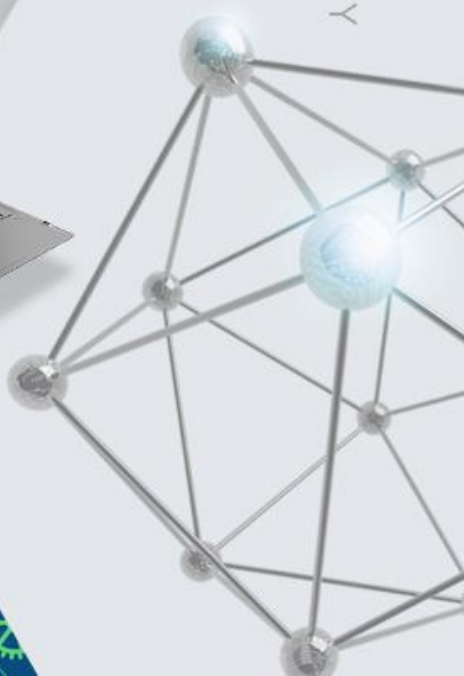
한국기술교육대학교
온라인평생교육원
OLEI

The 4th Industrial Revolution is characterized by super connectivity and super intelligence, where various products and services are connected to the network, and artificial intelligence and information communication technologies are used in 3D printing, unmanned transportation, robotics, Of the world's most advanced technologies.

TECHNOLOGY

딥러닝 입문

인공신경망과 퍼셉트론



The 4th Industrial Revolution is characterized by super connectivity and super intelligence, where various products and services are connected to the network, and artificial intelligence and information communication technologies are used in 3D printing, unmanned transportation, robotics, Of the world's most advanced technologies.

인공신경망과 퍼셉트론

학/습/목/표

1. 인공신경망의 개념에 대해 이해하고, 뉴런과 퍼셉트론에 대해 설명할 수 있다.
2. 딥러닝의 개념에 대해 이해하고, 직접 단층, 다층 퍼셉트론을 코딩할 수 있다.

학/습/내/용

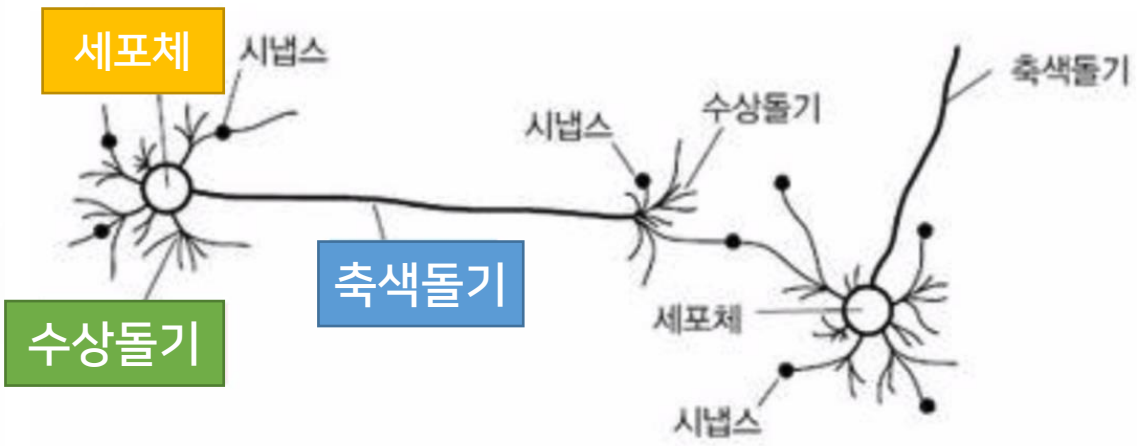
1. 인공신경망과 퍼셉트론
2. 논리회로와 다층 퍼셉트론

1. 인공신경망과 퍼셉트론

1) 뉴런(신경세포)이란?

(1) 뉴런(신경세포)

- 뉴런(신경세포)
  - 신경계를 구성하는 기본 세포 단위
  - 신경계의 모든 작용은 뉴런의 상호작용으로 인해 발생
  - 시냅스 구조를 통해 신호 전달



  - 인간의 뇌에는 뉴런이 약 1000억개, 시냅스는 약 100조개가 있음

2) 인공신경망이란?

(1) 인공신경망

- 기계학습의 알고리즘 중 하나로 인간의 뇌에서 영감을 얻어 만들어진 알고리즘
- 인간 뇌의 신경세포(뉴런)을 추상화하여 비슷하게 만든 인공신경망

(2) 인공신경망 등장 배경

인공지능을 만들자!

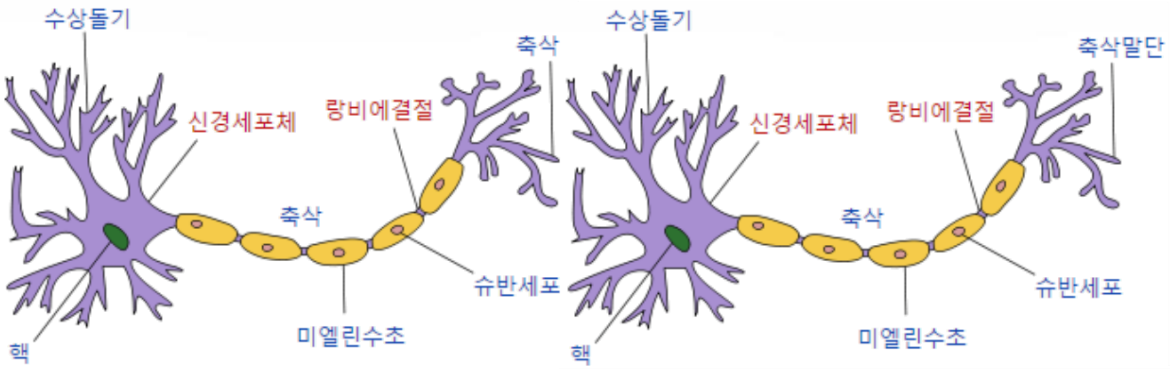
↓

인간처럼은 안 될까?

↓

인간의 뇌처럼 프로그래밍 해볼까?

- 뉴런의 구조를 본떠서 추상화해서 만들게 됨

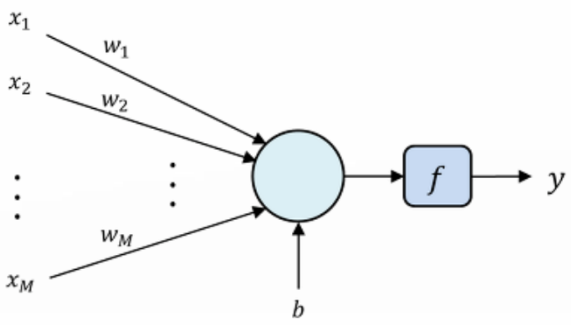


1. 인공신경망과 퍼셉트론

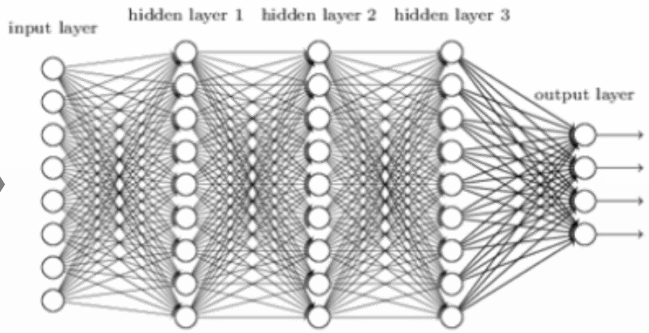
3) 퍼셉트론(인공뉴런)

(1) 퍼셉트론(인공뉴런)

- 인공신경망 모형 중 하나로 뉴런을 참고하여 만듦
- 현재 딥러닝의 기원
- 여러 개의 입력을 받을 수 있고, 하나의 출력을 내보낼 수 있음
- 다음의 뉴런으로 정보를 전달할지 여부를 결정함



인공뉴런



인공신경망

3) 퍼셉트론(인공뉴런)

(1) 퍼셉트론(인공뉴런)

$$y = \begin{cases} 0 & (w_1 x_1 + w_2 x_2 \le \theta) \\ 1 & (w_1 x_1 + w_2 x_2 > \theta) \end{cases}$$

: 흐르지 않는다

: 흐른다

출력

입력

가중치

가중치가 클수록?

- 입력이 결과에 많은 영향을 미침
- 입력이 전체 인공신경망에서 중요한 역할을 함

▪ θ를 넘겨 식을 정리하고 일반화하면…

입력 * 가중치 + 편향 > 0	: 흐른다
입력 * 가중치 + 편향 ≤ 0	: 흐르지 않는다

2. 논리회로와 다층 퍼셉트론

1) 퍼셉트론으로 구현하는 논리회로

(1) 논리회로 : 입력 2개(x1, x2), 출력 1개(1 또는 0)

- 퍼셉트론으로 구현 가능

x_1	x_2	y
0	0	0
1	0	0
0	1	0
1	1	1

AND

x_1	x_2	y
0	0	1
1	0	1
0	1	1
1	1	0

NAND

x_1	x_2	y
0	0	0
1	0	1
0	1	1
1	1	1

OR

- x1 * w1 + x2 * w2 + b > 0 을 만족하는 w1, w2 , b 찾기

AND : [0.5,0.5,-0.7], [0.6,0.7,-0.8], [0.3,0.3,-0.5], [1.0,1.0, -1.5] …

NAND : [-0.2,-0.2,0.3] …

OR : [0.7,0.7,-0.4] …

2. 논리회로와 다층 퍼셉트론

1) 퍼셉트론으로 구현하는 논리회로

(2) numpy 모듈을 활용한 파이썬으로 논리회로 구현

```
import numpy as np

def AND(x1, x2):
    x = np.array([x1, x2])
    w = np.array([0.5, 0.5])
    b = -0.7
    tmp = np.sum(w*x) + b
    if tmp <= 0:
        return 0
    else:
        return 1

print(AND(0,0))
print(AND(1,0))
print(AND(0,1))
print(AND(1,1))
```

```
0
0
0
1
```

AND

```
import numpy as np

def NAND(x1, x2):
    x = np.array([x1, x2])
    w = np.array([-0.2, -0.2])
    b = 0.3
    tmp = np.sum(w*x) + b
    if tmp <= 0:
        return 0
    else:
        return 1

print(NAND(0,0))
print(NAND(1,0))
print(NAND(0,1))
print(NAND(1,1))
```

```
1
1
1
0
```

NAND

```
import numpy as np

def OR(x1, x2):
    x = np.array([x1, x2])
    w = np.array([0.7, 0.7])
    b = -0.1
    tmp = np.sum(w*x) + b
    if tmp <= 0:
        return 0
    else:
        return 1

print(OR(0,0))
print(OR(1,0))
print(OR(0,1))
print(OR(1,1))
```

```
0
1
1
1
```

OR

(3) 학습이 아닌 무작위로 컴퓨터에게 만족하는 w, b 값 찾기

```
import numpy as np

def check(x1,x2,y):
    while True:
        cnt = 0
        w1 = np.random.normal()
        w2 = np.random.normal()
        b = np.random.normal()
        for i in range(4):
            x = np.array([x1[i], x2[i]])
            w = np.array([w1, w2])
            tmp = np.sum(w*x) + b
            if tmp <= 0:
                result = 0
            else:
                result = 1
            if y[i] == result:
                cnt +=1

        if cnt == 4:
            print('w1: ',w1)
            print('w2: ',w2)
            print('b: ',b)
            break
```

```
def AND():
    x1 = [0,1,0,1]
    x2 = [0,0,1,1]
    y  = [0,0,0,1]
    check(x1,x2,y)

def NAND():
    x1 = [0,1,0,1]
    x2 = [0,0,1,1]
    y  = [1,1,1,0]
    check(x1,x2,y)

def OR():
    x1 = [0,1,0,1]
    x2 = [0,0,1,1]
    y  = [0,1,1,1]
    check(x1,x2,y)
```

```
print('AND')
AND()
print()
print('NAND')
NAND()
print()
print('OR')
OR()
```

```
AND
w1:  0.6585951921435629
w2:  0.7835084762907474
b:  -0.856721503693574

NAND
w1:  -0.5015912811236347
w2:  -1.5247927756075694
b:  2.0173844147271445

OR
w1:  0.20221577635223334
w2:  2.2775719424387733
b:  -0.04371535274123168
```

2. 논리회로와 다층 퍼셉트론

2) 단층 퍼셉트론의 한계

(1) XOR 게이트 구현하기

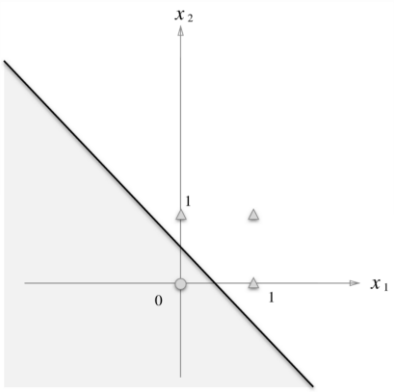
- 만족하는 w1, w2, b 찾기

x_1	x_2	y
0	0	0
1	0	1
0	1	1
1	1	0

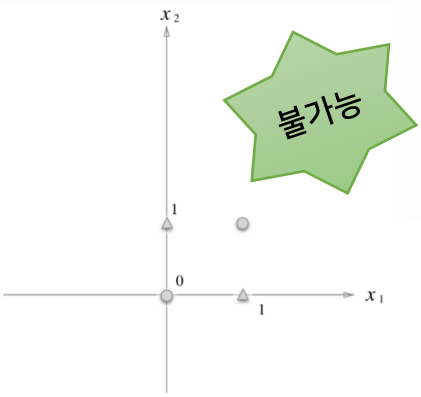
```
In [*]: def XOR():
            x1 = [0,1,0,1]
            x2 = [0,0,1,1]
            y  = [0,1,1,0]
            check(x1,x2,y)

        print('XOR')
        XOR()
```

x1 * w1 + x2 * w2 + b > 0 → 1
x1 * w1 + x2 * w2 + b ≤ 0 → 0

(2) 동그라미와 세모를 분리할 수 있는 하나의 직선 그리기



OR



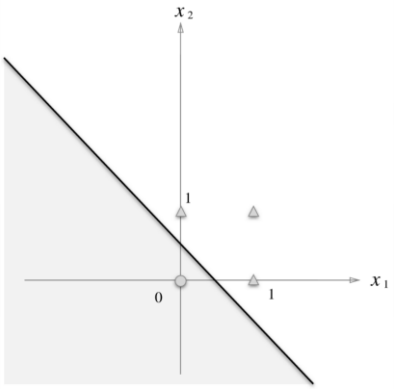
XOR

- 단층 퍼셉트론은 1차함수이고 직선으로만 표현 가능함

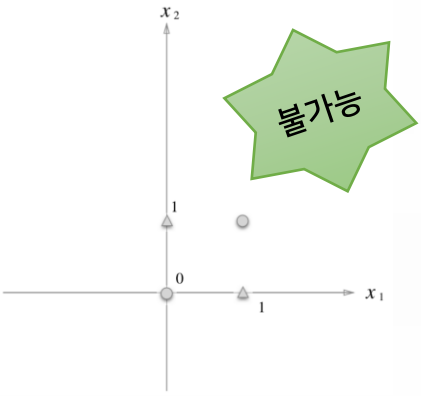
2. 논리회로와 다층 퍼셉트론

2) 단층 퍼셉트론의 한계

(2) 동그라미와 세모를 분리할 수 있는 하나의 직선 그리기



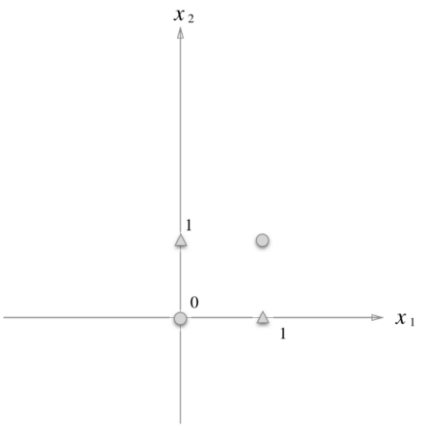
OR

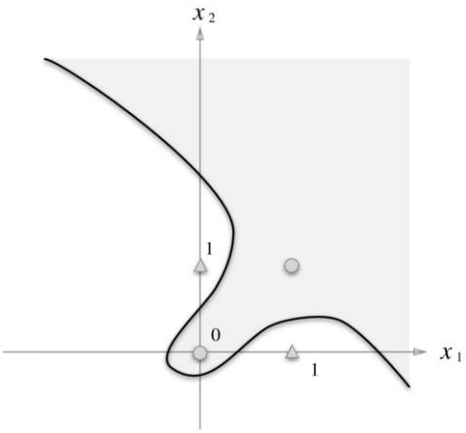


XOR

- 단층 퍼셉트론은 1차함수이고 직선으로만 표현 가능함

(3) 한 개의 퍼셉트론으로 표현 불가능

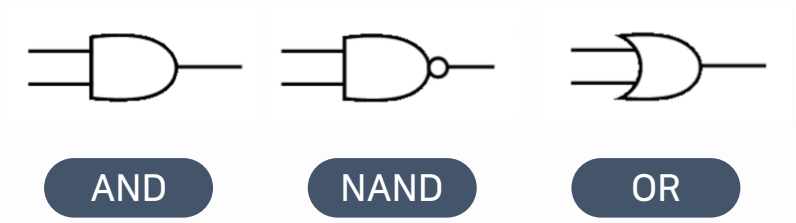




3) 다층 퍼셉트론

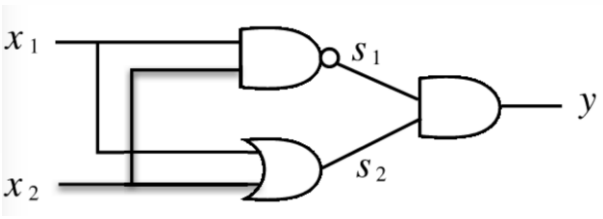
(1) XOR 게이트 만들기

- 기존 AND, OR, NAND 를 조합하여 생성



x_1	x_2	y
0	0	0
1	0	1
0	1	1
1	1	0

- 0층 : 입력 x1, x2
- 1층 : NAND, OR 계산
- 2층 : 1층의 결과를 입력으로 AND



x_1	x_2	s_1	s_2	y
0	0	1	0	0
1	0	1	1	1
0	1	1	1	1
1	1	0	1	0

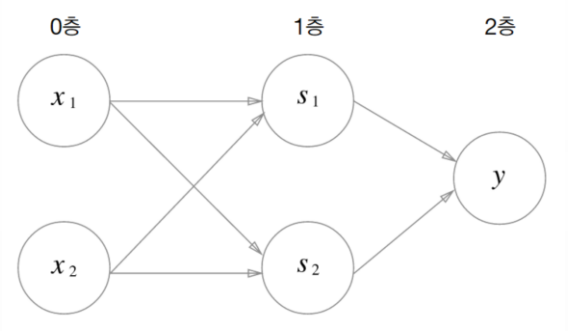
2. 논리회로와 다층 퍼셉트론

3) 다층 퍼셉트론

(2) XOR 게이트 만들기(파이썬 코드)

```
def XOR(x1, x2):
    s1 = NAND(x1, x2)
    s2 = OR(x1, x2)
    y = AND(s1, s2)
    return y
input_data = [(0,0),(1,0),(0,1),(1,1)]

print('XOR')
for i in input_data:
    print(XOR(i[0],i[1]))
```

```
XOR
0
1
1
0
```



(3) 논리회로 퍼셉트론으로 구현

AND

NAND

OR

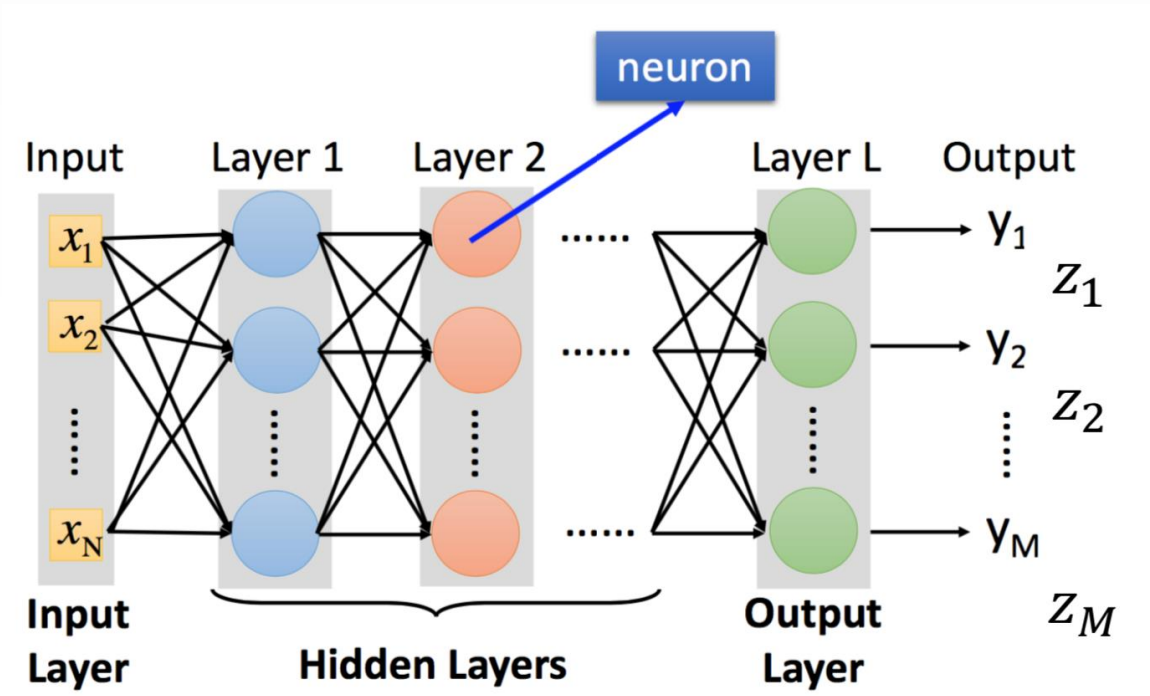
4) 인공신경망과 딥러닝

(1) 인공신경망과 딥러닝

- 실제 일상생활에선 논리회로(AND, OR 등), 퍼셉트론으로 해결할 수 있는 문제는 거의 없음
- 더 복잡한 문제를 해결하기 위해 층을 쌓다보니 인간의 뇌처럼 복잡한 인공신경망이 탄생함

(2) 딥러닝

- Deep은 '깊다' 라는 뜻으로 인공신경망의 층이 '아주 깊게 쌓여 있다' 라는 것에서 유래됨



Summary

1. 인공신경망과 퍼셉트론

- 인공신경망은 기계학습 알고리즘 중 하나로 인간의 신경계를 추상화하여 만든 알고리즘을 의미함
- 실제 인간의 신경계의 뉴런이 신호를 전달하는 것과 비슷하게 인공 뉴런(퍼셉트론)을 통해 입력과 출력을 구성했음
- 퍼셉트론
  - ✓ 여러 개의 입력과 하나의 출력을 가지고 있으며, 각 입력은 출력에 영향을 미치는 가중치가 곱해짐
  - ✓ 특정 임계 값을 넘으면 1 넘지 못하면 0으로 설정해 인간의 뉴런과 비슷한 역할을 하도록 함

Summary

2. 논리회로와 다층 퍼셉트론

- 논리회로(AND,OR,NAND)를 하나의 퍼셉트론으로 구현할 수 있음(두 개의 입력과 하나의 출력)
- XOR는 하나의 퍼셉트론으로 표현할 수 없고, 층을 쌓아 여러 개의 논리회로를 사용하여 구현할 수 있음
- 이처럼 복잡한 문제는 여러 층을 쌓아 다층 퍼셉트론으로 해결할 수 있음
- 점점 더 어려운 문제를 해결하기 위해 층을 쌓다보니 오늘 날 우리가 이야기하는 딥 러닝이 되었음