Lecture 17: Neural Network and Backpropagation I

[SCS4049-02] Machine Learning and Data Science

Seongsik Park (s.park@dgu.edu)

AI Department, Dongguk University

Tentative schedule

Machine learning, 일반적인
Neural network, Deep neural network.

week	topic	date (수 / 월)
1	Machine Learning Introduction & Basic Mathematics	09.02 / 09.07
2	Python Practice I & Regression	09.09 / 09.14
3	AI Department Seminar I & Clustering I	09.16 / 09.21
4	Clustering II & Classification I	09.23 / 09.28
5	Classification II	(추석) / 10.05
6	Python Practice II & Support Vector Machine I	10.07 / 10.12
7	Support Vector Machine II & Decision Tree and Ensemble Learning	10.14 / 10.19
8	Mid-Term Practice & **Mid-Term Exam**	10.21 / **10.26**
9	휴강 & Dimensional Reduction I	10.28 / 11.02
10	Dimensional Reduction II & Neural networks and Back Propagation I	11.04 / 11.09
11	Neural networks and Back Propagation II & Convolutional Neural Network	11.11 / 11.16
12	Model Optimization	11.18 / 11.23
13	Recurrent Neural network	11.25 / 11.30
14	Autoencoders	12.02 / 12.07
15	**Final exam**	(휴강) / **12.14**

① pdf로 제출하는 이유.

pdf = 종이로 하는 프린트를 대체하는 파일포맷. → SW종속성 ↓, 기기마다 같은모양, 폰트O

doc, hwp, ppt, ... = 문서 편집이 목적인애들 → SW종속성, 기기마다 다른게보일수있음
폰트가 포함X

★
② 어렵지만, 얻는것도 많다. 강의초반 → 어려웠 / 후반 → 나아지더라.
한번에 이해하기는 어려웠는데, 다시보면 대부분이해가 되더라.

③ 피드백이 좋다, 질문에 상세하게 답변한다, 편안하다. + 필기가 도움이 된다.

개선.
④ 수업분량·속도조절이 잘안된다. → 개선.
⑤ 실습이 크게 도움이 되지 않는다.

+ 예제가너무많다.

피드백 '비전공자' → (연계전공) → '전공자'

Neural Networks and Perceptron

Biological neuron

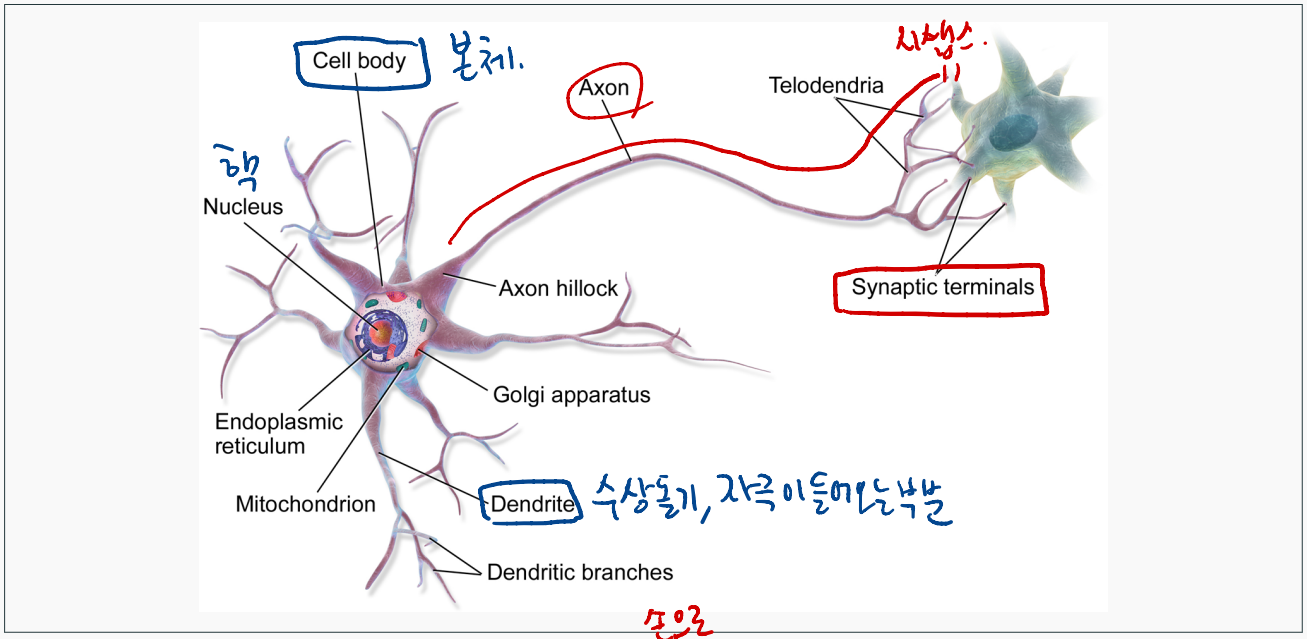


Figure 10-1. Biological neuron[3]

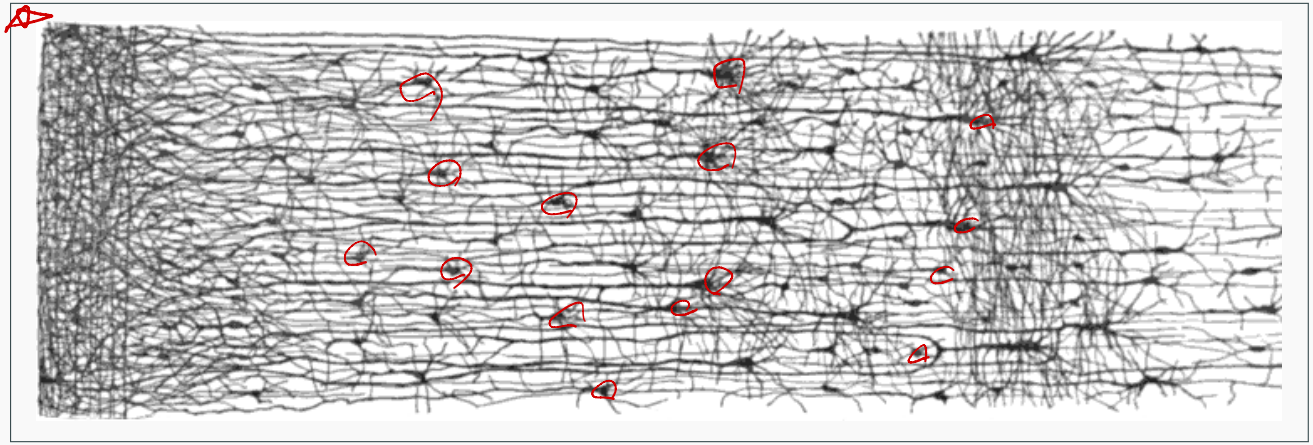
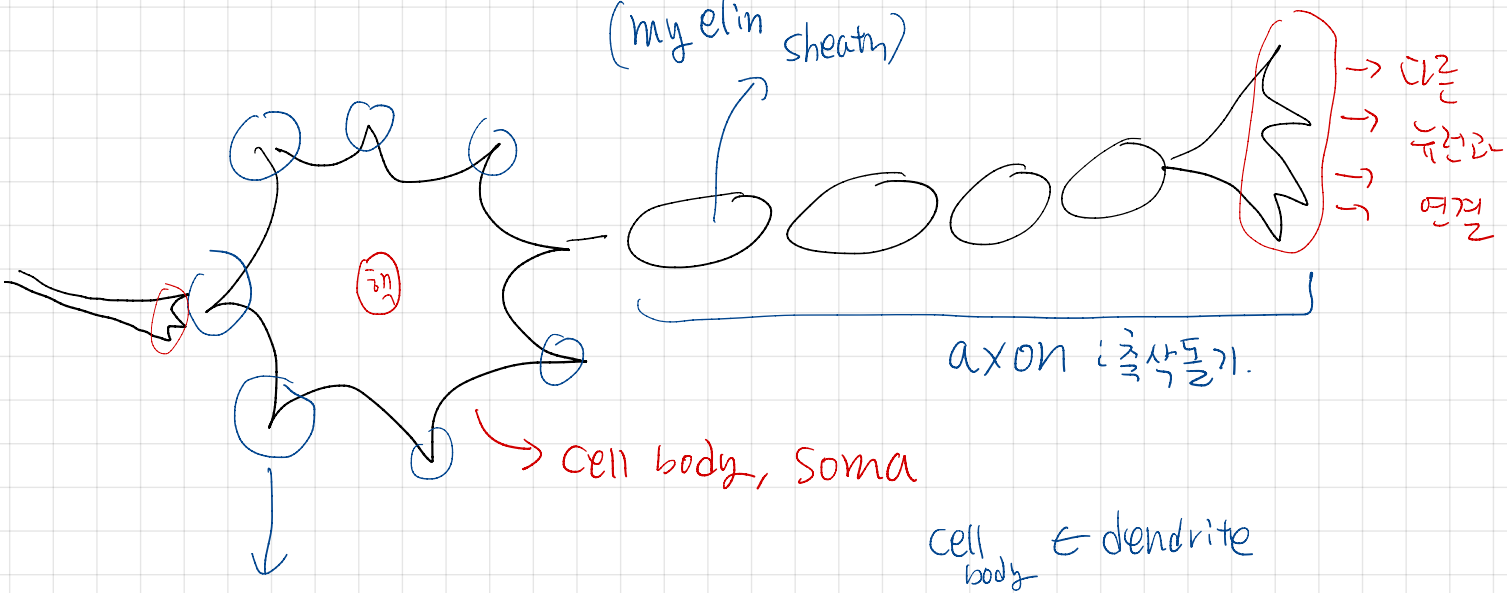


Figure 10-2. Multiple layers in a biological neural network (human cortex)[5]

neuron 신경세포.

⇒ 신경전달속도를 빠르게 만들어주는.

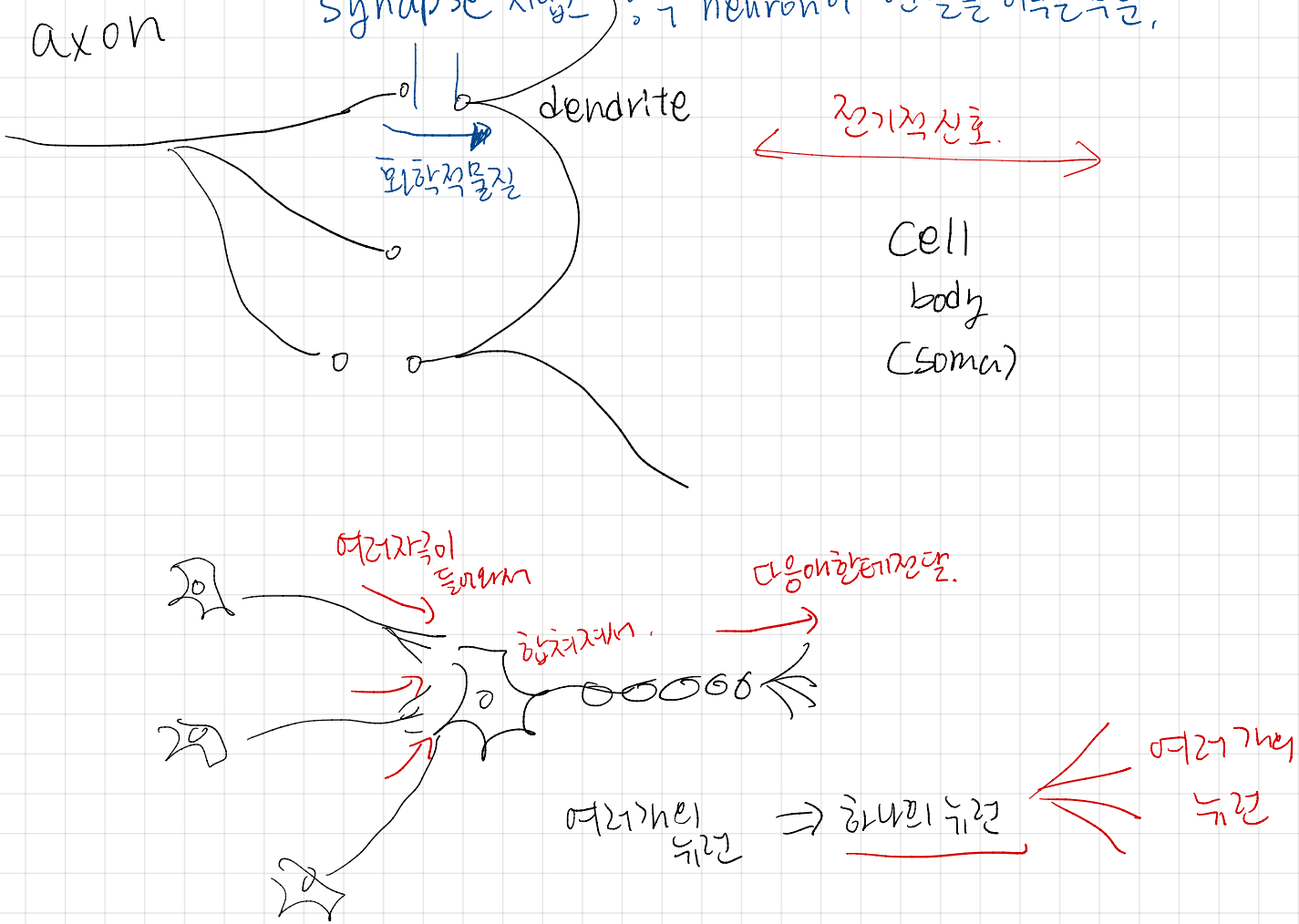
(myelin sheath)



cell body ← dendrite

axon

dendrite ; 자극을 받아들이는부분 수상돌기


axon
Synapse 시냅스 : 두 neuron이 연결을 이루는부분.
dendrite
화학적물질
전기적신호.
Cell
body
(soma)
여러자극이
들어옴
다음에 할지말지결정.
한뉴런에서.
여러개의
뉴런
⇒ 하나의 뉴런
여러개의
뉴런

Computing with the brain

An engineering perspective

- Compact
- Energy efficient (20 watts)
- 85 billion Glial cells (power, cooling, support)
- 86 billion Neurons (soma + wires)
- 69 billion Cerebellum neurons (soma + wires)
- 103~104 Connections (synapses) per neuron
- Volume = mostly wires

General computing machine?

- Slow for mathematical logic, arithmetic, etc
- Very fast for vision, speech, language, social interactions, etc
- Evolution: vision $\Longrightarrow$ language $\Longrightarrow$ logic

Artificial neural network 인공신경망, ANN

Bio-inspired, 생체모방 → Neural network.

Birds inspired us to fly, burdock plants inspired velcro, and countless more inventions were inspired by nature. It seems only logical, then, to look at the brain's architecture for inspiration on how to build an intelligent machine. This is the key idea that sparked artificial neural networks (ANNs). However, although planes were inspired by birds, they don't have to flap their wings. Similarly, ANNs have gradually become quite different from their biological cousins. Some researchers even argue that we should drop the biological analogy altogether (e.g., by saying "units" rather than "neurons"), lest we restrict our creativity to biologically plausible systems.

~40's 50's 제안.
기술적 장벽
학습
컴퓨터
90, 00, 10, 20's 대폭발.

→ NN →
학습? ∈ 역전파, Backpropagation

Artificial neural network

ANNs are at the very core of Deep Learning. They are versatile, powerful, and scalable, making them ideal to tackle large and highly complex Machine Learning tasks, such as classifying billions of images (e.g., Google Images), powering speech recognition services (e.g., Apple's Siri), recommending the best videos to watch to hundreds of millions of users every day (e.g., YouTube), or learning to beat the world champion at the game of Go by playing millions of games against itself (DeepMind's Alpha-Zero).

Threshold logic unit

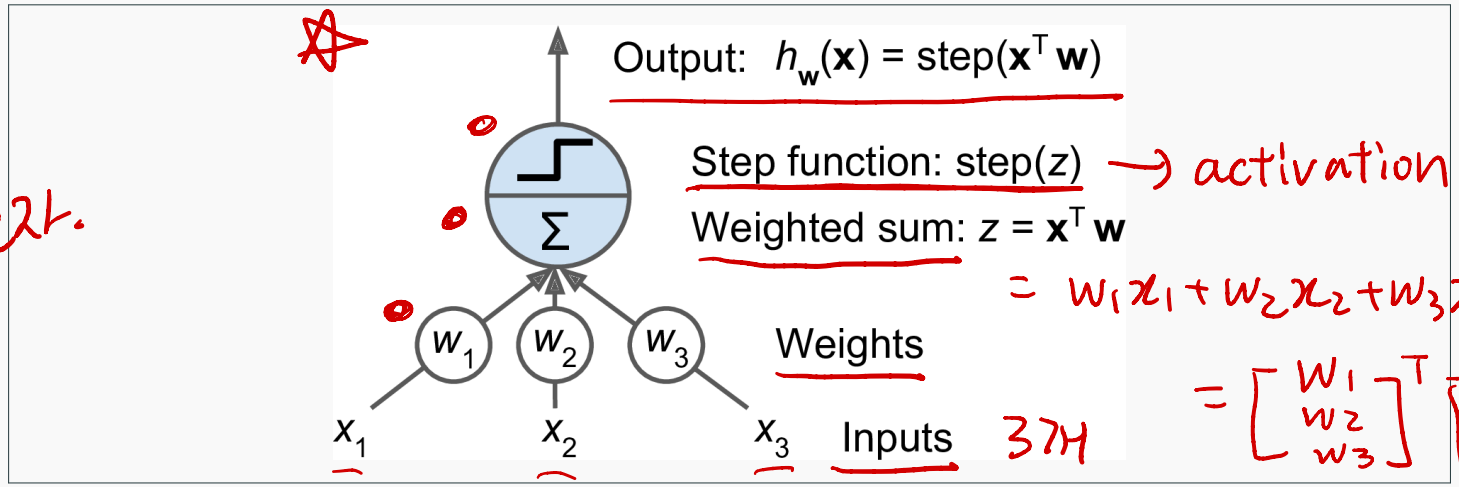


Figure 10-4. Threshold logic unit

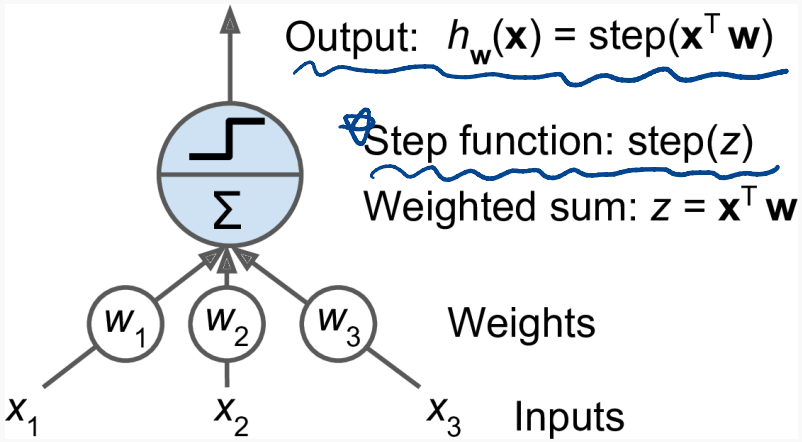
Perceptron $\Longrightarrow$ binary classifier

$$z = w_1 x_1 + w_2 x_2 + \cdots + x_m x_m = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{bmatrix}^T \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} = \mathbf{w}^T \mathbf{x} \quad (1)$$

Threshold logic unit (TLU) or linear threshold unit (LTU)

$$h_{\mathbf{w}}(\mathbf{x}) = \text{step}(z) = \text{step}(\mathbf{w}^T \mathbf{x}) \quad (2)$$

Threshold logic unit

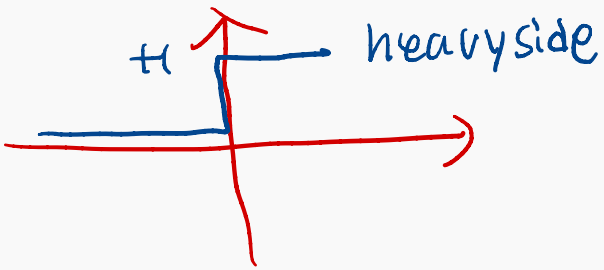


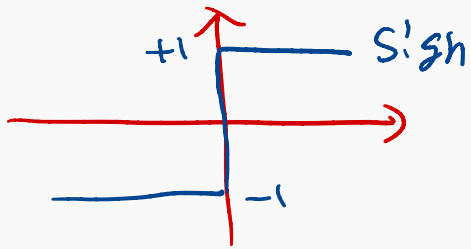
neuron이 하는 역할을 모사해서 만든.

Figure 10-4. Threshold logic unit

Common step functions used in Perceptrons

$$\text{heavyside}(z) = \begin{cases} 0 & \text{if } z < 0 \\ 1 & \text{otherwise} \end{cases} \qquad \text{sign}(z) = \begin{cases} -1 & \text{if } z < 0 \\ 0 & \text{if } z = 0 \\ +1 & \text{otherwise} \end{cases} \tag{3}$$





Perceptron algorithm

Computing the outputs of a fully connected layer

$$h_{\mathbf{W},\mathbf{b}} = \phi(\mathbf{XW} + \mathbf{b}) \qquad (4)$$

- Perceptron learning rule (weight update)

$$w_{i,j} \leftarrow w_{i,j} + \eta(y_j - \hat{y}_j)x_i \qquad (5)$$

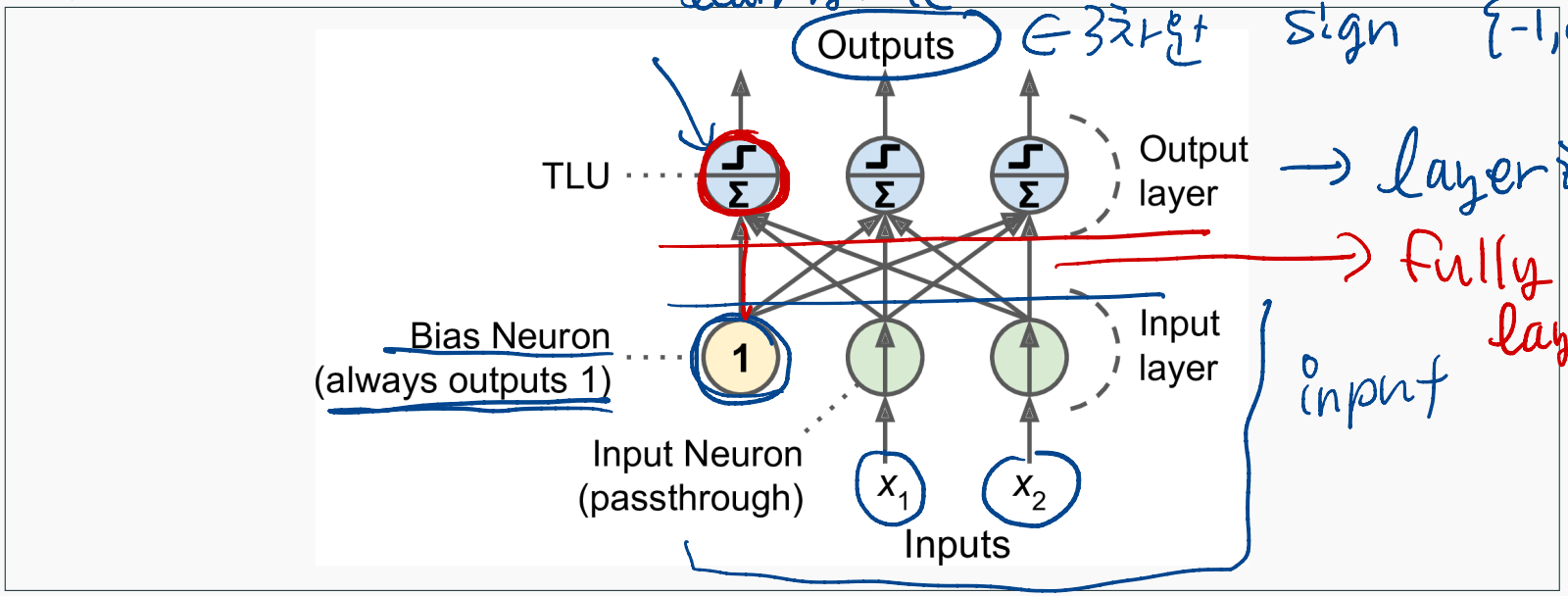
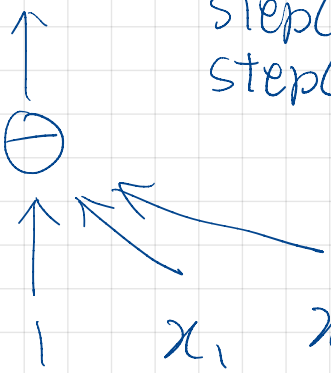


Figure 10-5. Perceptron diagram

output 1 $= \text{step}(_1w_1^T x + b_1) = \text{step}\left(\begin{bmatrix} w_{1,1} \\ w_{1,2} \end{bmatrix}^T \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + b_1\right)$

$\text{step}(_1w_2^T x + b_2)$

$\text{step}(_1w_3^T x + b_3)$

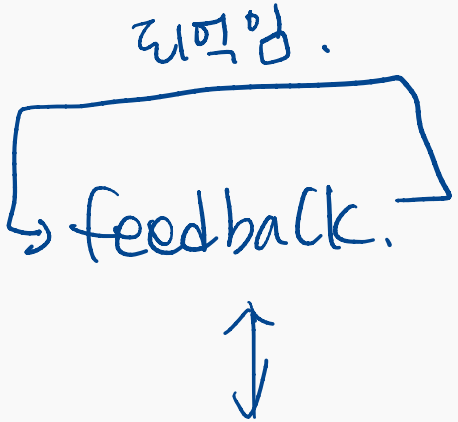


$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

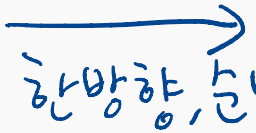
$$\text{output} = \begin{bmatrix} \text{output 1} \\ \text{output 2} \\ \text{output 3} \end{bmatrix} = \begin{bmatrix} \text{step}(_1w_1^T x + b_1) \\ \text{step}(_1w_2^T x + b_2) \\ \text{step}(_1w_3^T x + b_3) \end{bmatrix}$$

$$_1W = \begin{bmatrix} | & | & | \\ {}_1w_1 & {}_1w_2 & {}_1w_3 \\ | & | & | \end{bmatrix}^T \Rightarrow \text{step}\left(\underset{3\times2}{_1W}\ \underset{2\times1}{x} + \underset{3\times1}{b}\right)$$

$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$



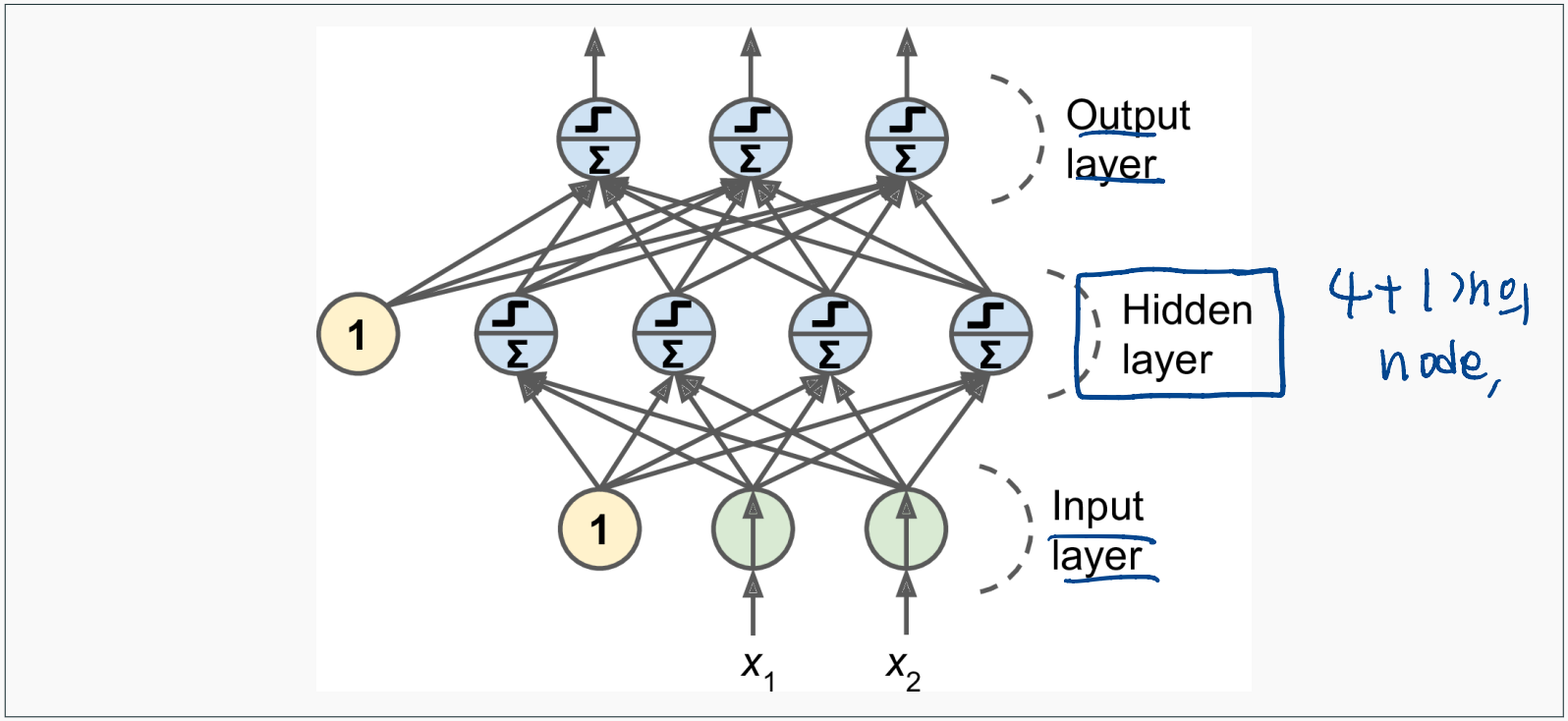
Deep Feedforward Networks



Multi-layer perceptron algorithm

(MLP)

모델을 복잡하게 → hidden layer를 여러개
→ hidden layer안의 perceptron의 수를 ↑



4+1개의 node, 숨어있는 neuron

Figure 10-7. Multi-Layer Perceptron

Activation functions and their derivatives

activation → step (heaviside, sign → discrete한 값)

↳ nonlinear한 activation function,

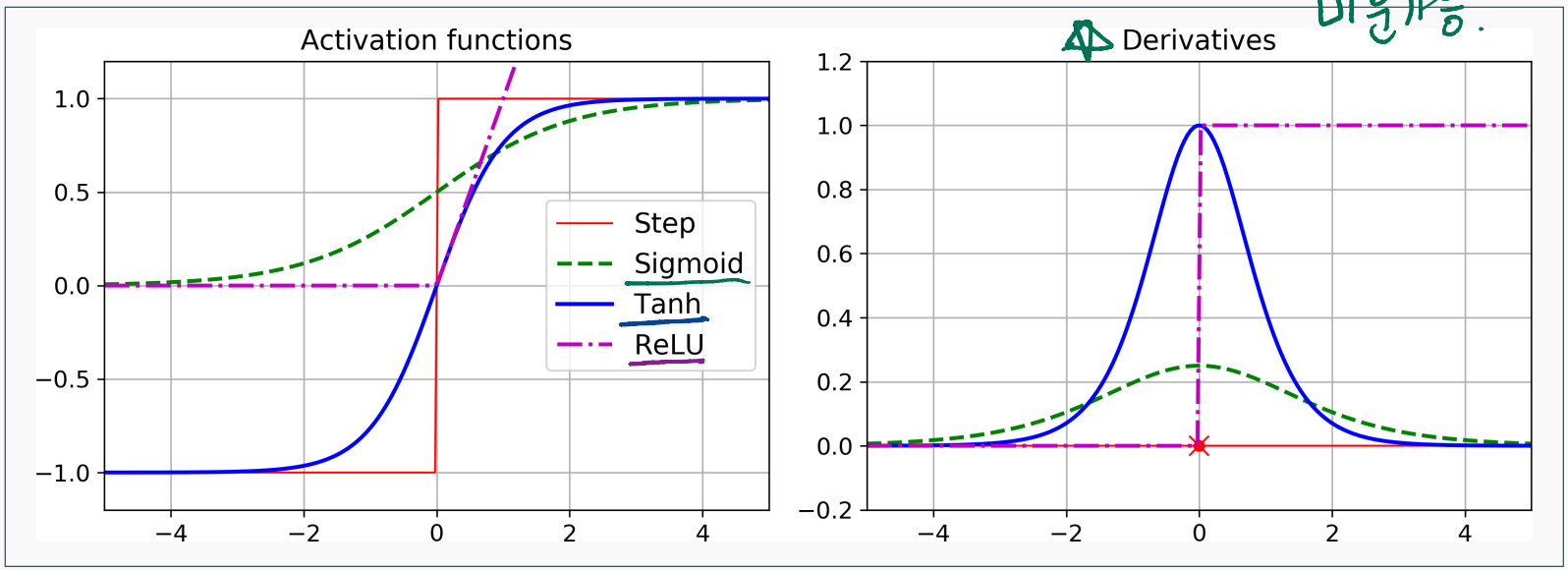


Figure 10-8. Activation functions and their derivatives

A modern MLP (including ReLU and softmax) for classification

[1, 2, 3] → softmax → [1/6, 2/6, 3/6] Σ=1로 만들었다.
⇒ 각각의 확률값.

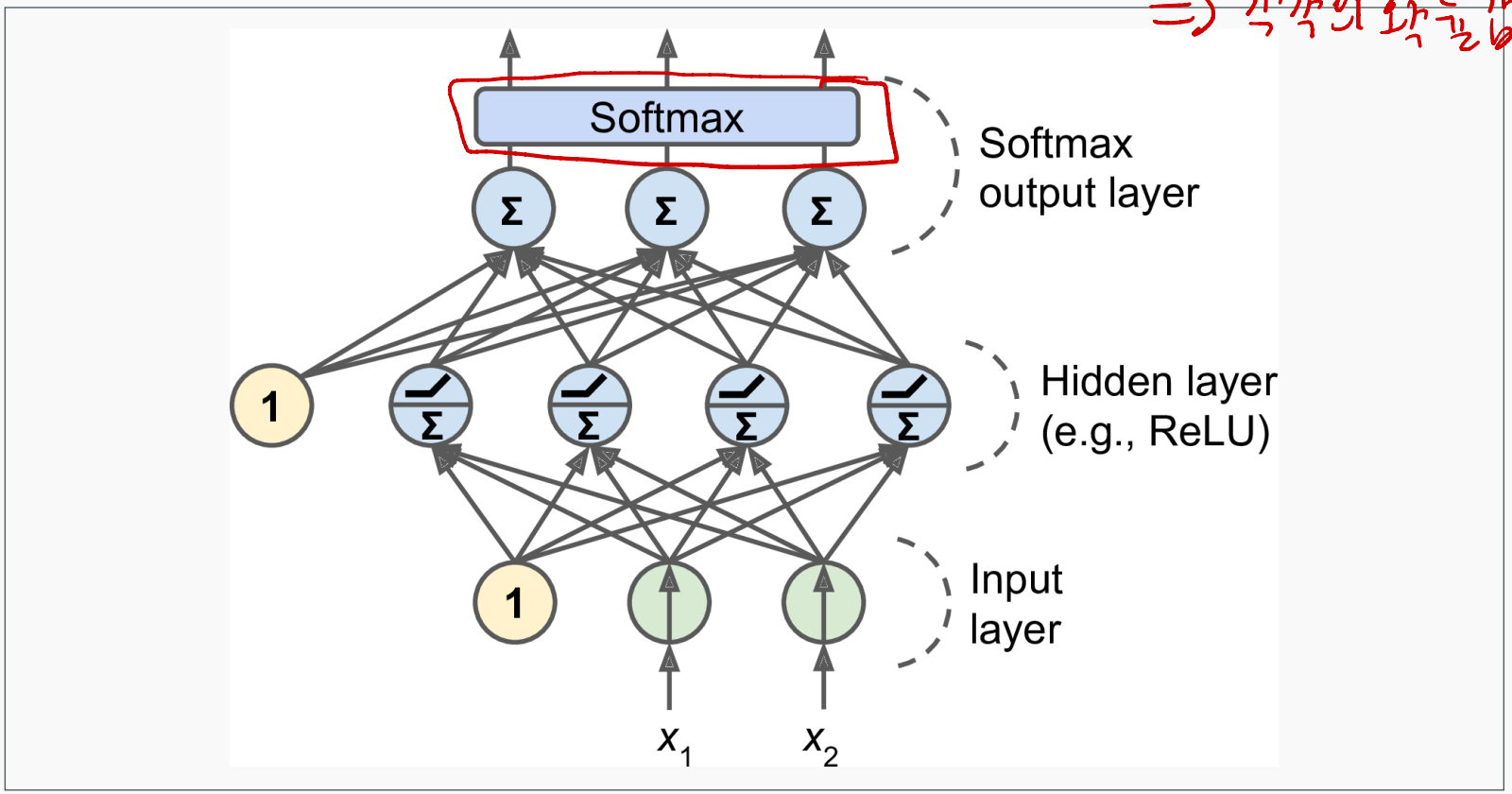


Figure 10-9. A modern MLP (including ReLU and softmax) for classification

class 5개인 경우.

output Softmax
#1일 probability 확률
#2
#3
#4
#5 ..

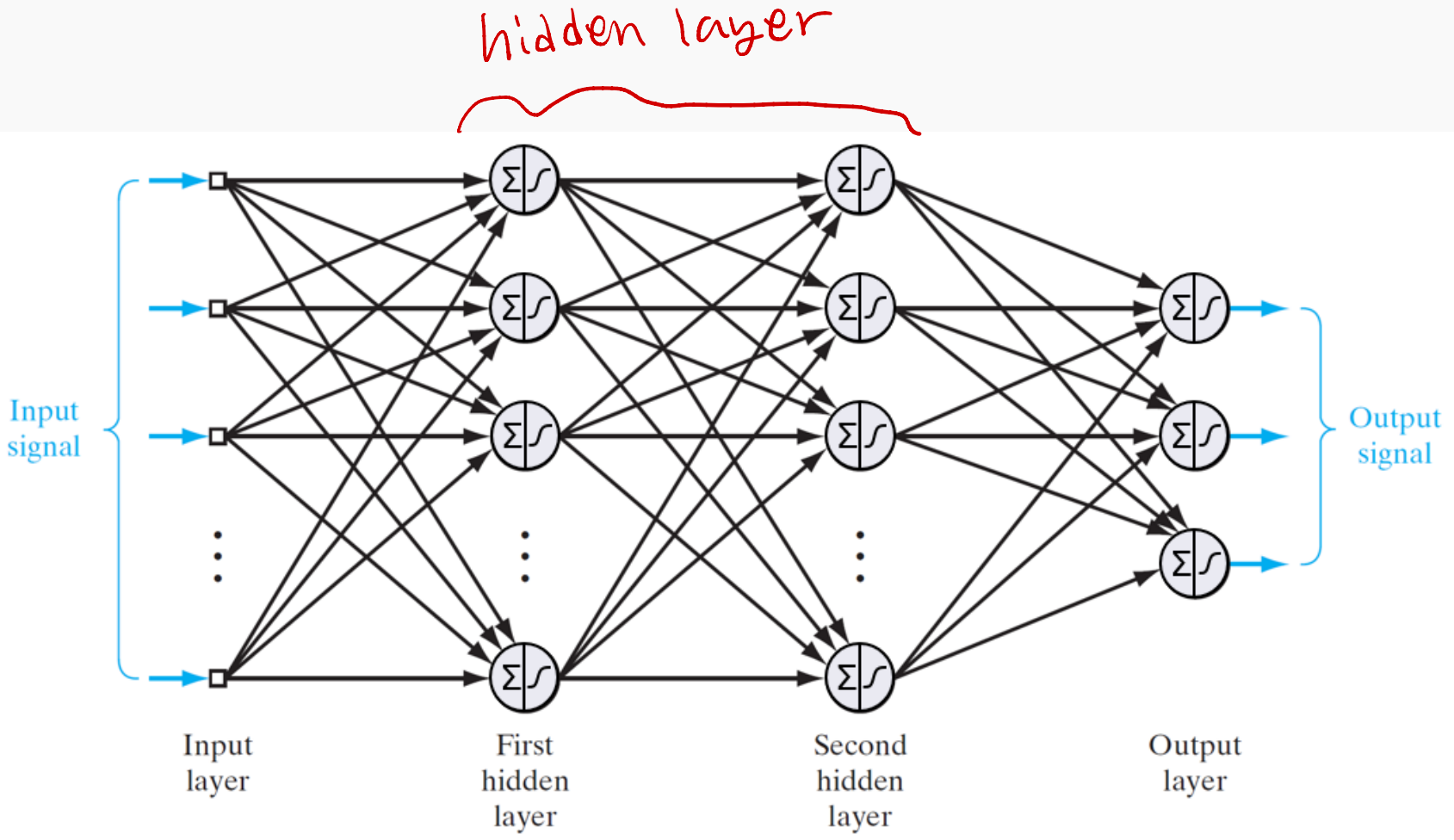
→ 가장 높은 확률값을 선택. ⇒ classification

Deep feedforward networks

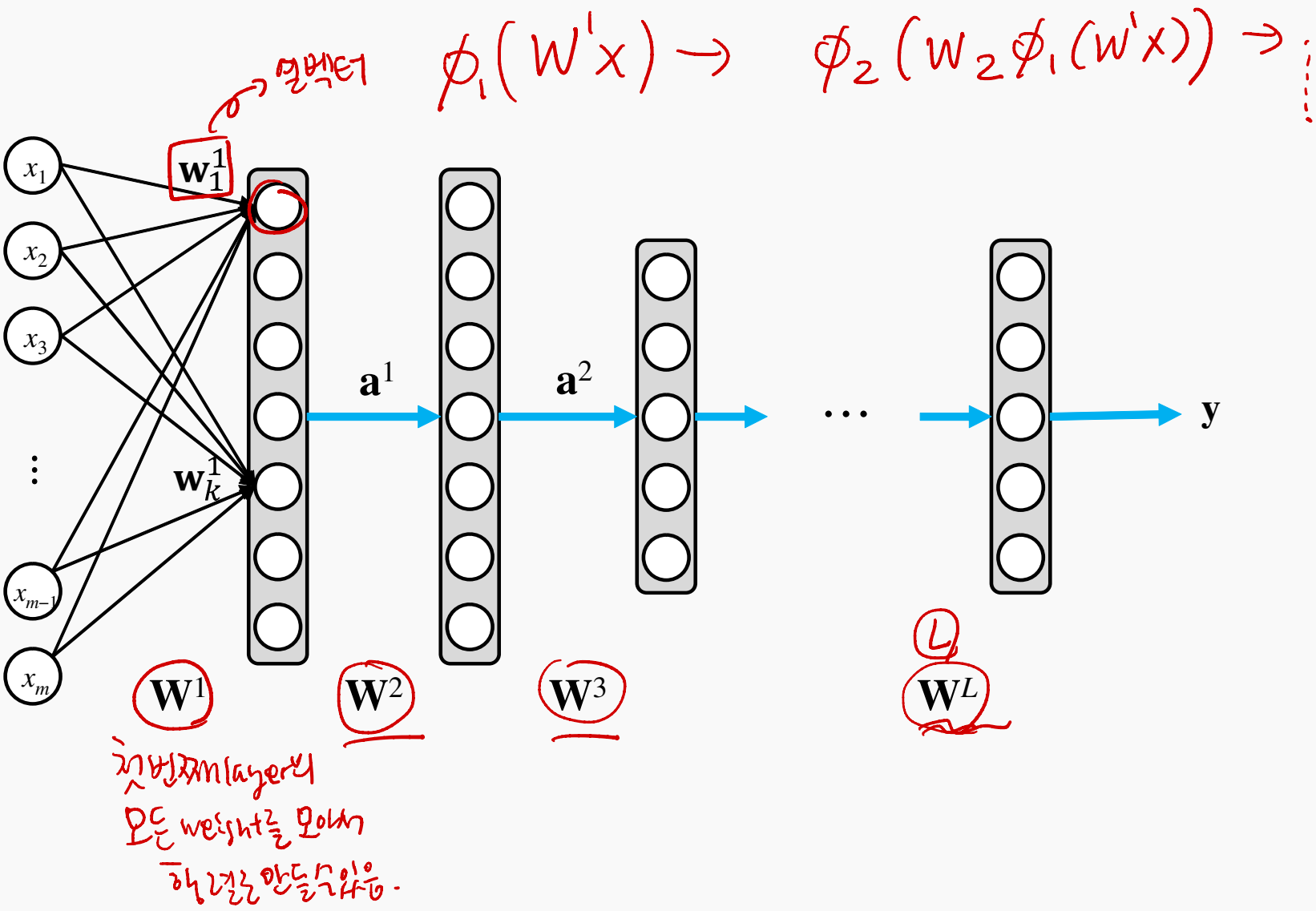
Deep Feedforward Networks, Multi-Layer Perceptron, or Feedforward Neural Networks

- Fully Connected Multi-layer
- Feedforward $\Longrightarrow$ Feedback이 없음
  (Feedback 이 있는 신경망: Recurrent Neural Networks)
- Nonlinear Activation Functions
  - Rumelhart (1986) 의 Backpropagation (역전파) 알고리즘 이후 집중적 연구
- Reverse-mode Autodiff 를 사용한 Gradient-based learning 방법

Deep feedforward networks



Goal of deep feedforward networks



Goal of deep feedforward networks

Let $\mathbf{W}^l = \begin{bmatrix} \mathbf{w}_1^l & \mathbf{w}_2^l & \cdots & \mathbf{w}_{m_l}^l \end{bmatrix}$ and $\boldsymbol{\varphi}^l$ be the activation function for layer l, then

$$\mathbf{y}^1 = \boldsymbol{\varphi}^1(\mathbf{W}^1\mathbf{x}) \tag{6}$$

$$\mathbf{y}^2 = \boldsymbol{\varphi}^2(\mathbf{W}^2\mathbf{y}^1) \tag{7}$$

$$= \boldsymbol{\varphi}^2(\mathbf{W}^2\boldsymbol{\varphi}^1(\mathbf{W}^1\mathbf{x})) \tag{8}$$

$$\mathbf{y}^3 = \boldsymbol{\varphi}^3(\mathbf{W}^3\mathbf{y}^2) \tag{9}$$

$$= \boldsymbol{\varphi}^3(\mathbf{W}^3\boldsymbol{\varphi}^2(\mathbf{W}^2\mathbf{y}^1)) \tag{10}$$

$$= \boldsymbol{\varphi}^3(\mathbf{W}^3\boldsymbol{\varphi}^2(\mathbf{W}^2\boldsymbol{\varphi}^1(\mathbf{W}^1\mathbf{x}))) \tag{11}$$

$$\cdots$$

$$\mathbf{y} = \boldsymbol{\varphi}^L(\mathbf{W}^L\boldsymbol{\varphi}^{L-1}(\mathbf{W}^{L-1}\boldsymbol{\varphi}^{L-2}(\cdots\boldsymbol{\varphi}^2(\mathbf{W}^2\boldsymbol{\varphi}^1(\mathbf{W}^1\mathbf{x}))\cdots))) \tag{12}$$

Very much complex nonlinear function

$\Longrightarrow$ require universal function approximator

Nonlinear activation functions

Logistic sigmoid function

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{13}$$

$$\sigma'(z) = \sigma(z)(1 - \sigma(z)) \tag{14}$$

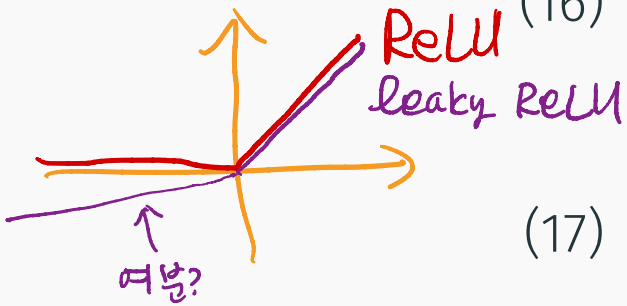
Hyperbolic tangent function tanh

$$\varphi(z) = \tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{15}$$

$$\varphi'(z) = 2\varphi(2z) - 1 \tag{16}$$

Rectified linear unit function ReLU

$$\mathrm{ReLU}(z) = \max(0, z) \tag{17}$$



$$\mathrm{ReLU}'(z) = \begin{cases} 0 & \text{if } z \leq 0 \\ 1 & \text{otherwise} \end{cases} \tag{18}$$

Nonlinear activation functions

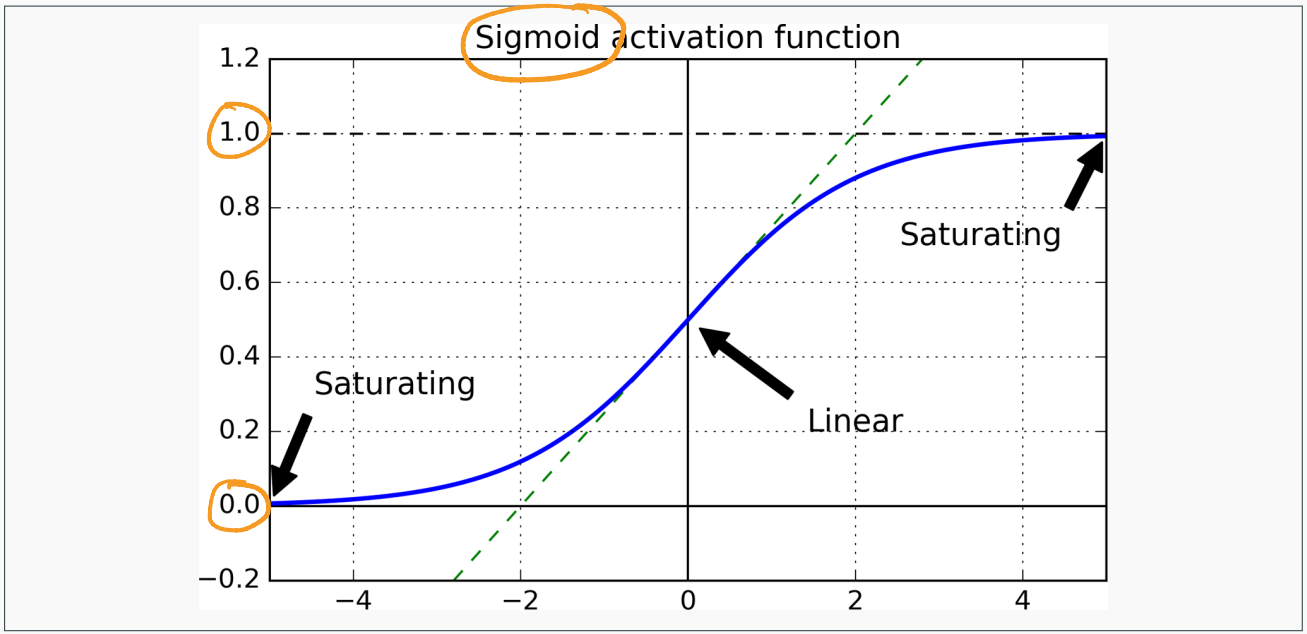


Figure 11-1. Logistic activation function saturation

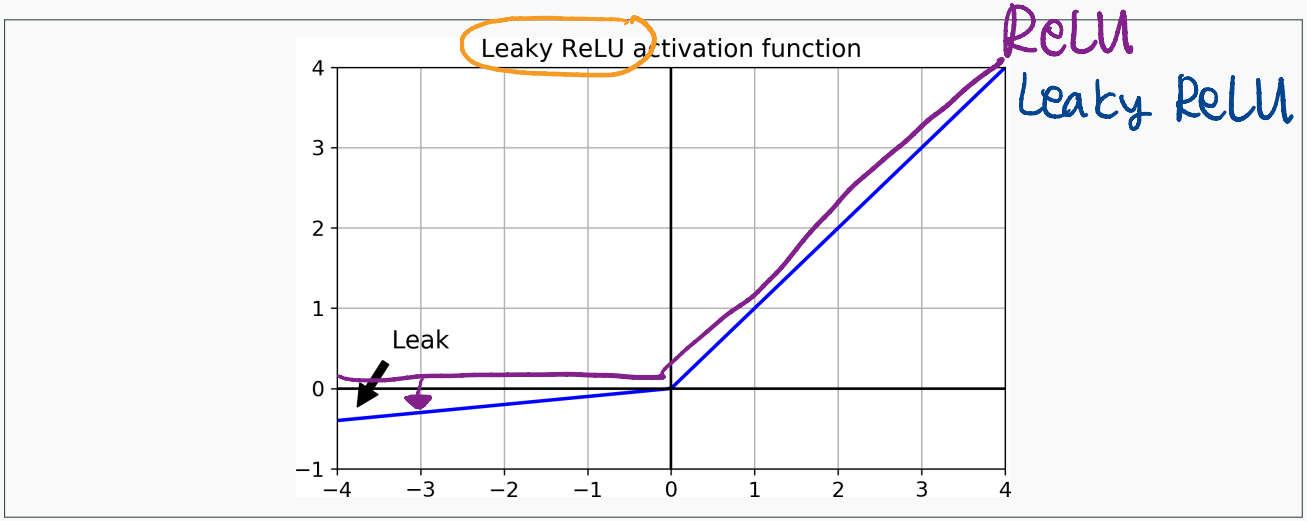


Figure 11-2. Leaky ReLU

Appendix

Reference and further reading

- “Chap 10 | Introduction to Artificial Neural Network with Keras” and “ Chap 11 | Training Deep Neural Networks” of A. Geron, Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow