과제명
과제 진행 일지 (연구노트)

대외비(Confidential)
관리번호 :
(Serial No.)

기 록 자 NAME	**황병준**			
연구과제명 PROJECT TITLE	심층학습 기반 스마트 축산 모델 개발			
소 속 LAB./DEPT.	컴퓨터소프트웨어공학과			
기록일자 RECORED	from	2023/04/01 (년/월/일)	to	2023/09/30 (년/월/일)

ContInued from page :

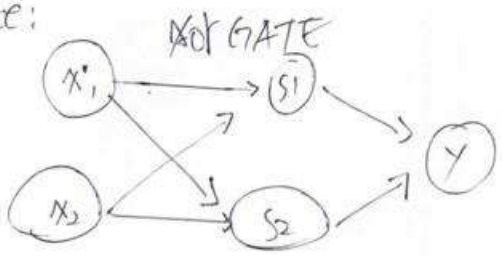
제목	다층 퍼셉트론 (Multi-Layer Perceptron)
목적	

ContInued from page : 다층 퍼셉트론: 층이 2개이상 쌓여있는 신경망

퍼셉트론의 한계:

기존의 퍼셉트론은 선형적으로 분리가 가능한 문제만 해결이 가능했었다.
이를 해결하기 위해서 XOR. (직선의 조건이 없다면)

XOR Gate:



다층 퍼셉트론은 다층 퍼셉트론과 다르게 비선형으로 분포하는 데이터들에 대해서 학습가능.
다층 퍼셉트론은 가중치에 대한 선형 방정식을 이용하기 때문에 층과 층 사이에서 선형으로 표현된
데이터를 비선형으로 바꿔주는 과정이 필요. 여기서 이 과정을 담당하는 것은 "활성화 함수" (Activation function)

기 록 자 Invented by	점 검 자 Witnessed & Understood by
황병준	오강한 오강한
일 자 Date	일 자 Date
2023/4/03	2023/04/06

제목	활성화 함수 (Activation function)
목적	

ContInued from page :

활성화 함수란:

퍼셉트론(Perceptron)의 출력값을 결정하는 비선형 함수

즉. 활성화 함수는 퍼셉트론에서 입력값의 총합을 출력할지 말지 결정하고 만약 출력을 한다면 어떤 값으로 변환해서 출력할지 결정하는 함수.

종류 (Sign, Sigmoid, Tanh, Softmax, Relu, Leaky Relu) 함수.

Sign func: Sign 함수의 퍼셉트론의 활성화 함수는 퍼셉트론 내 입력값의 총합이 양다 작을경우 -1을 출력, 반대로 0보다 클 경우 1을 출력하는 함수.

Sigmoid func: 모든 입력값에 대해 출력값이 실수값으로 정의,
값이 작아질수록 0, 클수록 1에 수렴,
출력의 0~1사이로 확률표현 가능

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

Tanh func: 모든 입력값에 대해 출력값이 실수값으로 정의,
작아질수록 -1, 커질수록 1에 수렴,
입력값이 0에 가까울수록 미분이 크기때문에 출력값이 빠르게 반응,

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

Soft max func: 출력값이 N개
입력값을 각각 지수함수로 취하고. 이를 총합으로 모든값을 정규화
정규화로 인해 0~1 값을 가질.
모든 출력값은 반드시 1

$$Softmax(x)_i = \frac{e^{x_i}}{\sum_j e^{x_j}}$$

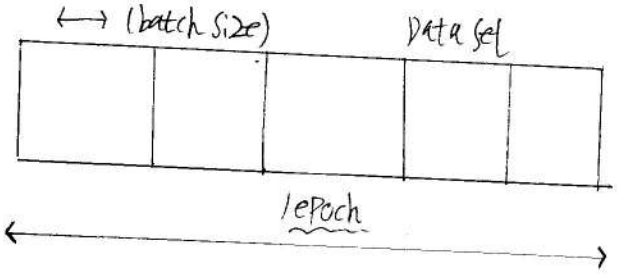
기 록 자 Invented by	점 검 자 Witnessed & Understood by
황병준	오강한 오강한
일 자 Date: 2023/4/12	일 자 Date: 2023/4/14

제목	배치(batch), 에폭(epoch)
목적	

ContInued from page :

batch: 가중치의 업데이트를 위해 한 번에 손실을 계산할 훈련데이터를 묶은것.

epoch: 전체 학습데이터를 한 번 돈것.



배치 크기와 모델 훈련의 관계

batch의 크기를 키우면 1에폭당 훈련수가 감소하고 이로 인해 전체 훈련 횟수가 감소하여 결과적으로 전체 훈련 시간이 감소하게 된다.

딥러닝에서는 입력데이터가 전체 훈련 데이터를 대표한다고 가정함. 즉 하나의 배치에 포함된 데이터가 전체훈련 데이터를 얼마나 잘 표현 하냐에 따라서 학습의 안정성이 결정된다.

Ex) 배치의 크기가 배수라면? (SGD)

데이터 셋의 크기 = N, 배치 크기 = 1이라면 최적화 함수는 하나의 학습 데이터마다 오차를 계산한뒤에 모델의 가중치를 1에폭당 N번 갱신하게 된다. 각각의 데이터 샘플을 계크하면 진행이 지리한 특성에 따라 가중치 업데이트 과정이 최적화 탐색 경로를 빠져나올 수 있다.

배치의 크기가 크다면? (BGD)

데이터셋의 크기=N, 배치크기=N이라면 모델 가중치는 1에폭당 1번 갱신 된다. 즉 주어진 모든 훈련데이터들의 특성을 바탕으로 한번의 최적의 가중치조합을 찾아간다.

주의: 배치 크기가 클수록 효과적으로 최적화가 되는것은 아니다

기 록 자 Invented by	점 검 자 Witnessed & Understood by
황병준	오강한 오강한
일 자 Date	일 자 Date
2023/4/18	2023/4/4

제목	활성화 함수 (Activation function)
목적	

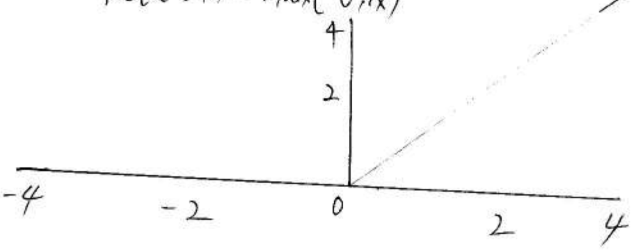
ContInued from page :

(Relu, Leaky)

Relu 함수 :

Relu (Rectified Linear Unit, Relu) 함수는 y=x 인 선형 함수가 입력값 0 이하에서부터 rectified(정류)된 함수

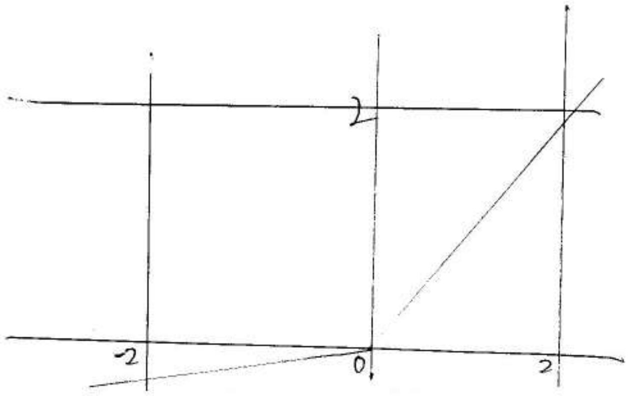
Relu (x) = max(0, x)



특징: 딥러닝 분야에서 가장 사용되는 활성화 함수. 구현이 단순하고 연산이 필요 없이 임계값 (양수/음수 여부)만 활용하므로 연산 속도 빠름

Leaky Relu 함수:

Relu 에서 발생하는 Dying Relu 현상을 보완하기 위하여 다양한 Relu가 제안됨. 그중 Leaky Relu func는 입력값이 음수일때 출력값이 아닌 0.01 과 같은 매우 작은 값을 출력하도록 설정 max(ax, x)



Leaky Relu func

기록자 Invented by	점검자 Witnessed & Understood by
황병준	오강한 오강한
일자 Date	일자 Date
2023/4/26	2023/4/28

제목	미니 배치 학습 (Mini-batch learning)
목적	

ContInued from page :

미니 배치 학습: 데이터의 일부를 추려 계산하는 방법

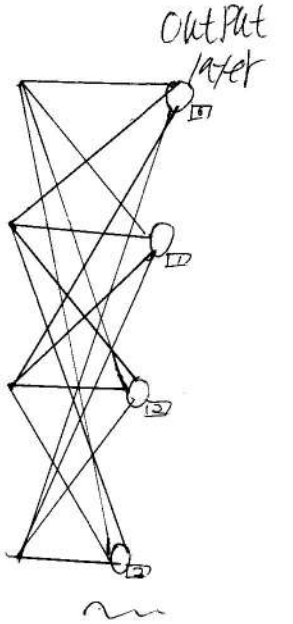
교차 엔트로피 오차 (CEE)

$$E = -\frac{1}{2}\sum_k (y_k - t_k)^2$$

→ 수많은 데이터를 모두 신경망에 한번씩 통과시켜 출력층의 값을 얻고 이를 정답 레이블 과 계산하여 평균 손실 함수 값을 얻을수 있다.

Y= 신경망 출력
T= 정답 레이블

· one-hot-encoding



Y: 신경망의 출력층 값 (확률 값)

0	0.17	0	03	→ Data 1이 신경망 통과
0	0.2	0.6	0.2	→ Data 2가 신경망 통과

→ y_{11} = 1번 데이터의 1번째의 값

t_{11} =

Soft max 통과.

t: 정답 레이블 (one-hot-encoding) 방식.

0	1	0	0	→ Data1의 정답(레이블) : 1
0	0	1	0	→ Data2의 정답(레이블) : 2

기 록 자 Invented by	점 검 자 Witnessed & Understood by
황병준	오강한 오강한
일 자 Date	일 자 Date
2023/5/3	2023/5/5

제목	Batch Gradient Descent (BDG) 배치 경사 하강법
목적	

ContInued from page :

Batch : Total Training Dataset

전체 데이터 셋의 오차를 구한뒤 기울기를 한번 계산한뒤 라라미터를 업데이트하는 방식

$$\theta^{(next\ step)} = \theta - \eta \nabla_\theta MSE(\theta)$$

특징: 전체 데이터에 대해 업데이트가 한번만 이루어 지므로 업데이트 횟수 자체는 적다. 하지만 업데이트를 한번만 하므로 계산 시간이 오래걸린다.

전체 데이터에 대해 미아 Gradient 계산한로 Optimal로 수렴이 안정적이다. 그러나 Local Optimal에 수렴할 경우 탈출이 어려움

GPU를 이용해 병렬처리에 유용

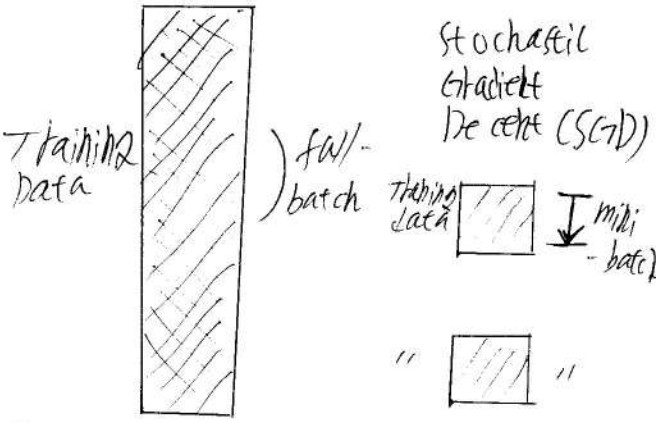
정리: 업데이트를 한번하기 위해서는 적지만 한 한번의 스텝으로 인하여 시간이 오래걸린다. 또한 학습데이터에 대해 한번에 처리를 해야하므로 메모리가 많이 필요하다.

기 록 자 Invented by	점 검 자 Witnessed & Understood by
황병준	오강한 오강한
일 자 Date	일 자 Date
2023/5/9	2023/5/11

제목	SGD (Stochastic Gradient Descent)
목적	

ContInued from page :

SGD: batch의 크기가 1인 경사하강법 알고리즘.



이용:
모델의 오차 또는 손실을 줄이기 위해 모델의 파라미터를 반복적으로 조정

경사하강법(Gradient Descent)은 전체 데이터 셋에 대한 손실함수의 경사를 구하지만 이런 방식은 데이터 셋이 큰 경우 계산 비용이 많이 들 수 있다.

↓

하지만 SGD의 경우에는 각 반복에서 무작위로 선택된 하나의 데이터 포인트나 작은 데이터 배치에 대한 Gradient을 계산하여 업데이트 수행.

핵심: 전체 데이터 셋을 사용하여 Gradient를 구하는것 보다 무작위로 선택된 하나의 샘플이나 작은 배치를 사용하여 Gradient를 계산함으로써 효율성과 확장성에 중점

Gradient Descent 보완.등

SGD 장점: 대규모 데이터셋에 효율적.

SGD 단점: 수렴이 불안정하고 느릴수 있고 lr(학습률)과 Parm(파라미터)를 조절하는데 어려움이 있다.

단점: 수렴이 불안정 할수 있음

해결을 위한 변형 → (Momentum, Nesterov Accelerated Gradient, Adagrad,

기 록 자 Invented by	점 검 자 Witnessed & Understood by
황병준	오강한 요한
일 자 Date	일 자 Date
2023/C/17	2023/C/19

제목	Gradient Descent (경사 하강법)
목적	

ContInued from page :

Gradient Descent : 딥러닝의 학습시 사용되는 최적화 (Optimizer) 중 하나이다.

Gradient가 descent 하는 방향으로 파라미터의 값을 찾는 방향으로 학습이 진행된다.

최적화 가정 :

1. 현재 current location을 기준으로 Gradient를 구하여 이동 방향을 찾는다.
2. 한번 이동할때 얼마나 이동할것인지 step size를 정하여 (learning rate)를 dot(곱)하여 그만큼 이동.
3. 최적의 값 (내려 갈수록 값이 없을때까지 찾는다. (반복)

Gradient descent 식

$$x_{i+1} = x_i - \alpha \frac{df}{dx}(x_i)$$

Gradient descent의 한계 :

local minimum에 빠지기 쉽다

↓

변형된 (momentum, SGD) 등의 방법이 있다.

기 록 자 Invented by	점 검 자 Witnessed & Understood by
황병준	오강한
일 자 Date	일 자 Date
2023/5/22	2023/5/25

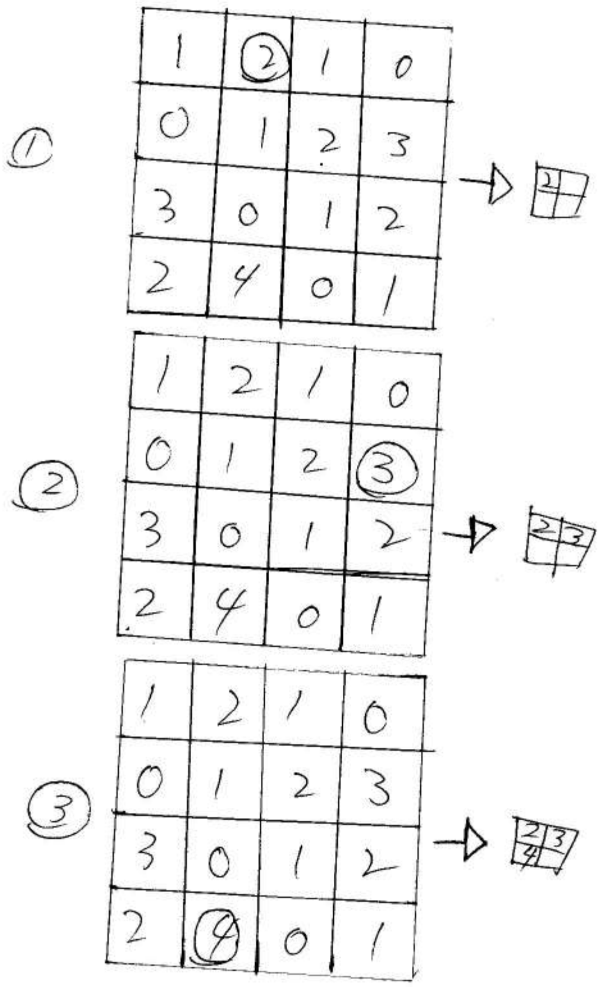
제목	Pooling Layer (풀링 계층)
목적	

ContInued from page :

Pooling:

풀링은 2차원 데이터의 세로 및 가로 방향을 줄이는 연산.
Pooling은 Max pooling, Average Pooling 등이 있다.
Max Pooling: 대상 영역에서 최대값을 취하는 연산.
Average Pooling: 대상 영역의 평균을 계산.

이미지 인식 분야에서 Max Pooling을 사용한다

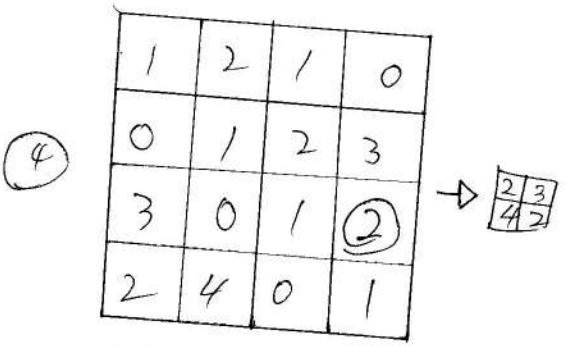


Pooling Layer의 특징:

풀링 계층은 합성곱 계층과 달리 학습해야 할 매개 변수가 없다. 풀링은 대상영역에서 최대값이나 평균을 취하는 명확한 처리이므로 특별히 학습 해야 할것이 없다.

풀링 연산은 입력 데이터의 채널 수 그대로 출력 데이터로 보낸다. 풀링은 2차원 데이터의 크기로 줄이는 연산이므로 3차원을 결정하는 채널 수는 건드리지 않는다. 따라서 채널마다 독립적 계산.

풀링 계층은 입력의 변화에 영향을 적게 받는다. 입력 데이터가 조금 변하더라도 풀링 계층 자체가 그 변화를 흡수하여 사라지게 된다.



기록자 Invented by	점검자 Witnessed & Understood by
황병준	오강한 오강한
일자 Date	일자 Date
2023/5/25	2023/5/29

제목	연쇄법칙(chain rule)
목적	

ContInued from page :

chain rule: 합성함수 미분법의 성질

$z = g(y),\ y = f(x)$

$z = (g \cdot f)(x) = g(f(x))$ 미분 $\rightarrow (g \cdot f)'(x) = g'(y) f'(x)$

chain rule $\searrow$ $\frac{dz}{dx} = \frac{dz}{dy} \times \frac{dy}{dx}$

$E = \frac{1}{2}[(o_1 - t_1)^2 + (o_2 - t_2)^2]$

$z_{h1} = w_1 x_1 + w_3 x_2 + w_5 x_3 + b_1$

$z_{h2} = w_2 x_1 + w_4 x_2 + w_6 x_3 + b_1$

$h_1 = a(z_{h1})$

$h_2 = a(z_{h2})$

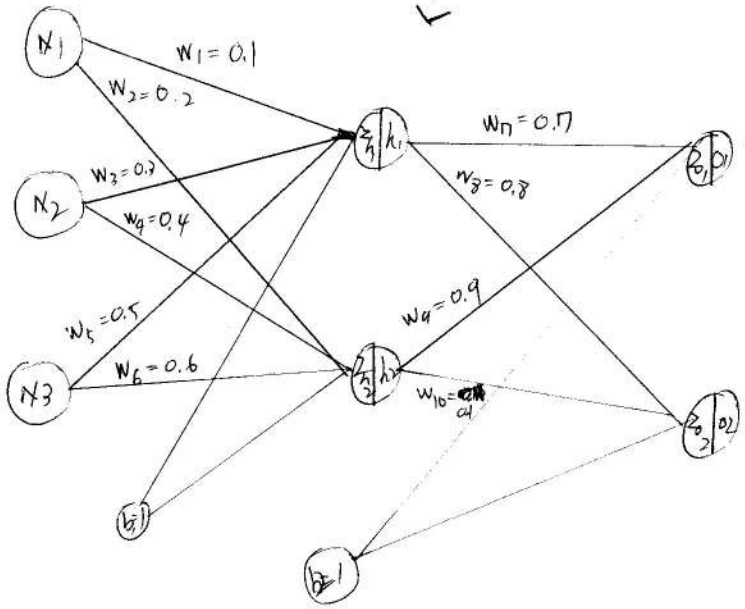
$z_{o1} = w_7 h_1 + w_9 h_2 + b_2$

$z_{o2} = w_8 h_1 + w_{10} h_2 + b_2$

$o_1 = a(z_{o1})$

$o_2 = a(z_{o2})$

신경망 개념 계산 적용



기 록 자 Invented by	점 검 자 Witnessed & Understood by
황병준	오강한 오강한
일 자 Date	일 자 Date
2023/5/31	2023/6/2

제목	Activation function Relu Layer
목적	

ContInued from page :

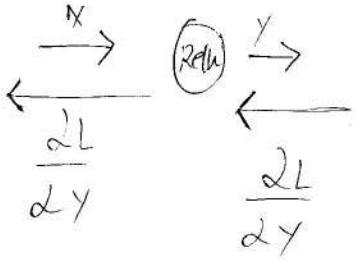
식:

$$y = \begin{cases} x & (x>0) \\ 0 & (x\le 0) \end{cases}$$

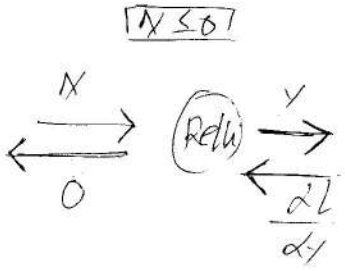
미분:

$$\frac{dy}{dx} = \begin{cases} 1 & (x>0) \\ 0 & (x\le 0) \end{cases}$$

$x>0$



$x\le 0$



Code:

```
class Relu:
    def __init__(self):
        self.mask = None

    def forward(self, x):
        self.mask = (x <= 0)
        out = x.copy()
        out[self.mask] = 0
        return out
```

```
    def backward(self, dout):
        dout[self.mask] = 0
        dx = dout
        return dx
```

기 록 자 Invented by	점 검 자 Witnessed & Understood by
황병준	오강한
일 자 Date	일 자 Date
2023/6/16	2023/6/19

제목	Activation function sigmoid layer
목적	

ContInued from page :

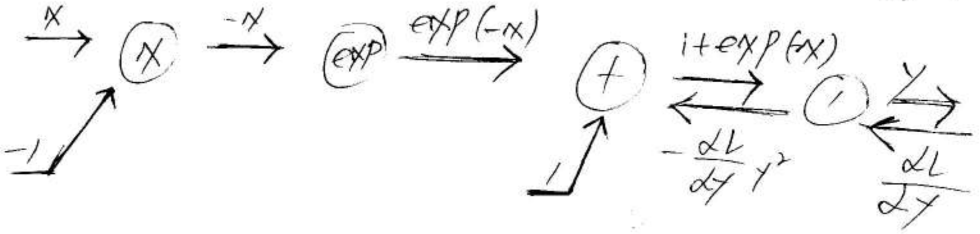
sigmoid

$$y = \frac{1}{1+\exp(-x)}$$

정리: $\frac{\partial L}{\partial y} y^2 \exp(-x) = \frac{\partial L}{\partial y} \frac{1}{(1+\exp(-x))^2} \exp(-x)$

$= \frac{\partial L}{\partial y} \frac{1}{1+\exp(-x)} \frac{\exp(-x)}{1+\exp(-x)}$

$= \frac{\partial L}{\partial y} y(1-y)$



1 step → 역전파 때는 순전파의 출력 제곱을 하류로 전달.

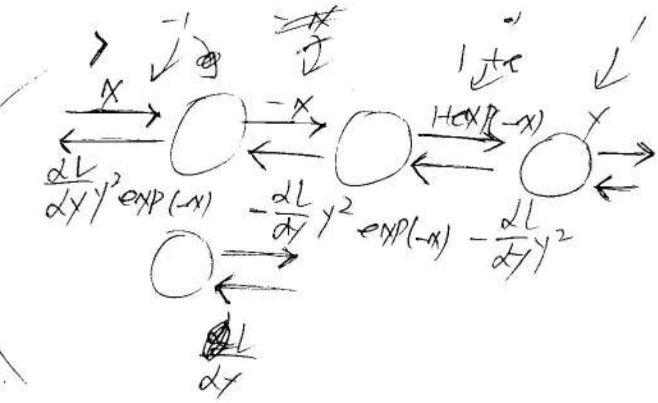
$y = 1/x$ 미분 → $\frac{\partial y}{\partial x} = -\frac{1}{x^2}$

$= -y^2$

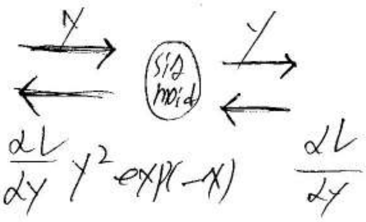
2 step

'+' 노드는 단순히 값을 하류로 보냄

3 step 'exp' 노드는 순전파 때의 값을 서로 곱해 하류로 전달한다.



simplify

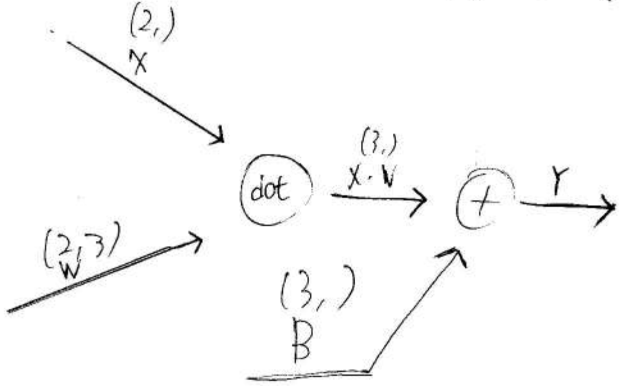


기 록 자 Invented by	점 검 자 Witnessed & Understood by
황병준	오강한
일 자 Date 2023/ 6 / 12	일 자 Date 2023/ 6 / 16

제목	Affine layer
목적	

ContInued from page :

수식 $Y = Wx + b$



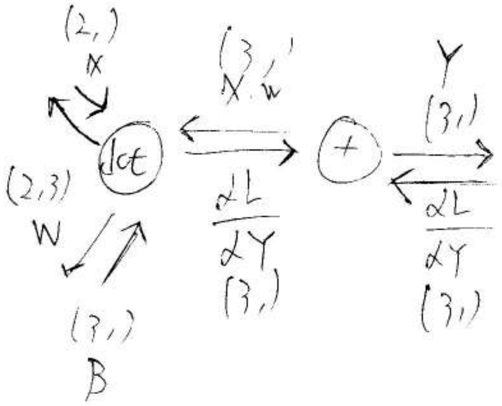
범용성이 [illegible] 머신러닝 모델이
학습 데이터에 맞춰져 있어
테스트 데이터 에서 성능 저하 할수 있다.

① $\frac{\partial L}{\partial X} = \frac{\partial L}{\partial Y} \cdot W^T$

(2,) (3,) (3,2)

② $\frac{\partial L}{\partial W} = X^T \cdot \frac{\partial L}{\partial Y}$

(2,3) (2,1) (,3)



기 록 자 Invented by	점 검 자 Witnessed & Understood by
황병준	오강한 오강한
일 자 Date 2023/ 6 / 20	**일 자** Date 2023/ 6/ 23

제목	퍼셉트론 (Perceptron)
목적	

ContInued from page :

퍼셉트론은 인공신경망(Artificial Neural Network, ANN)의 구성 요소로서 다수의 값을 입력 받아 하나의 알고리즘으로 출력하는 알고리즘.

특징 :

이진분류(Binary classification) 모델을 학습하기 위한 지도학습(Supervised learning) 기반의 알고리즘

이진분류:

EX) 인공지능(AI)에게 강아지와 고양이의 사진을 수각되고 제공되었을때 강아지와 고양이를 분류하는 문제

지도학습:

데이터(=feature)와 정답(=label)을 모두 활용하여 학습하는 방식

동작 과정:

다수의 값 X를 입력받고 가중치(Weight)를 곱한다. 편향(Bias, b)

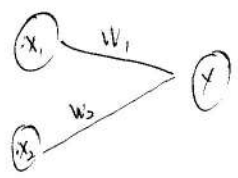
가중치의 값이 클수록 입력값이 중요.

일반적으로 입력값은 1로 고정하 bias를 곱한 변수로 표현

학습된 결과를 "가중합"이라 칭함.

$$W_1X_1 + W_2X_2 + \dots + W_nX_n > 0 \rightarrow 1 \quad \text{output}$$

$$W_1X_1 + W_2X_2 + \dots + W_nX_n \leq 0 \rightarrow 0$$



기 록 자 Invented by	점 검 자 Witnessed & Understood by
황 병 골	오강한 요함한
일 자 Date: 2023/6/28	일 자 Date: 2023/6/30

제목	손실 함수 (loss function)
목적	

ContInued from page :

손실함수(loss function) :

지도학습(Supervised learning) 시 알고리즘이 예측값과 실제 정답의 차이를 비교하기위한 함수.

즉, 학습중에 알고리즘이 얼마나 올바르게 예측하는 정도로 를 체크하기 위한 함수로써 최적화(Optimize)를 위해 최소화 하는 것이 목적인 함수.

손실함수 (loss function) = (objective function)

손실함수를 통하여 모델 학습중에 손실 (loss) 가 커질수록 학습이 잘 되지 않고 있다고 해석이 가능하다 이와 반대로 loss function 을 통하여 학습이 잘 되고 있다고 판단이 가능하다.

손실 (loss func) 가 작을수록 ↓

$$\tilde{\theta} = \arg\min_{\theta} L(x, y; \theta)$$

평균 (MSE) Mean Squared Error

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

예측한 값과 실제 값사이의 평균 오차 제곱 정의
오차가 커질수록 제곱 연산으로 인해서 값이 급격해진다.
제곱으로 인해 오차의 값이 양수로 음수를 보정해준다.

RMSE (Root Mean Squared Error)

MSE에 root 를 씌운것으로 MSE와 기본적으로는 동일하다 기본 MSE의 값은 오차의 제곱을 하기 때문에 실제 오차보다 더 커지는 특성이 있어 Root를 씌운 RMSE는 MSE의 값보다 오차를 줄여준다.

이외로 (Binary crossentropy, Binary crossentropy class, Categorical Crossentropy Class) 등이 있다.

기 록 자 Invented by	점 검 자 Witnessed & Understood by
황 병 을	오강한 요함한
일 자 Date	일 자 Date
2023/7/3	2023/7/6

제목	역전파(BackPropagation)
목적	

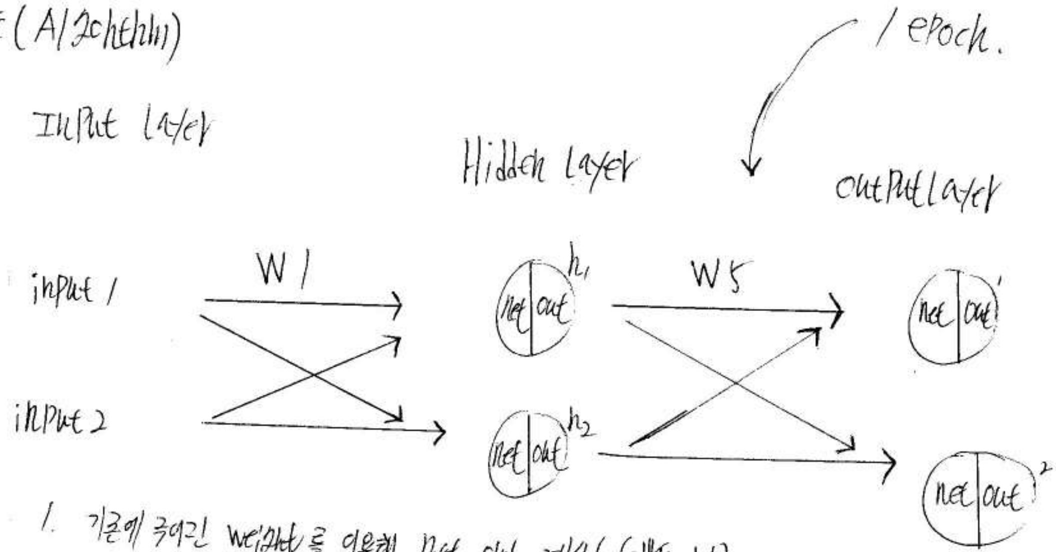
ContInued from page :

역전파 알고리즘: backPropagation algorithm)

Input, output을 알고 있는 상태에서 신경망을 학습시키는 방법.

특징: 1. 초기 Weight 값은 랜덤,
2. 노드를 지나갈때마다 활성화 함수 적용,
3. 이때 Activation func는 sigmoid 사용 ∴ 다른 func 사용해도 문제 없음

알고리즘(Algorithm)



1. 기존에 주어진 weight를 이용해 net, out 계산(forward)
2. 전체 오차를 각 가중치로 편미분한 값을 기존의 weight에서 빼낸다
3. 모든 가중치에 대하여 편미분 값을 기존 weight에서 빼낸다 ← output에 가까운 층에서 부터 먼 층으로
4. 1~3의 순서를 학습 횟수 만큼 반복.

한계:

경사 하강법의 문제에 관련된것이다
극소값에 따라서 작은 값으로 기울기가 작아지는 방향으로 이동하기에 start의 지점에 따라 달라질수 있다.

기록자 Invented by	점검자 Witnessed & Understood by
황 병 근	오강한 오강한
일자 Date 2023/7/10	일자 Date 2023/7/13

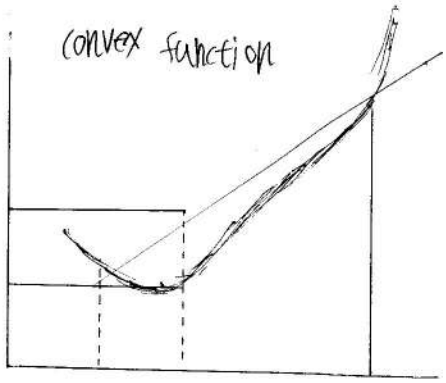
제목	Gradient Descent의 한계.
목적	

ContInued from page :

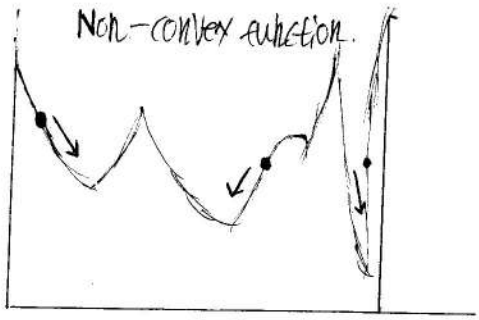
정수 최적법 단점: Gradient Descent의 단점

정수 최적법은 비 볼록 함수의 정수 파라미터의 최적의 위치에 따라서 최적의 값이 달라지는 한계가 있다.

1. local Minimum
2. Saddle Point



볼록 함수의 2가지 종류
(convex function, Non-convex function)
볼록 함수 비 볼록 함수



convex function VS Non-convex function

볼록함수(convex function)
어느 지점에서 시작하더라도 최적의 값에 도달 가능.

비 볼록 함수(Non-convex function)
시작 지점에 따라서 각기 다른 최적의 값을 찾는다 ← 국부 최적값(local minimum)에 빠질 가능성이 있다.

Q. Saddle Point(안장점)을 벗어나기 힘들다.

기 록 자 Invented by	점 검 자 Witnessed & Understood by
황 병 곤	오강한 요함한
일 자 Date	일 자 Date
2023/7/17	2023/7/19

제목	Adam
목적	

ContInued from page :

Adam OPtimizer?

1. 딥러닝의 최적화 방법중 하나
2. 경사 하강법의 기본한 알고리즘.

2종 원리:

1. 모델의 가중치를 무작위로 초기화
2. 훈련 데이터 에서 미니 배치를 무작위로 선택
3. 선택한 미니배치를 사용해 손실함수 계산
4. 손실함수의 기울기 계산
5. 기울기를 사용해 가중치 업데이트
6. 1~5 단계를 반복해 전체 미니 배치에 대해서 가중치 업데이트.

Adam optimizer의 장점, 한계?

장점:

1. 메모리 사용량이 크게 일시 제한이 있을수 있다.
2. 학습률 하이퍼 파라미터 조정 필요성.

한계:

1. 메모리 사용량이 크게 일시 제한이 있을수 있다.
2. 학습률 하이퍼 파라미터 조정 필요성.

기 록 자 Invented by	점 검 자 Witnessed & Understood by
황병준	오강한 오강한
일 자 Date	일 자 Date
2023/7/24	2023/7/28

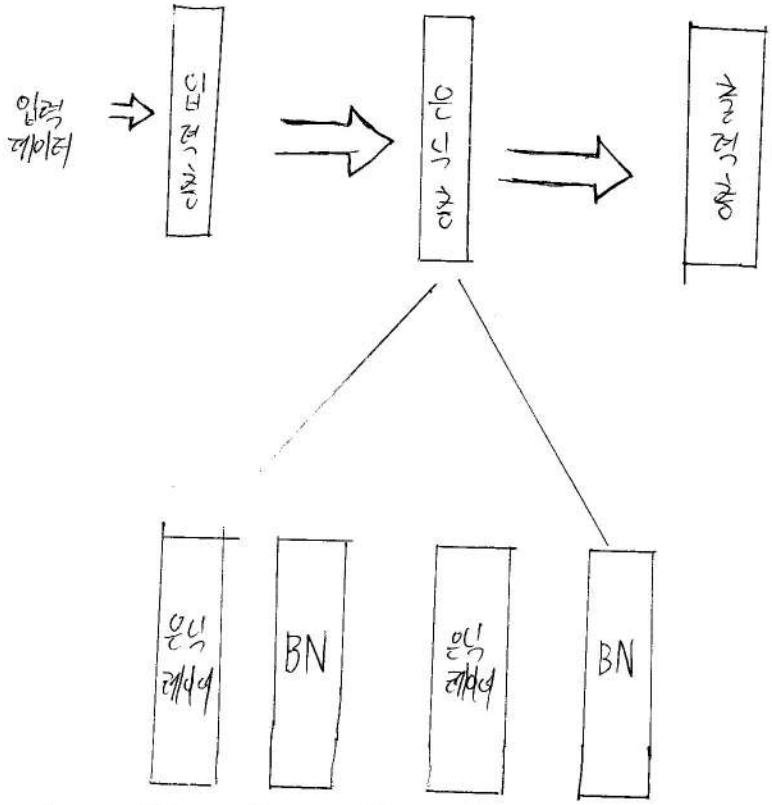
제목	배치 정규화 (Batch Normalization)
목적	

ContInued from page :

개념:
모델의 학습 과정에 여러 Feature 레이터 분포/상태를 조정하는 기술

필요성: 평균과 분산을 조정하는 과정이 별도의 과정이 아니라 신경망 안에 포함되어 학습시 평균과 분산을 조정 하는 과정역시 같이 조절된다는것.
↳ 레이어 마다 정규화 하는 레이어를 만들어 활성값 분포가 나오지 않도록 조절하는것.

> 미니배치의 평균과 분산을 이용해서 정규화 후 Scale 및 Shift를 감마(γ)값 베타(β) 값을 통해서 실행된다.
↳ BackPaPagation을 통해서 실행된다.



장점:
학습 속도 향상
오버피팅억제
가중치 초기화 의존도 해소

한계:
1. RNN에 적용하기 어렵다.
2. 미니배치 크기에 의존적이다.
3. 배치의 크기가 클 경우 각종이 잘 하지 않는다.

기 록 자 Invented by	점 검 자 Witnessed & Understood by
황병준	오강한 오강한
일 자 Date	일 자 Date
2023/8/1	2023/8/4

제목	학습 관련 기술 (Overfitting)
목적	

ContInued from page :

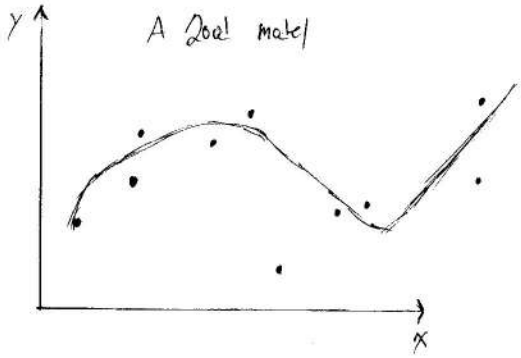
오버피팅(과적합):

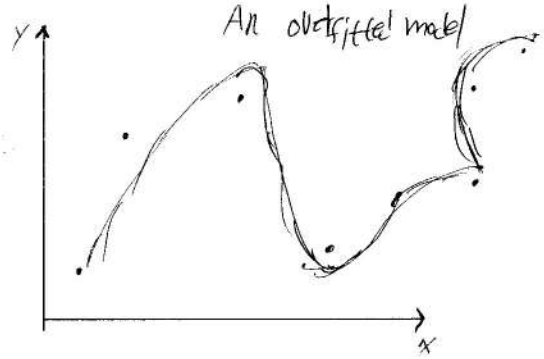
신경망이 훈련 데이터에만 지나치게 학습되어 그 외의 데이터에는 제대로 대응하지 못하는 상태

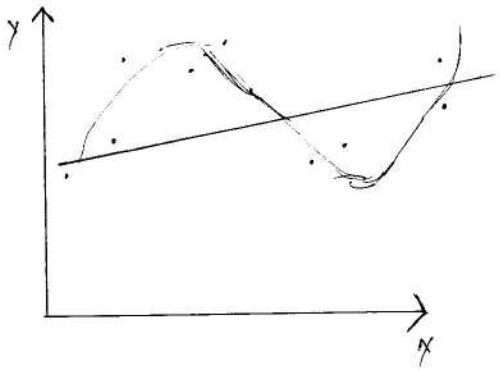
과적합 발생 예:

feature(→parameter)가 많고 표현력이 높은 모델인 경우(상대적으로 데이터 수가 적은데 비해)

train 데이터가 적은 경우(상대적으로 feature → parameter가 많은데 비해)







과적합 방지 대안들

1. Input 데이터 늘리기.
2. parameter 줄이기 → 데이터를 더 늘리는 방법은 힘들기 때문에 feature를 줄여보는 것으로 중점적으로 고려.

기 록 자 Invented by	점 검 자 Witnessed & Understood by
황병준	오강한 오강한
일 자 Date	일 자 Date
2023/8/8	2023/8/11

제목	가중치 감소 (weight decay)
목적	

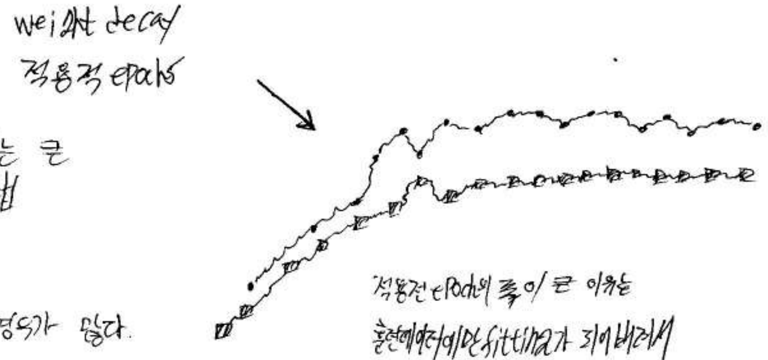
ContInued from page :

weight decay:

학습 과정에서 큰 가중치에 대해서는 그에 맞는 큰 패널티를 부과하여 오버피팅을 억제하는 방법

↓

매개변수의 값이 커서 발생하는 경우가 많다.

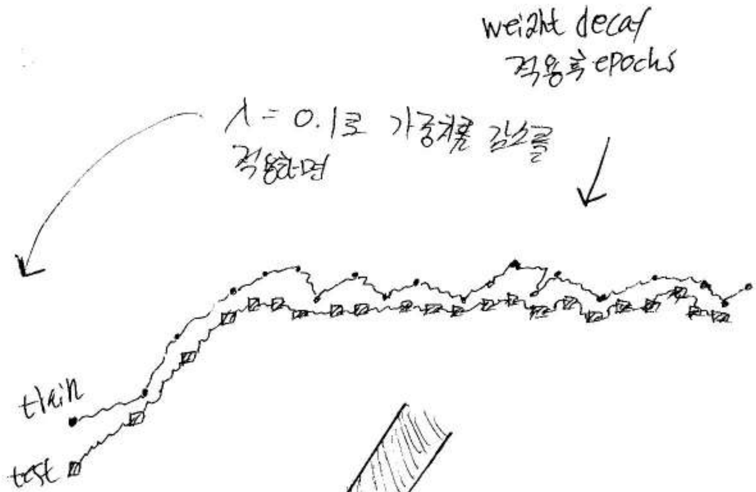


적용전 epoch의 폭이 큰 이유는 훈련데이터에만 fitting가 되어버려서 그렇다 ← 이러한 현상을 오버피팅이라 한다.

Ex) 가중치의 제곱 노름(L2 노름)을 손실 함수에 더한다. ← 가중치가 커지는 것을 억제 할수 있다.

가중치 감소에서는 모든 가중치 각각의 손실 함수(loss function)에 $L(W)+\frac{1}{2}\lambda W^2$ 가중치의 제곱 놈(L2-norm)을 더하여 가중치의 기울기를 계산할때 BackPropagation에 따른 결과에 정규화 항을 미분한 λW를 더한다.

$\lambda = 0.1$로 가중치를 감소를 적용하면



train 데이터와 test 데이터의 정확도 차이가 감소한 것이 확인 가능

기록자 Invented by	점검자 Witnessed & Understood by
황병준	오강한 오강한
일자 Date	일자 Date
2023/8/15	2023/8/18

제목	AdaGrad
목적	

ContInued from page :

AdaGrad 개념:

일정한 learning rate를 사용하지 않고 변수, 스텝마다 learning rate가 바뀐다.

↓

시간이 지날수록 learning rate는 줄어드는데 큰 변화를 겪은 변수의 learning rate는 대폭 작아지게 되고 작은 변화를 겪은 학습률은 소폭으로 작아진다.

점화식:

$$h_n = h_{n-1} + \nabla f(x_n) \odot \nabla f(x_n), \quad h_{-1} = 0$$

$$x_{n+1} = x_n - \eta \frac{1}{\sqrt{h_n}} \odot \nabla f(x_n),$$

수식:

G_t : t번째 time step까지의 기울기 누적 크기

ϵ : 분모가 0이 되는 것을 방지하기 위해 작은 값 $\approx 10^{-6}$

$$G_t = G_{t-1} + (\nabla J(x_{t-1}))^2$$

$$x_t = x_{t-1} - \frac{\eta}{\sqrt{G_t+\epsilon}} \cdot \nabla J(x_{t-1})$$

장점: feature마다 다른 학습률을 적용함으로써 feature별 특성을 고려하여 학습을 효율적으로 하는 장점이 있다.

↓

큰 기울기를 갖게 되어 학습이 많이 된 변수는 학습률을 감소시킨다.
반대로 학습이 적게 된 변수는 학습률을 높게 설정하여 조절이 가능하다.

단점: G_t 값은 점차 커지기 때문에 학습이 오래 진행되면 learning rate가 $\left(\frac{\eta}{\sqrt{G_t+\epsilon}}\right)$이 0에 가까워지기 때문에 이 이상의 학습이 이뤄지지 않을 수도 있다.
즉 → model의 학습이 진행될 때 학습이 잘 이루어져 더이상 변수의 값이 업데이트가 되지 않는 것인지, G_t 값이 지나치게 커져서 추가적으로 학습이 되지 않는 것인지 알기 어려운 문제점이 있다.

기 록 자 Invented by	점 검 자 Witnessed & Understood by
황 병 준	오강한 오강한
일 자 Date 2023/ 8/20	일 자 Date 2023/8/25

제목	합성곱 신경망 (CNN)
목적	

ContInued from page :

완전 연결 계층:

입력한 계층의 모든 뉴런과 연결되어 있는
신경망, 입력 데이터와 가중치 1대1로
대응하는 값을 모두 연결(FULL-connected)이라 한다.
행렬의 내적(Affine)으로 표현됨.

플래튼:

데이터의 형상이 5시된다. 입력 데이터가 다차원 shape 일때
FULL-connected에 Input 하기 위해서는 입력 데이터를
1차원으로 평탄화 해줄 필요 입력해야 한다.

합성곱 계층:

완전 연결 신경망 과는 다르게
합성곱 신경망(CNN) 계층은 형상을 유지한다.
즉 이미지와 같은 3차원 데이터를 입력받으면
다음 계층에도 3차원 데이터로 전달한다.

convolutional layer 의
입출력 데이터는 다차원 이기에
(Feature Map)이라함

입력데이터 → Input Feature Map
출력데이터 → Output Feature Map

합성곱 연산

2차원의 입력데이터가 들어오면, 필터(커널)의 윈도우 를 일정 간격으로
이동하며 입력데이터에 적용한다.

1	2	3	0
0	1	2	3
3	0	1	2
2	3	0	1

Input Data

X

2	0	1
0	1	2
1	0	2

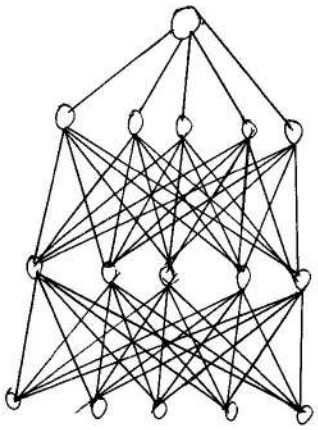
→

15	16
6	15

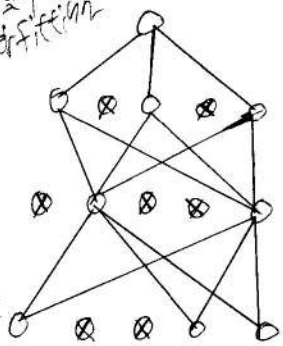
기 록 자 Invented by	점 검 자 Witnessed & Understood by
황 병 준	오강한 오(확)한
일 자 Date: 2023/8/28	일 자 Date: 2023/8/31

제목	드롭 아웃 (DropOut)
목적	

ContInued from page :



DROP OUT:
딥러닝 학습에 있어서 ~~Overfitting을 해소하기~~
우한 방법.



Validation 데이터의 일부 유닛이
동작하지 않게 하여 overfitting
을 방지하는것.

DropOut:

뉴런을 임의로 삭제 하면서 학습하는 방법.
신경망 구조 학습시, 레이어간 연결중 일부를 랜덤하게
삭제하면, 더러에서 네트워크를 양상하는 효과를 볼수 있으
이로 인해 일반화 성능이 올라간다.

기 록 자 Invented by	점 검 자 Witnessed & Understood by
황병준	오강한 오강한
일 자 Date	일 자 Date
2023/9/4	2023/9/18

제목	HyperParameter (하이퍼파라미터)
목적	

ContInued from page :

HyperParameter : 알고리즘 사용자가 직접 세팅하는 값
← 얻어낸 수행결과의 최적의 값을 찾는다.,

2. 정해진 최적의 값 없음
3. HyperParameter는 모델에서 외적인 요소, 데이터 분석을 통해 얻어지는 값 아님.
4. 모델의 parameter 값을 측정하기 위해 알고리즘 구현 과정에서 사용
5. 예측 알고리즘 모델링의 복잡성을 대체 조절함

ex) 신경망 학습에서 learning rate, SVM에서의 코스트 값, KNN에서의 K의 개수, epoch

HyperParameter tuning :

개념: 1~5와 같이 사용자의 입력값이고 정해진 최적의 값은 없다. 모델과 데이터에 따라서 달라지기만 이를 반복 시도하면서 데이터와 모델에 맞는 하이퍼 라미터를 찾아 나가는 과정을 일컫는다.

네트워크 구조와 관련된 HyperParameter

은닉층의 뉴런개수 (Hidden unit)

1. 문제에 따른 학습 최적화 결정 방식
2. 첫 Hidden Layer의 뉴런 수가 Input Layer 보다 큰 것이 효과적

Dropout:

1. 과적합을 피하기 위한 정규화 기법

가중치 초기화:

학습 성공에 대한 결정 방식

학습에 사용되는 HyperParameter:

(learning rate, momentum, epoch, training epochs, batch size, Iteration, cost function, Regularization parameter)

기 록 자 Invented by	점 검 자 Witnessed & Understood by
황병준	오강한 오강한
일 자 Date	일 자 Date
2023/11/11	2023/11/15

제목	합성곱 신경망(CNN)
목적	

ContInued from page :

패딩(Padding):

합성곱연산을 수행하기 앞서, 입력데이터 주변을 특정값(0, 1)등으로 채우는것.

1	2	3	0
0	1	2	3
3	0	1	2
2	3	0	1

×

2	0	1
0	1	2
1	0	2

padding을 적용하는 이유.

출력데이터 크기를 조절하기 위해서이다.

padding을 적용하지 않으면 입력데이터에 비해 출력데이터는 합성곱 연산을 수행할수록 출력데이터는 입력데이터에 비해 작아지게 되어진다.

합성곱 연산을 반복하는 깊은 신경망에서는 출력크기가 1이 되고, 더이상 합성곱 연산을 수행할 수 없게된다.

×

7	17	10	7
4	15	16	10
10	6	15	6
8	10	4	3

3차원 데이터의 합성곱 연산

채널까지 고려한 3차원 데이터도 2차원의 데이터와 동일한 방식으로 각중된다.

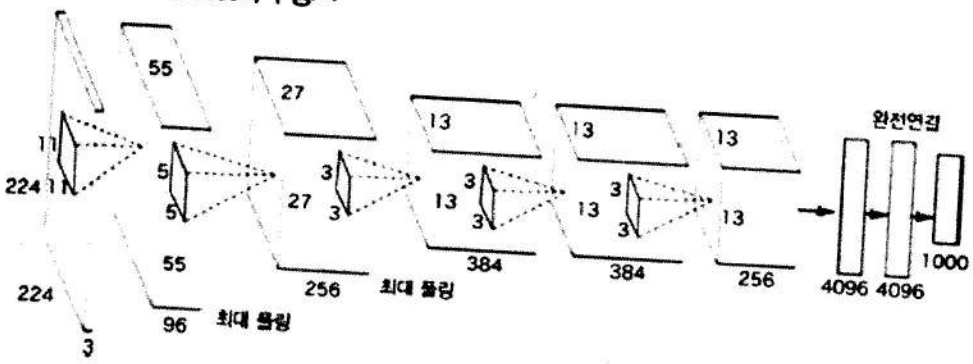
주의할: 3차원의 합성곱 연산에서 입력데이터의 채널수와 필터의 채널수가 같아야 한다. 또한 필터의 크기는 임의로 설정이 가능하나 모든 채널의 크기는 같아야 한다.

기 록 자 Invented by	점 검 자 Witnessed & Understood by
황병준	오강한 오강한
일 자 Date	일 자 Date
2023/9/18	2023/9/22

제목	AlexNet
목적	

ContInued from page :

그림 7-28 AlexNet의 구성도



AlexNet은 합성곱 계층과 풀링 계층을 거듭하여 마지막으로 완전 연결 계층을 거쳐 결과를 출력한다. LeNet에서 크게는 바뀌지 않았지만 AlexNet에서는

→ 1. 활성화 함수로 ReLU 사용.
2. LRN (Local Response Normalization) 이라는 국소적 정규화를 실시하는 계층을 이용
3. Dropout 사용.

기 록 자 Invented by	점 검 자 Witnessed & Understood by
황병준	오강한 오강한
일 자 Date	일 자 Date
2023/9/25	2023/9/29