

# 합성곱 신경망

---

Convolution Neural Network (CNN)

# Agenda

- DNN 모델 클래스화 하기
  - MNIST DNN 모델 클래스화 연습
- CNN 모델 클래스로 만들기
  - **MNIST CNN 모델 학습 연습**

# 1. How to make NN Model (MNIST)

- Pytorch 모델 클래스화 하기

## 기본 모델 정의

```
[ ] 1 linear1 = torch.nn.Linear(784,512,bias=True)
2 linear2 = torch.nn.Linear(512,512,bias=True)
3 linear3 = torch.nn.Linear(512,512,bias=True)
4 linear4 = torch.nn.Linear(512,512,bias=True)
5 linear5 = torch.nn.Linear(512,10,bias=True)
6 relu = torch.nn.ReLU()
```

```
[ ] 1 # Random Init => Xavier Init
2 torch.nn.init.xavier_uniform_(linear1.weight)
3 torch.nn.init.xavier_uniform_(linear2.weight)
4 torch.nn.init.xavier_uniform_(linear3.weight)
5 torch.nn.init.xavier_uniform_(linear4.weight)
6 torch.nn.init.xavier_uniform_(linear5.weight)
```

```
[ ] 1 # =====
2 # relu는 맨 마지막 레이어에서 빼는 것이 좋다.
3 # =====
4 model = torch.nn.Sequential(linear1,relu,
5                             linear2,relu,
6                             linear3,relu,
7                             linear4,relu,
8                             linear5).to(device)
```



## 모델 클래스화[2]

```
[15] 1 class NN(torch.nn.Module):
2     def __init__(self):
3         super(NN,self).__init__() # 이부분 빼먹기 쉬운데 조심할 것
4
5         # 주의) self 빼먹지 말자!
6         self.linear1 = torch.nn.Linear(784,512,bias=True)
7         self.linear2 = torch.nn.Linear(512,512,bias=True)
8         self.linear3 = torch.nn.Linear(512,512,bias=True)
9         self.linear4 = torch.nn.Linear(512,512,bias=True)
10        self.linear5 = torch.nn.Linear(512,10,bias=True)
11        self.relu = torch.nn.ReLU()
12
13        torch.nn.init.xavier_uniform_(self.linear1.weight)
14        torch.nn.init.xavier_uniform_(self.linear2.weight)
15        torch.nn.init.xavier_uniform_(self.linear3.weight)
16        torch.nn.init.xavier_uniform_(self.linear4.weight)
17        torch.nn.init.xavier_uniform_(self.linear5.weight)
18
19    def forward(self,x):
20        out = self.linear1(x)
21        out = self.relu(out)
22        out = self.linear2(out)
23        out = self.relu(out)
24        out = self.linear3(out)
25        out = self.relu(out)
26        out = self.linear4(out)
27        out = self.relu(out)
28        out = self.linear5(out)
29        return out
30
[16] 1 model = NN().to(device)
```

# 1. How to make NN Model

## • Pytorch 모델 클래스화 하기

```
1 class NN(torch.nn.Module):
2     def __init__(self):
3         super(NN,self).__init__() # 이부분 빼먹기 쉬운데 조심할 것
4
5         # 주의) self 빼먹지 말자!
6         self.linear1 = torch.nn.Linear(784,512,bias=True)
7         self.linear2 = torch.nn.Linear(512,512,bias=True)
8         self.linear3 = torch.nn.Linear(512,512,bias=True)
9         self.linear4 = torch.nn.Linear(512,512,bias=True)
10        self.linear5 = torch.nn.Linear(512,10,bias=True)
11        self.relu = torch.nn.ReLU()
12
13        torch.nn.init.xavier_uniform_(self.linear1.weight)
14        torch.nn.init.xavier_uniform_(self.linear2.weight)
15        torch.nn.init.xavier_uniform_(self.linear3.weight)
16        torch.nn.init.xavier_uniform_(self.linear4.weight)
17        torch.nn.init.xavier_uniform_(self.linear5.weight)
18
19    def forward(self,x):
20        out = self.linear1(x)
21        out = self.relu(out)
22        out = self.linear2(out)
23        out = self.relu(out)
24        out = self.linear3(out)
25        out = self.relu(out)
26        out = self.linear4(out)
27        out = self.relu(out)
28        out = self.linear5(out)
29        return out
30
```

```
1 model = NN().to(device)
```

[L1] torch.nn.Module을 상속받는 파이썬 클래스

[L2] \_\_init\_\_()은 모델의 구조와 동작을 정의하는 생성자 객체가 생성될 때 호출됩니다.

[L3] super() 함수를 부르면 여기서 만든 클래스는 nn.Module 클래스의 속성들을 가지고 초기화 됩니다

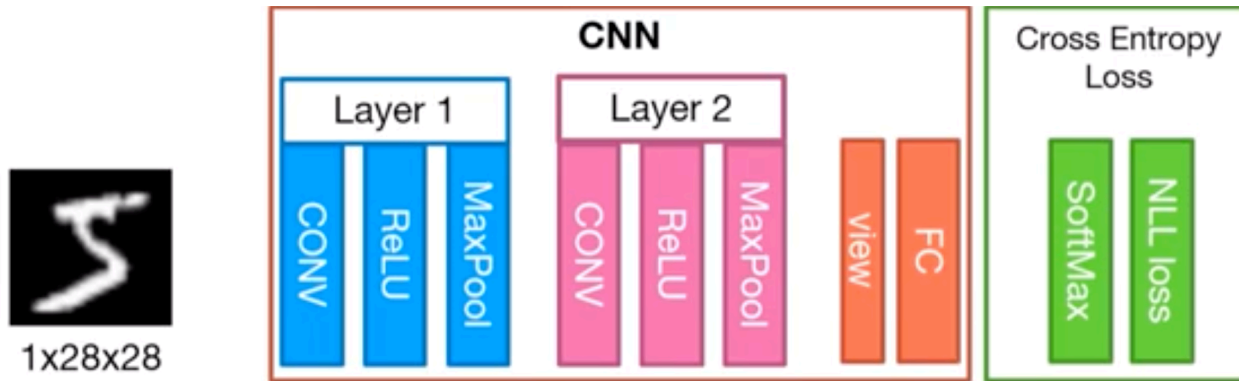
[L19] foward() 함수는 모델이 학습데이터를 입력 받아서 forward 연산을 진행시키는 함수입니다.

이 forward() 함수는 model 객체를 데이터와 함께 호출하면 자동으로 실행이됩니다.

예를 들어 model이란 이름의 객체를 생성 후, model(입력 데이터)와 같은 형식으로 객체를 호출하면 자동으로 forward 연산이 수행됩니다.

\*\*  $H(x)$  식에 입력  $x$ 로부터 예측된  $y$ 를 얻는 것을 forward 연산이라고 합니다.

## 2. How to make CNN Model (MNIST)



```
1 ## 버전 1.0
2 class CNN(torch.nn.Module):
3     def __init__(self):
4         super(CNN, self).__init__() # 이부분 빼먹기 쉬운데 조심할 것
5
6         self.conv1 = torch.nn.Conv2d(1, 32, kernel_size=3, stride=1, padding=1)
7         self.relu1 = torch.nn.ReLU()
8         self.maxpool1 = torch.nn.MaxPool2d(kernel_size=2, stride=2)
9         self.conv2 = torch.nn.Conv2d(32, 64, kernel_size=3, stride=1, padding=1)
10        self.relu2 = torch.nn.ReLU()
11        self.maxpool2 = torch.nn.MaxPool2d(kernel_size=2, stride=2)
12        self.fc = torch.nn.Linear(7*7*64, 10, bias=True)
13        torch.nn.init.kaiming_uniform(self.fc.weight)
14
15    def forward(self, x):
16        out = self.conv1(x)
17        out = self.relu1(out)
18        out = self.maxpool1(out)
19        out = self.conv2(out)
20        out = self.relu2(out)
21        out = self.maxpool2(out)
22        out = out.view(out.size(0), -1) # out = Batch(0) x 7(1) x 7(2) x 64(3)
23        out = self.fc(out)
24        return out
```

```
1 # 버전 2.0
2 class CNN(torch.nn.Module):
3     def __init__(self):
4         super(CNN, self).__init__() # 이부분 빼먹기 쉬운데 조심할 것
5
6         self.layer1 = torch.nn.Sequential(
7             torch.nn.Conv2d(1, 32, kernel_size=3, stride=1, padding=1),
8             torch.nn.ReLU(),
9             torch.nn.MaxPool2d(kernel_size=2, stride=2)
10        )
11
12        self.layer2 = torch.nn.Sequential(
13            torch.nn.Conv2d(32, 64, kernel_size=3, stride=1, padding=1),
14            torch.nn.ReLU(),
15            torch.nn.MaxPool2d(kernel_size=2, stride=2)
16        )
17
18        self.fc = torch.nn.Linear(7*7*64, 10, bias=True)
19        torch.nn.init.kaiming_uniform(self.fc.weight)
20
21    def forward(self, x):
22        out = self.layer1(x)
23        out = self.layer2(out)
24        out = out.view(out.size(0), -1) # out = Batch x 7 x 7 x 64
25        out = self.fc(out)
26        return out
```

## 2. How to make CNN Model (MNIST)

```
1 # 버전 2.0
2 class CNN(torch.nn.Module):
3     def __init__(self):
4         super(CNN, self).__init__() # 이부분 빼먹기 쉬운데 조심할 것
5
6         self.layer1 = torch.nn.Sequential(
7             torch.nn.Conv2d(1, 32, kernel_size=3, stride=1, padding=1),
8             torch.nn.ReLU(),
9             torch.nn.MaxPool2d(kernel_size=2, stride=2)
10        )
11
12        self.layer2 = torch.nn.Sequential(
13            torch.nn.Conv2d(32, 64, kernel_size=3, stride=1, padding=1),
14            torch.nn.ReLU(),
15            torch.nn.MaxPool2d(kernel_size=2, stride=2)
16        )
17
18        self.fc = torch.nn.Linear(7*7*64, 10, bias=True)
19        torch.nn.init.kaiming_uniform(self.fc.weight)
20
21    def forward(self, x):
22        out = self.layer1(x)
23        out = self.layer2(out)
24        out = out.view(out.size(0), -1) # out = Batch x 7 x 7 x 64
25        out = self.fc(out)
26        return out
```

[L18]

FC 이전 feature Map은 2D 데이터임 (Nx7x7x64)  
FC 레이어 입력은 7x7x64로 1D 데이터를 입력 받음

따라서, MaxPool2D → FC 로 데이터가 넘어갈 때  
데이터 변형이 필요함 (2D → 1D)

[L24] View 함수를 사용하여 변형 가능함

[L24]

Out 데이터는 [배치크기x7x7x64] 이다.

Out.size(0)은 배치크기

Out.size(1)은 7

Out.size(2)은 7

Out.size(3)은 64이다.

# [실습 연습]

- MNIST DNN 코드 클래스화 연습
  - <https://colab.research.google.com/drive/12Kcx-wTr0PMF82eCGGtPDzGPlwtX-DKF?usp=sharing>
- MNIST CNN 모델 학습 연습
  - <https://colab.research.google.com/drive/13rNXo-OqmII95tl2Iec76SgWahfy3jZ9?usp=sharing>

# 추천 도서

- <https://wikidocs.net/55580>


📖 PyTorch로 시작하는 딥 러닝 입문

## PyTorch로 시작하는 딥 러닝 입문



지은이 : 원준

최종 편집일시 : 2020년 6월 2일 9:08 오전

저작권 : 

👍 136 명이 추천

이 책은 딥 러닝 프레임워크 PyTorch를 사용하여 딥 러닝에 입문하는 것을 목표

현재 작성 중에 있습니다.

많은 피드백 부탁드립니다.

감사합니다.

최근 댓글 (16)

최근 수정글 (10)

🕒 최근변경 RSS

다음글 : 00. 파이토치 공식 문서 링크



**END**

---