Pytorch

torch.nn

발표 주제

이 동헌

2018.03.02.

torch.nn

- 지원하는 기능
  - Parameters
  - Containers
  - Conv
  - Pooling
  - Padding
  - Non-linear Activation
  - Normalization
  - Recurrent
  - Linear
  - Dropout
  - Loss
  - Vision
  - 등등

torch.nn.functional

- 지원하는 기능
  - Conv
  - Pooling
  - Non-linear activation
  - Normalizaion
  - Linear function(=fully connected layer)
  - Dropout
  - Distance
  - Loss
  - Vision
  - 등등

torch.nn.Con2d

- nn.Conv2d(in_channels, out-channels, kernel_size, ...)
  - in_channels : 들어가는 이미지의 채널
  - out_channels : 나오는 이미지의 채널(사용하고자 하는 filter의 갯수)
  - kernel_size : 커널의 크기(filter의 크기, 3 -> filter = 3x3)
  - weight를 직접 설정해주지 않는다, -> 자동으로 설정 된다.

```
input = torch.ones(1,1,3,3)
input = Variable(input, requires_grad=True)

func = nn.Conv2d(1,1,3)
print(func.weight)

out = func(input)
print('=======================================')
print(out)
```

```
Parameter containing:
tensor([[[[ 0.0182,  0.1602,  0.0071],
          [ 0.3156, -0.0377,  0.2567],
          [ 0.0589,  0.0302,  0.0760]]]])
=======================================
tensor([[[[ 0.9719]]]])
```

  - 실제로 연산을 실행했을대 weight에 1을 곱한값을 다 더했을 때 0.9719와 값이 다르게 나온다.
  - 그 이유는 nn.conv2d는 bias를 가지고 있기때문에 이 값이 더해져서 나오기 때문이다.

torch.nn.functional.Conv2d

- functional.con2d(input, weight, ...)
  - input과 weight를 직접 넣어줘야한다.
  - weight에는 직접만든 filter를 넣어준다.

```
input = torch.ones(1,1,3,3)
filter = torch.ones(1,1,3,3)

input = Variable(input, requires_grad=True)
print(input)

filter = Variable(filter)
out = F.conv2d(input, filter)
print(out)
```

```
tensor([[[[ 1.,  1.,  1.],
          [ 1.,  1.,  1.],
          [ 1.,  1.,  1.]]]])
tensor([[[[ 9.]]]])
```