:\Users\admin\Desktop\BigData DeepLearning\DeepLearning

소스코드\실습\18-cnn>python train.py --model_fn ./model.pth --model cnn

Train: torch.Size([48000, 28, 28]) torch.Size([48000])

Valid: torch.Size([12000, 28, 28]) torch.Size([12000])

ConvolutionalClassifier(
 (blocks): Sequential(
 (0): ConvolutionBlock(
 (layers): Sequential(
 (0): Conv2d(1, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
 (1): ReLU()
 (2): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
 (3): Conv2d(32, 32, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
 (4): ReLU()
 (5): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
)
)
 (1): ConvolutionBlock(
 (layers): Sequential(
 (0): Conv2d(32, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
 (1): ReLU()
 (2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
 (3): Conv2d(64, 64, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
 (4): ReLU()
 (5): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
)
)
 (2): ConvolutionBlock(
 (layers): Sequential(
 (0): Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
 (1): ReLU()
 (2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True,

```

track_running_stats=True)
    (3): Conv2d(128, 128, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
    (4): ReLU()
    (5): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  )
)
(3): ConvolutionBlock(
  (layers): Sequential(
    (0): Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): ReLU()
    (2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (3): Conv2d(256, 256, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
    (4): ReLU()
    (5): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  )
)
(4): ConvolutionBlock(
  (layers): Sequential(
    (0): Conv2d(256, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): ReLU()
    (2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (3): Conv2d(512, 512, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
    (4): ReLU()
    (5): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  )
)
)
(layers): Sequential(
  (0): Linear(in_features=512, out_features=50, bias=True)
  (1): ReLU()
  (2): BatchNorm1d(50, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  (3): Linear(in_features=50, out_features=10, bias=True)
  (4): LogSoftmax(dim=-1)
)
)
Adam (
Parameter Group 0
  amsgrad: False
  betas: (0.9, 0.999)
  capturable: False
  differentiable: False
  eps: 1e-08
  foreach: None
  fused: None

```

```

lr: 0.001
maximize: False
weight_decay: 0
)
NLLLoss()
Epoch(1/10): train_loss=1.8603e-01  valid_loss=8.7814e-02  lowest_loss=8.7814e-02
Epoch(2/10): train_loss=5.3109e-02  valid_loss=8.4416e-02  lowest_loss=8.4416e-02
Epoch(3/10): train_loss=3.6915e-02  valid_loss=4.5671e-02  lowest_loss=4.5671e-02
Epoch(4/10): train_loss=2.4024e-02  valid_loss=4.1718e-02  lowest_loss=4.1718e-02
Epoch(5/10): train_loss=2.2342e-02  valid_loss=4.2429e-02  lowest_loss=4.1718e-02
Epoch(6/10): train_loss=2.2036e-02  valid_loss=4.5503e-02  lowest_loss=4.1718e-02
Epoch(7/10): train_loss=1.4721e-02  valid_loss=6.0025e-02  lowest_loss=4.1718e-02
Epoch(8/10): train_loss=1.4759e-02  valid_loss=4.6336e-02  lowest_loss=4.1718e-02
Epoch(9/10): train_loss=1.4324e-02  valid_loss=3.6083e-02  lowest_loss=3.6083e-02
Epoch(10/10): train_loss=7.2854e-03  valid_loss=6.8029e-02  lowest_loss=3.6083e-02

```

python train.py --model_fn ./model.pth --model cnn을 사용하여 트레이닝을 했을 때 이와 같은 형태가 나온다.

<predict.ipynb 수정>

```

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

```

```

from mnist_classifier.utils import load_mnist
from mnist_classifier.utils import get_model
이 부분에서 에러가 발생함.

```

<에러 해결>

Colab에서 train.py로 만든 model.pth content로 이동 시키기
mnist_classifier 폴더 생성 trainer.py utils.py 이동
하위 폴더로 models 폴더 생성 cnn.py fc.py 이동

```

├── content
│   ├── mnist_classifier
│   │   ├── models
│   │   │   ├── cnn.py
│   │   │   ├── fc.py
│   │   │   ├── trainer.py
│   │   │   └── utils.py
│   │   └── sample_data
│   └── model.pth

```

이와 같이 하면 된다.

MNIST 테스트 세팅

```

# Load MNIST test set.
x, y = load_mnist(is_train=False, flatten=(train_config.model == "fc"))
x, y = x.to(device), y.to(device)

```

```
print(x.shape, y.shape)
```

```
input_size = int(x.shape[-1])  
output_size = int(max(y)) + 1
```

```
model = get_model(  
    input_size,  
    output_size,  
    train_config,  
    device,  
)
```

```
model.load_state_dict(model_dict)
```

```
test(model, x, y, to_be_shown=False)
```

```
Downloading http://yann.lecun.com/exdb/mnist/train-images-idx3-ubyte.gz  
Downloading http://yann.lecun.com/exdb/mnist/train-images-idx3-ubyte.gz to  
../data/MNIST/raw/train-images-idx3-ubyte.gz  
100%|████████████████████| 9912422/9912422 [00:00<00:00, 128978523.84it/s]Extracting  
../data/MNIST/raw/train-images-idx3-ubyte.gz to ../data/MNIST/raw
```

```
Downloading http://yann.lecun.com/exdb/mnist/train-labels-idx1-ubyte.gz  
Downloading http://yann.lecun.com/exdb/mnist/train-labels-idx1-ubyte.gz to  
../data/MNIST/raw/train-labels-idx1-ubyte.gz  
100%|████████████████████| 28881/28881 [00:00<00:00, 32554607.32it/s]  
Extracting ../data/MNIST/raw/train-labels-idx1-ubyte.gz to ../data/MNIST/raw
```

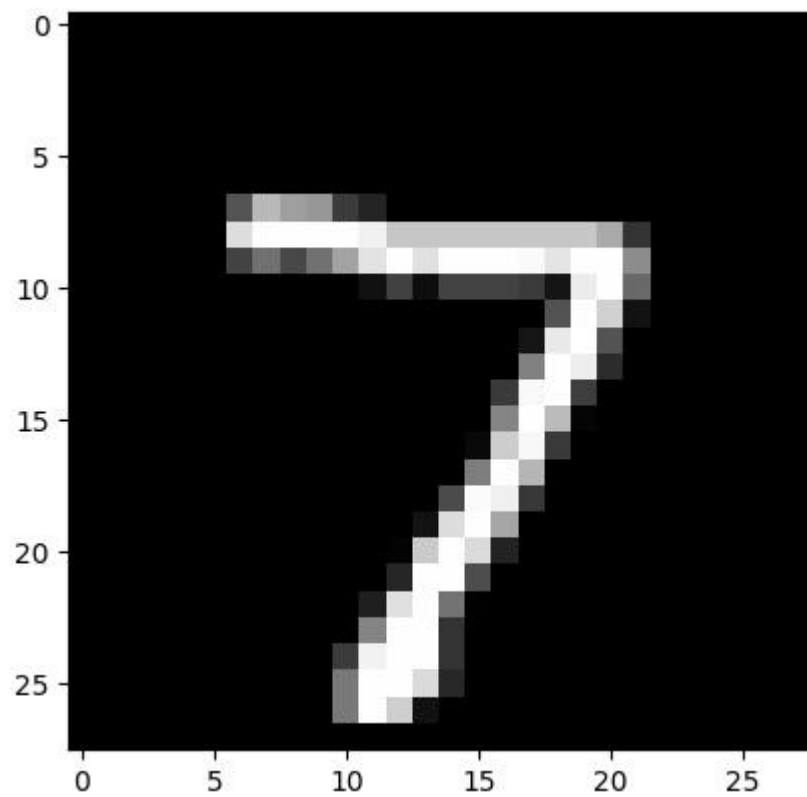
```
Downloading http://yann.lecun.com/exdb/mnist/t10k-images-idx3-ubyte.gz  
Downloading http://yann.lecun.com/exdb/mnist/t10k-images-idx3-ubyte.gz to  
../data/MNIST/raw/t10k-images-idx3-ubyte.gz  
100%|████████████████████| 1648877/1648877 [00:00<00:00, 68804570.43it/s]Extracting  
../data/MNIST/raw/t10k-images-idx3-ubyte.gz to ../data/MNIST/raw
```

```
Downloading http://yann.lecun.com/exdb/mnist/t10k-labels-idx1-ubyte.gz  
Downloading http://yann.lecun.com/exdb/mnist/t10k-labels-idx1-ubyte.gz to  
../data/MNIST/raw/t10k-labels-idx1-ubyte.gz  
100%|████████████████████| 4542/4542 [00:00<00:00, 3428203.85it/s]  
Extracting ../data/MNIST/raw/t10k-labels-idx1-ubyte.gz to ../data/MNIST/raw
```

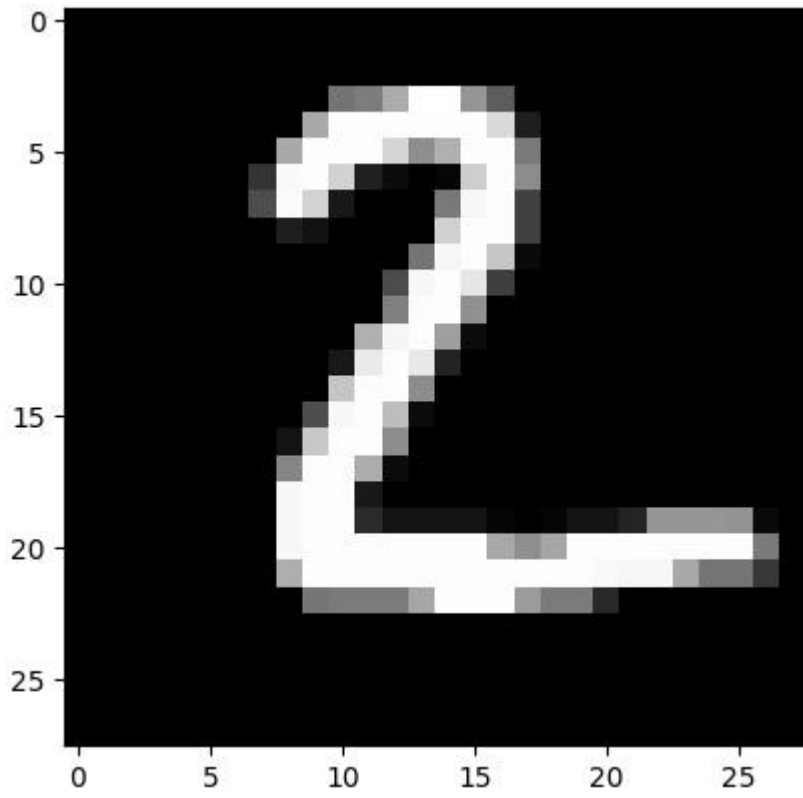
```
torch.Size([10000, 28, 28]) torch.Size([10000])  
Accuracy: 0.9916
```

```
n_test = 20  
test(model, x[:n_test], y[:n_test], to_be_shown=True)
```

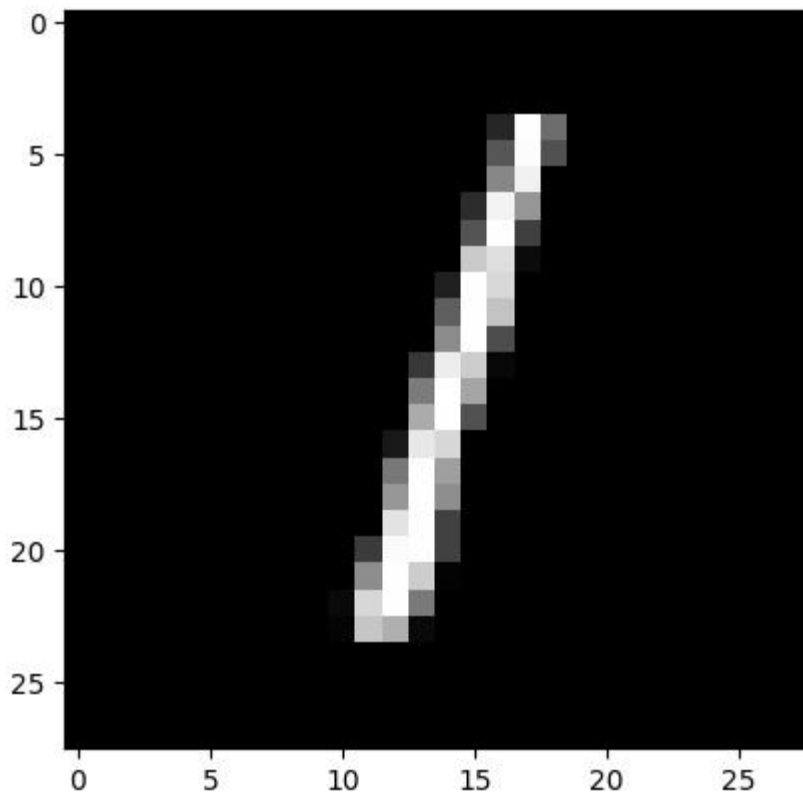
Accuracy: 1.0000



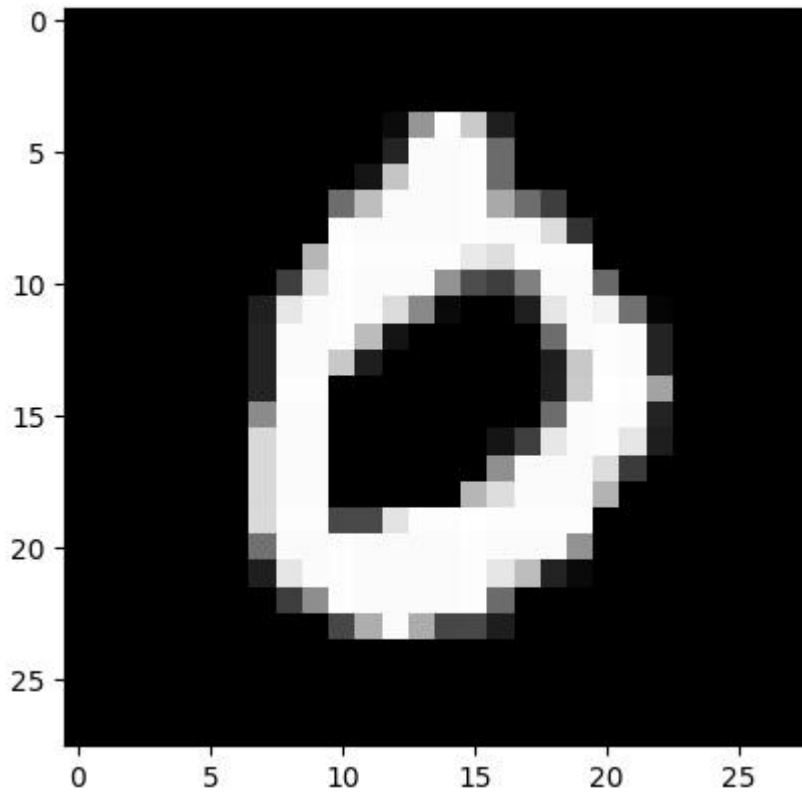
Predict: 7.0



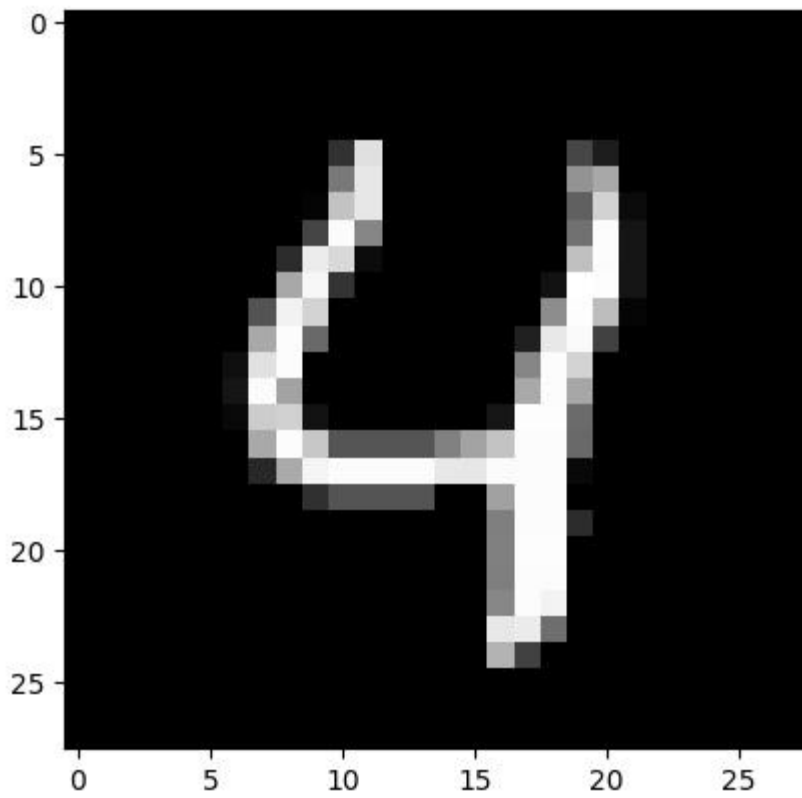
Predict: 2.0



Predict: 1.0



Predict: 0.0



이와 같은 형태로 나타난다.