2022학년도 1학기 컴퓨터언어학

제7강 신경망 언어 모형 (1)

박수지

서울대학교 인문대학 언어학과

2022년 3월 28일 월요일

오늘의 목표

1. “deep learning”에서 “deep”의 의미를 설명할 수 있다.
2. 인공 신경망을 이루는 계산 단위를 해독하고 계산할 수 있다.
3. 활성화함수의 정의를 설명하고 예시를 들 수 있다.
4. 다층 퍼셉트론이 필요한 이유를 설명할 수 있다.

인공 신경망(ANN: Artificial Neural Network)

순방향 신경망(FFNN/FNN: Feed-forward Neural Network)

1. 계산 단위(Computing/Computational Units)
   - 선형 함수(Linear function): 가중치(weights) 행렬 & 편향(bias) 벡터
   - 활성화 함수(Activation function): 비선형 변환
     - logistic(sigmoid), softmax, ReLU, tanh, ⋯
2. Shallow learning vs. Deep learning

   로지스틱회귀 입력층 → 출력층

   신경망분류기 입력층 → 은닉층(hidden layer) → 출력층
3. 은닉층 도입의 필요성
   - 퍼셉트론(Perceptron)
   - XOR 문제

복습: 로지스틱(이항) 회귀분석

입력 $\mapsto$ 출력

$$\vec{x} \longmapsto y = \sigma\left(\vec{w} \cdot \vec{x} + b\right)$$

복습: 소프트맥스(다항) 회귀분석

입력 $\mapsto$ 출력

$$\vec{x} \longmapsto \vec{y} = \mathrm{softmax}\left(\mathbf{W} \cdot \vec{x} + \vec{b}\right)$$

사실

출력은 입력에 대한 선형함수와 분류함수의 합성함수이다.

입력 $\vec{x} \in \mathbb{R}^{f}$ (f개의 특성값)

선형함수 $\vec{z} = \mathbf{W} \cdot \vec{x} + \vec{b} \in \mathbb{R}^{K}$

분류함수 $\vec{y} = \mathrm{softmax}(\vec{z}) \in \mathbb{R}^{K}$ (K개의 확률)

연습: 위 식에서 행렬 $\mathbf{W}$의 크기는 어떻게 되는가?

계산 단위

$$\text{입력} \cdots \vec{x} \xmapsto[\text{linear function}]{\text{선형함수}} z = \vec{w}\cdot\vec{x} + b \xmapsto[\text{activation function}]{\text{활성화함수}} y = a = f(z) \cdots \text{출력}$$

참고: 편향 b를 쓰기 귀찮을 때

입력값이 $\vec{x} = (x_1, x_2, x_3)$ 이고 가중치가 $\vec{w} = (w_1, w_2, w_3)$, 편향이 b일 때
$\vec{x}' = (x_1, x_2, x_3, 1), \vec{w}' = (w_1, w_2, w_3, b)$ 로 놓으면 다음이 성립한다.

$$\begin{aligned}\vec{w}\cdot\vec{x} + b &= w_1x_1 + w_2x_2 + w_3x_3 + b \\ &= w_1x_1 + w_2x_2 + w_3x_3 + 1b \\ &= \vec{w}'\cdot\vec{x}'\end{aligned}$$

편향은 가중치에 포함시킬 수 있으므로 편의에 따라 생략해도 된다.

활성화함수(activation function)

선형함숫값 z에 적용하는 비선형함수 f

활성홧값 $a = f(z)$

주요 활성화함수

1. 로지스틱 함수 $y = \dfrac{1}{1 + \exp(-z)}$
2. ReLU(Rectified Linear Unit) 함수 $y = \max(z, 0)$
3. tanh(hyperbolic tangent) 함수 $y = \dfrac{\exp(z) - \exp(-z)}{\exp(z) + \exp(-z)}$

주요 활성화함수의 그래프

로지스틱(시그모이드) 함수

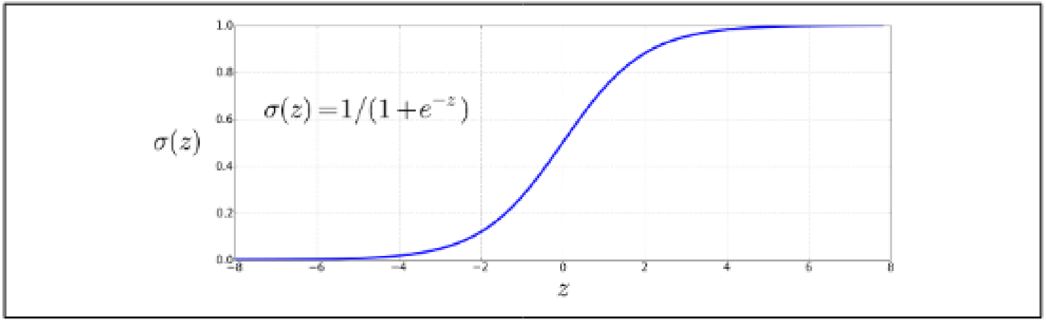


Figure 7.1 The sigmoid function takes a real value and maps it to the range $[0, 1]$. It is nearly linear around 0 but outlier values get squashed toward 0 or 1.

주요 활성화함수의 그래프

tanh 및 ReLU 함수

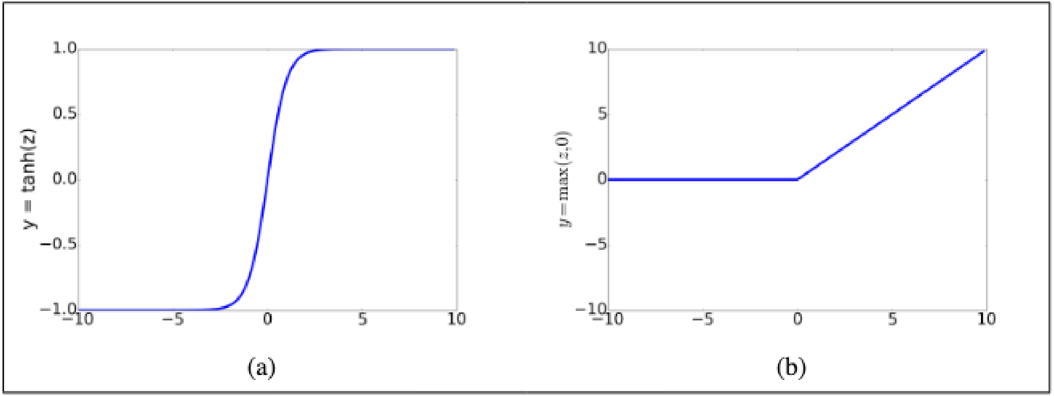


Figure 7.3 The tanh and ReLU activation functions.

신경 계산 단위의 도식

활성화함수가 로지스틱 함수인 경우

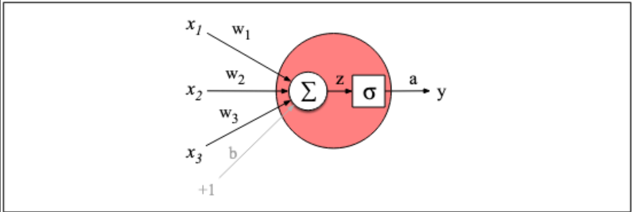


Figure 7.2 A neural unit, taking 3 inputs x_1, x_2, and x_3 (and a bias b that we represent as a weight for an input clamped at +1) and producing an output y. We include some convenient intermediate variables: the output of the summation, z, and the output of the sigmoid, a. In this case the output of the unit y is the same as a, but in deeper networks we'll reserve y to mean the final output of the entire network, leaving a as the activation of an individual node.

입력 $\vec{x} = (x_1, x_2, x_3, 1)$

가중치 $\vec{w} = (w_1, w_2, w_3, b)$

가중합 $z = \vec{w} \cdot \vec{x}$

활성홧값 $a = \sigma(z)$

"Shallow" vs. "Deep" learning

첫번째 활성홧값 $a = f(z)$ 가…

- 곧바로 출력 y가 된다. …Shallow learning
- 새로운 계산 단위의 입력으로 들어간다. …Deep learning

의문: 계산 단위를 굳이 여러 개 쌓을 필요가 있는가?

심층 신경망의 필요성

단층 신경망으로는 XOR 문제를 풀 수 없다.

퍼셉트론(Perceptron)

출력 y의 값이 0 또는 1이고 비선형 활성화함수를 가지지 않는 신경 단위

$$y = \begin{cases} 0, & \text{if } \vec{w} \cdot \vec{x} + b \leq 0 \\ 1, & \text{if } \vec{w} \cdot \vec{x} + b > 0 \end{cases}$$

목표

(단층)퍼셉트론으로 간단한 진리함수를 구현해 보자!

진리함수

AND

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

OR

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

XOR

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

사실

논리함수 AND와 OR는 단층퍼셉트론으로 구현할 수 있다.

논리함수 AND와 OR의 퍼셉트론

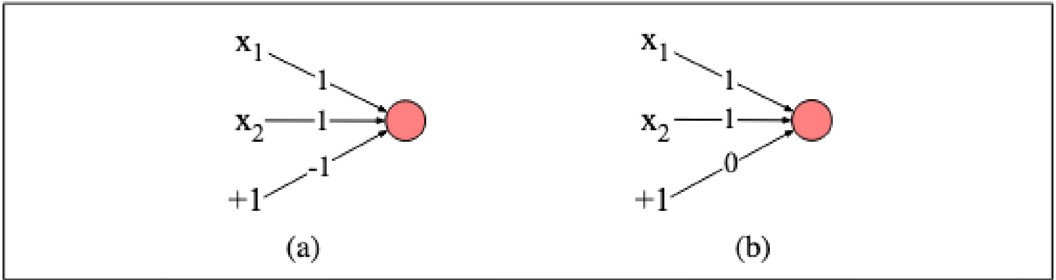


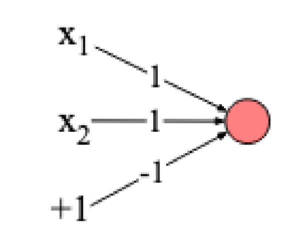
Figure 7.4 The weights w and bias b for perceptrons for computing logical functions. The inputs are shown as x_1 and x_2 and the bias as a special node with value $+1$ which is multiplied with the bias weight b. (a) logical AND, showing weights $w_1 = 1$ and $w_2 = 1$ and bias weight $b = -1$. (b) logical OR, showing weights $w_1 = 1$ and $w_2 = 1$ and bias weight $b = 0$. These weights/biases are just one from an infinite number of possible sets of weights and biases that would implement the functions.

논리함수 AND의 퍼셉트론

매개변수 $\vec{w} = (w_1, w_2) = (1, 1), b = -1$

가중합 $z = \vec{w} \cdot \vec{x} + b = w_1 x_1 + w_2 x_2 + b = x_1 + x_2 - 1$

출력 $y = \begin{cases} 0, & \text{if } x_1 + x_2 - 1 \le 0 \\ 1, & \text{if } x_1 + x_2 - 1 > 0 \end{cases}$



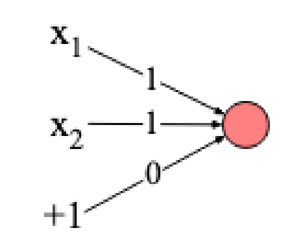
입력값		가중합	출력값
x_1	x_2	$x_1 + x_2 - 1$	y
0	0	$0 + 0 - 1 = -1$	0
0	1	$0 + 1 - 1 = 0$	0
1	0	$1 + 0 - 1 = 0$	0
1	1	$1 + 1 - 1 = 1$	1

논리함수 OR의 퍼셉트론

매개변수 $\vec{w} = (w_1, w_2) = (1, 1), b = 0$

가중합 $z = \vec{w} \cdot \vec{x} + b = w_1x_1 + w_2x_2 + b = x_1 + x_2$

출력 $y = \begin{cases} 0, & \text{if } x_1 + x_2 \leq 0 \\ 1, & \text{if } x_1 + x_2 > 0 \end{cases}$



입력값		가중합	출력값
x_1	x_2	$x_1 + x_2$	y
0	0	$0 + 0 = 0$	0
0	1	$0 + 1 = 1$	1
1	0	$1 + 0 = 1$	1
1	1	$1 + 1 = 2$	1

관찰: 퍼셉트론은 선형 분류기와 같다.

예시: $\vec{x} = (x_1, x_2)$ 일 때

$$\vec{w} \cdot \vec{x} + b = 0 \iff w_1 x_1 + w_2 x_2 + b = 0 \iff x_2 = -\frac{w_1}{w_2} x_1 - \frac{b}{w_2}$$

그러므로 아래의 동치 관계가 성립한다.

1. $y = 0 \cdots$ 입력 $\vec{x} = (x_1, x_2)$ 가 0으로 분류된다.
2. $\vec{w} \cdot \vec{x} + b \leq 0$ (퍼셉트론의 정의에 의해 성립)
3. $x_2 \leq -\frac{w_1}{w_2} x_1 - \frac{b}{w_2} \cdots (x_1, x_2)$ 가 좌표평면에서 결정 경계 직선보다 아래에 있다.

(단층)퍼셉트론으로 구현 가능하다.
$\Leftrightarrow$ 입력값에 대응하는 점들을 분류하는 결정 경계 직선(평면, 초평면)이 존재한다.

문제: 단층 퍼셉트론은 XOR 함수를 표현할 수 없다.

결정 경계 직선을 그을 수 없다.

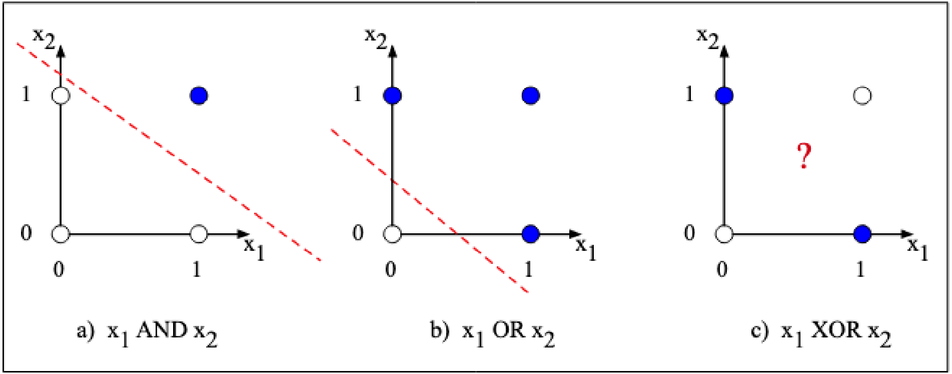


Figure 7.5 The functions AND, OR, and XOR, represented with input x_1 on the x-axis and input x_2 on the y axis. Filled circles represent perceptron outputs of 1, and white circles perceptron outputs of 0. There is no way to draw a line that correctly separates the two categories for XOR. Figure styled after Russell and Norvig (2002).

해결책: 신경망

입력층과 출력층 사이에 은닉층을 도입한다.

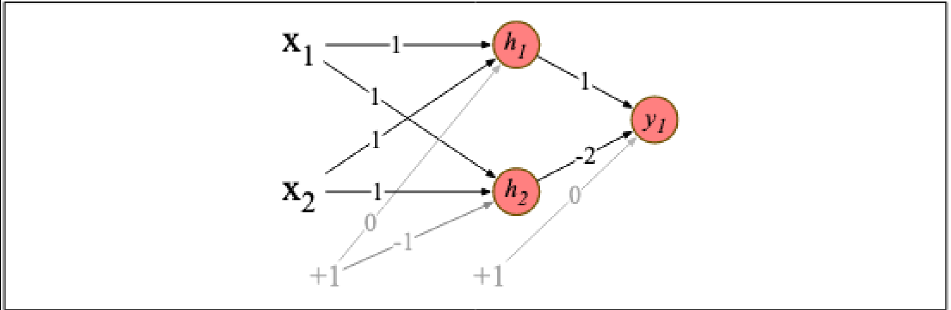
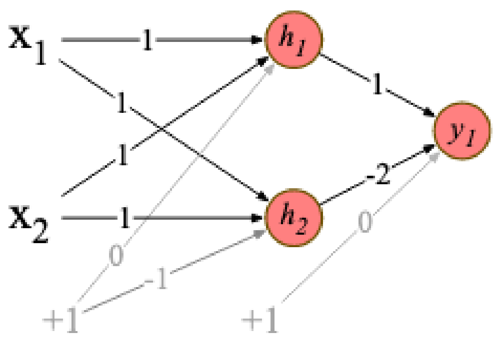


Figure 7.6 XOR solution after Goodfellow et al. (2016). There are three ReLU units, in two layers; we've called them h_1, h_2 (h for "hidden layer") and y_1. As before, the numbers on the arrows represent the weights w for each unit, and we represent the bias b as a weight on a unit clamped to +1, with the bias weights/units in gray.

크기가 2인 은닉층이 1개 있는 신경망



은닉층의 크기와 개수는 초매개변수에 해당한다.

XOR의 은닉층 (활성화함수 ReLU)

입력	$\vec{x} = (x_1, x_2)$
가중합1	$z_1 = x_1 + x_2 + 0$
활성홧값1	$h_1 = \mathrm{ReLU}(z_1)$
가중합2	$z_2 = x_1 + x_2 - 1$
활성홧값2	$h_2 = \mathrm{ReLU}(z_2)$

XOR의 출력층 (활성화함수 ReLU)

입력	$\vec{h} = (h_1, h_2)$
가중합	$z_3 = h_1 - 2h_2 + 0$
활성홧값	$y_1 = \mathrm{ReLU}(z_3)$

논리함수 XOR의 다층 신경망

입력값		가중합1	활성홧값1	가중합2	활성홧값2
x_1	x_2	$z_1 = x_1 + x_2$	$h_1 = \mathrm{ReLU}(z_1)$	$z_2 = x_1 + x_2 - 1$	$h_2 = \mathrm{ReLU}(z_2)$
0	0	$0+0=0$	0	$0+0-1=-1$	0
0	1	$0+1=1$	1	$0+1-1=0$	0
1	0	$1+0=1$	1	$1+0-1=0$	0
1	1	$1+1=2$	2	$1+1-1=1$	1

(입력값)		$\cdots$	은닉값		가중합	출력값
$(x_1$	$x_2)$	$\cdots$	h_1	h_2	$z_3 = h_1 - 2h_2$	$y_1 = \mathrm{ReLU}(z_3)$
(0	0)	$\cdots$	0	0	$0-0=0$	0
(0	1)	$\cdots$	1	0	$1-0=1$	1
(1	0)	$\cdots$	1	0	$1-0=1$	1
(1	1)	$\cdots$	2	1	$2-2=0$	0

논리함수 XOR의 다층 신경망

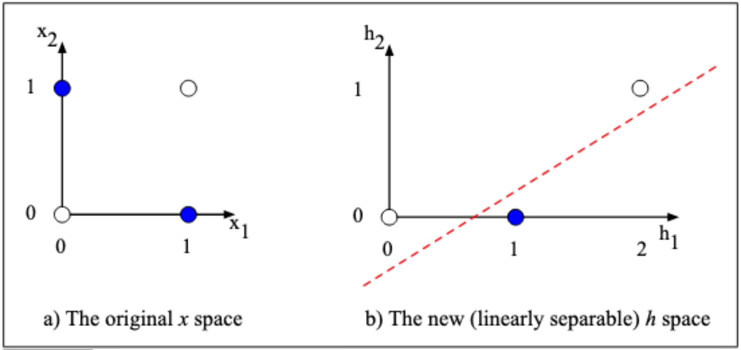


Figure 7.7 The hidden layer forming a new representation of the input. (b) shows the representation of the hidden layer, **h**, compared to the original input representation **x** in (a). Notice that the input point [0, 1] has been collapsed with the input point [1, 0], making it possible to linearly separate the positive and negative cases of XOR. After Goodfellow et al. (2016).

은닉층의 역할

입력값의 새로운 표상

$$
\begin{aligned}
\vec{x} &\mapsto \vec{h} &&\mapsto y \\
(0,0) &\mapsto (0,0) &&\mapsto 0 \\
(0,1) &\mapsto (1,0) &&\mapsto 1 \\
(1,0) &\mapsto (1,0) &&\mapsto 1 \\
(1,1) &\mapsto (2,1) &&\mapsto 0
\end{aligned}
$$

오늘 배운 것

- 계산 단위
- 활성화함수
- (단층)퍼셉트론
- (다층)신경망

이번 주의 남은 일

- SLP3 Sec.7.3 읽기