1 [Back propagation]

20170243 심재윤

$f_w(x)$, $w = (w_1, w_2, w_3)^T \in \mathbb{R}^3$

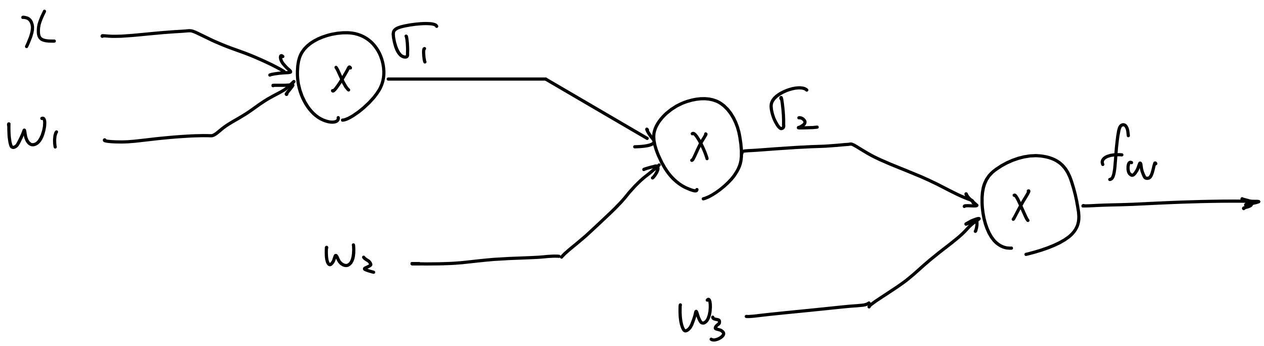
$$f_w(x) := w_3 \sigma_2 \big(\underbrace{w_2 \sigma_1 (\overbrace{w_1 x}^{x_1})}_{x_2} \big)$$

where $\sigma_1(u) = \sigma_2(u) = \dfrac{1}{1+\exp(-u)}$

$x_1 = w_1 x$

$x_2 = w_2 \sigma_1 (x_1)$

(a)



(b) $\dfrac{\partial \sigma_1}{\partial u}$

(i) $$\frac{\partial \sigma_1}{\partial u} = \frac{-\exp(-u)(-1)}{(1+\exp(-u))^2}$$

$$= \frac{\exp(-u)}{(1+\exp(-u))^2}$$

(ii) $$\frac{\partial \sigma_1}{\partial u} = \sigma_1(u)\,(1-\sigma_1(u))$$

(c) forward pass : process of receiving the inputs(x) and current parameters(w) and calculating the output (f_w)

backward pass : process of back-propagate error from the final output (f_w) and intermediate results to obtain the derivatives after forward pass

(d) $$\frac{\partial f_w}{\partial w_3} = \sigma_2(w_2 \sigma_1(w_1 x))$$

(e) $$\frac{\partial f_w}{\partial w_2} = \frac{\partial f_w}{\partial \sigma_2} \cdot \frac{\partial \sigma_2}{\partial x_2} \cdot \frac{\partial x_2}{\partial w_2}$$

$$= w_3 \sigma_2(x_2)(1-\sigma_2(x_2)) \cdot \sigma_1(x_1)$$

$$\frac{\partial f_w}{\partial w_1} = \frac{\partial f_w}{\partial \sigma_2} \frac{\partial \sigma_2}{\partial x_2} \frac{\partial x_2}{\partial \sigma_1} \frac{\partial \sigma_1}{\partial x_1} \frac{\partial x_1}{\partial w_1}$$

$$= w_3 \sigma_2(x_2)(1-\sigma_2(x_2)) \cdot w_2 \sigma_1(x_1)(1-\sigma_1(x_1)) \cdot x$$

(f) $\mathcal{L}(w) := \sum_{(x,y)\in D} \left(y - f_w(x)\right)^2 \qquad D = \{(x^1, y^1), (x^2, y^2)\}$

$$\nabla_w \mathcal{L}(w) = \frac{\partial \mathcal{L}(w)}{\partial f_w(x)} \cdot \frac{\partial f_w(x)}{\partial w}$$

$$= \sum_{(x,y)\in D} 2\left(y - f_w(x)\right)(-1) \cdot \frac{\partial f_w(x)}{\partial w}$$

$$= \left(2f_w(x^1) - 2y^1\right) \cdot \frac{\partial f_w(x^1)}{\partial w} + \left(2f_w(x^2) - 2y^2\right) \cdot \frac{\partial f_w(x^2)}{\partial w}$$

$$= \left(2f_w(x^1) - 2y^1\right) \begin{bmatrix} \frac{\partial f_w(x^1)}{\partial w_1} \\ \frac{\partial f_w(x^1)}{\partial w_2} \\ \frac{\partial f_w(x^1)}{\partial w_2} \end{bmatrix} + \left(2f_w(x^2) - 2y^2\right) \begin{bmatrix} \frac{\partial f_w(x^2)}{\partial w_1} \\ \frac{\partial f_w(x^2)}{\partial w_2} \\ \frac{\partial f_w(x^2)}{\partial w_2} \end{bmatrix}$$

$$\begin{bmatrix} w_1^{new} \\ w_2^{new} \\ w_3^{new} \end{bmatrix} \leftarrow \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} - \alpha \begin{bmatrix} \left(2f_w(x^1) - 2y^1\right)\frac{\partial f_w(x^1)}{\partial w_1} + \left(2f_w(x_2) - 2y_2\right)\frac{\partial f_w(x^2)}{\partial w_1} \\ \left(2f_w(x^1) - 2y^1\right)\frac{\partial f_w(x^1)}{\partial w_2} + \left(2f_w(x_2) - 2y_2\right)\frac{\partial f_w(x^2)}{\partial w_2} \\ \left(2f_w(x^1) - 2y^1\right)\frac{\partial f_w(x^1)}{\partial w_3} + \left(2f_w(x_2) - 2y_2\right)\frac{\partial f_w(x^2)}{\partial w_3} \end{bmatrix}$$

s.t. $\frac{\partial f_w(x^i)}{\partial w_1} = w_3 \sigma_2(x_2)\left(1 - \sigma_2(x_2)\right) \cdot w_2 \sigma_1(x_1)\left(1 - \sigma_1(x_1)\right) \cdot x^i$

$$\frac{\partial f_w(x^i)}{\partial w_2} = w_3 \sigma_2(x_2)\left(1 - \sigma_2(x_2)\right) \sigma_1(x_1)$$

$$\frac{\partial f_w(x^i)}{\partial w_3} = \sigma_2\left(w_2 \sigma_1(w_1 x^i)\right) \qquad , \forall i = 1, 2$$

$\boxed{2}$ [MNIST]

(a) input dimension : 28×28

filter dimension : $5 \times 5 \times 1 \times 20$ → # of filters

output dimension after 2D-convolution : $24 \times 24 \times 20$

output dimension after 2x2 max-pooling : $12 \times 12 \times 20$

(b) (conv + pooling) $28 \times 28 \times 1 \xrightarrow[(5\times5\times1)\times20]{} 24 \times 24 \times 20 \rightarrow 12 \times 12 \times 20$

(conv + pooling) $12 \times 12 \times 20 \xrightarrow[(5\times5\times20)\times50]{} 8 \times 8 \times 50 \rightarrow 4 \times 4 \times 50$

$$\frac{N + \cancel{2P} - f}{S} - 1 = \underset{8}{\text{output size}}$$

$$\frac{12 - f}{S} = 9$$

→ $f = 5$, $s = 1$

$f = -6$, $s = 2$ (X)

⋮

filter size = 5×5

stride = 1

channel dimension = 20

(assuming padding 0)

(c)

conv1 : $(5\times5\times1+1)\times20 = 520$

relu1 : 0

pool1 : 0 (pooling has no parameters.)

Conv2 : $(5\times5\times20+1)\times50 = 25050$

relu2 : 0

pool 2 : 0

FC1 : $(4\times4\times50+1)\times500 = 400500$

FC2 : $(500+1)\times10 = 5010$

$\therefore$ # of parameters $= 431080$ (including biases)

And I observed the test set accuracy as 99.25.

```python
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()

        # [your task1]
        ##############################
        ## declare the layers of the network which have parameters
        self.conv1 = nn.Conv2d(1, 20, 5, 1)
        self.conv2 = nn.Conv2d(20, 50, 5, 1)
        self.fc1 = nn.Linear(50*4*4, 500)
        self.fc2 = nn.Linear(500 ,10)
        ##############################


    def forward(self, x):
        # [your task1]
        ##############################
        ## combine the layers; don't forget the relu and pooling operations
        x = F.relu(self.conv1(x))     # the first conv layer + ReLU
        x = F.max_pool2d(x, 2, 2)     # the first pooling layer
        x = F.relu(self.conv2(x))         # the second conv layer + ReLU
        x = F.max_pool2d(x, 2, 2)         # the second pooling layer
        x = x.view(-1, 50*4*4)
        x = F.relu(self.fc1(x))         # the first linear layer + ReLU
        return self.fc2(x)            # the second linear layer
        ##############################
```

```
(assn) simjaeyoon@simjaeyoonui-MacBookPro code % python DeepMNIST.py
10000
60000
torch.Size([20, 1, 5, 5])
torch.Size([20])
torch.Size([50, 20, 5, 5])
torch.Size([50])
torch.Size([500, 800])
torch.Size([500])
torch.Size([10, 500])
torch.Size([10])
431080
Test Accuracy: 9.330000
Test Accuracy: 98.320000
Test Accuracy: 98.380000
Test Accuracy: 98.920000
Test Accuracy: 98.900000
Test Accuracy: 98.940000
Test Accuracy: 99.100000
Test Accuracy: 99.100000
Test Accuracy: 99.250000
Test Accuracy: 98.950000
Test Accuracy: 99.080000
```