Multilayer Perceptron

POSTECH A.I

Seungbeom Lee

The contents is made based on moonjeong's notes

Table of Contents

Table of Contents

- Review the key points
  - Multilayer Perceptron (MLP)
  - Representation Power
  - Backpropagation
- Practice
  - Implement Backpropagation
  - Multilayer Perceptron

Multilayer Perceptron

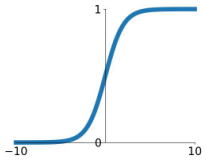
- In a neural network, given an input $\mathbf{x} \in \mathbb{R}^{D_0}$ a neuron is defined as

$$f = \sigma(\mathbf{w}^\top \mathbf{x} + b)$$

where σ is nonlinear activation function, $\mathbf{w} \in \mathbb{R}^{D_0}$ is a weight, $b \in \mathbb{R}$ is a bias.

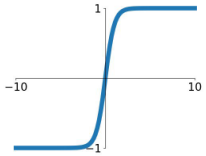
Sigmoid

$\sigma(x) = \frac{1}{1+e^{-x}}$



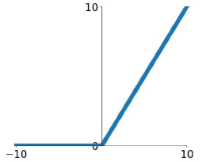
tanh

$\tanh(x)$



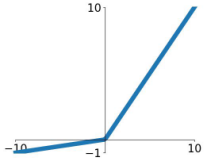
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

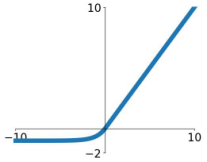


Maxout

$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$

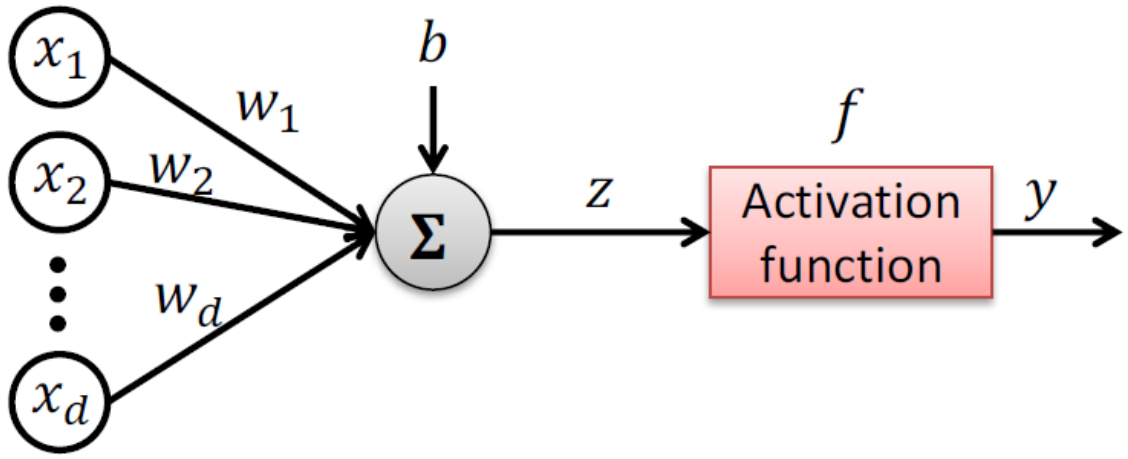


Multilayer Perceptron

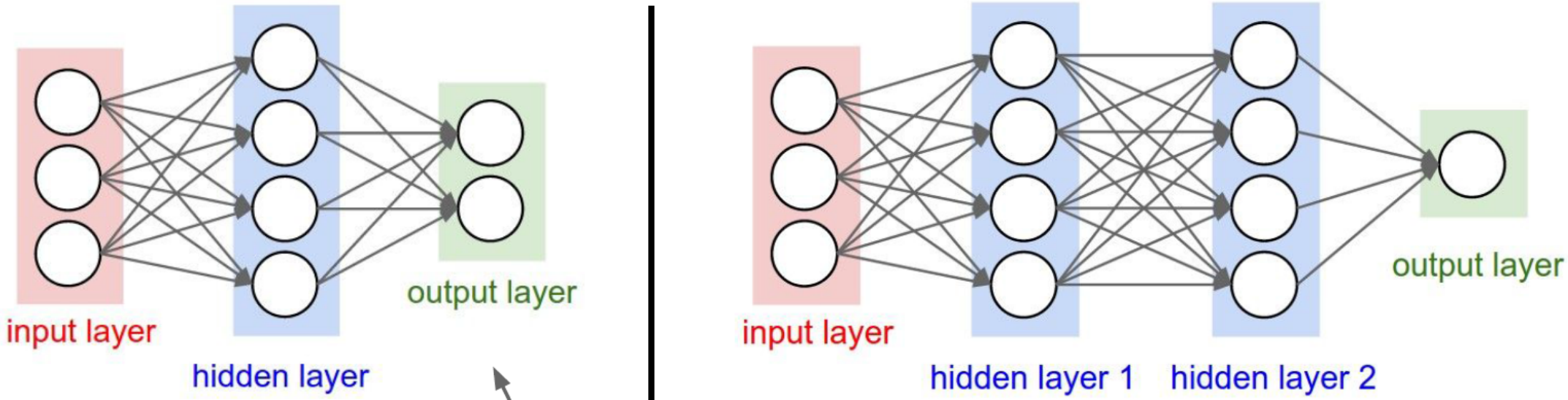
Perceptron: Single-Layer Neural Net

- Framework
  - Input: $\boldsymbol{x} = (x_1, x_2, \dots, x_d)^{\mathrm{T}}$
  - Output: y
  - Model: weight vector $\boldsymbol{w} = (w_1, w_2, \dots, w_d)^{\mathrm{T}}$ and bias b

$$y = f(z) = f\left(\sum_i w_i x_i + b\right) = f(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b)$$



Multilayer Perceptron



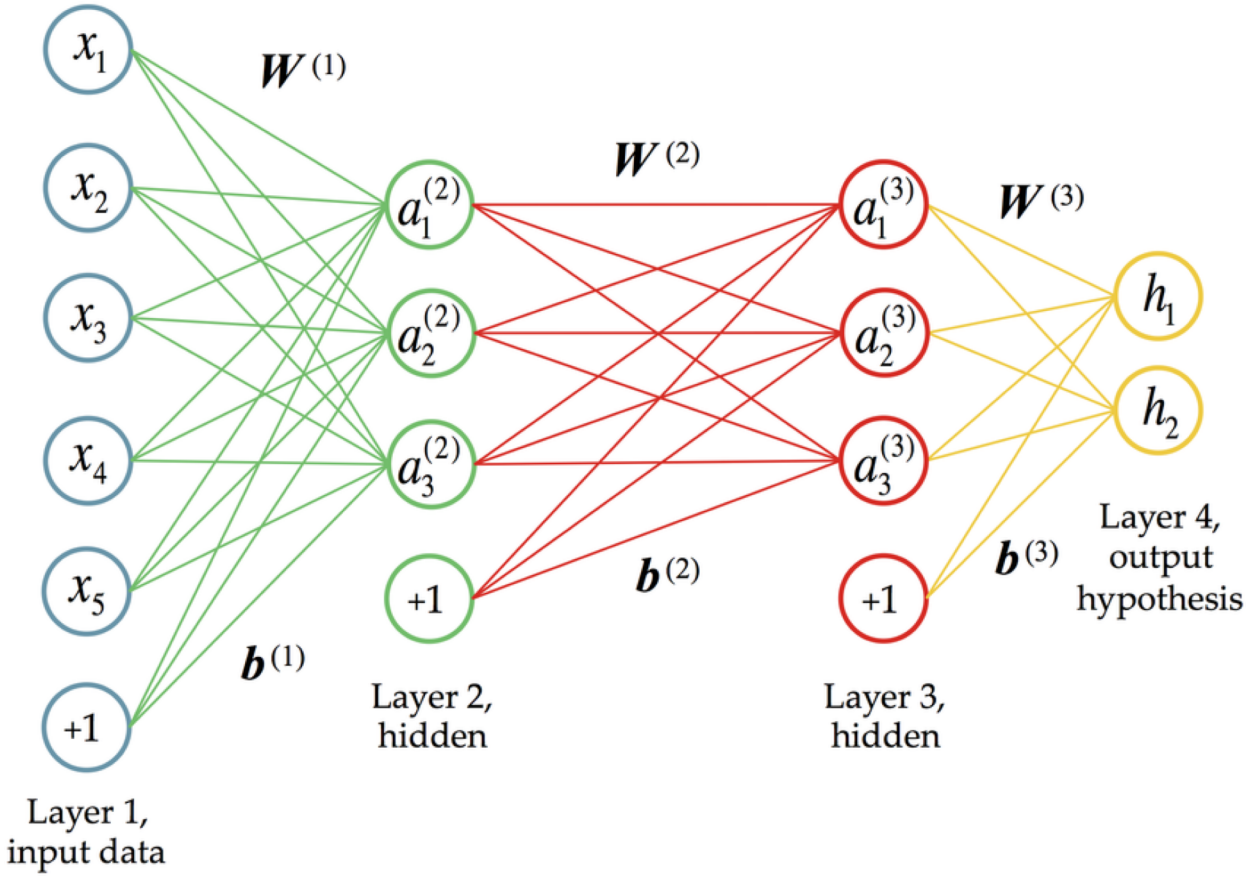
"2-layer Neural Net", or
"1-hidden-layer Neural Net"

"3-layer Neural Net", or
"2-hidden-layer Neural Net"

"Fully-connected" layers

Multilayer Perceptron

- Bias term



Multilayer Perceptron

- We can expand it to multiple neurons (hidden units) as

$$\mathbf{f} = \sigma(\mathbf{W}\mathbf{x} + \mathbf{b})$$

  where D_1 is the number of hidden units, $\mathbf{W} \in \mathbb{R}^{D_1 \times D_0}$ and $\mathbf{b} \in \mathbb{R}^{D_1}$. σ is applied element-wise.

- 2-layer neural network:

$$\mathbf{f} = \mathbf{W}_2\sigma(\mathbf{W}_1\mathbf{x} + \mathbf{b})$$

- 3-layer neural network:

$$\mathbf{f} = \mathbf{W}_3\sigma(\mathbf{W}_2\sigma(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1) + \mathbf{b}_2)$$

Table of Contents

- Review the key points
  - Multilayer Perceptron (MLP)
  - Representation Power
  - Backpropagation
- Practice
  - Implement Backpropagation
  - Multilayer Perceptron

Representation Power

- Zero-hidden layer ($\mathbf{f} = \sigma(\mathbf{W}\mathbf{x} + \mathbf{b})$): Hyperplanes

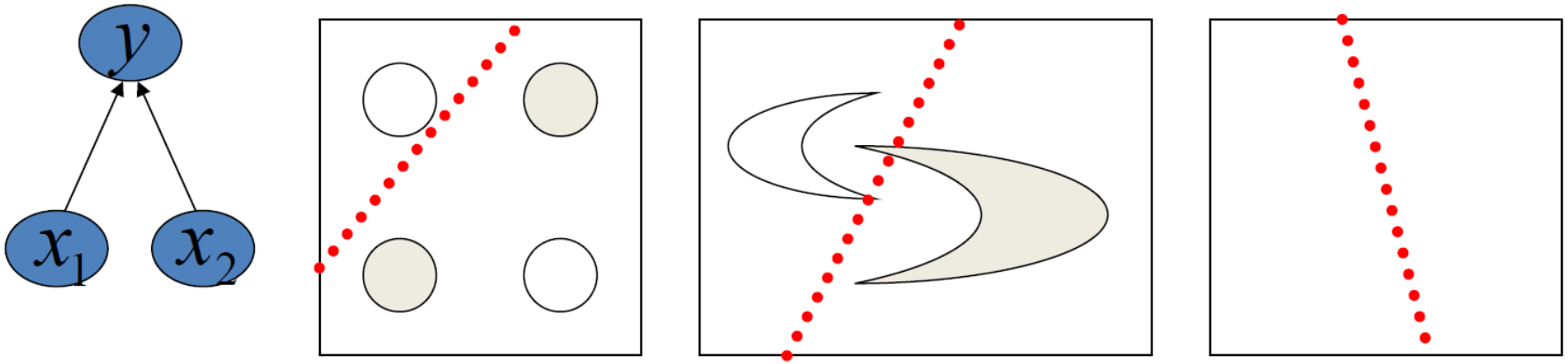


Figure: Let x_1 and x_2 be the inputs and y is the output from zero-hidden layer net. The network can represent any hyperplane separating input dimensions into two halves.

Representation Power

- One-hidden layer ($\mathbf{f} = \mathbf{W}_2\sigma(\mathbf{W}_1\mathbf{x} + \mathbf{b})$): Open or close boundary of convex region

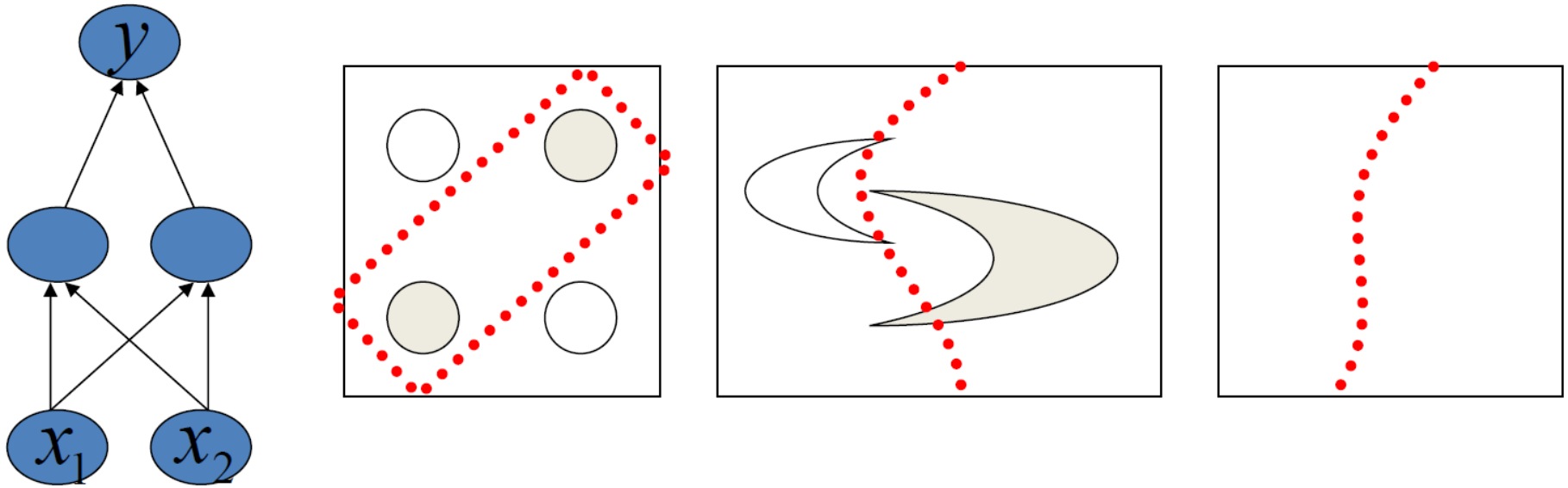
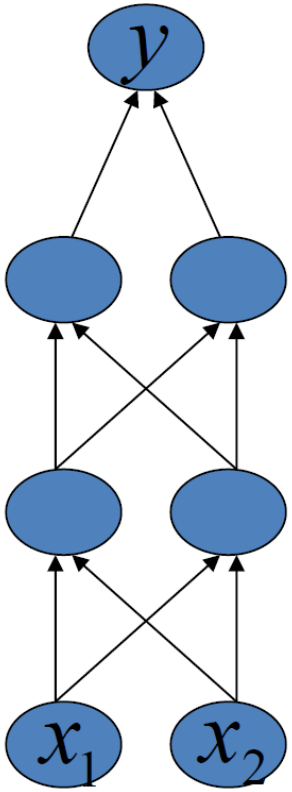


Figure: The one-hidden layer can represent any bounded convex region.

Representation Power



- 2 hidden layers
  - Combinations of convex regions

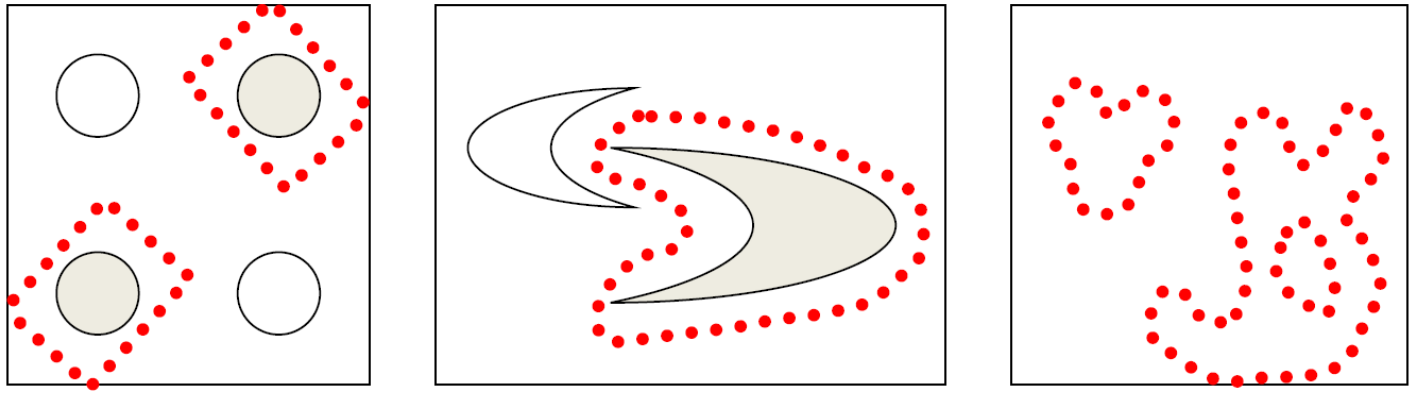


Figure: The two-hidden layers can represent any combination of convex region.

Representation Power

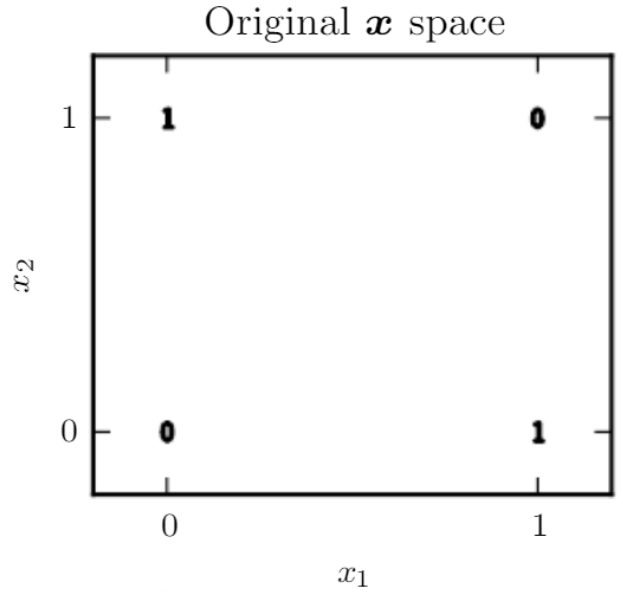
- Activation function의 역할?
  - What if there is no activation function?
  - Then, it becomes a linear function.

$$\mathbf{f} = \mathbf{W}_2(\mathbf{W}_1\mathbf{x} + \mathbf{b})$$
$$\mathbf{f} = \mathbf{W}_2\mathbf{W}_1\mathbf{x} + \mathbf{W}_2\mathbf{b}$$
$$\mathbf{f} = \mathbf{W}'\mathbf{x} + \mathbf{b}'$$

where $\mathbf{W}' \in \mathbb{R}^{D_2 \times D_0}$

Representation Power

- Activation function의 역할?
  - Ex) 다음과 같은 **X**와 **y**가 데이터셋으로 주어졌다고 가정



$$X = \begin{bmatrix} \vec{x}_1 \\ \vec{x}_2 \\ \vec{x}_3 \\ \vec{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix} \qquad y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016). http://www.deeplearningbook.org

Representation Power

- Activation function의 역할?
  - Ex) $y = XW + b$ 로 **X**와 **y** 사이의 관계를 모델링한다면, best **W**와 **b**는 아래와 같다.

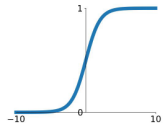
$$best\ (W^*, b^*) = \left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \end{bmatrix} \right)$$

$$\Rightarrow \quad XW^* + b^* = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \end{bmatrix} \quad \text{<--비교-->} \quad y = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

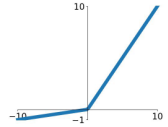
Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016). http://www.deeplearningbook.org

Representation Power

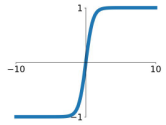
Sigmoid
$\sigma(x) = \frac{1}{1+e^{-x}}$



Leaky ReLU
$\max(0.1x, x)$

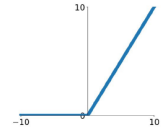


tanh
$\tanh(x)$

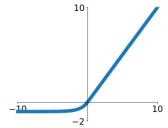


Maxout
$\max(w_1^T x + b_1, w_2^T x + b_2)$

ReLU
$\max(0, x)$



ELU
$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$



- Activation function의 역할?
  - Ex) 만약 activation function(ex. ReLU)을 활용해 x와 y 사이의 관계를 아래와 같이 모델링한다면?

$$y = w^T \max\{0, XW + c\} + b$$

Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016).
http://www.deeplearningbook.org

Representation Power

- Activation function의 역할?
  - Ex) 만약 activation function을 활용해 x와 y 사이의 관계를 아래와 같이 모델링 한다면?

$$y = w^T \max\{0, XW + c\} + b$$

$$W = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \quad c = \begin{bmatrix} 0 & -1 \\ 0 & -1 \\ 0 & -1 \\ 0 & -1 \end{bmatrix}, \quad w = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$ 넣고 계산해보기

Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016). http://www.deeplearningbook.org

Representation Power

$$X = \begin{bmatrix} \vec{x}_1 \\ \vec{x}_2 \\ \vec{x}_3 \\ \vec{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix} \qquad y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

$$W = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \quad c = \begin{bmatrix} 0 & -1 \\ 0 & -1 \\ 0 & -1 \\ 0 & -1 \end{bmatrix}, \quad w = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

- Activation function의 역할?
  - Ex) 만약 activation function을 활용해 x와 y 사이의 관계를 아래와 같이 모델링 한다면?

$$y = w^T \max\{0, XW + c\} + b$$

$$XW = \begin{bmatrix} 0 & 0 \\ 1 & 1 \\ 1 & 1 \\ 2 & 2 \end{bmatrix}$$

Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016). http://www.deeplearningbook.org

Representation Power

$$X = \begin{bmatrix} \vec{x}_1 \\ \vec{x}_2 \\ \vec{x}_3 \\ \vec{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix} \qquad y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

$$W = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \quad c = \begin{bmatrix} 0 & -1 \\ 0 & -1 \\ 0 & -1 \\ 0 & -1 \end{bmatrix}, \quad w = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

- Activation function의 역할?
  - Ex) 만약 activation function을 활용해 x와 y 사이의 관계를 아래와 같이 모델링 한다면?

$$y = w^T \max\{0, XW + c\} + b$$

$$XW = \begin{bmatrix} 0 & 0 \\ 1 & 1 \\ 1 & 1 \\ 2 & 2 \end{bmatrix} \qquad XW + c = \begin{bmatrix} 0 & -1 \\ 1 & 0 \\ 1 & 0 \\ 2 & 1 \end{bmatrix}$$

Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016). http://www.deeplearningbook.org

Representation Power

$$X = \begin{bmatrix} \vec{x}_1 \\ \vec{x}_2 \\ \vec{x}_3 \\ \vec{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix} \qquad y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

$$W = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \quad c = \begin{bmatrix} 0 & -1 \\ 0 & -1 \\ 0 & -1 \\ 0 & -1 \end{bmatrix}, \quad w = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

- Activation function의 역할?
  - Ex) 만약 activation function을 활용해 x와 y 사이의 관계를 아래와 같이 모델링 한다면?

$$y = w^T \max\{0, XW + c\} + b$$

$$XW + c = \begin{bmatrix} 0 & -1 \\ 1 & 0 \\ 1 & 0 \\ 2 & 1 \end{bmatrix} \qquad \max\{0, XW + c\} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 1 & 0 \\ 2 & 1 \end{bmatrix}$$

Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016). http://www.deeplearningbook.org

Representation Power

$$X = \begin{bmatrix} \vec{x}_1 \\ \vec{x}_2 \\ \vec{x}_3 \\ \vec{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix} \qquad y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

$$W = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \quad c = \begin{bmatrix} 0 & -1 \\ 0 & -1 \\ 0 & -1 \\ 0 & -1 \end{bmatrix}, \quad w = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

- Activation function의 역할?
  - Ex) 만약 activation function을 활용해 x와 y 사이의 관계를 아래와 같이 모델링 한다면?

$$y = w^T \max\{0, XW + c\} + b$$

$$\max\{0, XW + c\} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 1 & 0 \\ 2 & 1 \end{bmatrix} \qquad w^T \max\{0, XW + c\} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016). http://www.deeplearningbook.org

Representation Power

$$X = \begin{bmatrix} \vec{x}_1 \\ \vec{x}_2 \\ \vec{x}_3 \\ \vec{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix} \qquad y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

$$W = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \quad c = \begin{bmatrix} 0 & -1 \\ 0 & -1 \\ 0 & -1 \\ 0 & -1 \end{bmatrix}, \quad w = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

- Activation function의 역할?
  - Ex) 만약 activation function을 활용해 x와 y 사이의 관계를 아래와 같이 모델링 한다면?

$$y = w^T \max\{0, XW + c\} + b$$

$$w^T \max\{0, XW + c\} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \quad \text{<--비교-->} \quad y = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016). http://www.deeplearningbook.org

Table of Contents

- Review the key points
  - Multilayer Perceptron (MLP)
  - Representation Power
  - Backpropagation
- Practice
  - Implement Backpropagation
  - Multilayer Perceptron

Backpropagation

- All we need to train the model is the partial derivatives $\frac{\partial \mathcal{L}}{\partial \mathbf{W}_\ell}$ and $\frac{\partial \mathcal{L}}{\partial \mathbf{b}_l}$ for all $\ell = 1, \ldots, N$.
- You may wish to compute the partial derivative analytically (by writing down the equations on a paper).
- This doesn't seem good idea since
  - You need lots of matrix calculus (and paper)
  - What if you want to change the loss? you need to start from beginning
  - Not feasible for complex models.
- *Back-propagation* is the solution.

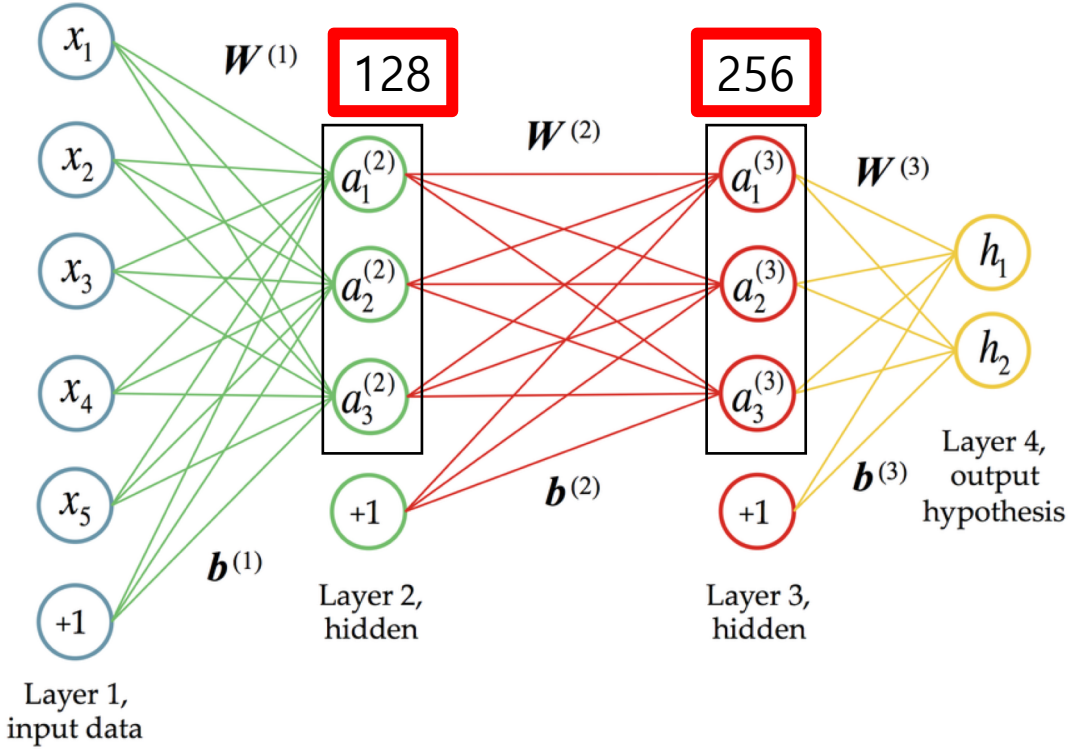
Table of Contents

- Review the key points
  - Multilayer Perceptron (MLP)
  - Representation Power
  - Backpropagation

- Practice
  - Implement Backpropagation
  - Multilayer Perceptron

Multilayer Perceptron

- MNIST 데이터를 classify하는 multilayer neural network 구현하기
  - Network design : 2 hidden layer (1st hidden layer : 128, 2nd hidden layer : 256)

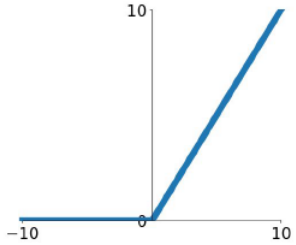


Implement Backpropagation

- single layer neural network에 대한 backpropagation 구현하기
  - Activation function :

ReLU

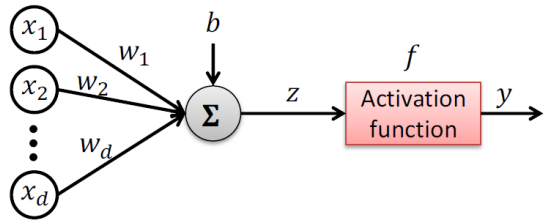
$$\max(0, x)$$



  - Loss function :

$$\frac{1}{N}\sum_{i=1}^{N}\{y_i - \text{relu}(wx_i + b)\}^2$$

$$y = f(z) = f\left(\sum_i w_i x_i + b\right) = f\left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b\right)$$

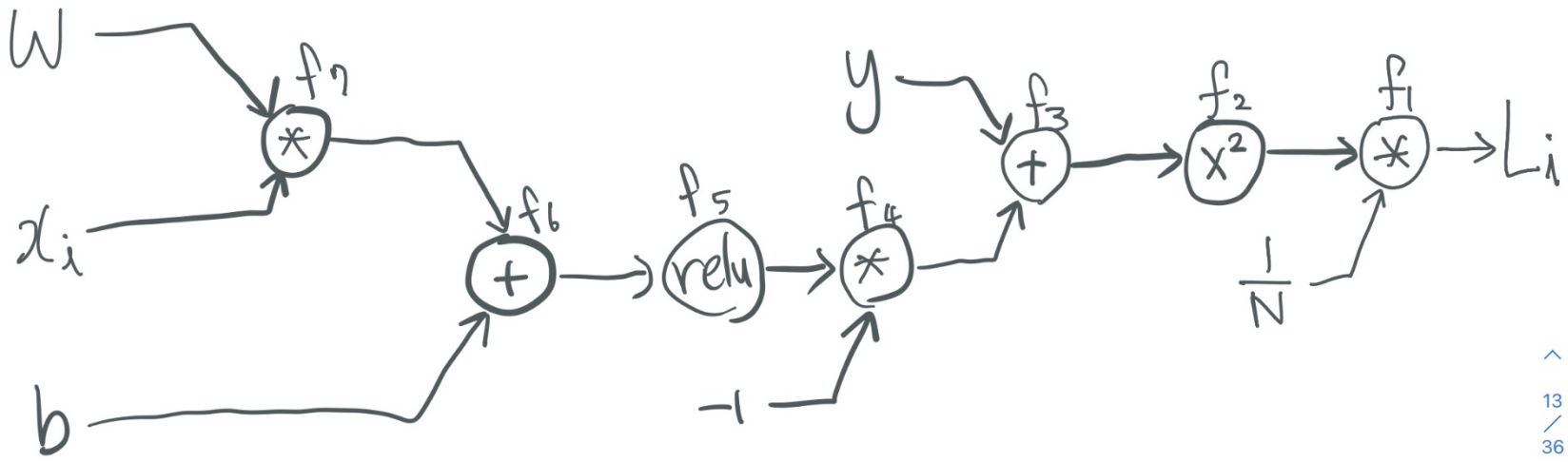


Implement Backpropagation

- single layer neural network에 대한 backpropagation 구현하기

$$loss = \frac{1}{N}\sum_{i=1}^{N}\{y_i - relu(Wx_i + b)\}^2$$

$$= \sum_{i=1}^{N} L_i \quad \left(L_i = \frac{1}{N}\{y_i - relu(Wx_i + b)\}^2\right)$$

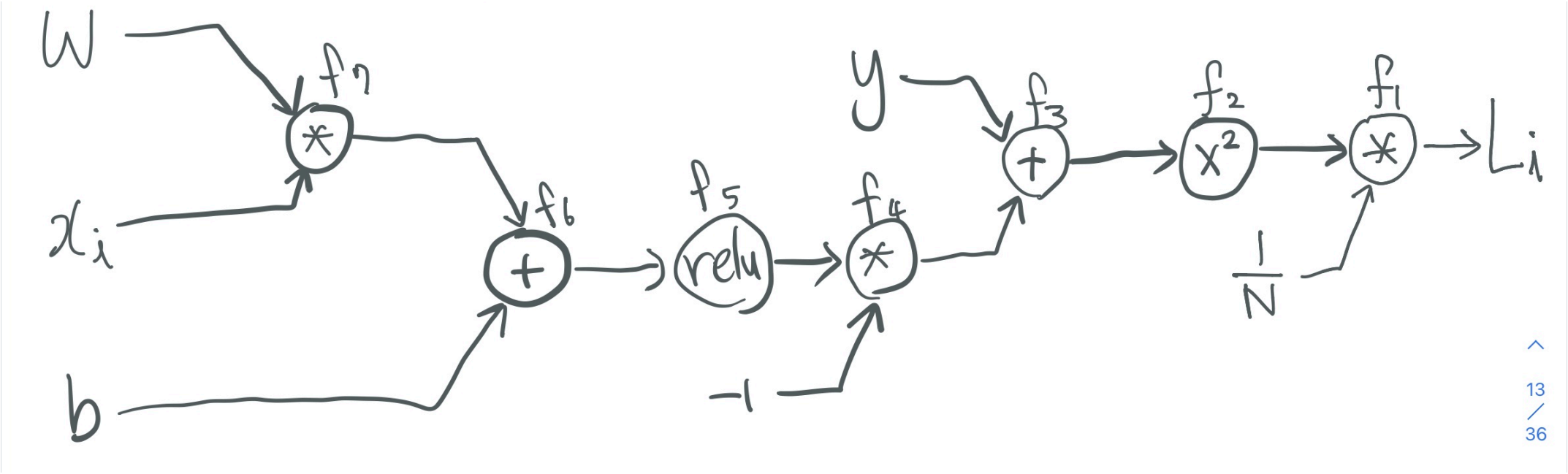


13 / 36

Implement Backpropagation

$$\frac{\partial}{\partial x}\max(0, x) = \begin{cases} 1 & : \text{if } x > 0 \\ 0 & : \text{otherwise} \end{cases}$$

- single layer neural network에 대한 backpropagation 구현하기



Thank You :)

slee2020@postech.ac.kr