6 주차 조별보고서 (Default)

작성일: 2019 년 10 월 11 일	작성자: 이충현
조 모임 일시: 2019 년 10 월 11 일 9 교시	모임장소: 학교 앞 카페
참석자: 위성조, 이충현, 최진성, 이재은, 김영연	조원: 위성조, 이충현, 최진성, 이재은, 김영연

구 분	내 용
학습 범위와 내용	기계학습 3장 3.1절 신경망의 역사와 종류 3.2절 퍼셉트론의 동작, 학습 알고리즘 3.3 절 다층 퍼셉트론
논의 내용	**<과제: 하이퍼볼릭 탄젠트, 로지스틱 시그모이드 함수의 도함수 도출과정>** **1. 하이퍼볼릭 탄젠트의 도함수 추출** $\mathrm{f(x)} = \mathrm{tanh(x)}$ $= \dfrac{\mathrm{sinh\ (x)}}{\mathrm{cosh\ (x)}}$ $* \ \left(\dfrac{f}{g}\right)' = \dfrac{fg' - gf'}{g^2}$ $* \ \mathrm{sinh(x)''} = \mathrm{cosh(x)'} = \mathrm{sinh(x)}$ $* \ cosh^2(x) - sinh^2(x) = 1$

f'(x) = tanh'(x)

$$= \left(\frac{\sinh(x)}{\cosh(x)}\right)' = \left(\frac{\cosh(x)*\cosh(x)-\sinh(x)*\sinh(x)}{\cosh(x)^2}\right) = \frac{cosh^2(x)- sinh^2(x)}{\cosh^2(x)}$$

$$= \frac{1}{cosh^2(x)}$$

$$= sech^2(x)$$

2. 로지스틱 시그모이드의 도함수 추출

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

$$\frac{d}{dx}\mathrm{Sigmoid}(x) = \frac{d}{dx}(1 + e^{-x})^{-1}$$

$$= (-1)\frac{1}{(1+e^{-x})^2}\frac{d}{dx}(1 + e^{-x})$$

$$= (-1)\frac{1}{(1+e^{-x})^2}e^{-x}\frac{d}{dx}(-x)$$

$$= (-1)\frac{1}{(1+e^{-x})^2}-e^{-x}$$

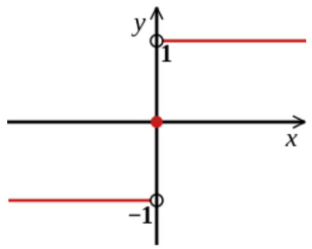
$$= \frac{e^{-x}}{(1+e^{-x})^2} = \frac{1+e^{-x}-1}{(1+e^{-x})^2} = \frac{1+e^{-x}}{(1+e^{-x})^2} - \frac{1}{(1+e^{-x})^2}$$

$$= \frac{1}{1+e^{-x}} - \frac{1}{(1+e^{-x})^2} = \frac{1}{1+e^{-x}}\left(1 - \frac{1}{1+e^{-x}}\right)$$

= sigmoid(x)(1-sigmoid(x))

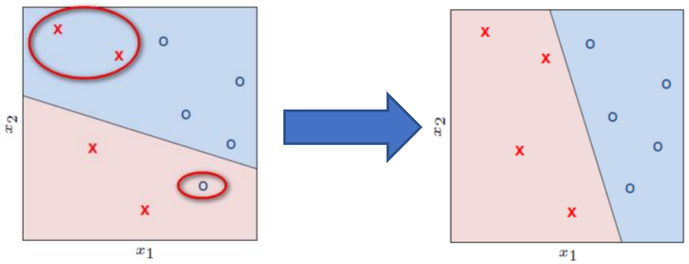
<과제: 그레이디언트 계산>

퍼셉트론은 주어진 값을 토대로 어떤 것에 대한 binary output을 내보낼 때 쓰이기 때문에 sign function을 사용한다. Sign function은 x의 입력에 따라 y가 1 또는 -1의 형태로 출력되는 것을 말한다.



이를 이용하면 $\mathrm{h(x)} = sign\left(\left(\sum_{i=1}^{d} w_i x_i\right) + bias\right) = sign(W^T X)$이 된다.

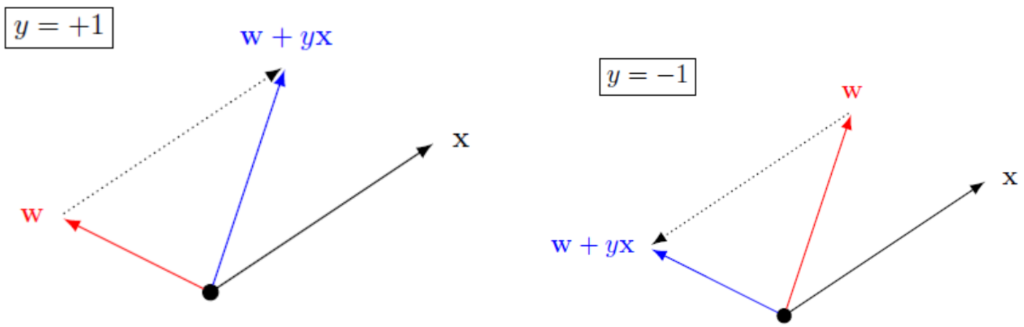
위의 식을 이용하여 가설을 수정하여 다음과 같이 바꿀 수 있어야 한다.



퍼셉트론에서 잘못 분류된 데이터에 대해 제대로 분류할 수 있도록 선형분류기의 방향을 틀어줘야 하는데

$sign(W^T X) \neq y_n$일 때, misclassified point를 찾아 델타규칙 w=w+y(n)x(n)에 대입하여 값을 업데이트 하면 된다.

Y의 값이 -1과 1일 때의 그림은 다음과 같다.



논의 내용	**<조원들의 질문>** Q1. 진성: Sigmoid 함수의 문제점이 발견된 이후 딥러닝에선 ReLU 함수와 같이 Softplus 나 Recrifier 함수를 채용하였는데, 그렇다면 기존의 다층 퍼셉트론에서는 계속 Sigmoid 활성 함수를 사용하는건가? 사용한다면 왜인가? A1. 신경망 초기에는 sigmoid가 많이 사용되었지만, 최근에는 1) Gradient Vanishing 현상이 발생한다. 2) 함수 값 중심이 0이 아니다. 3) exp 함수 사용시 비용이 크다. 등의 단점등으로 인하여 거의 사용하지 않는다. 최근 가장 많이 사용하는 함수는 간단하고 사용이 쉬운 ReLU 함수이다. Q2. 재은: 퍼셉트론에서 바이어스항을 추가하는 이유는 무엇인가요? A2. 바이어스가 없는 퍼셉트론의 경우, 계수를 변경하며 그래프의 회전은 가능하지만 수평이동이 불가능하여, x=0일 때, 결과값이 2인 점을 구분하는 문제 등을 해결할 수 없다. 이러한 문제들을 해결하기 위해, 바이어스항을 이용한다.
질문 내용	조원들이 질문하였던 것을 모두 해결하였기 때문에, 이번 주차는 질문 내용이 없습니다.

기타	

<첨부 개인 레포트>

-201402033 위성조

구분	내용
학습 범위	신경망의 역사와 종류 퍼셉트론 다층 퍼셉트론
학습 내용	1958년 로젠블렛의 퍼셉트론 제안 1986년 루멜하트의 다층 퍼셉트론 제안 2000년대 딥 러닝이 기계학습의 주류 기술로 자리매김 신경망의 종류 전방 신경망과 순환 신경망 얕은 신경망과 깊은 신경망 결정론 신경망과 스토캐스틱 신경망 퍼셉트론의 구조 입력층과 출력층을 가짐 – 입력층은 연산을 하지 않으므로 퍼셉트론은 단일 층 구조라고 간주 i번째 입력층 노드와 출력층을 연결하는 에지는 가중치 w_i를 가짐

	계단함수를 활성화함수로 이용. 퍼셉트론의 동작 해당하는 특징값과 가중치를 곱한 결과를 모두 더하여 s를 구하고, 활성함수 t를 적용함 – 활성함수 t로 계단함수를 사용하므로, 최종 출력 y는 +1 또는 -1 결정 직선은 전체 공간을 +1과 -1의 두 부분공간으로 분할하는 분류기 역할 d차원 공간에서는 d(x) = w1x1+w2x2+…+wdxd+w0 = 0 2차원은 결정 직선, 3차원은 결정 평면, 4차원 이상은 결정 초평면 학습 - 결정 직선 혹은 결정 평면의 파라메터 w1, w2,…,wd의 값을 결정하기 위한 과정 목적함수의 조건 – w가 최적이면, 즉 모든 샘플을 맞히면 J(w) = 0이다 틀리는 샘플이 많은 w일수록 J(w)는 큰 값을 가진다 다층 퍼셉트론 퍼셉트론은 선형 분류기라는 한계를 가지고 있어서, 선형 분리 불가능한 상황에서는 항상 일정한 양의 오류를 가지고 있다. Ex) XOR 문제에서는 75%가 정확률 한계 ➔ 1986년 루멜하트의 저서에서 다층 퍼셉트론 이론이 정립됨. 다층 퍼셉트론의 핵심 아이디어 - 은닉층을 둔다 - 활성함수로 시그모이드 함수를 사용한다 - 오류 역전파 알고리즘을 사용한다. 활성함수 계단함수는 딱딱한 의사결정(영역을 점으로 변환), 나머지 활성함수(로지스틱 시그모이드, 하이퍼볼릭 탄젠트 시그모이드,

	softplus, recrifier)는 부드러운 의사결정(영역을 영역으로 변환)한다.
질문 내용	

-201402665 이충현

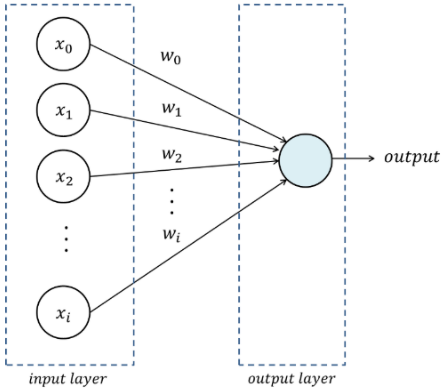
구분	**내용**
학습 범위	기계학습 3장 3.1절 신경망의 역사와 종류 3.2절 퍼셉트론의 동작, 학습 알고리즘 3.3절 다층 퍼셉트론
학습 내용	<3.1절 신경망의 역사와 종류> ● 1958년 로젠블렛이 '퍼셉트론'제안, 1986년 '루멜하트'는 다층 퍼셉트론 제안하므로서 2000년대 '딥러닝'이 실현되어 신경망이 기계 학습의 주류 기술로 자리 매겨줬다. ● 신경망의 종류는 전방, 순환, 얕은, 깊은 신경망이 있다. 그 외에도 결정론 신경망과 스토캐스틱 신경망이 있다. <3.2절 퍼셉트론의 동작, 학습 알고리즘> ● 퍼셉트론은 노드, 가중치, 층과 같은 개념을 도입하고 학습 알고리즘을 만든다. 임계치(threshold)=어떠한 값이 활성화되기 위한 최소값. 가중치(weight)=선형 경계의 방향성 또는 형태를 나타내는 값. 바이어스(bias)=선형

Net값=입력값과 가중치의 곱을 모두 합한 값.

$$net = \sum_{i}^{N} w_i x_i + w_0 x_0$$

활성함수(activation function)=뉴런에서 계산된 net값>임계치이면 1을 출력 net값<임계치이면 0을 출력한다. 이 정의는 단층 퍼셉트론에만 유효하다.

뉴런=인공신경망을 구성하는 가장 작은 요소로써, net값이 임계치보다 크면 활성화되면서 1을 출력하고, 반대의 경우에는 비활성화되면서 0을 출력한다.



[그림 2] 출력층의 크기가 1인 단층 퍼셉트론

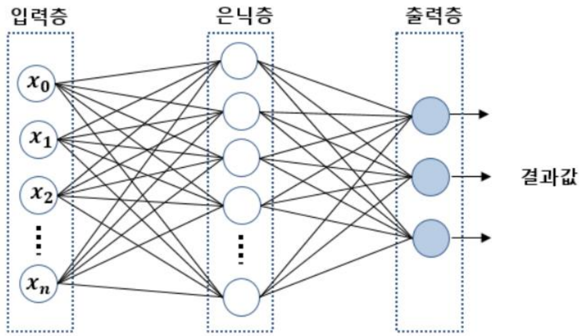
- 학습 연산의 정의는 출력층 뉴런의 출력값과 목표값의 차이가 허용오차보다 크면 출력층 뉴런의 가중치를 조정해야 한다.

$$w_i = w_i + \eta x_i (t - f(net))$$

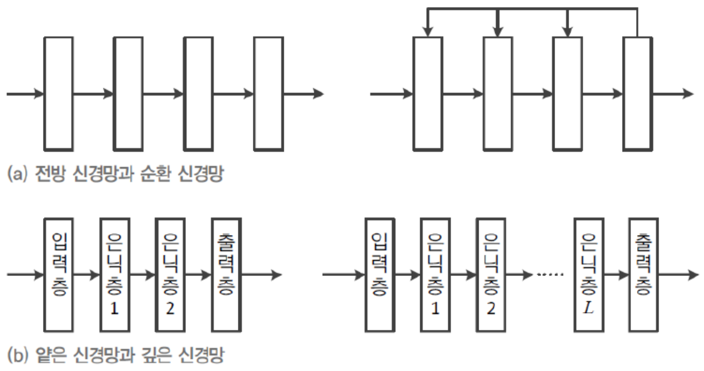
$$\eta = learning\ rate$$
$$t = target\ value$$

<3.3절 다층 퍼셉트론>

- 단층 퍼셉트론으로 XOR에 대해서는 학습이 불가능하다. 이를 극복하기 위한 방안으로 입력층과 출력층 사이에 하나 이상의 중간층(은닉층)을 두어 비선형적으로 분리되는 데이터에 대해서도 학습이 가능하도록 다층 퍼셉트론이

	고안되었다.  ● 그러나 XOR문제를 퍼셉트론을 여러 층을 쌓아 해결하고 역전파를 통해 학습의 문제를 해결했지만 더욱 복잡한 문제를 해결하기 위해 Hidden Layer를 늘려가는 도중 다음과 같은 문제가 발생한다. Overfitting=트레이닝 셋에 너무 과최적화 되어 실제 데이터를 집어넣어 분류를 하려고 하면 정확도가 좀 떨어지는 것=>정규화, 훈련 데이터 증가, 특징 개수 감소등으로 해결 Vanishing Gradient=layer가 deep해지면서 backpropagation으로 에러를 뒤로 전파하게 되는데 문제가 생긴다. 경사하강법을 통해 에러에 대한 값을 새로운 w에 업데이트 시켜줘야 하는데 각각의 레이어를 계속 거치면서 계속 경사하강법을 통해 미분하다 보면 뒤로 전해지면서 에러 값이 현저히 작아져 학습이 제대로 안된다=> 활성화 함수를 변경한다(Sigmoid함수을 Relu함수로 변경).
질문 내용	

-201403474 최진성

구분	내용
학습 범위	기계학습 3장 다층 퍼셉트론 3.1 신경망 기초 3.2 신경망의 간략한 역사 3.3 신경망의 종류
학습 내용	기계학습 3장, 다층 퍼셉트론 3.1 신경망 기초 신경망의 역사 – 로젠블렛의 퍼셉트론 제안, 이후 1960년에 퍼셉트론에 대한 기대감이 높아졌다가 당시 단층 퍼셉트론의 근본적인 한계가 밝혀진 이후 퇴보하게 됨. 이후 1986년 루멜하트가 다층 퍼셉트론을 제안을 계기로 신경망에 대한 연구 부활, 딥 러닝의 대두로 신경망 연구 또한 크게 진보하게 됨. 신경망의 종류 전방 신경망, 순환 신경망, 얇은 신경망, 깊은 신경망, 결정론 신경망, 스토캐스틱 신경망…  (a) 전방 신경망과 순환 신경망 (b) 얇은 신경망과 깊은 신경망

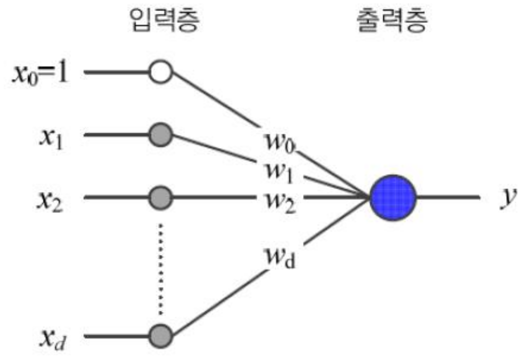
퍼셉트론

퍼셉트론은 이전의 지식 기반 학습과는 달리 새로운 개념들을 이용하여 학습 알고리즘을 창안하였다.

초기의 퍼셉트론은 원시적인 신경망이었으나 현대의 딥러닝에 쓰이는 퍼셉트론은 병렬구조와 순차구조를 결합하여 만듦

단층 퍼셉트론의 구조

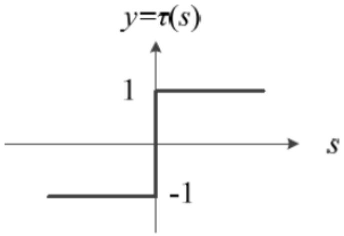
입력층 – 출력층으로 이루어져 있으며, 입력층은 연산의 대상이 아니기 때문에 단일 층 구조라고 간주한다.



퍼셉트론의 동작

해당하는 특징값과 가중치를 모두 곱한 결과(x0*w0+x1w1+...+xdwd)를 모두 더하여 s를 구하고 활성함수 r을 적용한다.

r(s)는 계단함수를 통해서 출력값을 조정하므로 최종 출력 값은 +-1



$$y = \tau(s)$$

$$\text{이때 } s = w_0 + \sum_{i=1}^{d} w_i x_i, \qquad \tau(s) = \begin{cases} 1 & s \geq 0 \\ -1 & s < 0 \end{cases}$$

*3.6의 학습의 경우에는 w1의 수치는 마이너스를, w2와 w3의 수치를 x2와 x3 이상으로 설정하면 된다.

목적함수

어떤 목적을 가진 설계에서 최적의 성능을 만족시키고자 하기 위해서 사용하는 함수. 기계학습에서는 샘플 분류에서 오류를 최소화하기 위해 사용한다.

$J(\mathbf{w})$로 표기하며 0에 가까울수록 틀리는 샘플이 적은 반면, 0에서 멀어질수록 오류가 많음을 의미한다.

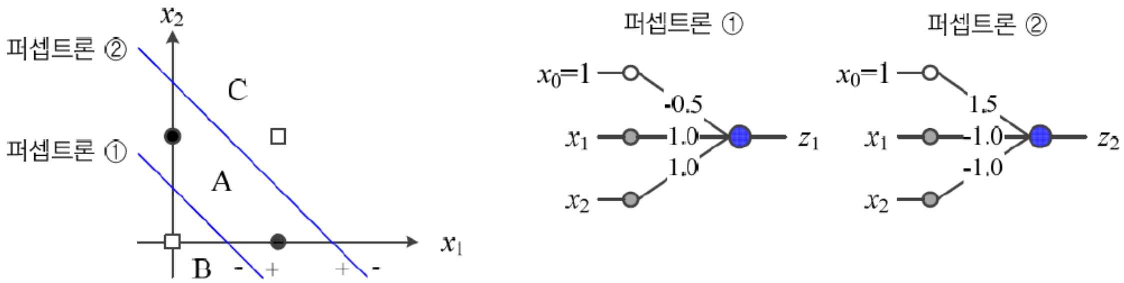
다층 퍼셉트론

입력층 – 은닉층 – 출력층으로 구성되어 있으며, 이 은닉층을 이용해 기존의 단층 퍼셉트론으로는 구분할 수 없었던 특징 공간을 훨씬 유리한 특징 공간으로 변환하여 구분한다.

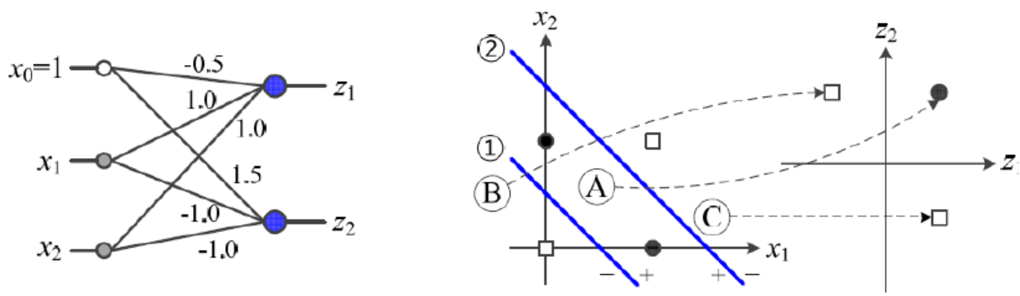
Sigmoid 활성함수를 도입하였었으나 이후 ReLU 함수를 도입하는 경우가 많아지고 있다.

오류 역전파 알고리즘을 사용하여 한 번에 한 층씩 Gradient를 계산, 가중치를 갱신한다.

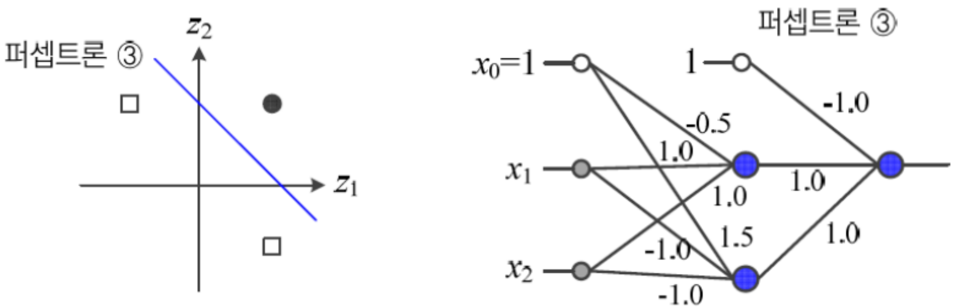
기존에 해결하지 못했던 XOR 문제를 해결하는 다층 퍼셉트론의 형태



● 2개의 퍼셉트론을 병렬로 연결했을 때의 다층 퍼셉트론의 형태



● 2개의 병렬 퍼셉트론에 1개의 순차 퍼셉트론을 연결했을 때의 다층 퍼셉트론의 형태(딥러닝)

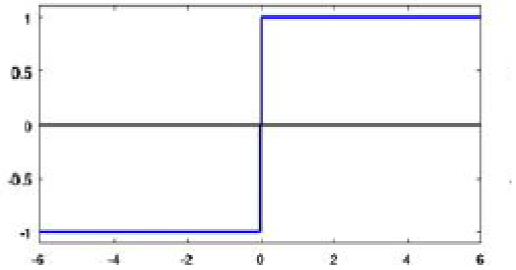


● 다층 퍼셉트론의 용량

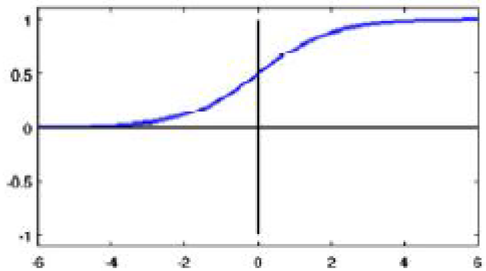
- $1 + \sum_{i=1}^{p} i$개의 영역으로 분할됨.
- 3일경우 1 + 1 + 2 + 3 = 7개의 영역

활성함수

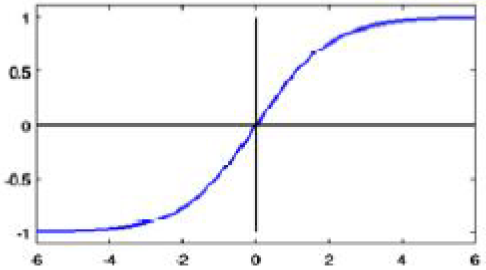
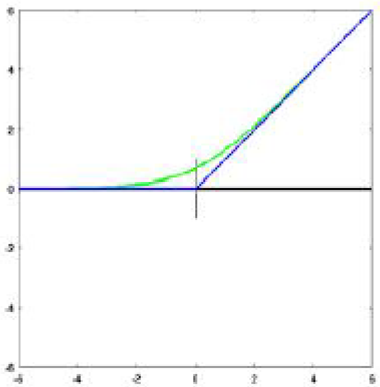
계단형 활성함수 : -1 or 1의 비연속적인 딱딱한 의사결정



Logistic Sigmoid : 초기에 사용되었던 Sigmoid. 연속적인 0 ~ 1 값의 부드러운 의사결정



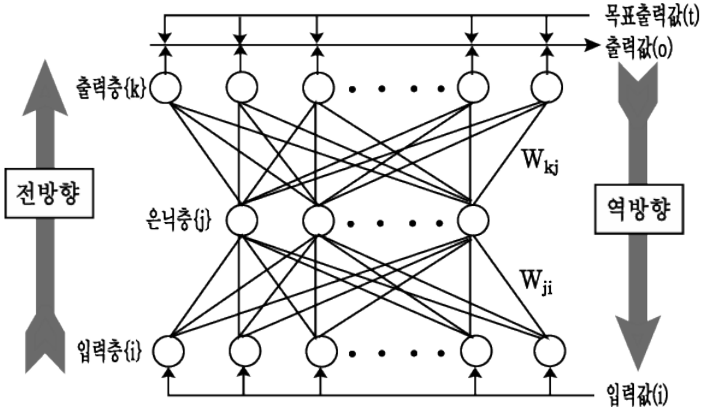
Hyperbolic Tangent Sigmoid : 기존의 Logistic Sigmoid의 단점을 개선하여 음수 결과까지 출력하는 부드러운 의사결정

	 Softplus, Recrifier : ReLU와 같이 음수일 때의 값은 통제하고 양수일 때의 값은 그대로 출력하거나 1차 함수와 같이 수렴하게 하는 부드러운 의사결정 
질문 내용	Sigmoid 함수의 문제점이 발견된 이후 딥러닝에선 ReLU 함수와 같이 Softplus나 Recrifier 함수를 채용하였는데, 그렇다면 기존의 다층 퍼셉트론에서는 계속 Sigmoid 활성 함수를 사용하는건가? 사용한다면 왜?

-201502469 이재은

구분	내용
학습 범위	3.1절: 신경망의 역사와 종류 소개 3.2절: 퍼셉트론의 구조와 동작, 학습 알고리즘 3.3절: 다층 퍼셉트론
학습 내용	- 퍼셉트론(perceptron): 인공신경망의 한 종류로서, 1957년에 코넬 항공 연구소(Cornell Aeronautical Lab)의 프랑크 로젠블라트(Frank Rosenblatt)에 의해 고안되었다. 이것은 가장 간단한 형태의 선형분류기로 볼 수 있다. - 퍼셉트론의 동작 방식: 각 노드의 가중치와 입력치를 곱한 것을 모두 합한 값이 활성함수에 의해 판단되는데, 그 값이 임계치(보통 0)보다 크면 뉴런이 활성화되고 결과값으로 1을 출력한다. 뉴런이 활성화되지 않으면 결과값으로 -1을 출력한다. $y = \tau(s)$ 이때 $s = w_0 + \sum_{i=1}^{d} w_i x_i, \quad \tau(s) = \begin{cases} 1 & s \geq 0 \\ -1 & s < 0 \end{cases}$ - 기울기, 절편 개념으로 퍼셉트론 이해: y = ax + b (a는 기울기, b는 y 절편) → y = wx + b (w는 가중치, b는 바이어스) 먼저 기울기 a는 퍼셉트론에서는 가중치를 의미하는 w(weight)로 표기한다. y 절편 b는 편향, 선입견이라는 뜻인 바이어스(bias)에를 의미한다. 가중합(weighted sum)이란 입력 값(x)과 가중치(w)의 곱을 모두 더한 다음 거기에 바이어스(b)를 더한 값을 가중합이라고 한다. 가중합의 결과를 놓고 1 또는 -1을 출력해서 다음으로 보낸다. 여기서 1과 -1을 판단하는 함수가 활성화 함수(activation function)이다. 대표적으로 시그모이드 함수가 있다. - XOR문제를 해결하기 위해 여러 개의 퍼셉트론을 한 번에 계산하여 좌표 평면 자체에 변화를 주는 것이 다층 퍼셉

<table>
<tr><td></td><td>
트론이다. 다층 퍼셉트론은 퍼셉트론이 가지고 있는 문제점을 해결하기 위하여 중간에 은닉층을 추가한 전방향(Feedforward) 네트워크이다.



- 다층 퍼셉트론 구조:

(1) 입력층(Input-Layer): 데이터를 입력 받는 뉴런들로 구성된 층을 말한다. 예를 들어 훈련 데이터(Training data set)가 8개의 속성으로 구성된 데이터라고 가정하면 입력층의 뉴런은 총 9개(8개 속성 + bias)의 뉴런으로 구성될 것이다.

(2) 은닉층(Hidden-Layer): 입력층을 통해 전달된 신호를 받아 특징을 추출하고 학습한다. (정확히는 output layer의 오차를 최소화하는 가중치를 학습한다.)

(3) 출력층(Output-Layer): 은닉층을 통해 계산된 신호들을 받아 최종적으로 분류를 실행하여 결과를 반환한다.

(4) 깊이(Depth): 몇개의 층이 존재하는가를 깊이(Depth)로 표현한다.
</td></tr>
<tr><td>질문 내용</td><td>
퍼셉트론에서 바이어스항을 추가하는 이유는 무엇인가요? 그리고 3장 15,16p 에 나와있는 목적 함수에 관한 설명에 대해 교수님이 "J(w)>=0부터 잘못되었고, 맞으면 양수 틀리면 음수로 최소화 되는 방향으로 가는 건 맞지만 sine 함수 자체가 미분이 되지 않아 (3.8) 모두 옳지 않다."라 말씀 하셨습니다. 이 부분에 대하여 자세히 알고 싶습니다.
</td></tr>
</table>

-201500629 김영연

구분	내용
학습 범위	3.1 신경망의 기초 3.2 퍼셉트론 3.3 다층 퍼셉트론
학습 내용	● 신경망의 종류 1) 전방 신경망과 순환 신경망 -전방신경망: 계산을 왼쪽에서 오른쪽으로 진행 -순환신경망: 오른쪽에서 왼쪽으로 진행하는 피드백 계산도 포함 2) 얕은 신경망과 깊은 신경망 -얕은 신경망: 은닉층이 1~2대 정도인 신경망 -깊은 신경망 3) 결정론 신경망과 스토캐스틱 신경망 -결정론 신경망: 입력이 같으면 항상 같은 출력이 나옴 -스토캐스틱 신경망: 계산이 확률에 따른 난수를 계산하므로 입력이 같아도 매번 다른 출력이 나옴

- 퍼셉트론의 구조

-입력층과 출력층이 있으나 입력층은 연산을 하지 않으므로 제외

-노드: 층을 구성하고 입력층에 있는 입력 노드 하나는 특징 벡터의 특징 하나에 해당.

$\mathbf{x} = (x_1, x_{2,\dots,} x_d)^T$ 로 표시하고 입력층은 d개의 노드를 가진다.

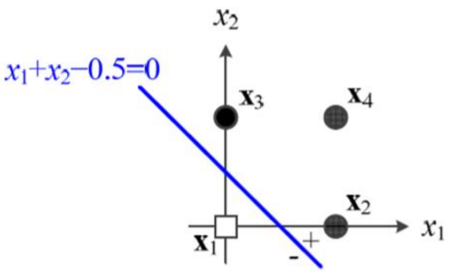
-바이어스: 맨 위에 x0로 표시된 여분의 노드

- 퍼셉트론 동작

-입력층에 특징 벡터가 들어오면, 서로 연결된 특징 값과 가중치를 곱한 결과를 모두 더한다.

$y = \tau(s)$, $s = w_0 + \sum_{i=1}^{d} w_i x_i$, 이고 $\tau(s)$는 -1 또는 1이 나온다.

- 분류기로 해석



-퍼셉트론은 2차원 특징 공간을 파란색 직선에 따라 +영역과 – 영역으로 나눈다. +영역은 모두 +1로, -영역은 모두 -1로 변환되므로 이진분류기로 작용한다.

-결정 경계: 위처럼 특징공간은 2개의 영역으로 나눔으로써 패턴의 부류를 결정하는 경계

-결정 직선: 2차원에서는 결정 경계가 직선이 되고, 이를 결정 직선이라고 한다.

-결정 평면: 3차원에서는 공간이 둘로 나누는 평면이 되어 결정 평면이라고 한다.

-결정 초평면: 4차원 이상에서 나타내는 형식

- 목적함수 설계와 델타 규칙

목적함수는 다음을 만족해야 한다.

J(w)>=0이다.

W가 최적이면, 즉 모든 샘플을 맞히면 J(w)=0이다.

틀리는 샘플이 많은 w일수록 J(w)는 큰 값을 가진다.

$\mathrm{J(w)} = -\sum_{x_k \in Y} -y_k(w^T x_k)$

델타 규칙: $w_i = w_i + \rho y_k(w^T x_k)$

- 다층 퍼셉트론은 여러 개의 퍼셉트론을 결합한 다층 구조를 이용하여 선형분리가 불가능한 상황을 해결하는데 이 때 도입한 기법은 다음과 같다.

-은닉층을 둔다.

-시그모이드 활성함수를 도입한다.

-오류 역전파 알고리즘을 사용한다.

	● 다층 퍼셉트론의 용량 – 원래 특징 공간이 2차원이고 퍼셉트론을 p개 결합한 상황이면 신경망은 2차원 공간을 p차원 공간으로 변환한다. 새로운 공간에는 좌푯값이 -1 또는 +1인 점이 2의 p승 개가 있는데, 원래의 공간을 이 점들로 매핑한다. 따라서 p가 클수록 신경망의 용량은 커지마 무턱대고 크게 하면 과잉적합 될 가능성이 높아지므로 적당한p를 찾아야 한다.
질문 내용	