기계학습
(Machine Learning)
이론 및 실습

6. Artificial Neural Network

Biological Neurons and Neural network

What is Neural network in Computer Science domain?

- **A computer modeling approach** to computation that is **loosely based upon the architecture of the brain**.

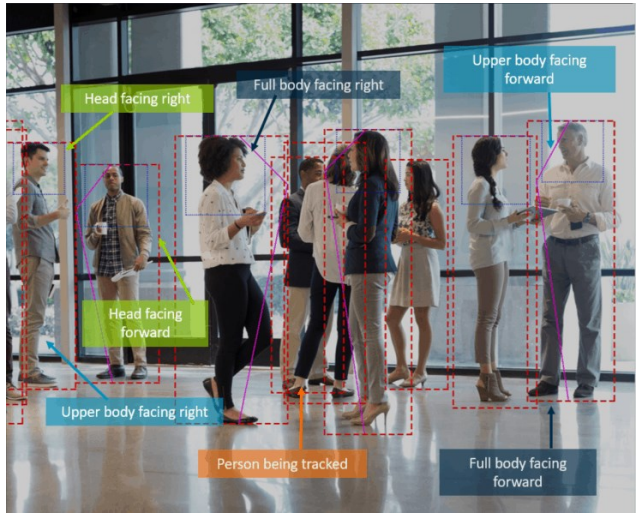
- **Many different models, but all include:**
  - Multiple, individual “nodes” or “units” that operate at the same time (in parallel)
  - **A network that connects the nodes together**
  - **Information is stored** in a distributed fashion **among the links that connect the nodes**
  - **Learning** can occur with gradual **changes in connection strength**

Applications

Autonomous vehicle

Image source: https://www.sciencetimes.co.kr/news/

Image Processing.



https://www.analyticsinsight.net/deep-learning-to-analyse-human-activities-recorded-on-videos/

Game.

Image Generation.

Deep fake

Image source : twitter.com/bornmiserable

Comparison of Brains and Traditional Computers

- 200 billion neurons (G), 32 trillion (T) synapses
- Element size: 10^{-6} m
- Energy use: 25W
- Processing speed: 100 Hz
- Parallel, Distributed
- Fault Tolerant
- Learns: Yes
- Intelligent/Conscious: Usually

- 16~256 billion bytes (GB) RAM but trillions of bytes (TB) on disk
- Element size: 10^{-9} m
- Energy watt: 30-90W (CPU)
- Processing speed: 10^{9} Hz (GHz)
- Serial, Centralized
- Generally not Fault Tolerant
- Learns: Some
- Intelligent/Conscious: Generally No

This page is based on Lecture note of Prof. Kenrick Mock.
http://www.math.uaa.alaska.edu/~afkjm

Applications

Autonomous vehicle

Image source: https://www.sciencetimes.co.kr/news/

Image Processing.

https://www.analyticsinsight.net/deep-learning-to-analyse-human-activities-recorded-on-videos/

Game.

Image Generation.

Deep fake

Image source : twitter.com/bornmiserable

Why neural network?

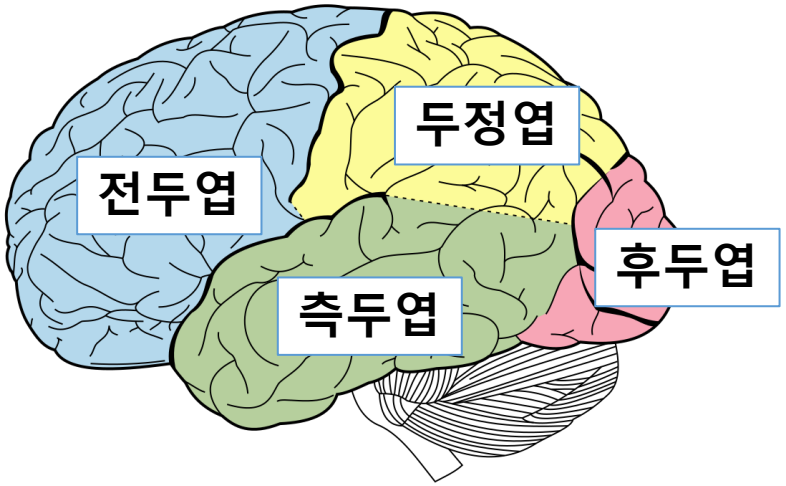
Artificial Intelligence :

- **Elaine Rich**
- Artificial Intelligence is the study of how to make computers do things at which, at the moment, people are better. (1983, Elaine Rich)
- 인공지능이란 컴퓨터에게 "현 시점(연구가 이루어지는 시점)에서" 컴퓨터보다 인간이 더 잘한다고 생각되는 일을 시키는 방법을 연구하는 것이다.

Idea : To make the computer more robust, intelligent, and learn, …

Let's **model our computer software (and/or hardware) after the brain.**

뇌 구조



전두엽: 기억력·사고력 등의 고등행동을 관장하며 다른 연합영역 으로부터의 정보를 조정하고 행동을 조절.

두정엽: 기관에 **운동명령**을 내리는 운동중추.
체감각 피질과 감각연합영역이 있어 촉각, 압각, 통증등의 체감각의 처리에 관여하며 피부, 근골격계, 내장, 미뢰로부터의 감각신호를 담당한다.

측두엽: 청각정보의 처리. 일차시각 피질에서 유래한 정보가 도달해 색, 모양등이 인지. 내측두엽 부분은 해마와 함께 **기억형성**에 주요한 역할을 수행.

후두엽: 시각정보의 처리. 눈으로 들어온 시각정보가 시각피질에 도착하면 사물의 위치, 모양, 운동 상태를 분석.

Image source: https://ko.wikipedia.org/wiki/%EB%87%8C

Neuron (cont'd)

Although heterogeneous, at a low level **the brain is composed of neurons.**

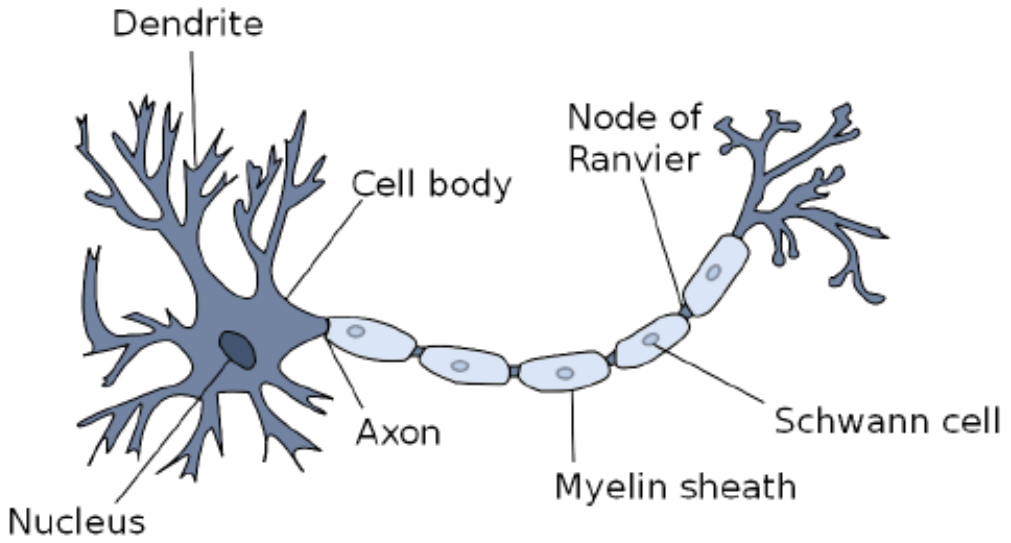
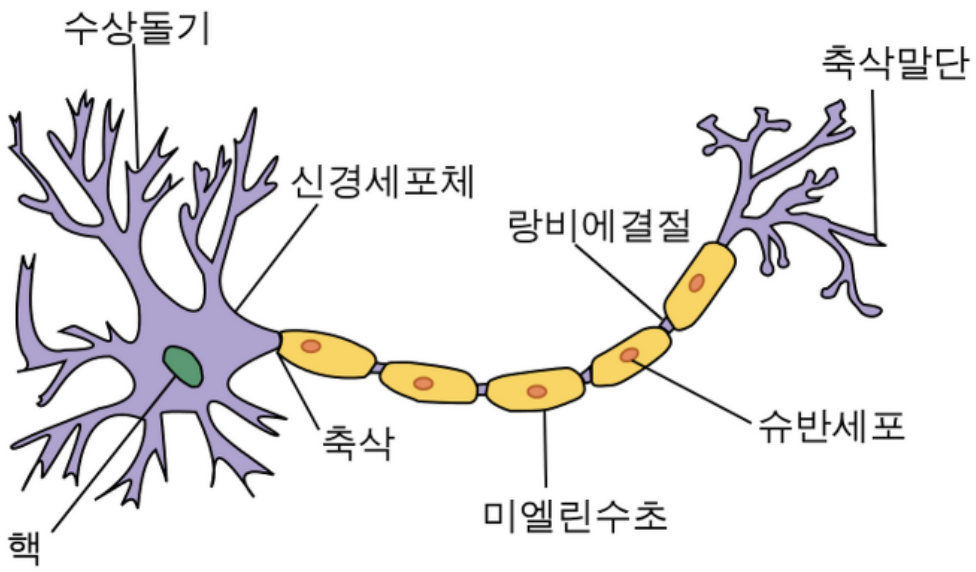


Figure 2.3: Illustration of a biological neuron with the components discussed in this text.

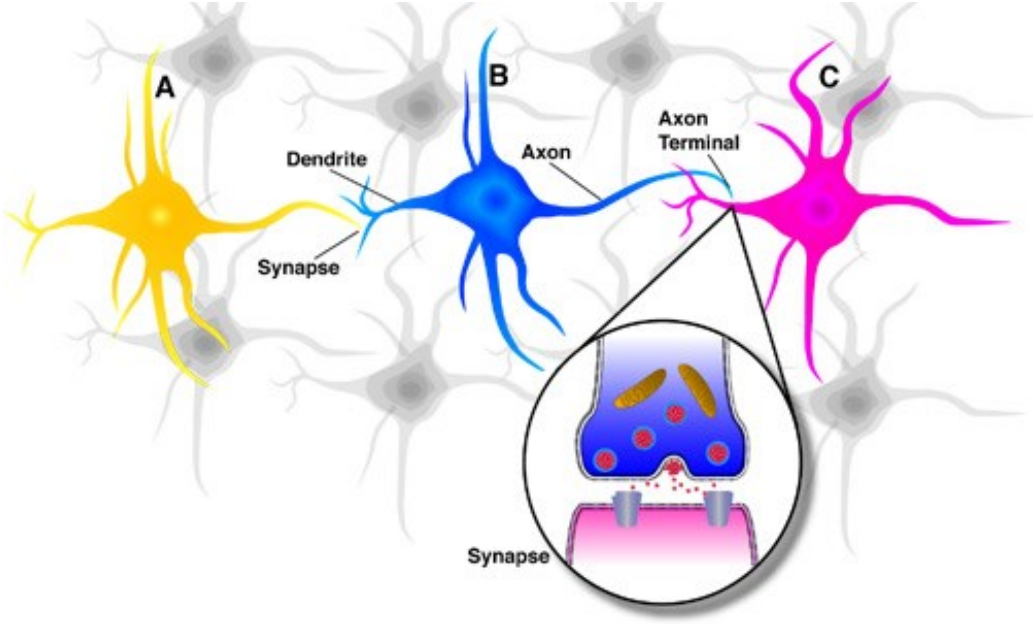
출처: A Brief introduction to Neural Networks.
http://www.dkriesel.com/en/science/neural_networks



(출처/ 한글 위키피디아, '신경세포')

Neuron (cont'd)

- A neuron **receives input from other neurons** (generally thousands) from its synapses
- **Inputs are** approximately **summed.**
- When the input **exceeds a threshold** the **neuron sends an electrical spike** that travels from the body, down the axon, **to the next neuron(s)**



https://www.youtube.com/watch?v=A9Xru1ReRwc

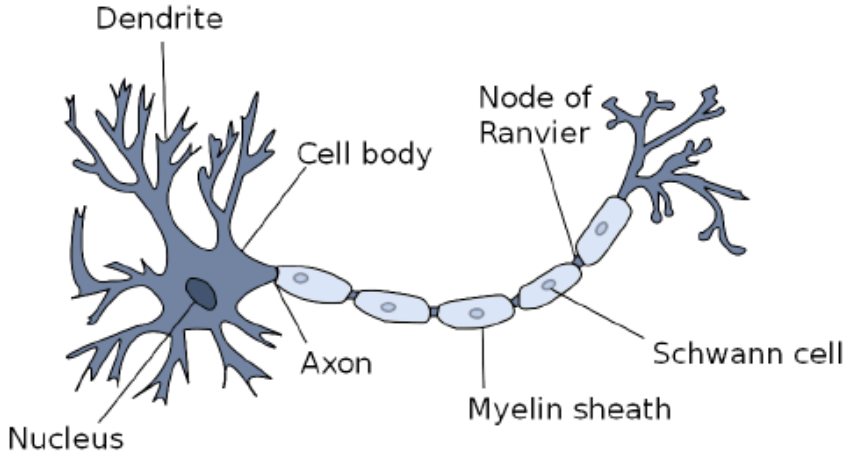


Figure 2.3: Illustration of a biological neuron with the components discussed in this text.

출처: A Brief introduction to Neural Networks.
http://www.dkriesel.com/en/science/neural_networks

Neuron Activation

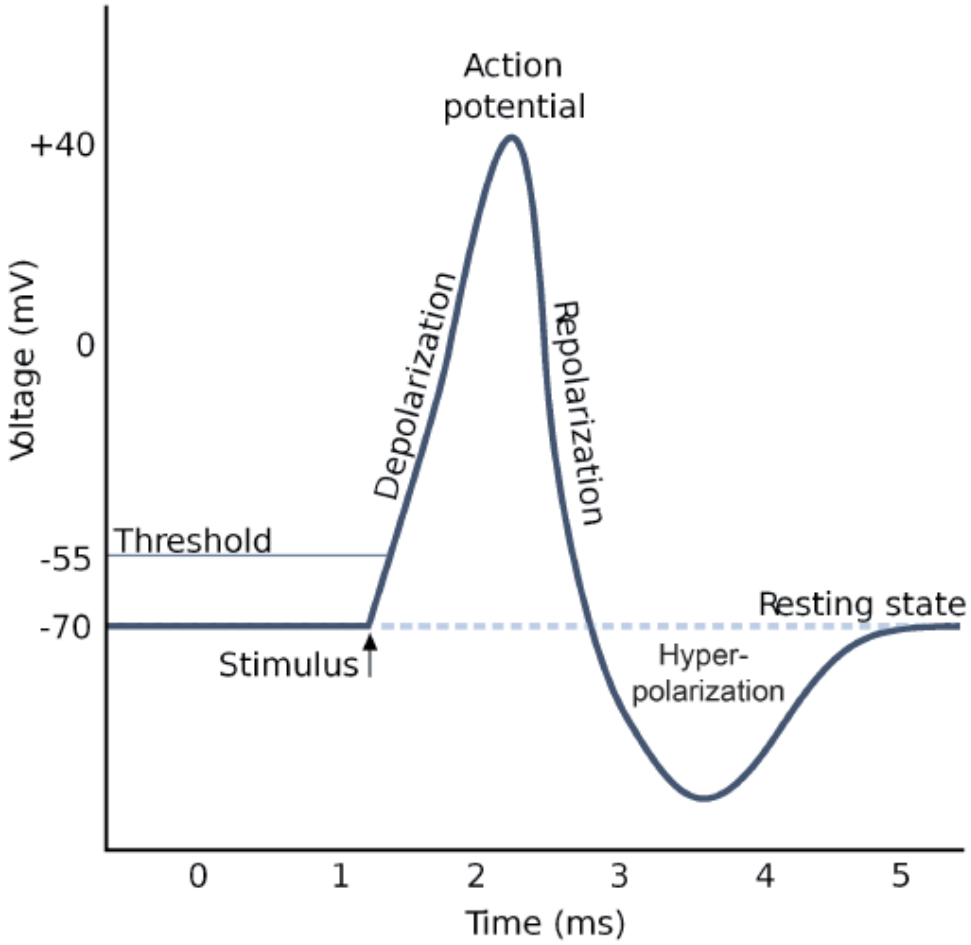


Figure 2.4: Initiation of action potential over time.

출처: A Brief introduction to Neural Networks.
http://www.dkriesel.com/en/science/neural_networks

Learning in Brain

- **Brains learn**
  - 뉴런 사이의 **연결 강도 변화**
  - 뉴런 사이의 새 연결 생성/기존 연결 삭제

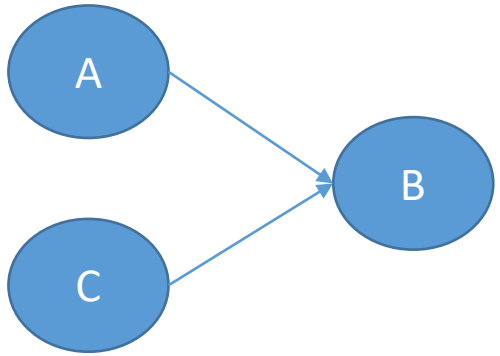
- **Hebb's Postulate (Hebbian Learning)**
  - 뉴런A의 축삭이 뉴런B를 흥분시키기(Exicited)에 충분히 가깝고
  - 반복적으로 또는 지속적으로 B를 발화(Firing)시키면
  - B를 발화시키는 뉴런 중 하나로서 A의 효율이 향상되도록 세포 연결구조가 변한다. **(연결 강도 변화)**
    - 한쪽 또는 양쪽 뉴런에서 성장과정 또는 대사변화가 일어난다.

- **Long Term Potentiation (LTP)**
  - 자극의 결과, 2개의 신경 세포 사이의 연결 강도 강화/약화된 상태가 장기간 유지되는 것.
  - 학습 및 기억을 위한 세포 기반.

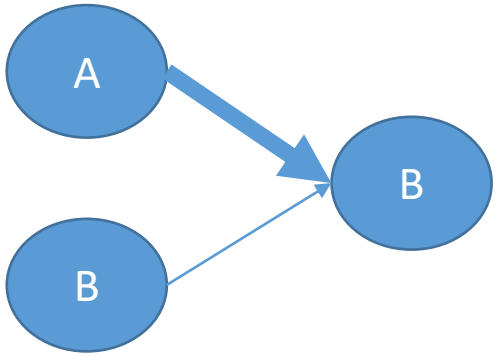
Learning in Brain: 예

- 뉴런 사이의 **연결 강도 변화**

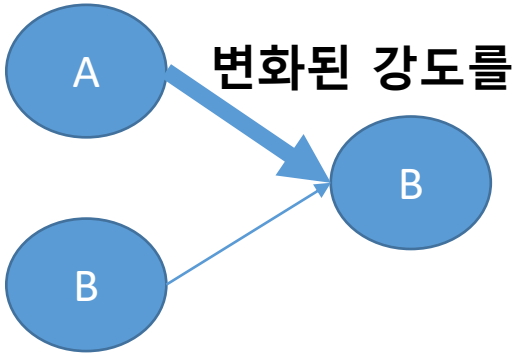
초기: 뉴런 A와 C가 다음 뉴런 B와 같은 강도로 연결되어 있다.
A가 Firing 하여 B가 Firing 하는 일이
C가 Firing하여 B가 Firing하는 일보다 훨씬 많이 반복되면,



후기: 뉴런 A와 뉴런 B 연결 강도가 커져
A가 Firing하는 즉시 B가 Firing하도록 변경된다.



- **Long Term Potentiation**



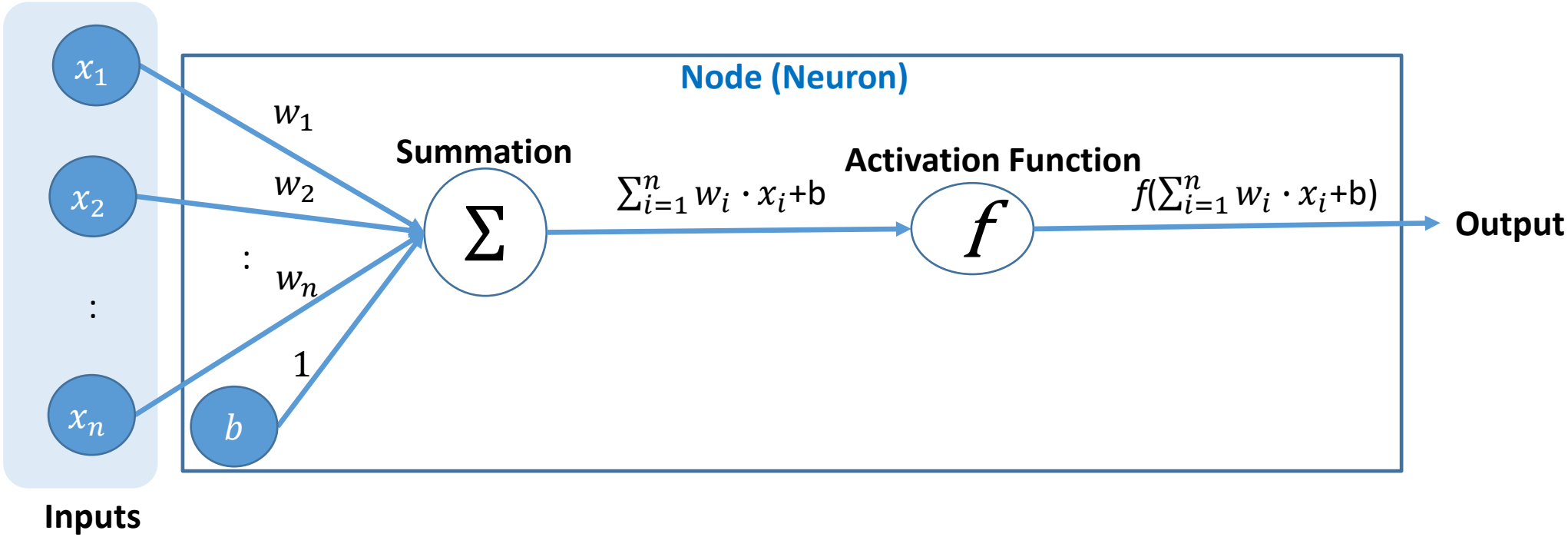
변화된 강도를 오랜 기간동안 유지

Summary

- **A network that connects the nodes (Neuron) together**
- **Information is stored** in a distributed fashion **among the links that connect the nodes**
- **Learning** can occur with gradual **changes in connection strength.**

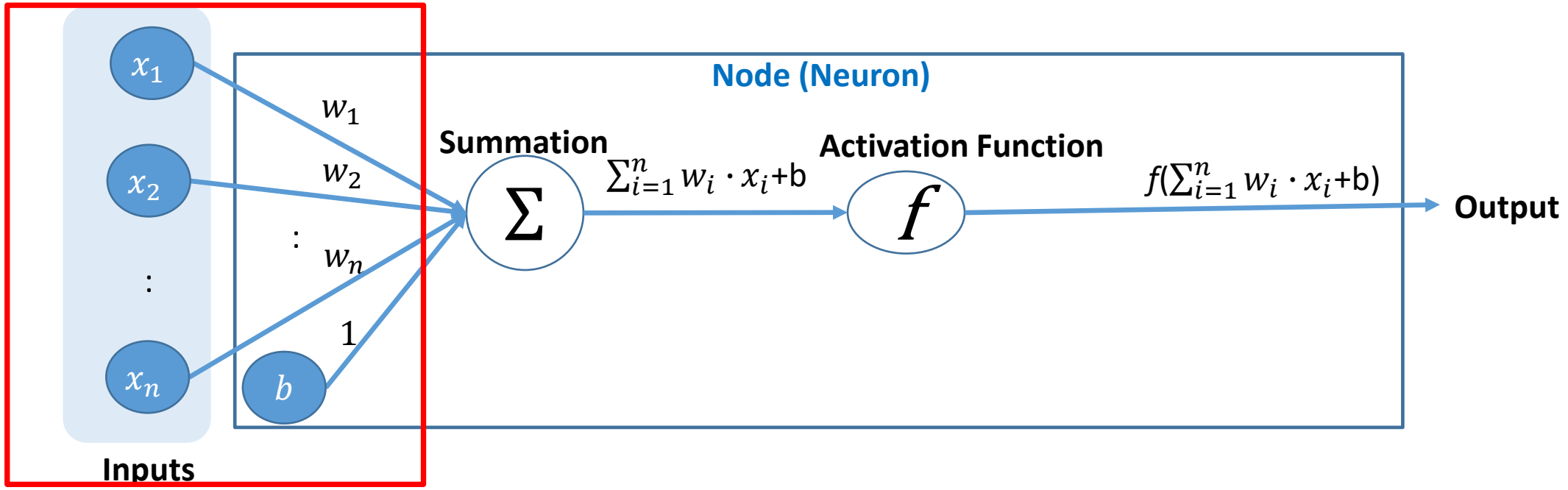
Neuron in Artificial Neural Network, Network Connection

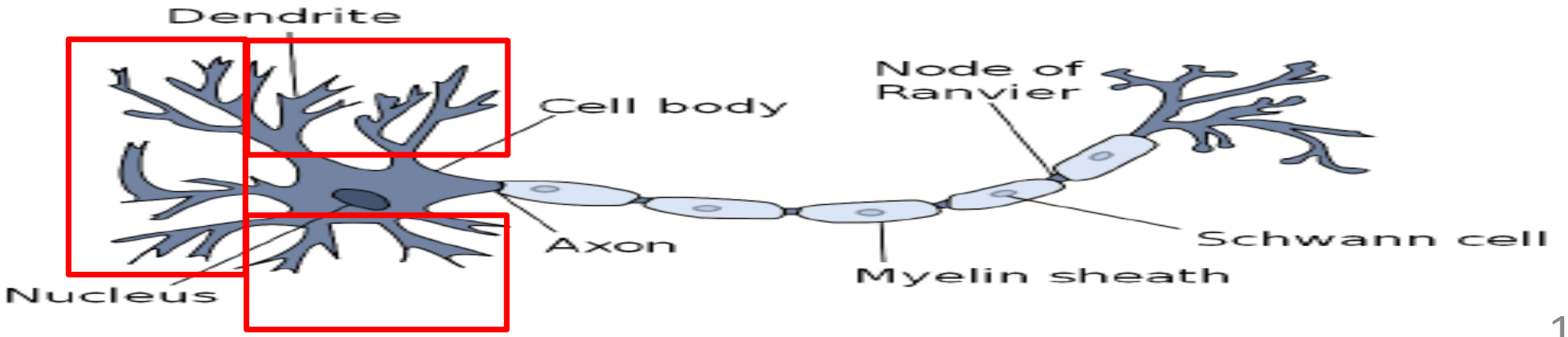
Node (Artificial Neuron)



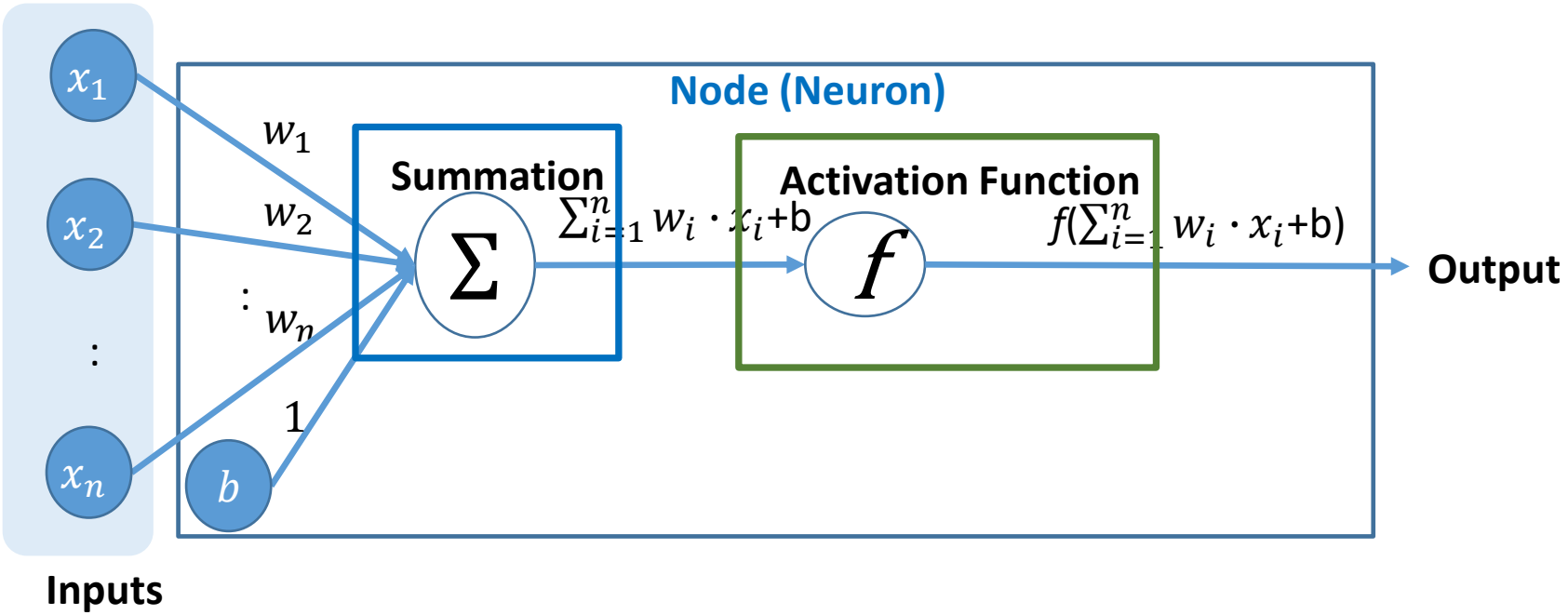
$x_1, x_2, \ldots, x_n$: 입력 값. 다른 Node 1, 2, … , n의 출력 값 .
$w_1, w_2, \ldots, w_n$: 입력 값에 대한 weight. (연결 강도)
b : bias.
f : activation function,

Node (Artificial Neuron) VS. Biological Neuron





Node (Artificial Neuron) VS. Biological Neuron (cont'd)



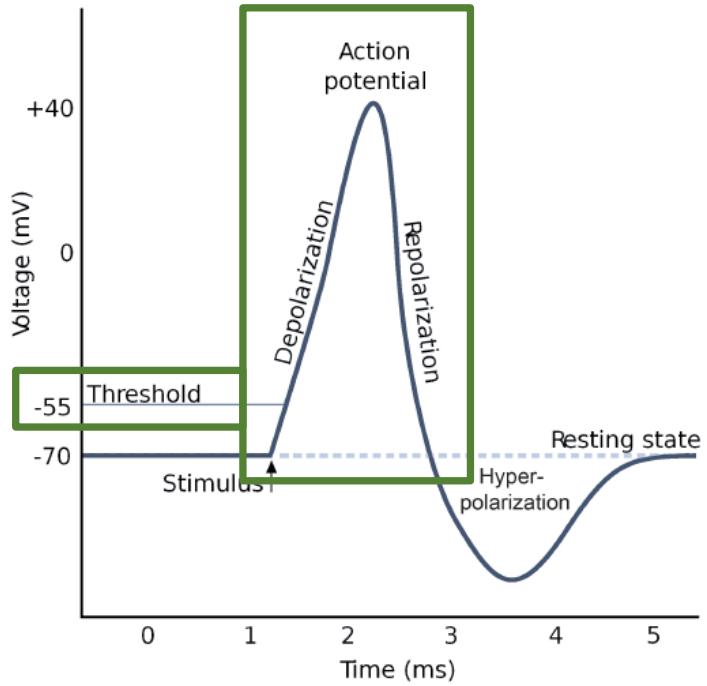
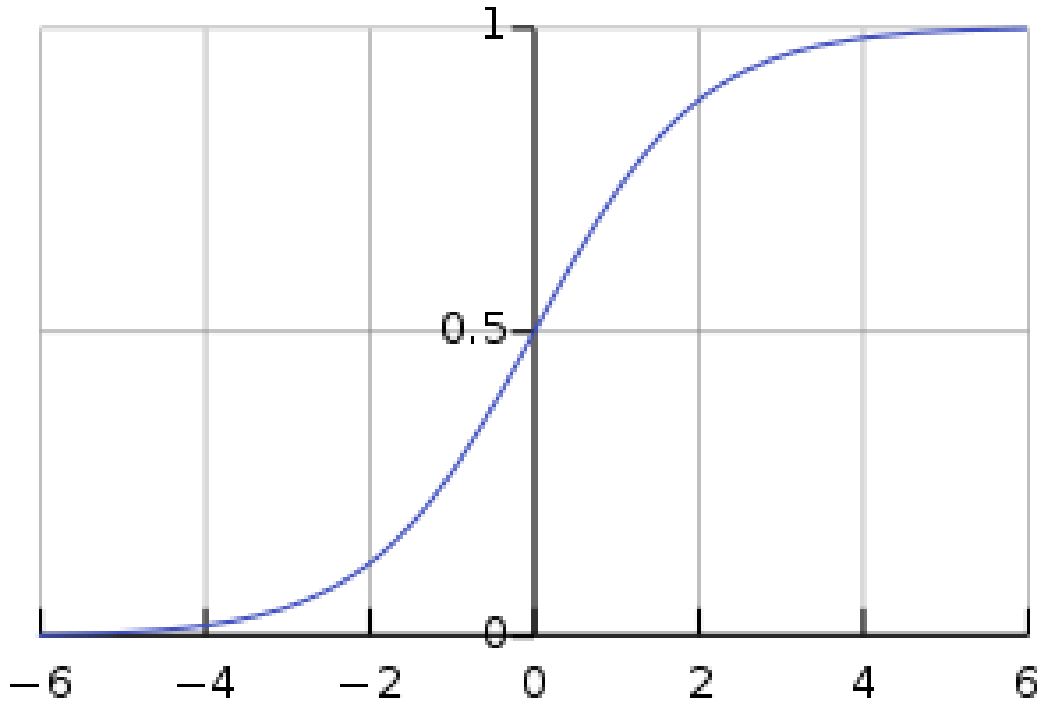


Figure 2.4: Initiation of action potential over time.

- **Inputs (Stimulus) are** approximately **summed.**
- When the input **exceeds a threshold** the **neuron sends an electrical spike** that travels from the body, down the axon, **to the next neuron(s)**

Activation Function : Logistic Function (Sigmoid)

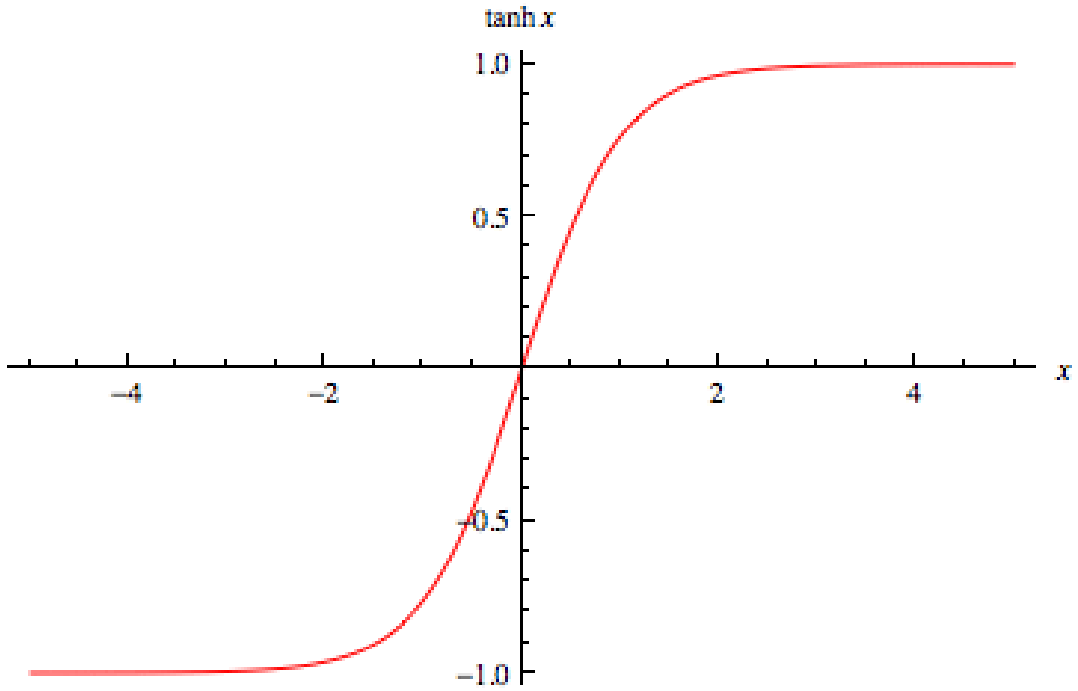
$$\sigma(x) = \frac{1}{1 + e^{-x}}$$



Imager source: Wikipedia

Activate Function : Hyperbolic Tangent (tanh)

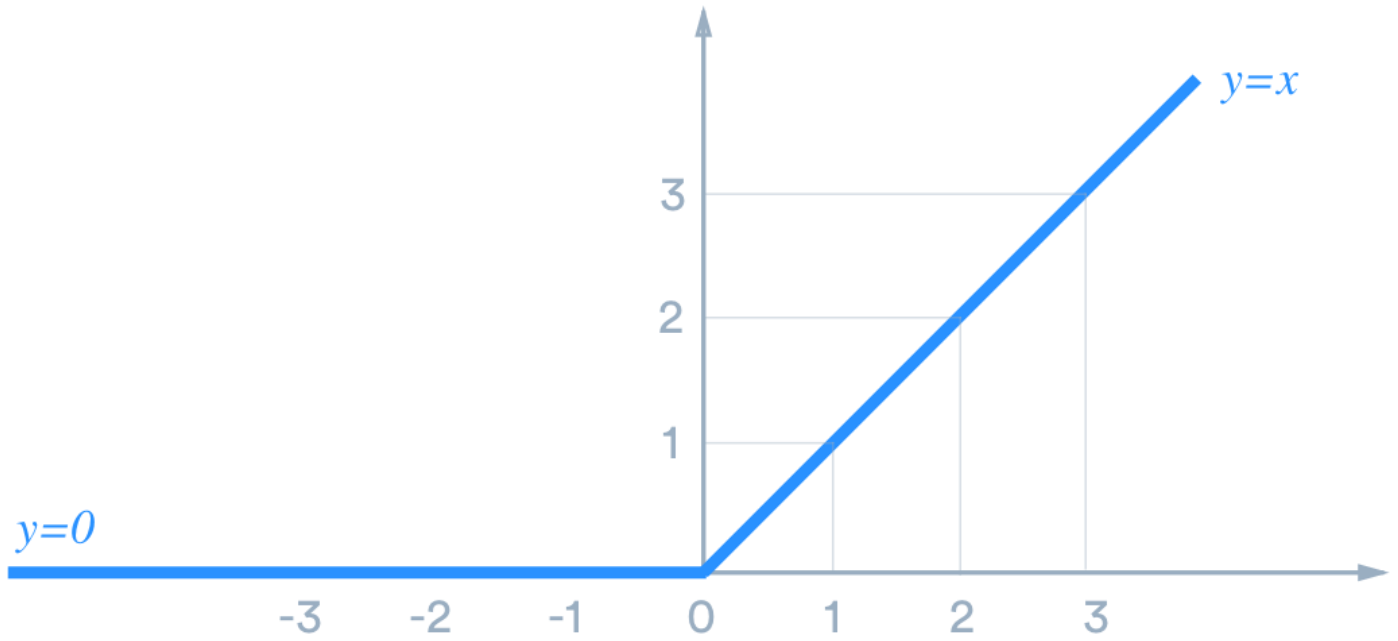
$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$



Imager source: https://mathworld.wolfram.com/HyperbolicTangent.html

Rectified Linear Unit

$$\text{ReLU(x)} = \begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$$
$$= \max\{0, x\} = x\mathbf{1}_{x>0}$$



Imager source: https://medium.com/@danqing/a-practical-guide-to-relu-b83ca804f1f7

Network Connection – Connection Example

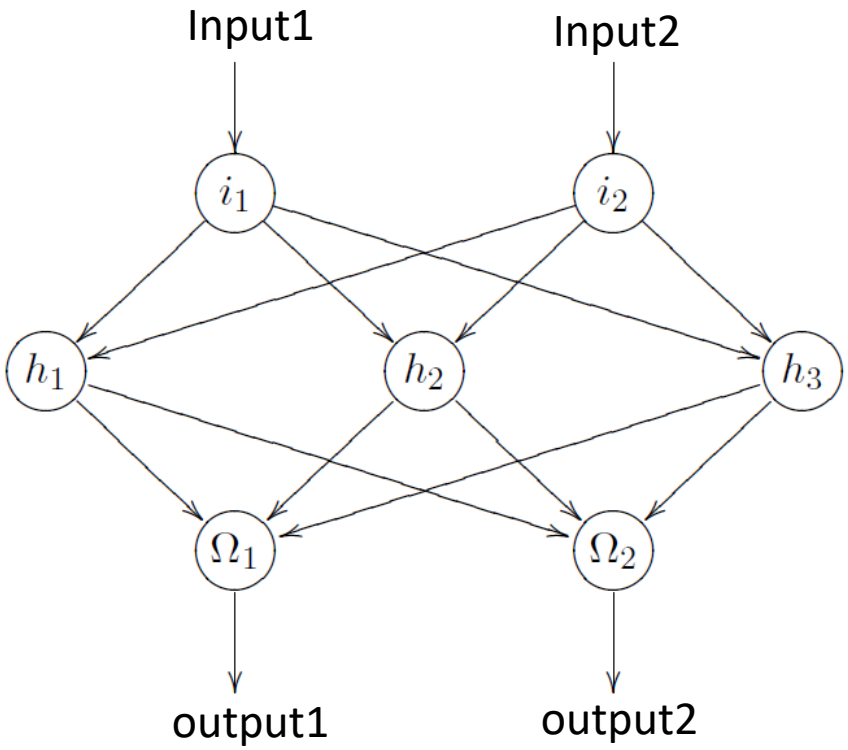
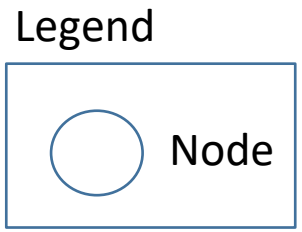


Figure 3.3: A feedforward network with three layers: two input neurons, three hidden neurons and two output neurons.



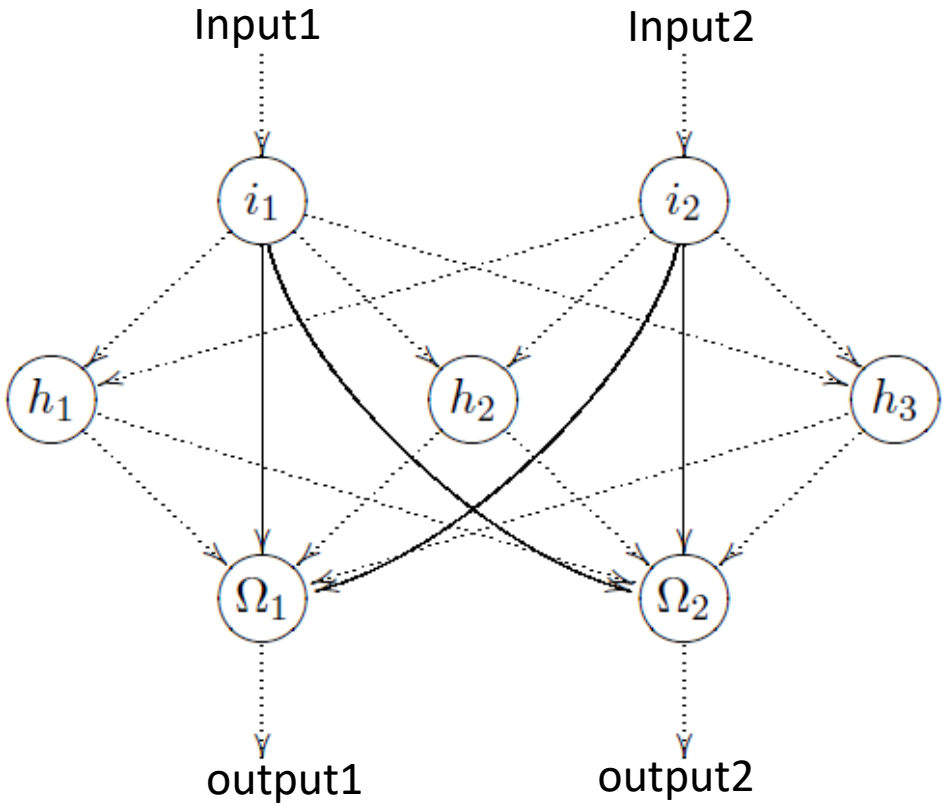


Figure 3.4: A feedforward network with shortcut connections, which are represented by solid lines.

출처: A Brief introduction to Neural Networks.
http://www.dkriesel.com/en/science/neural_networks

Network Connection – Connection Example

출처: A Brief introduction to Neural Networks.
http://www.dkriesel.com/en/science/neural_networks

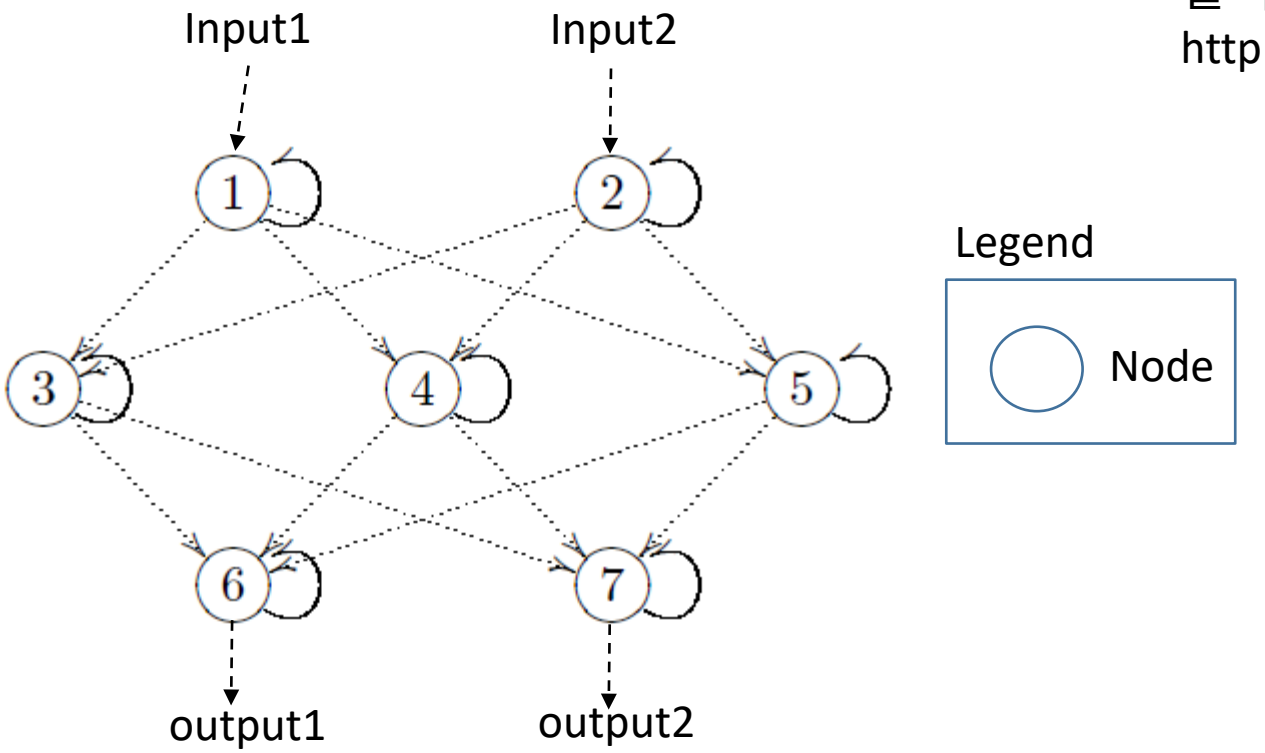
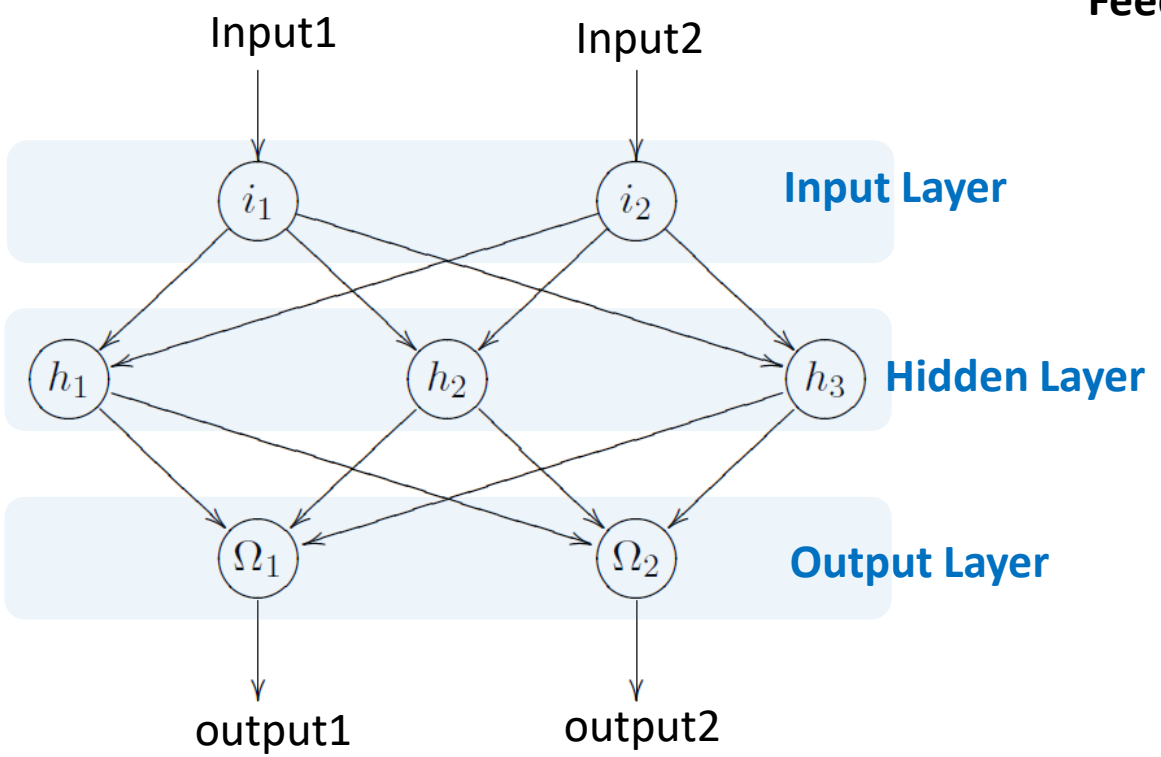


Figure 3.5: A network similar to a feedforward network with directly recurrent neurons. The direct recurrences are represented by solid lines.

Feedforward Network (Fully Connected Network)



3 Layer로 이루어진 Neural network

Feedforward Network

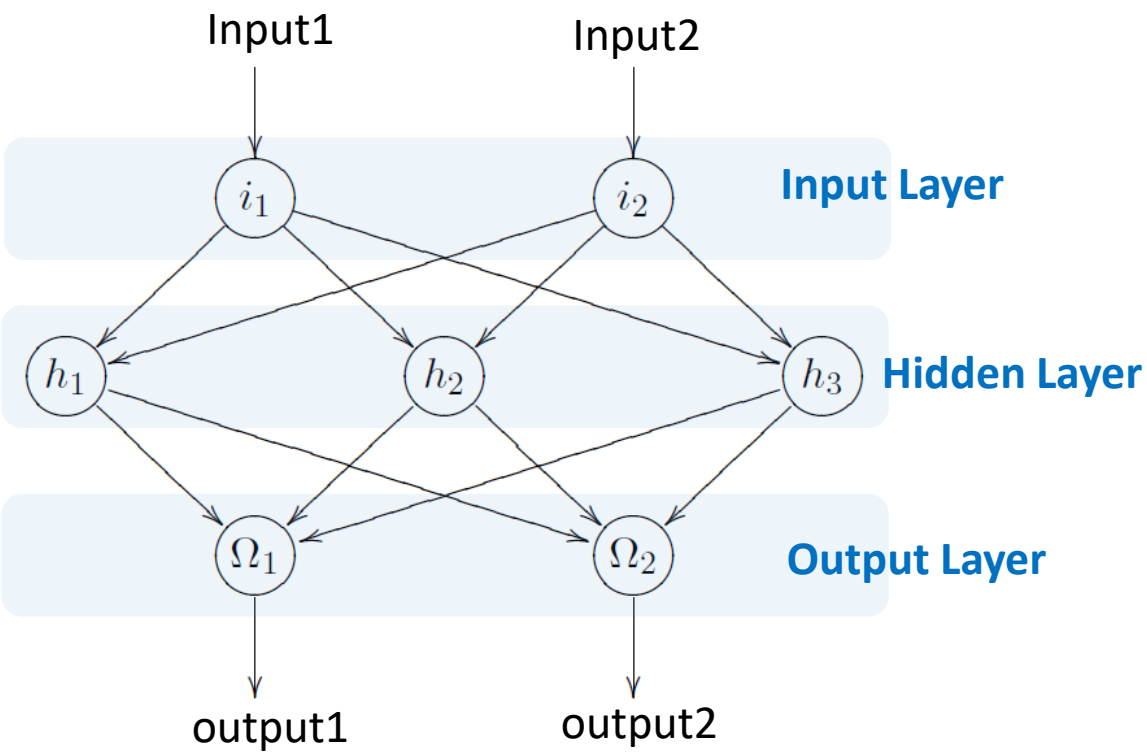
Network가 **여러 개의 Layer**로 구성된다.

하나의 Layer는 여러 개의 node로 구성된다.

- 제일 처음 Layer(Input Layer)를 제외한 모든 Layer의 각 node는 **이전 Layer에 속하는 모든 node의 output을 input으로 받는다.**
- Input Layer에 속하는 하나의 node가 feature vector 하나의 element(dimension)를 input으로 받는다.
- 같은 Layer에 속하는 node 사이, 혹은 node 자신에게는 연결하지 않는다.

출처: A Brief introduction to Neural Networks.
http://www.dkriesel.com/en/science/neural_networks

Feedforward Network (Fully Connected Network) (Cont'd)



3 Layer로 이루어진 Neural network

Input Layer : Data 입력을 받기 위한 Layer.
- Network의 첫 Layer.
- Input Layer에 속하는 하나의 node가 feature vector 하나의 element(dimension)를 input으로 받는다.

Output Layer : 최종 결과를 출력하는 Layer.
- Network의 제일 마지막 Layer.
- Output layer의 모든 node의 output을 모아 output vector로 활용한다.

Hidden Layer : Input Layer와 Output Layer사이의 Layer.
- Layer가 여러 개(층) 존재할 수 있다.
- Hidden Layer 층 수가 많아지면 Deep이란 수식어가 붙는다.

출처: A Brief introduction to Neural Networks.
http://www.dkriesel.com/en/science/neural_networks

Feedforward Network : Input, Output 예1 (사과 분류)

Table 8.1 Training data for the apple sorting agent

Feature vector

Size [cm]	8	8	6	3	…
Color	0.1	0.3	0.9	0.8	…
Merchandise class	B	A	A	B	…

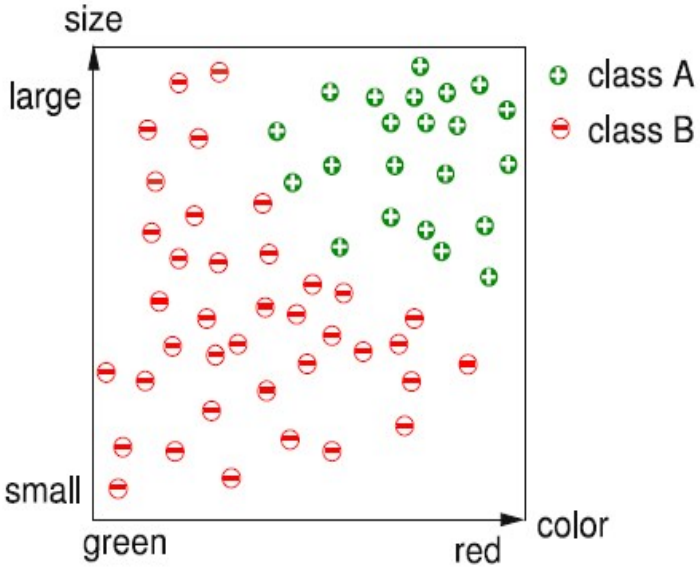
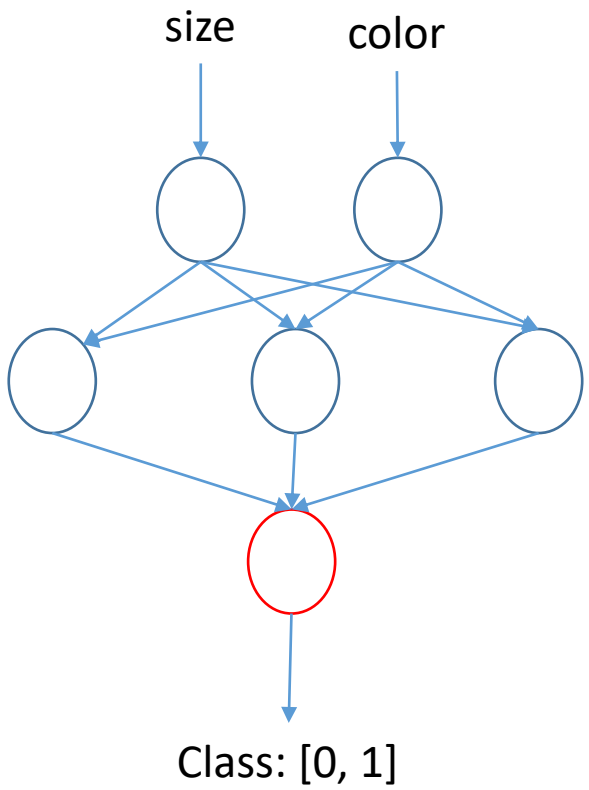


Fig. 8.2 BayWa company apple sorting equipment in Kressbronn and some apples classified into merchandise classes A and B in feature space (Photo: BayWa)



Class가 0이냐? 1 이냐?

Feedforward Network : Input, Output 예2 (사과 분류)

Table 8.1 Training data for the apple sorting agent

Feature vector

Size [cm]	8	8	6	3	…
Color	0.1	0.3	0.9	0.8	…
Merchandise class	B	A	A	B	…

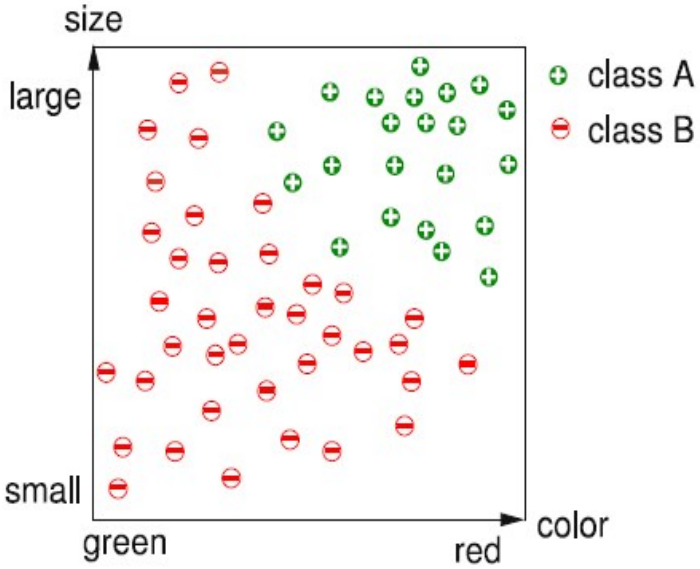
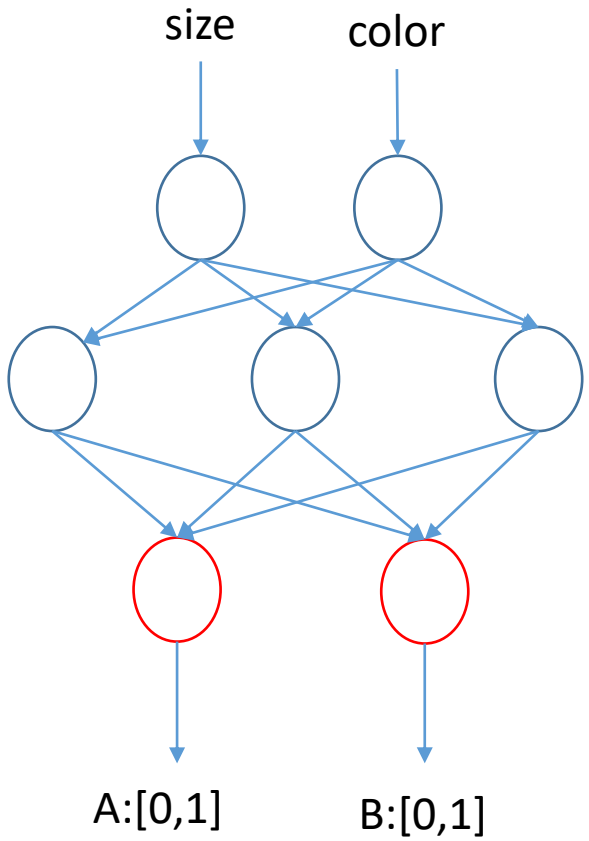


Fig. 8.2 BayWa company apple sorting equipment in Kressbronn and some apples classified into merchandise classes A and B in feature space (Photo: BayWa)

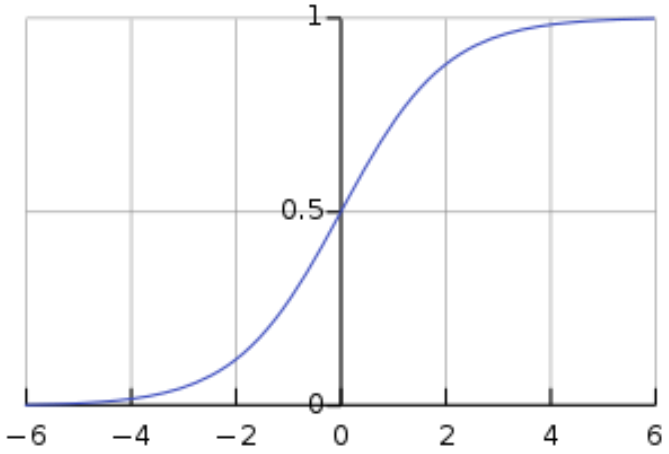


Class가A일 확률 **Class가B일 확률**

SLP(Single Layer Perceptron), MLP (Multi-Layer Perceptron)

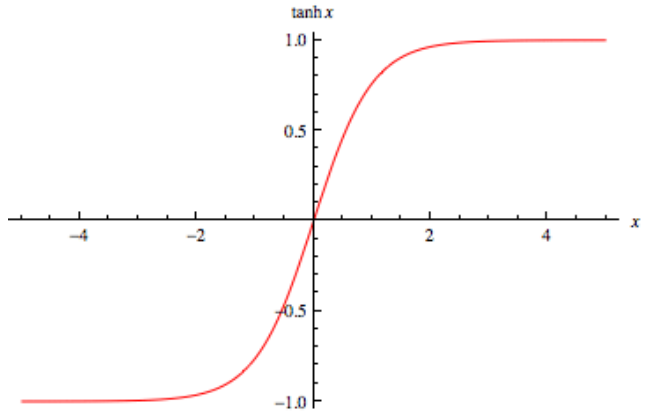
Sigmoid와 같은 함수로 Activation함수를 변경함으로써 생기는 이점

- GD(Gradient Descent)를 사용하여 Objective Function 최적화 가능
  - 미분 가능하기 때문에
- Non-linearly separable problem도 풀 수 있는 가능성이 생김.
  - Sigmoid, Tanh, ReLU 모두 **Non-linear function**



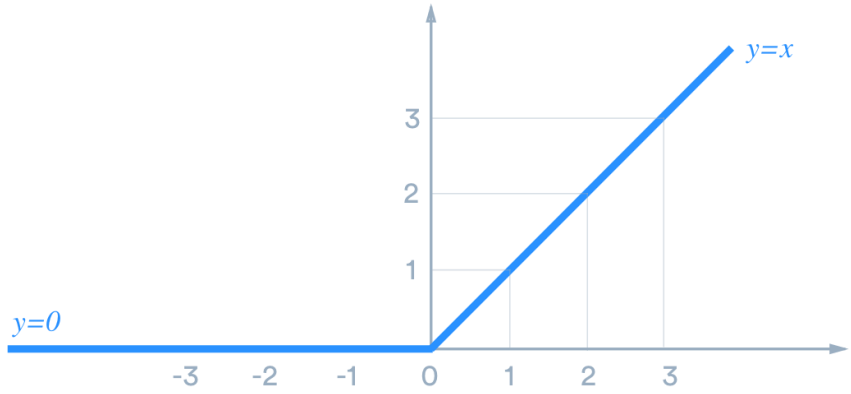
Sigmoid

Imager source: wikipedia



tanh

Imager source:
https://mathworld.wolfram.com/HyperbolicTangent.html

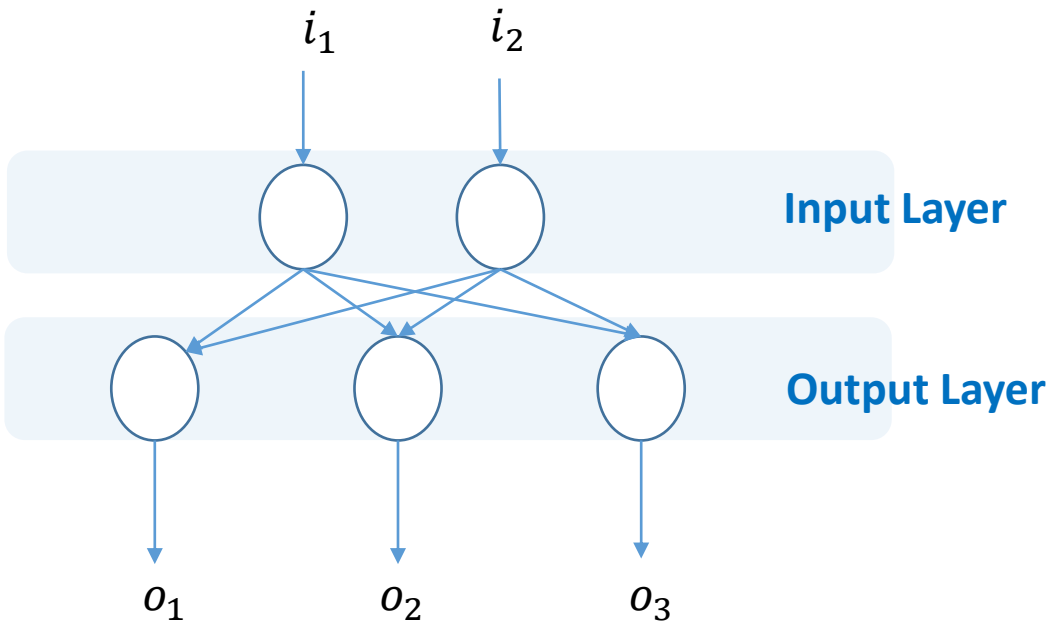


ReLU

Imager source: https://medium.com/@danqing/a-practical-guide-to-relu-b83ca804f1f7

위 선은 모두 **직선이 아님**.

Single Layer Perceptron

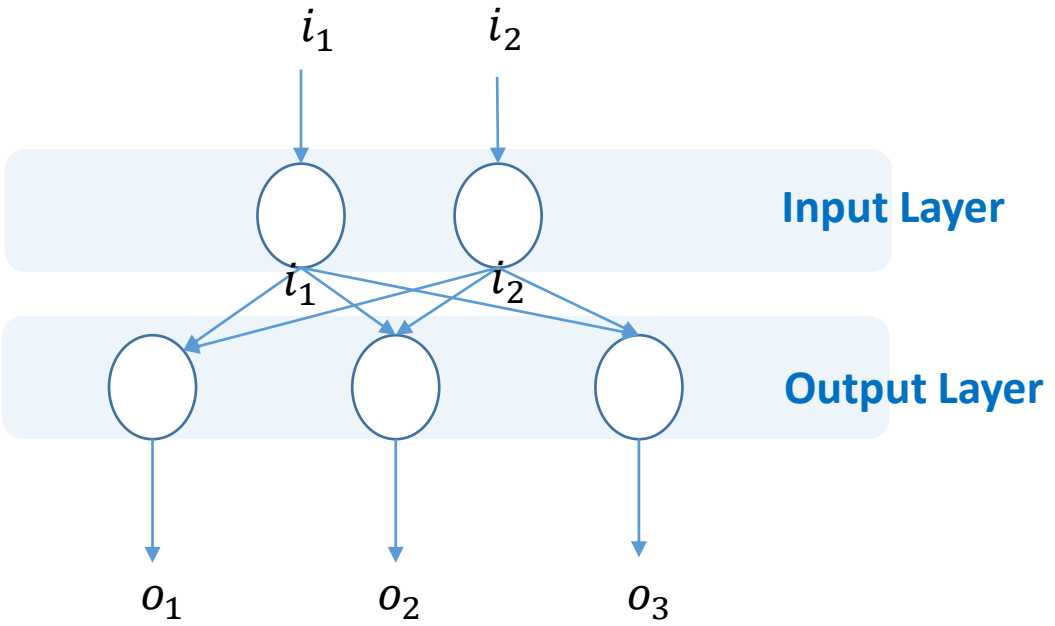


Single Layer Perceptron :
Node 하나를 하나의 Perceptron이라 볼 수 있으므로, 왼쪽 그림의 Network는 하나의 Perceptron 층(Output Layer)으로 이루어진 네트워크이다.

질문:
왼쪽 그림에서는 Input Network가 있는데 Single Layer?

Input Layer

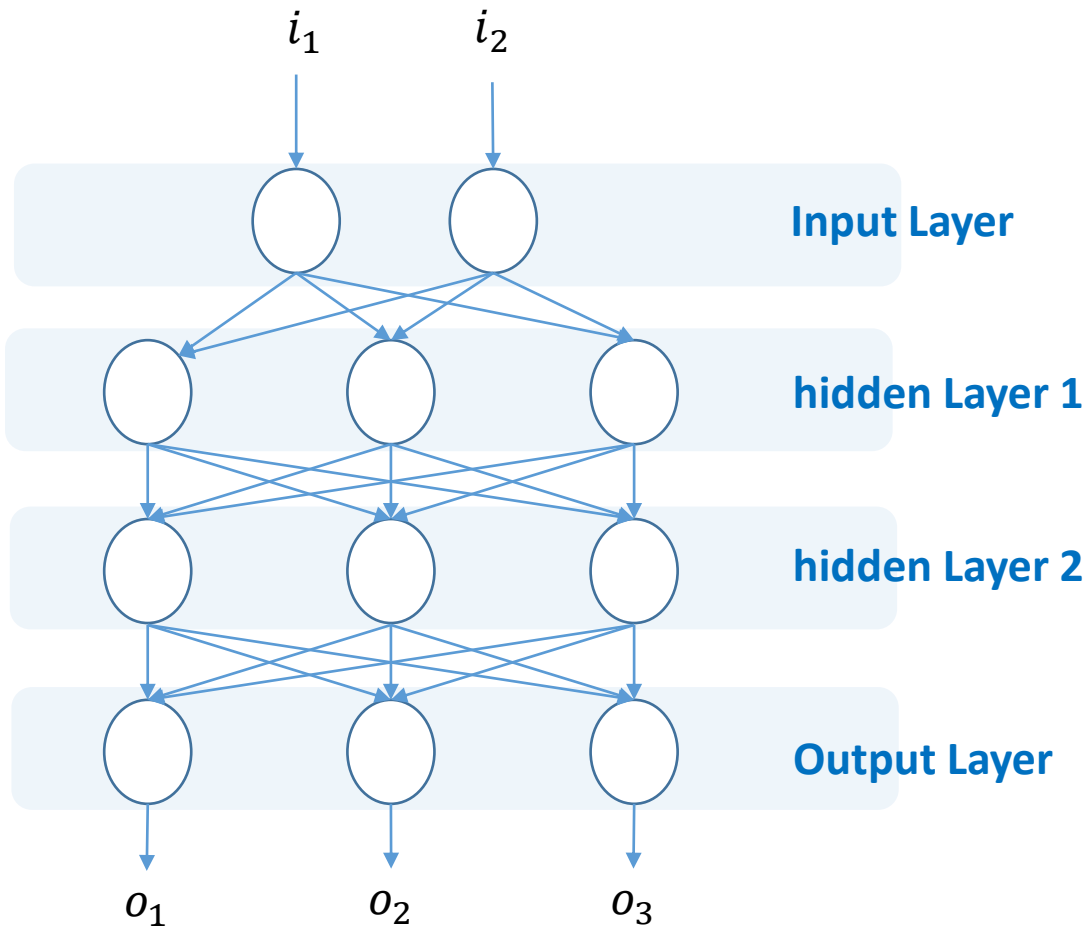
- Neural Network에서 Input Layer는
  - **Input을 다음 Layer로 전달**하기 위한 특수 목적의 Layer라 볼 수 있다.
  - Input을 다음 Layer로 전달하는 것이 목적이므로 **학습에는 기여하지 않는다.**
    - Weight vector가 update 되지 않는다.
  - Input Layer의 node는 아래와 같은 특수 Node라 생각할 수 있다.
    - Input 이 x_1 하나만 존재
    - Weight $w_1 = 1$이며 변하지 않는다.
    - Bias b = 0 이다.
    - Activation Function 이 Identity Function 이다.
      - f(x) = x



Single Layer Perceptron: Linear Problem Solver

- Activation Function이 non-linear 임에도 불구하고, Single layer perceptron은 Non-linear 한 문제를 풀기 어렵다.
  - 왜? Perceptron이 1 층이라서.

Multi Layer Perceptron

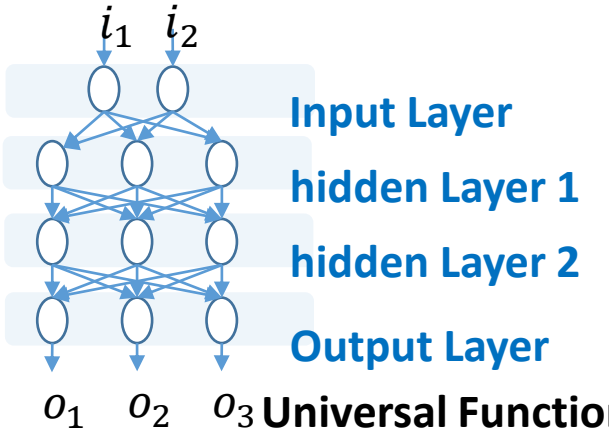


Multi Layer Perceptron (MLP):
Hidden Layer을 Input layer와 Output layer 사이에 최소 1 Layer 끼워 넣으면
(Hidden Layer 1층 + Output Layer 1층 = Multi)
Multi Layer Perceptron이 된다.

Multi Layer Perceptron (Cont'd)

- **Universal Function Approximator**
  - **Pros**
    - Very powerful - can **learn any function (물론 non-linear function도)**, given enough hidden units!
      - With enough hidden units, we can generate any function.
    - Inherently **parallel algorithm**, ideal for multiprocessor hardware.
  - **Cons**
    - **Have the same problems of Generalization vs. Memorization.**
      - With too many units, we will tend to memorize the input and not generalize well.
      - Some schemes exist to "prune" the neural network.
  - Despite the cons, a very powerful algorithm that has seen **widespread successful deployment**.

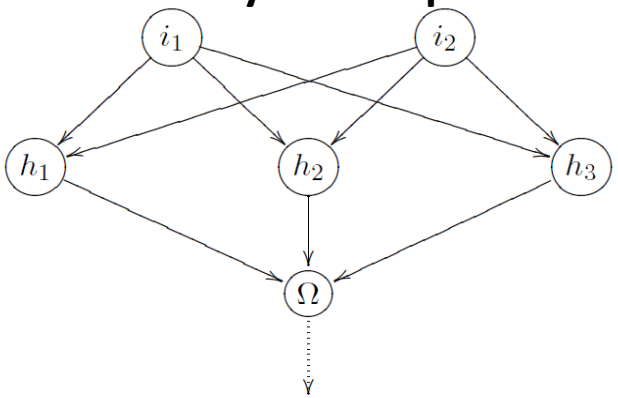
Universal Cutter

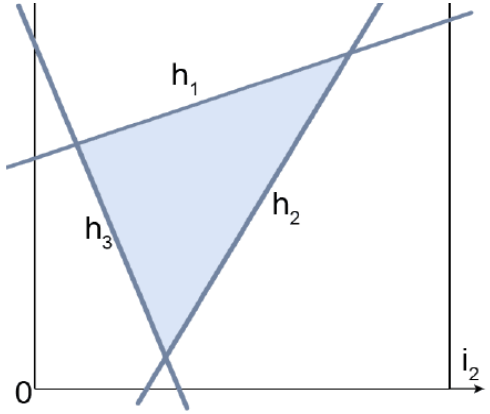


Universal Function Approximator

Multi-Layer Perceptron : Concept

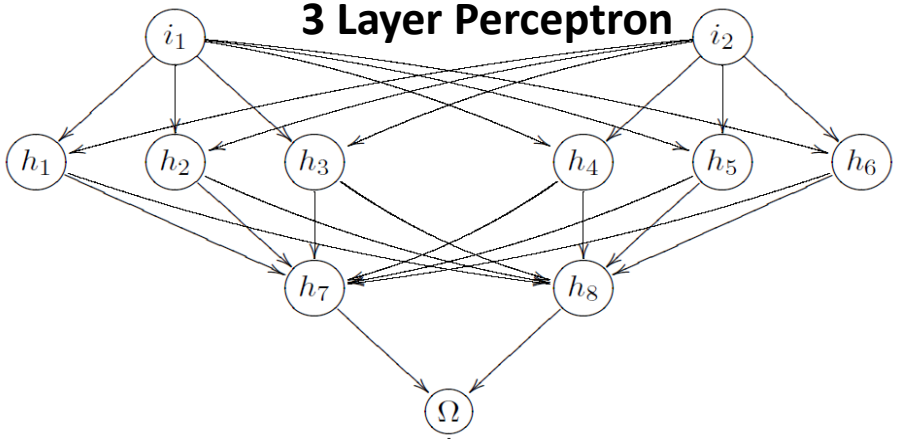
2 Layer Perceptron

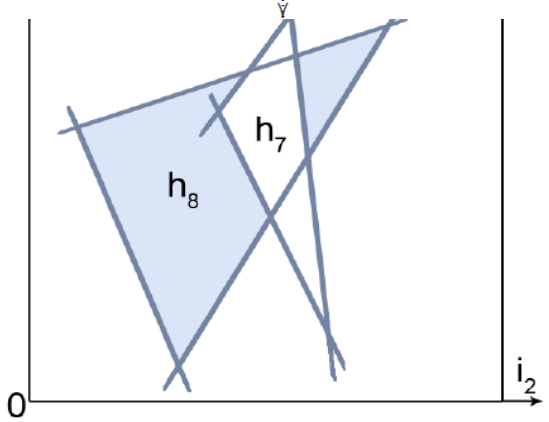




Non-linearly separable

3 Layer Perceptron





Non-linearly separable

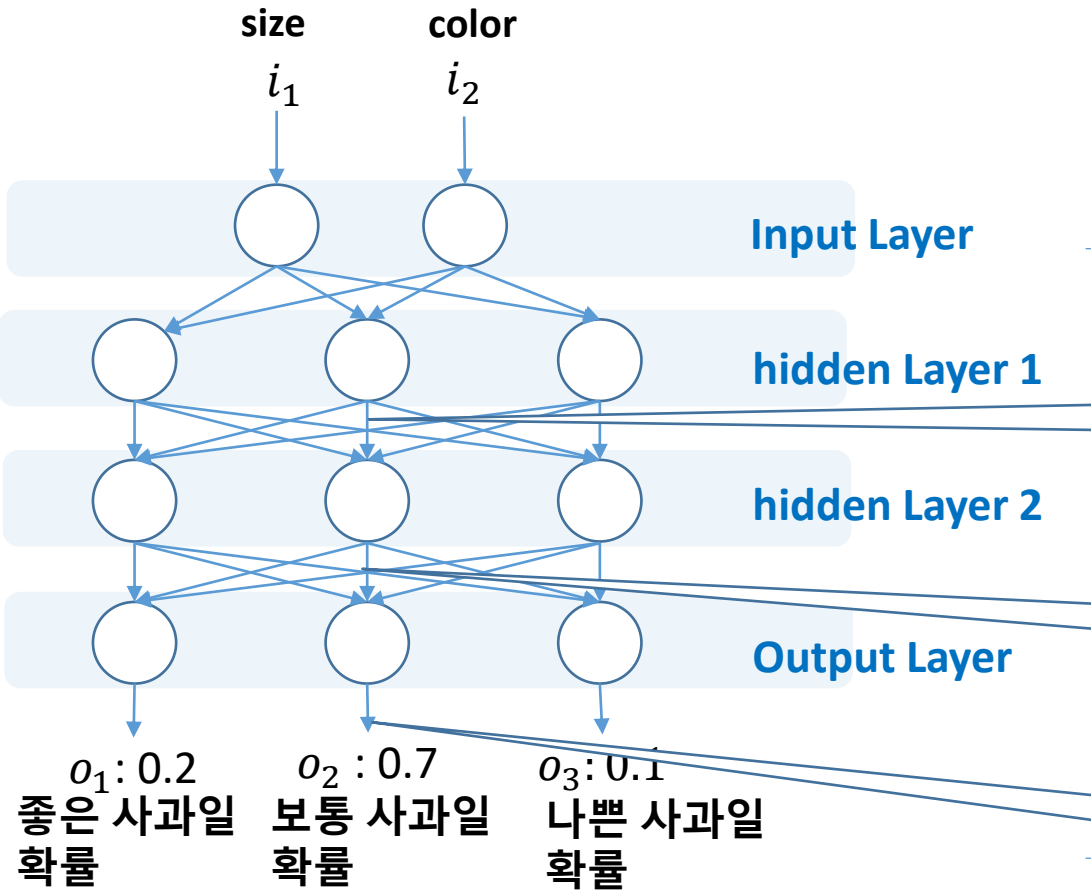
***1 Layer Perceptron은 선 1개**

Figure 5.10: We know that an SLP represents a straight line. With 2 trainable weight layers, several straight lines can be combined to form convex polygons (left). By using 3 trainable weight layers several polygons can be formed into arbitrary sets (right).

출처: A Brief introduction to Neural Networks.
http://www.dkriesel.com/en/science/neural_networks

Back Propagation

Back Propagation : MLP를 어떻게 Training 할 것인가? (Cont'd)



정답 <1.0, 0.0. 0.0>

Training 할 Layer가 여럿.

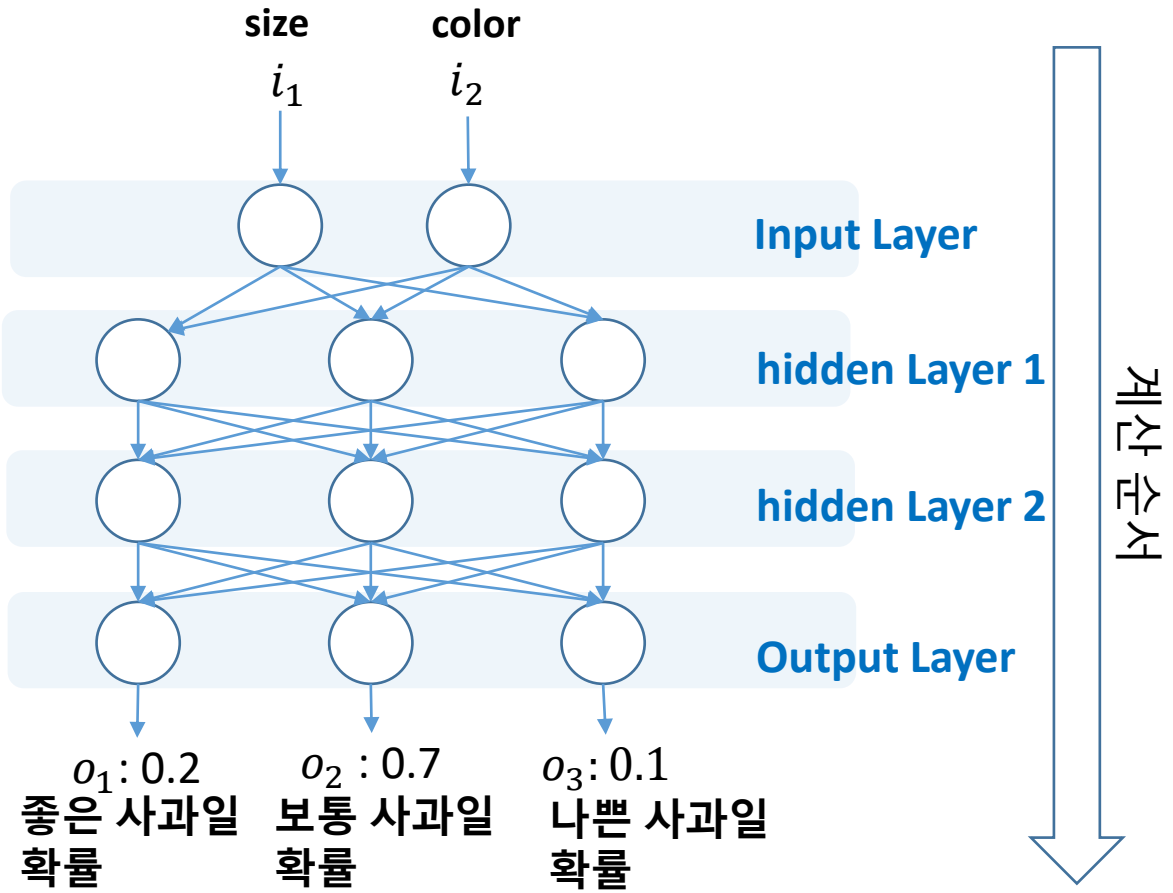
Hidden Layer1 Node들의 출력의 정답에 대한 오차
<?, ?, ?>

Hidden Layer2 Node들의 출력의 정답에 대한 오차
<?, ?, ?>

Output Layer Node들의 출력의 정답에 대한 오차
<0.8, 0.7, 0.1>

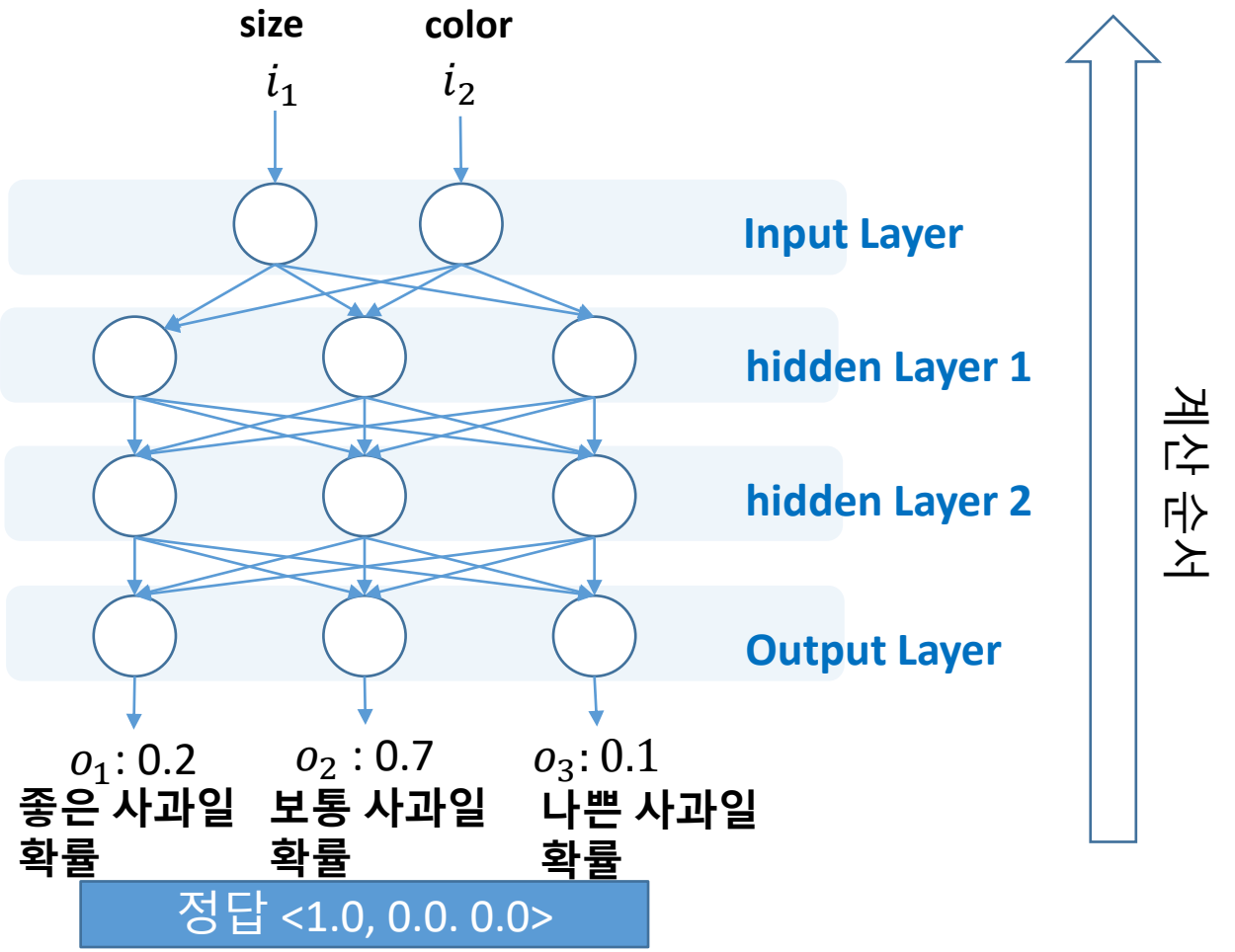
오차 전파 방향: Inference(추론)과 반대 방향으로 전파

Inference 방향 : 입력에 대해 Hidden Layer 1부터 시작하여 점차적으로 아래로 내려가며 최종적으로 Output Layer에서 결과가 계산됨. (**Feed Forward**)



오차 전파 방향: Inference(추론)과 반대 방향으로 전파 (Cont'd)

Train (오차 교정)방향 : 오차를 사용하여 model parameter를 보정하는 과정을 출력에 가까운 Output Layer 부터 시작해 점차적으로 위 Layer로 올라가 최종적으로 Hidden Layer 1이 보정됨. (**Back Propagate**)



- 따라서 **Back Propagation 계산은 Output Layer에서 부터 시작하여 순차적으로 위 Layer로 올라간다.**

Back Propagation

- Node가 존재하는 층에 관계 없이 (Hidden Layer나 Output Layer나 관계 없이) Gradient Descent를 사용하여 model parameter (w_i, b)를 학습하는 방법
  - 자세한 설명은 수업 범위 밖이므로 생략.