딥러닝 기본 원리의 이해

컨설팅실 박희원 대리

신경망

Momentum

VGG

Softmax

ReLU

pooling

dropout

Learning rate

backpropagation

Sigmoid

활성화 함수

딥러닝

생소하다면
기본 원리부터 이해하자

수요샘 발표 내용

딥러닝을 이해하기 위한 첫 걸음

단층/다층 신경망의 구조

Backpropagation

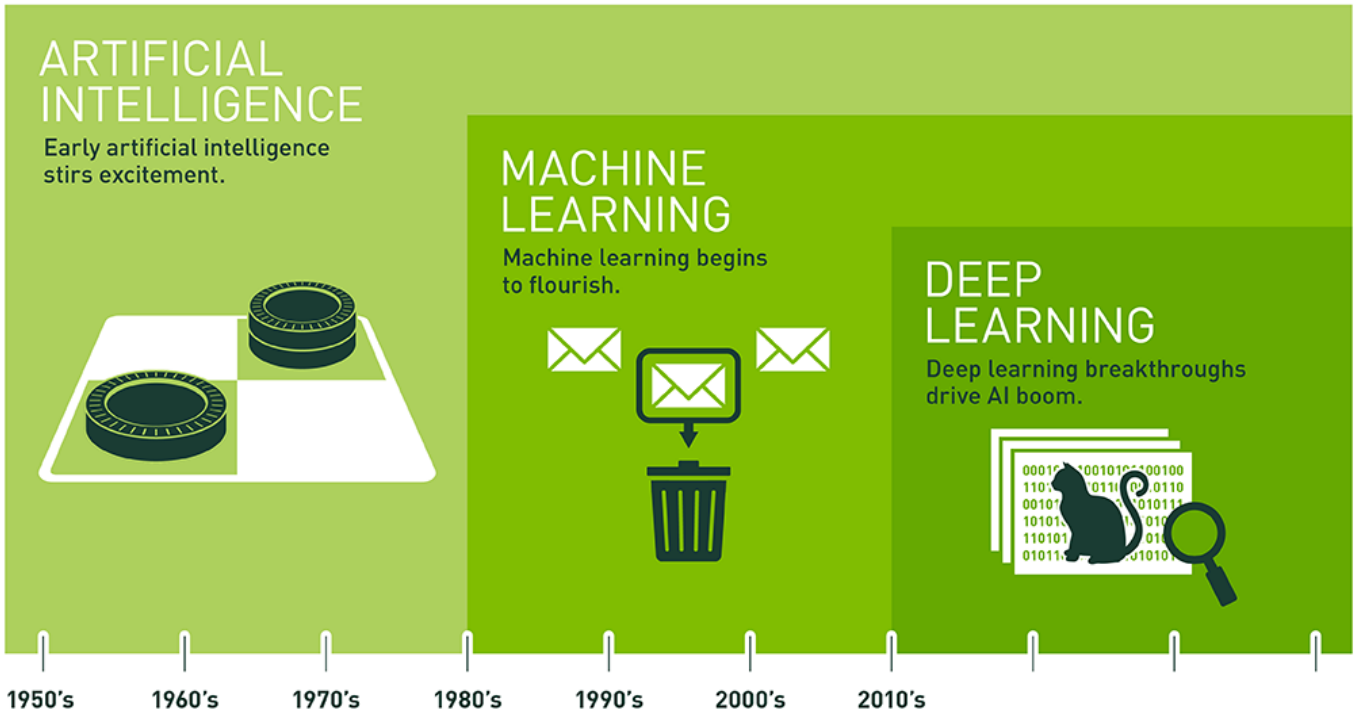
3가지 한계점

CNN, RNN

딥러닝이란?

‘심층 신경망을 이용한 머신러닝 기법’

Data가 Model을 스스로 찾아내는



딥러닝의 발전 과정

1957
최초 신경망 모델
(Perceptron) 등장

ARCHIVES | 1958

NEW NAVY DEVICE LEARNS BY DOING; Psychologist Shows Embryo of Computer Designed to Read and Grow Wiser

SPECIAL TO THE NEW YORK TIMES JULY 8, 1958

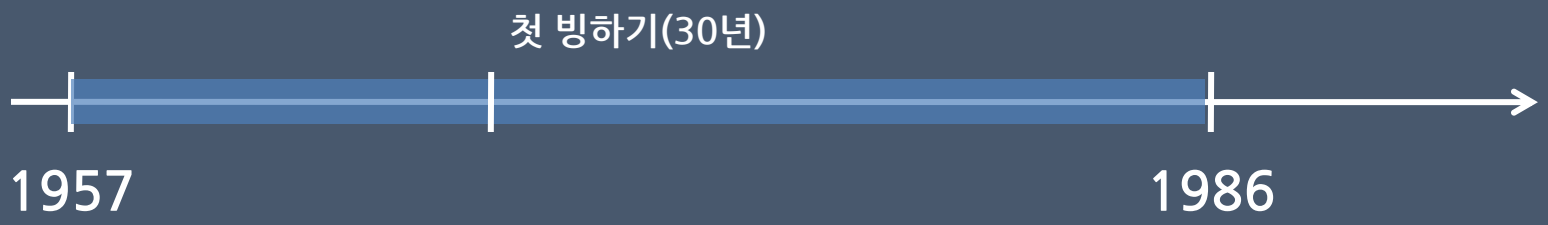
WASHINGTON, July 7 (UPI) -- The Navy revealed the embryo of an electronic computer today that it expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence.

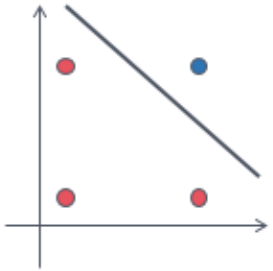
"걷고, 말하고, 보고, 쓰고, 스스로 만들어 내고 자아를 인식할 수 있을 것으로 예측"

Subscribe and see the full article in TimesMachine

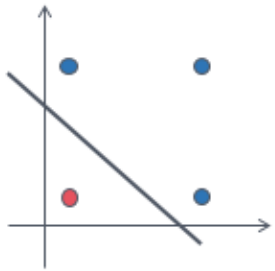
New York Times subscribers enjoy full access to TimesMachine—view 129 years of New York Times journalism, as it originally appeared. **99¢ for your first 4 weeks.**

딥러닝의 발전 과정

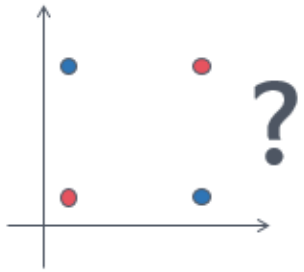




AND		
Input_A	Input_B	Output
0	0	0
0	1	0
1	0	0
1	1	1

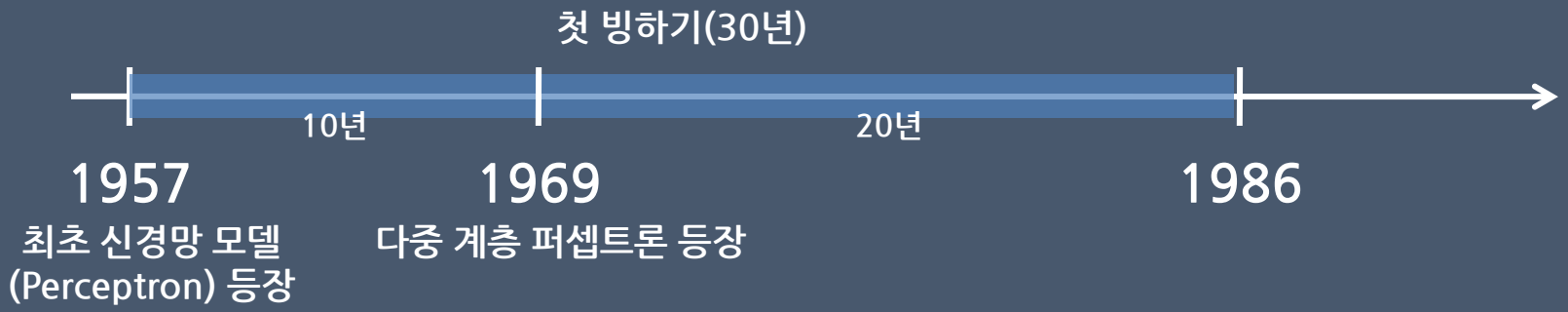


OR		
Input_A	Input_B	Output
0	0	0
0	1	1
1	0	1
1	1	1



XOR		
Input_A	Input_B	Output
0	0	0
0	1	1
1	0	1
1	1	0

딥러닝의 발전 과정



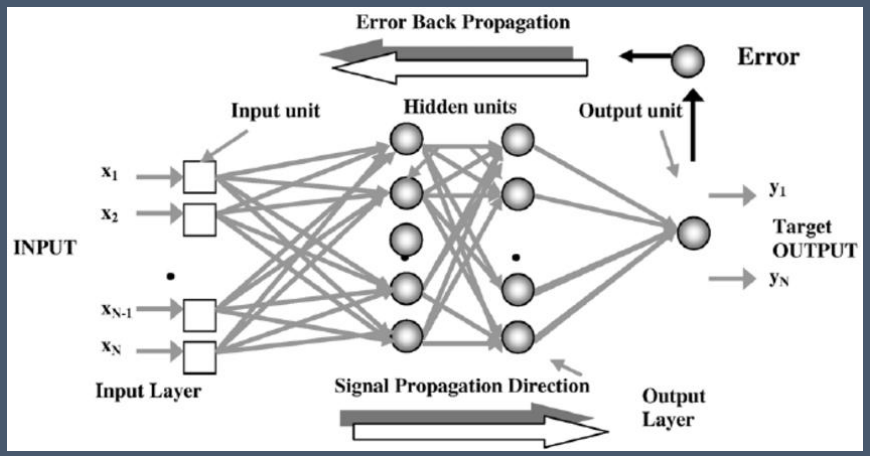
Perceptron을 다중으로 겹치면 이 문제를 해결할 수 있음을 증명

하지만 레이어가 복잡해질수록, 연산이 복잡해져서 현실적으로 parameter 값을 구하는 것이 불가능

딥러닝의 발전 과정

1986

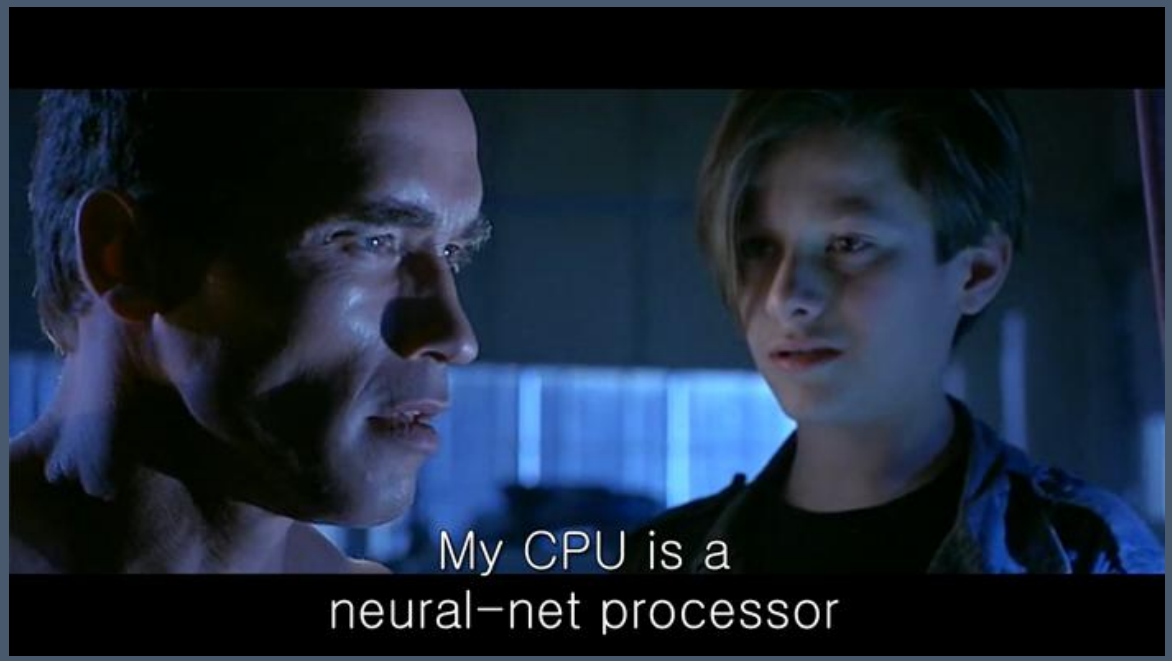
새로운 학습 방법 등장



Data가 Model을 스스로 찾아내는 Backpropagation 등장

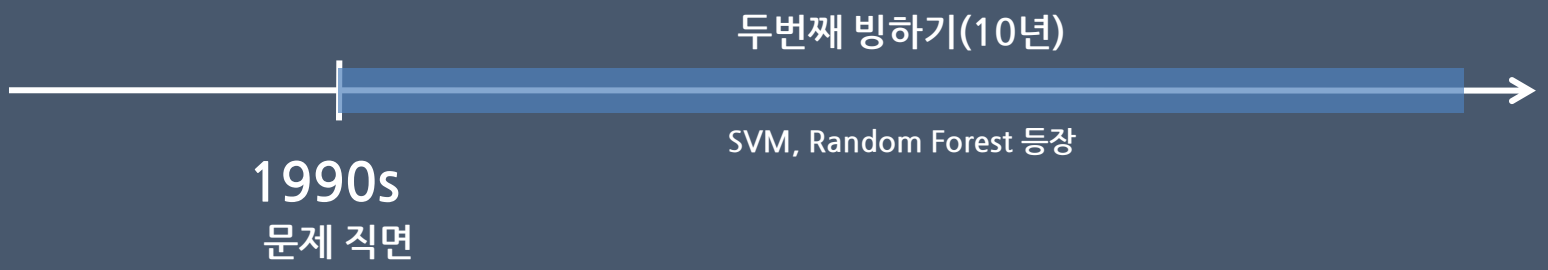
즉 앞의 진행방향에서 고쳐가는 것이 아니라 결과를 보고 뒤로 가면서 weight와 bias를 조정하는 방법 고안

딥러닝의 발전 과정



터미네이터2 심판의 날(1991)

딥러닝의 발전 과정

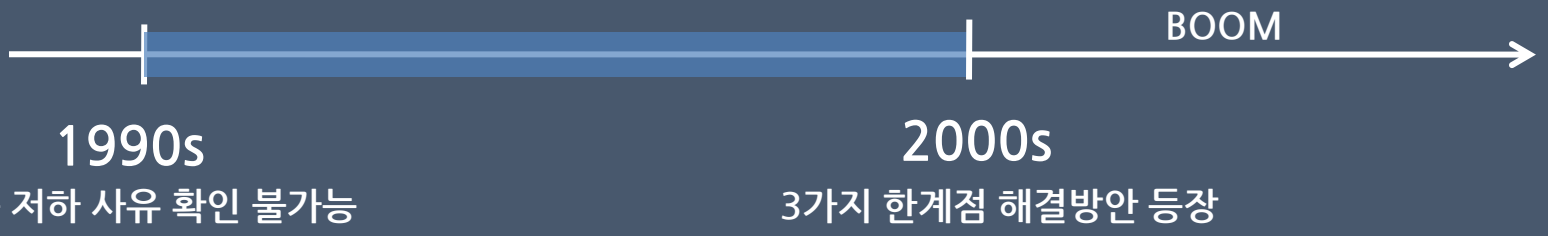


신경망이 깊어질수록 원하는 결과를 얻을 수 없다.
오히려 성능이 저하되는 경우 발생

다음의 문제를 해결 방안을 찾는데 10년이 걸림

1. Overfitting
2. Vanishing gradient
3. Too slow

딥러닝의 발전 과정



1. Overfitting
2. Vanishing Gradient
3. Too slow

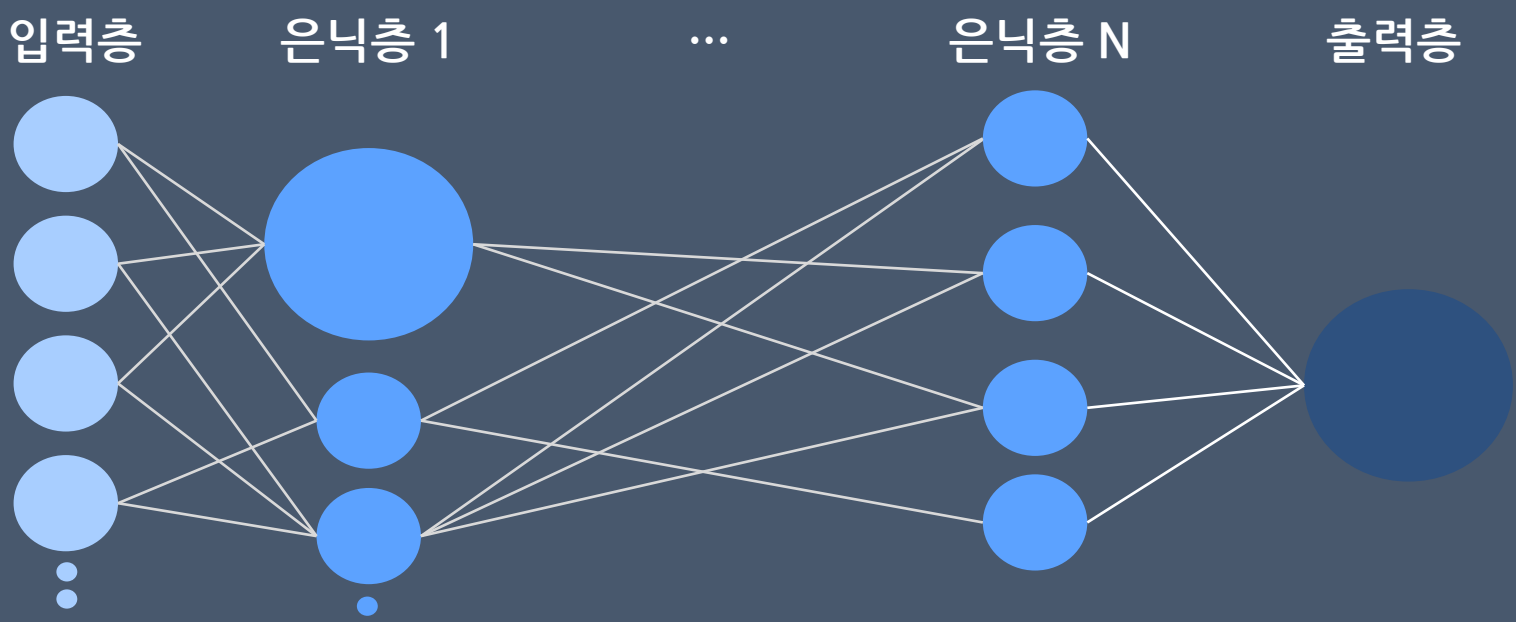
✖

GPU를 활용한 연산 시도

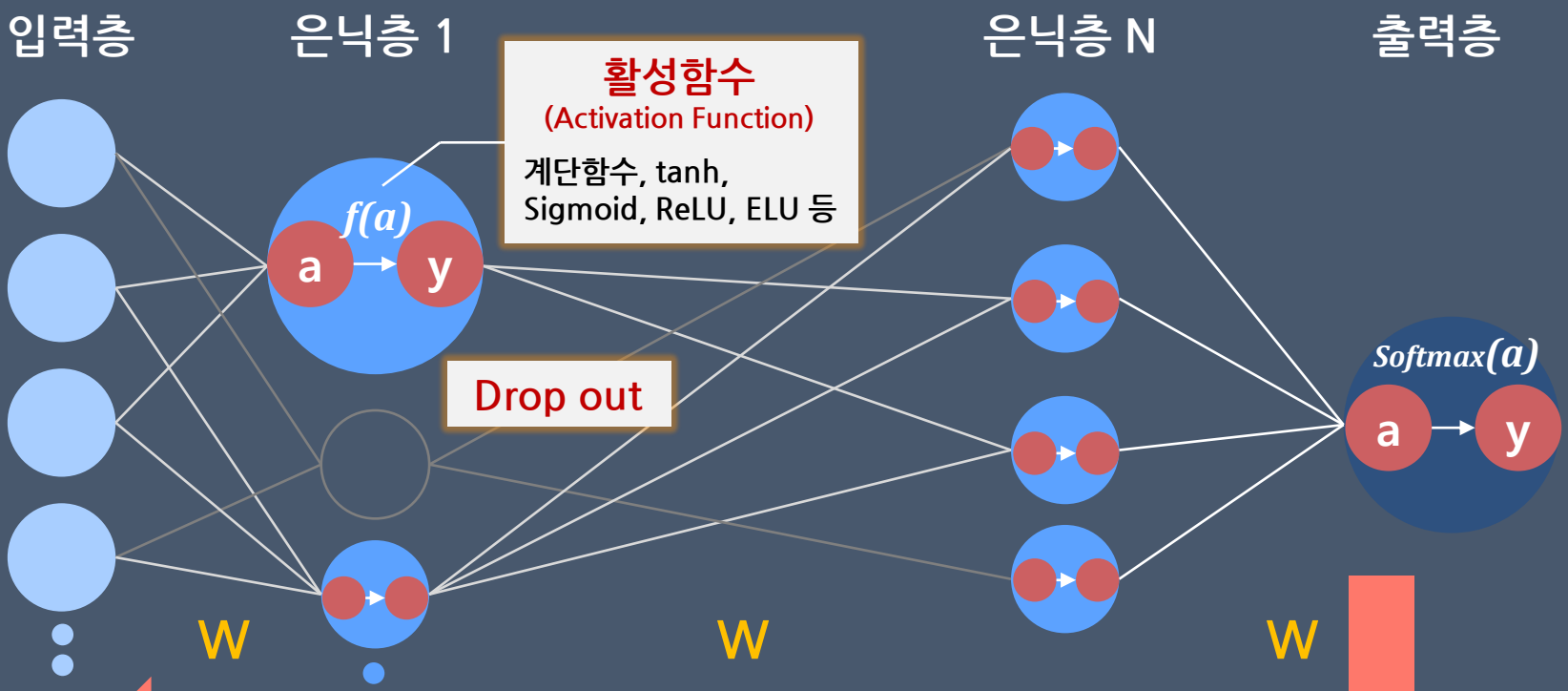
알고리즘의 발전

딥러닝 프로세스 개요


1957
최초 신경망 모델
(Perceptron) 등장
1969
다중 계층 퍼셉트론 등장
1986
새로운 학습방법
(Backpropagation)
2000s
3가지 한계점 해결방안


입력층
은닉층 1
…
은닉층 N
출력층

딥러닝 프로세스 개요



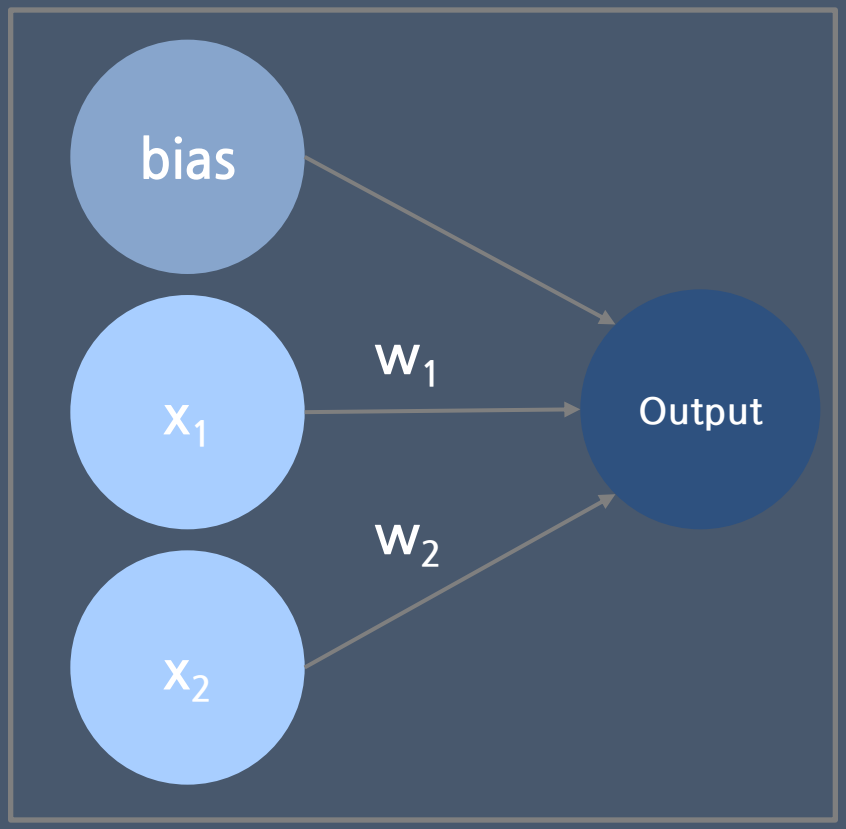
층 개수 (Layer size)	정규화 (Normalization)	Batch Size	epoch
노드 개수 (Unit size)	손실함수 (Loss Function)	Learning rate	

단층/다층 신경망의 구조

단층/다층 신경망의 구조

신경망의 기본 구조부터 알아보자

키워드 | #input | #weight | #bias | #output | #activation_function

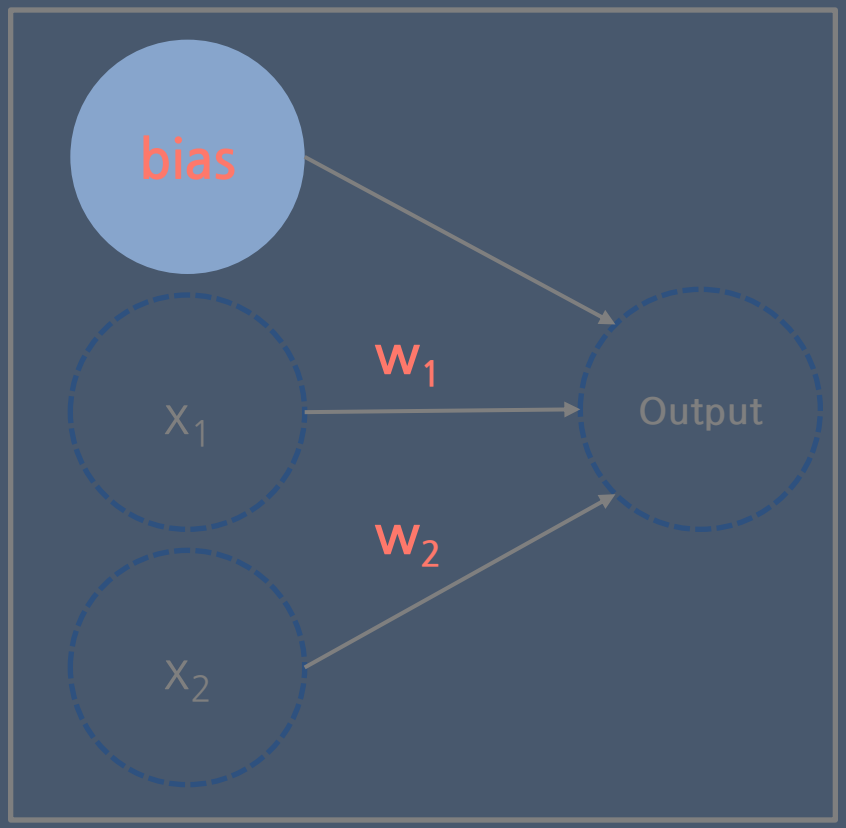


$$Output = x_1*w_1 + x_2*w_2 + bias$$

단층/다층 신경망의 구조

신경망의 기본 구조부터 알아보자

키워드	#input	#weight	#bias	#output
	#activation_function			

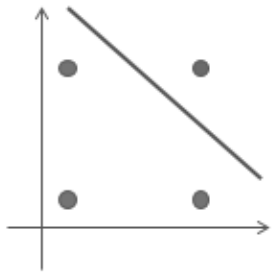


Weight와 bias를 알아내는 것이 신경망의 학습 목표!

단층/다층 신경망의 구조

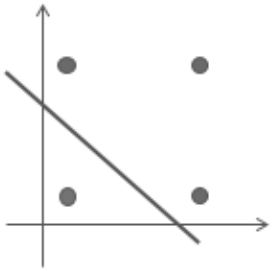
신경망의 기본 구조부터 알아보자

키워드 | #input | #weight | #bias | #output | #activation_function



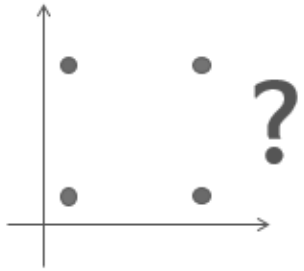
AND

AND		
Input_A	Input_B	Output
0	0	0
0	1	0
1	0	0
1	1	1



OR

OR		
Input_A	Input_B	Output
0	0	0
0	1	1
1	0	1
1	1	1



XOR

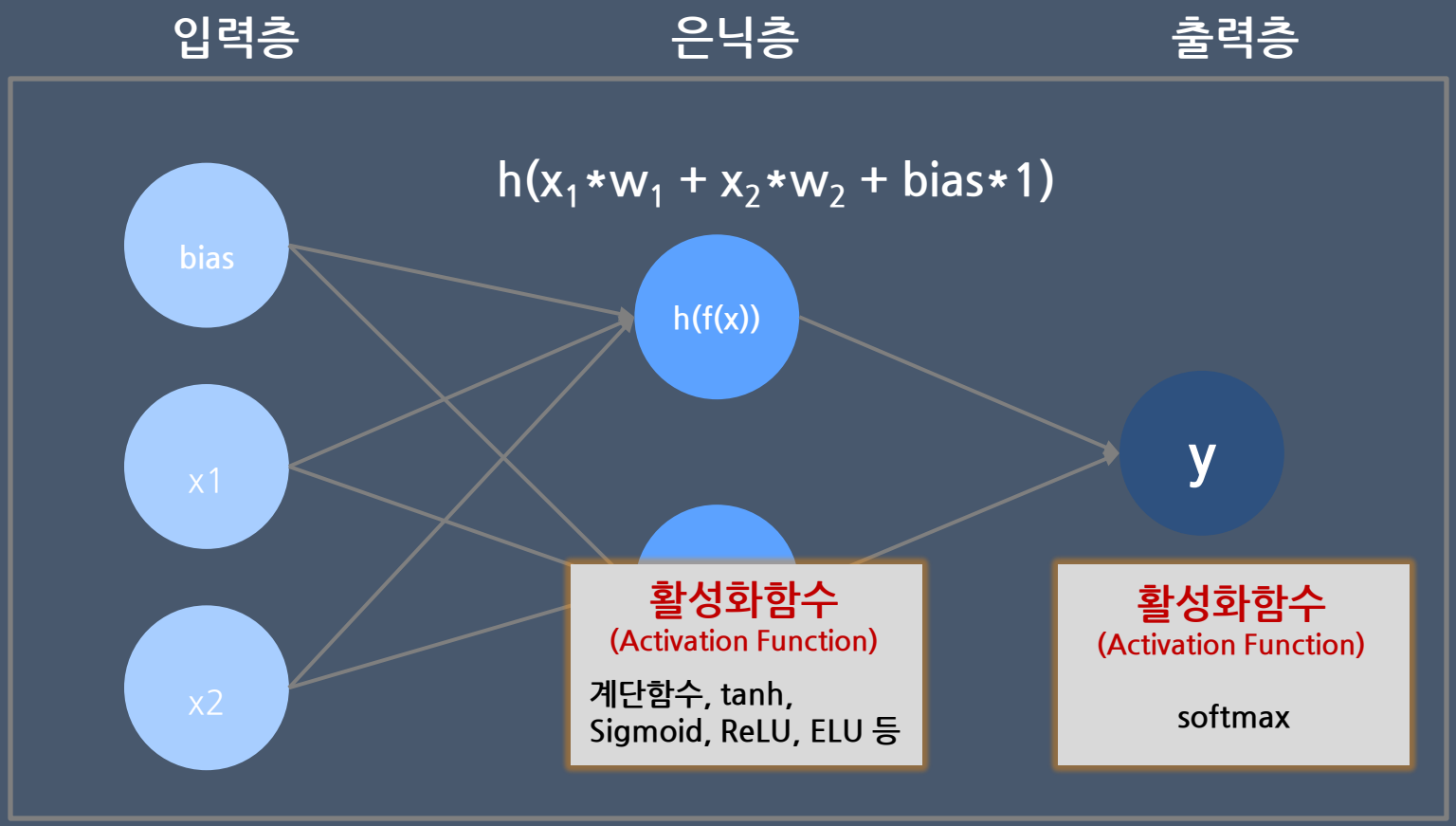
XOR		
Input_A	Input_B	Output
0	0	0
0	1	1
1	0	1
1	1	0

Weight와 bias를 알아내는 것이 신경망의 학습 목표!

단층/다층 신경망의 구조

은닉층의 추가, Deep Learning

키워드	#hidden_layer	#activation_function

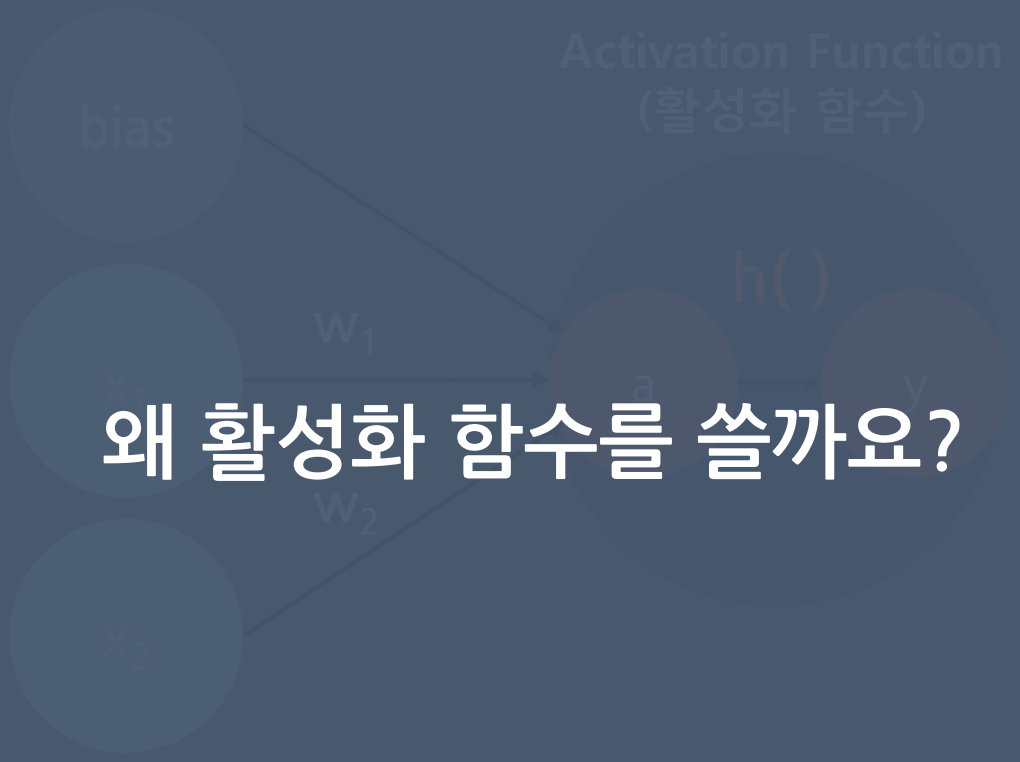


은닉층이 2개 이상 = Deep Learning

단층/다층 신경망의 구조

신경망의 기본 구조부터 알아보자

키워드 | #input | #weight | #bias | #output | #activation_function



왜 활성화 함수를 쓸까요?

단층/다층 신경망의 구조

신경망의 기본 구조부터 알아보자

키워드 | #input | #weight | #bias | #output | #activation_function

Activation Function
(활성화 함수)

bias

활성화 함수를 쓰지 않으면,

Linear regression과 똑같다.

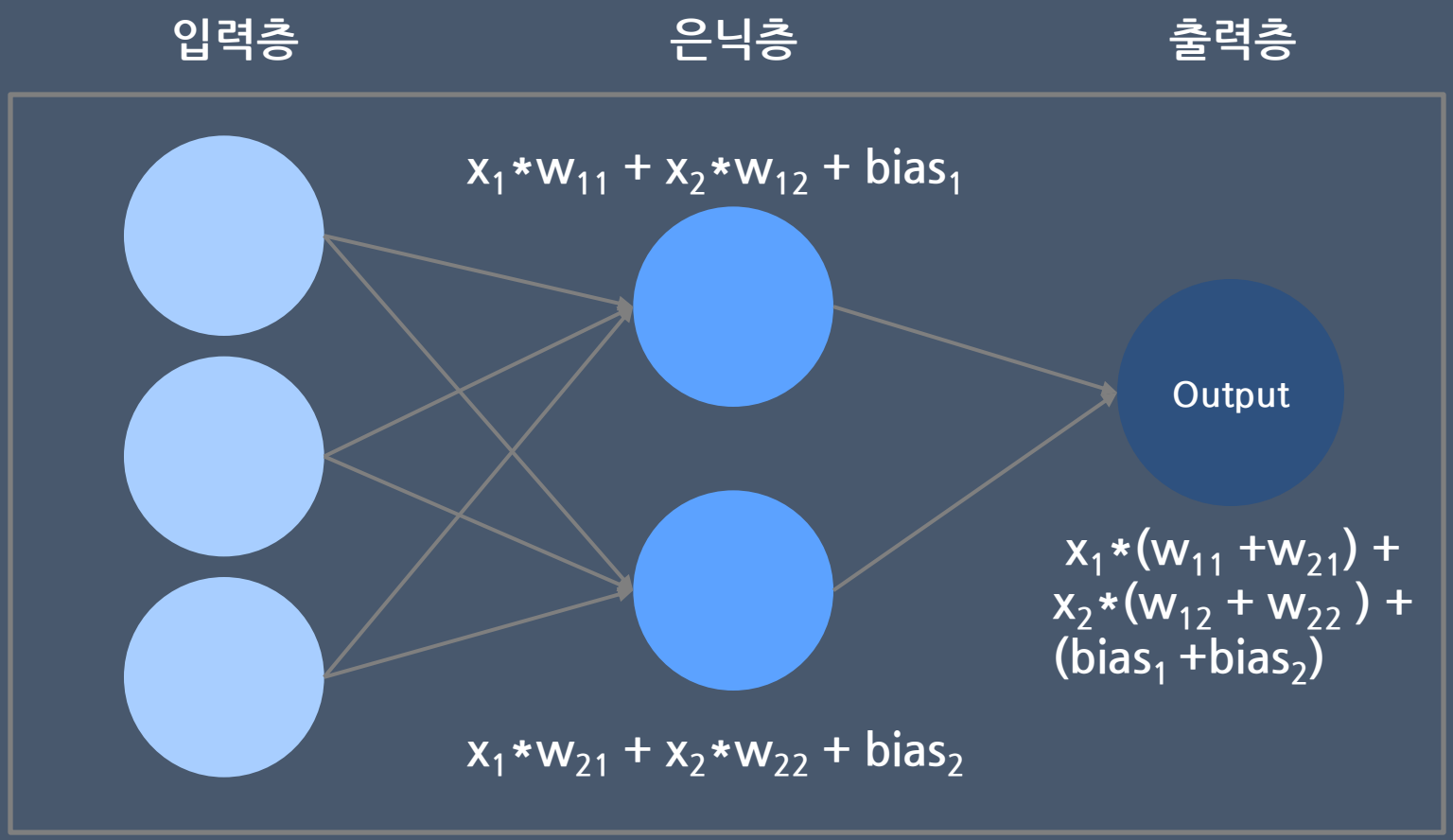
(결국엔 y = A • X + b의 형태)

Output = $h(x_1*w_1 + x_2*w_2 + x_3*w_3 + bias)$

단층/다층 신경망의 구조

은닉층의 추가, Deep Learning

키워드	#hidden_layer	#activation_function



단층/다층 신경망의 구조

신경망의 기본 구조부터 알아보자

키워드 | #input | #weight | #bias | #output | #activation_function

활성화 함수를 선형 함수로 적용해도,

Linear regression과 똑같다.

(결국엔 y = A • X + b의 형태)

단층/다층 신경망의 구조

신경망의 기본 구조부터 알아보자

키워드 | #input | #weight | #bias | #output | #activation_function

비선형 함수(non-linear function)

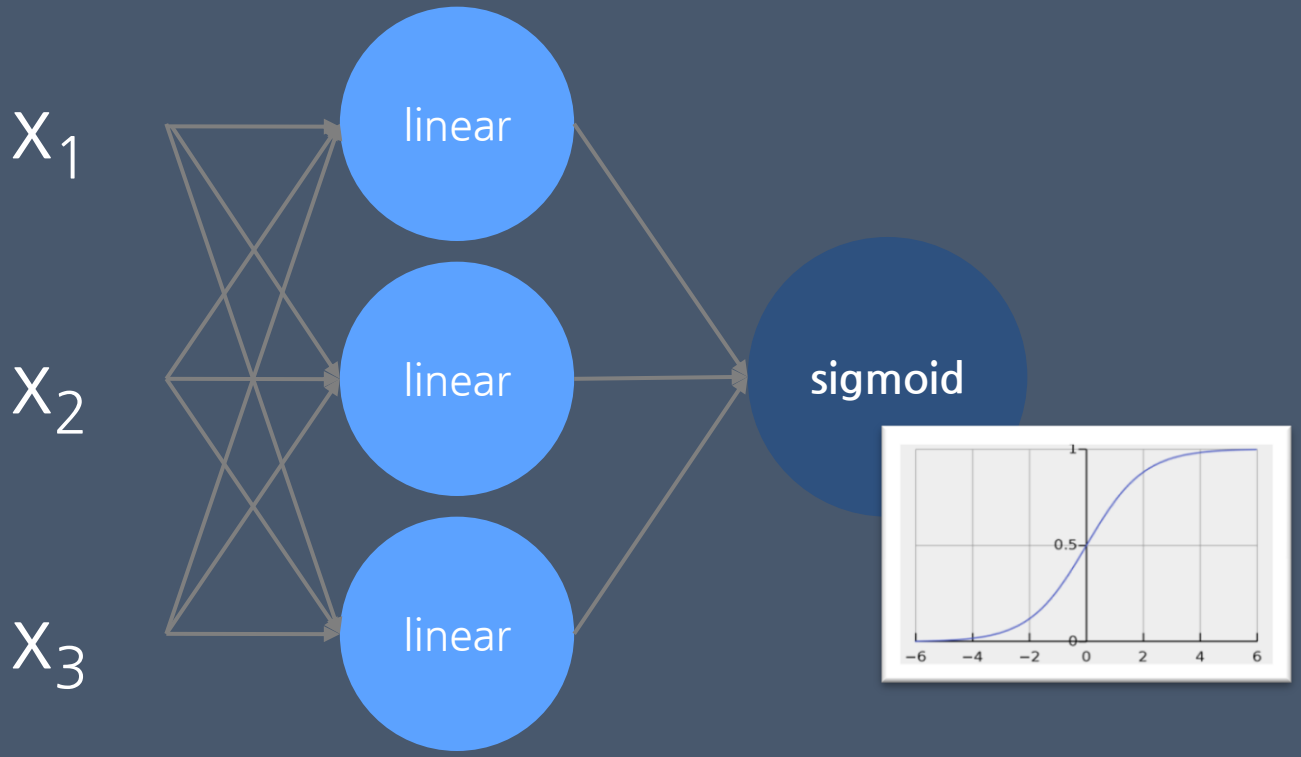
딥러닝의 핵심 내용은
선형 함수로 표현하지 못하던 비선형 영역을 표현

단층/다층 신경망의 구조

신경망의 기본 구조부터 알아보자

키워드 | #input | #weight | #bias | #output | #activation_function

Q. 은닉층에 선형함수 적용, 출력층에 비선형함수 적용

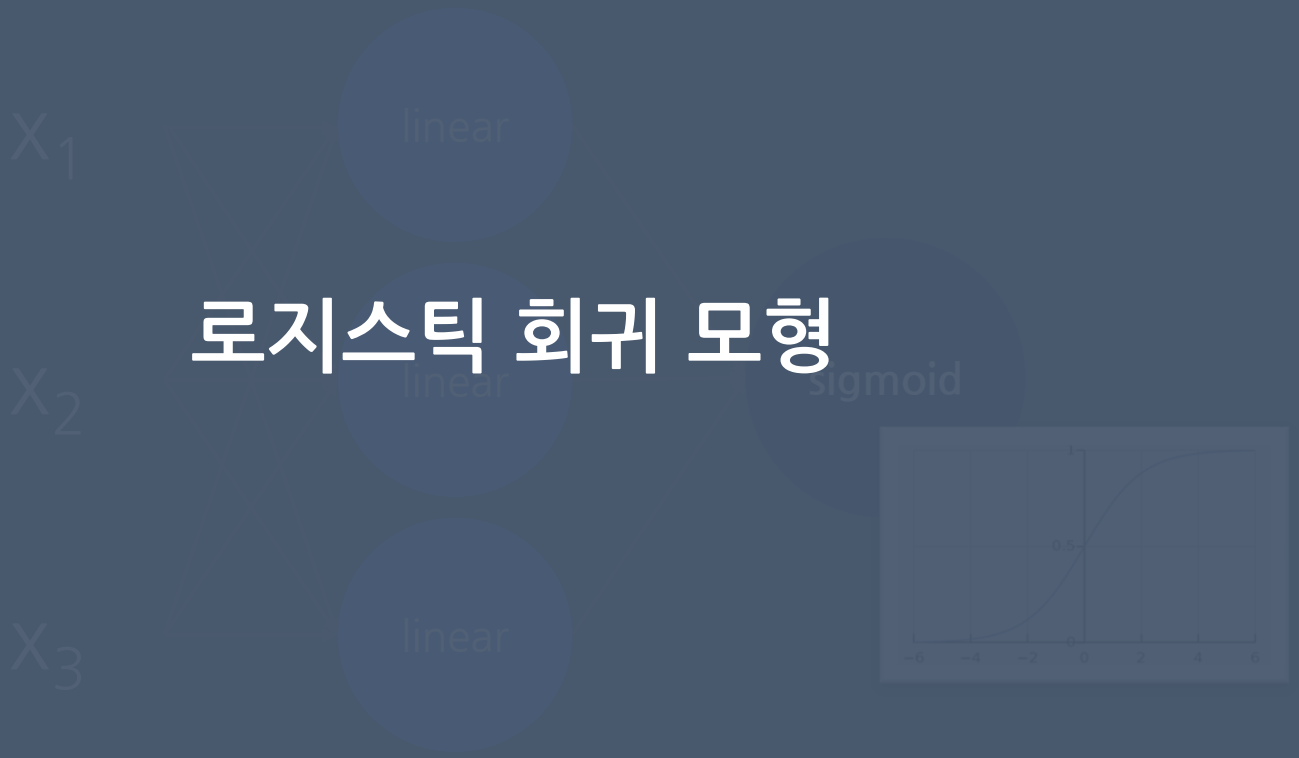


단층/다층 신경망의 구조

신경망의 기본 구조부터 알아보자

키워드 | #input | #weight | #bias | #output | #activation_function

Q. 은닉층에 선형함수 적용, 출력층에 비선형함수 적용



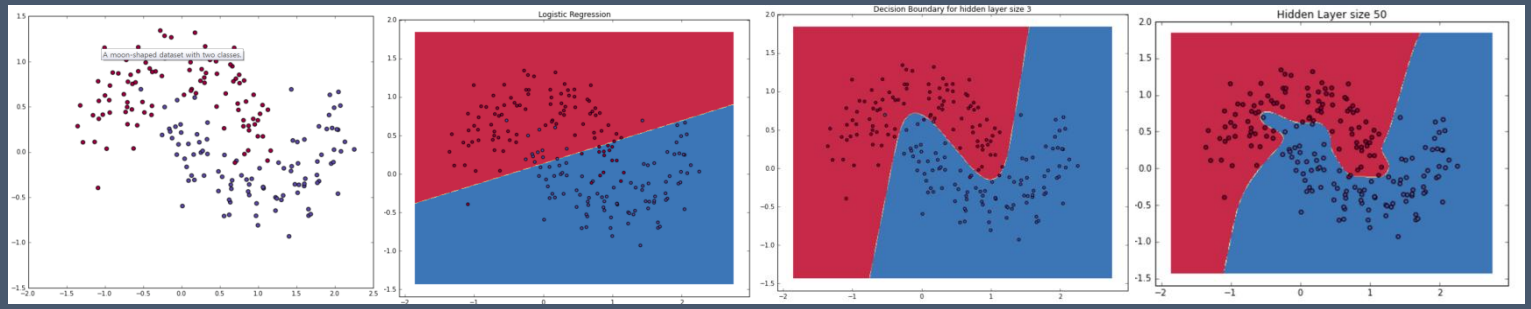
로지스틱 회귀 모형

Non-linear 활성화 함수로 이루어진

여러 은닉층의 결합의

결과가 비선형 영역을 표현

그렇다면 레이어를 더하고 더하면
높은 성능의 모델을
만들 수 있지 않을까요?



데이터 로지스틱 회귀모형 NN(은닉층3개) NN(은닉층50개)

3가지 한계점

1. Overfitting 과하거나
2. Vanishing Gradient 덜하거나
3. Too slow 느리거나

1. Overfitting 과하거나
2. Vanishing Gradient 덜하거나
3. Too slow 느리거나

3가지 한계점

Overfitting 이란?

키워드 | #overfitting | #vanishing_gradient

3가지 한계점

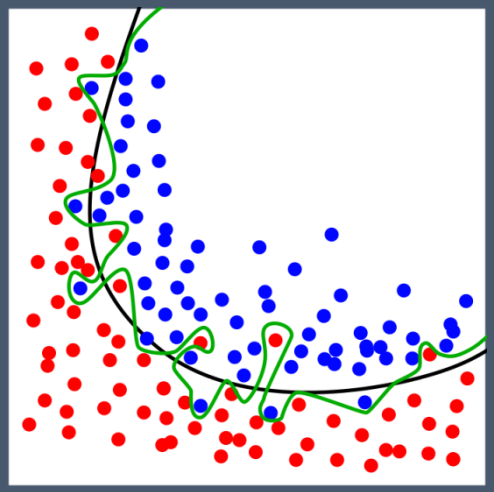
Overfitting 이란?

키워드 | #overfitting | #vanishing_gradient

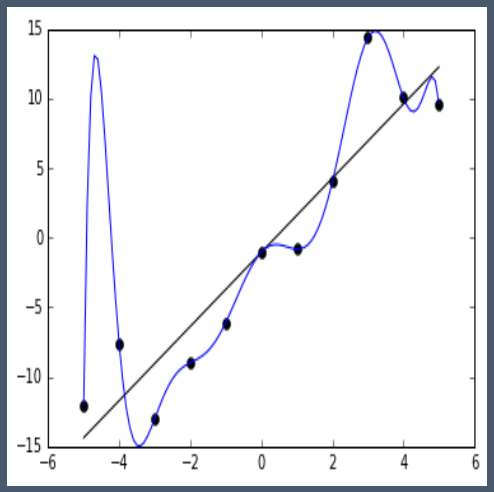
학습 데이터에만 잘 맞는 모델

왜 생길까?

CASE 1



CASE 2



3가지 한계점

Overfitting 해결 방안

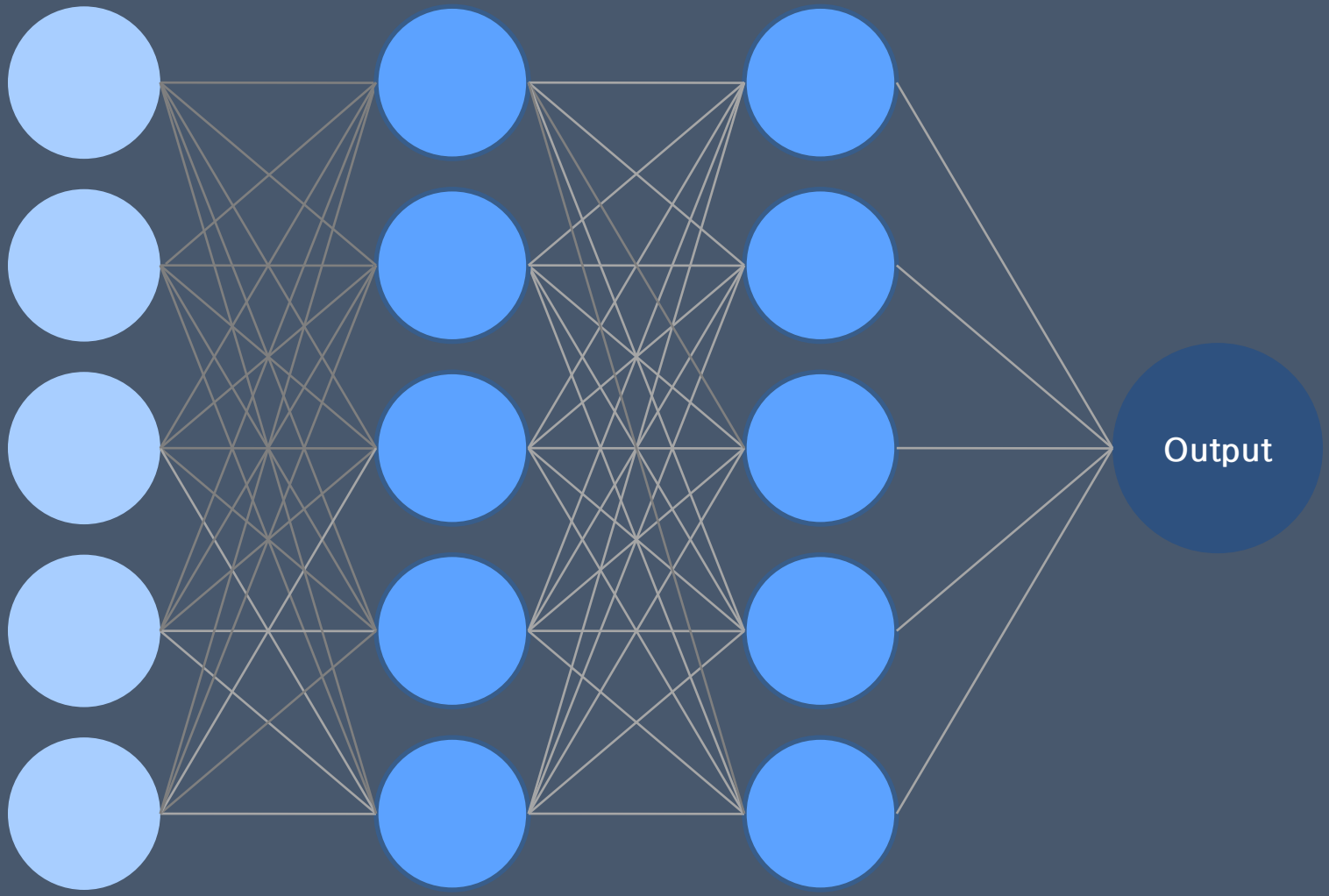
키워드 | #dropout

랜덤하게 뉴런을 끊음으로써, 모델을 단순하게 만든다

3가지 한계점

Overfitting 해결 방안

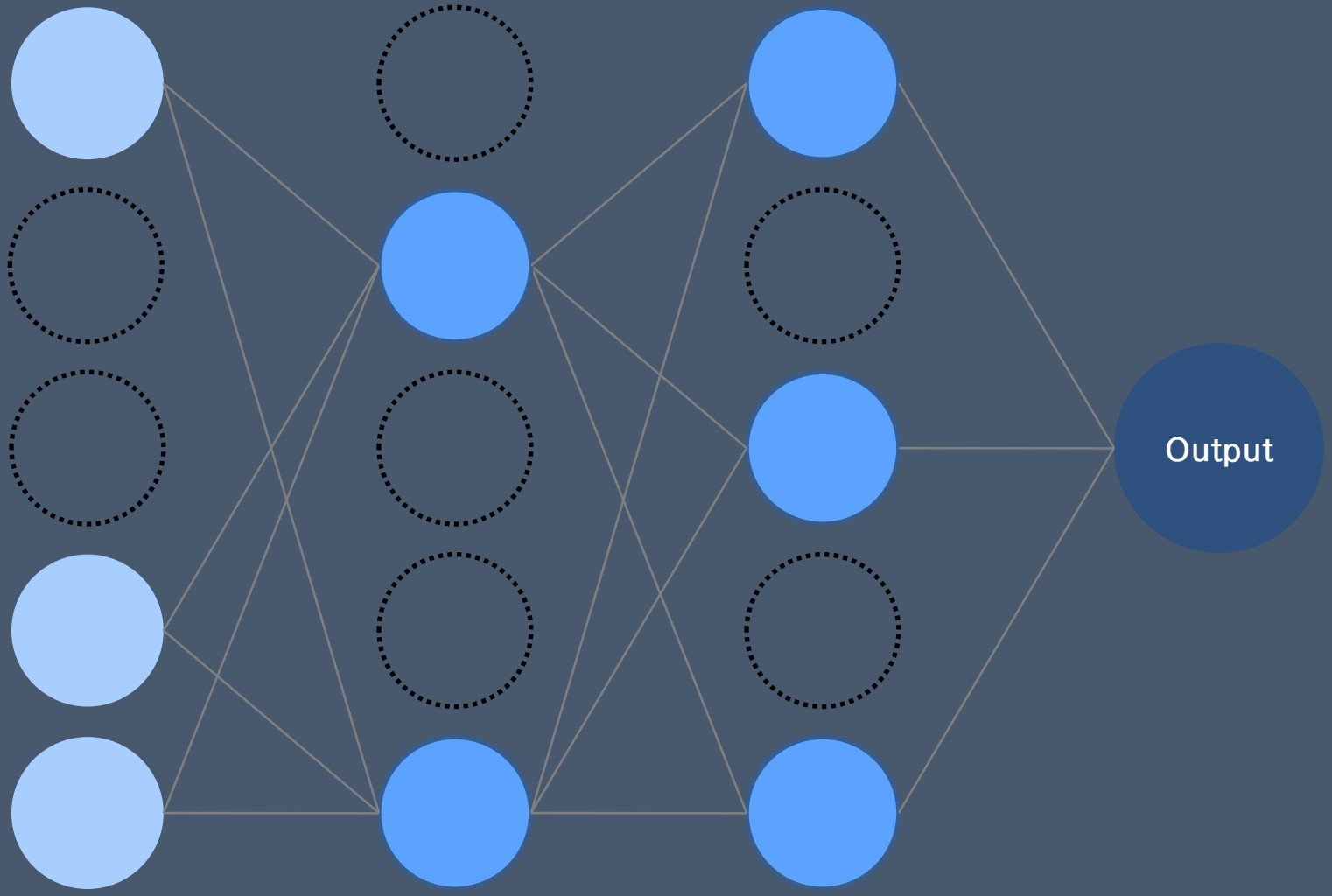
키워드 #dropout



3가지 한계점

Overfitting 해결 방안

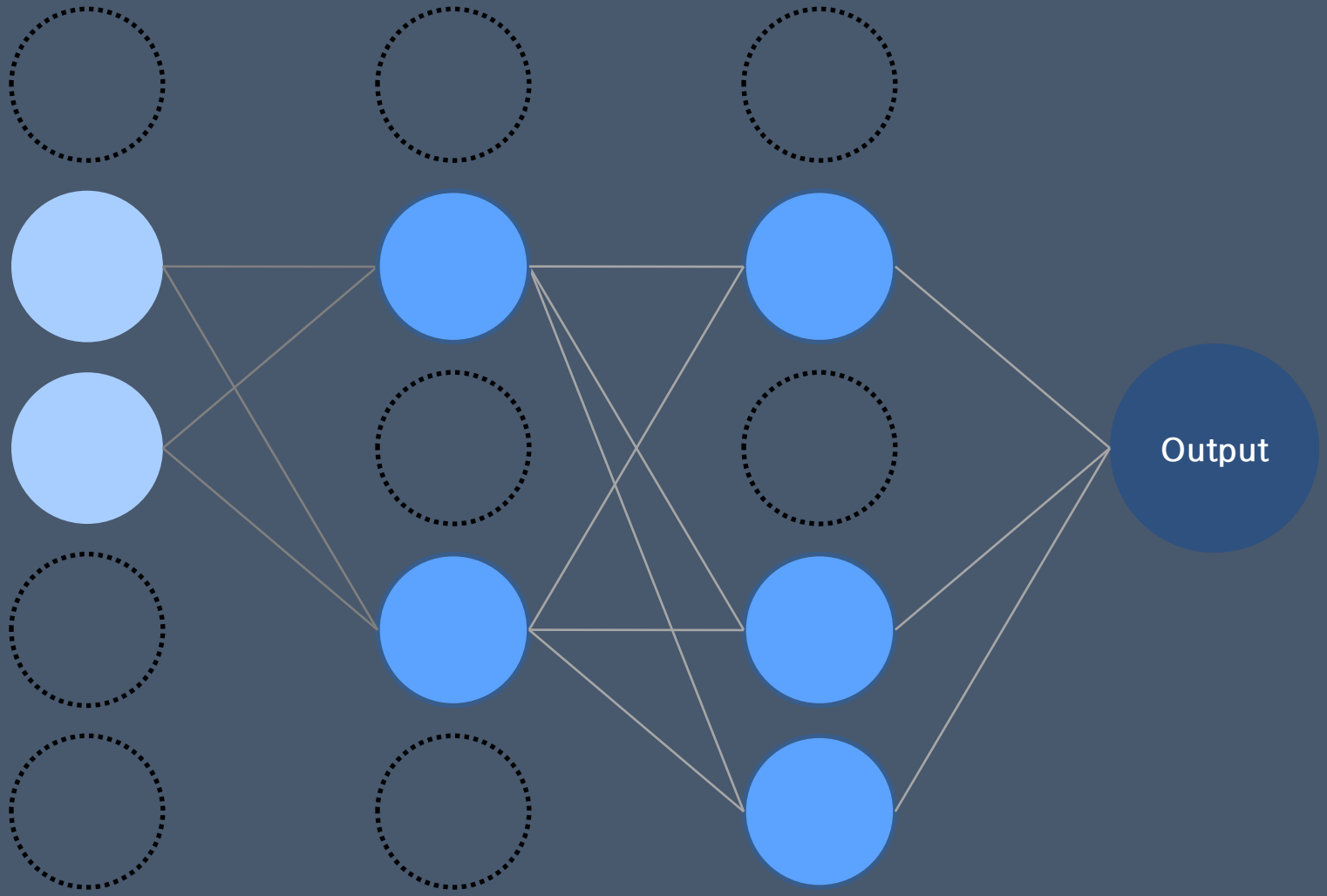
키워드 #dropout



3가지 한계점

Overfitting 해결 방안

키워드 #dropout



1. Overfitting 과하거나

2. Vanishing Gradient 덜하거나

3. Too slow 느리거나

1. Overfitting 과하거나

2. Vanishing Gradient 덜하거나

사라지는 기울기

3. Too slow 느리거나

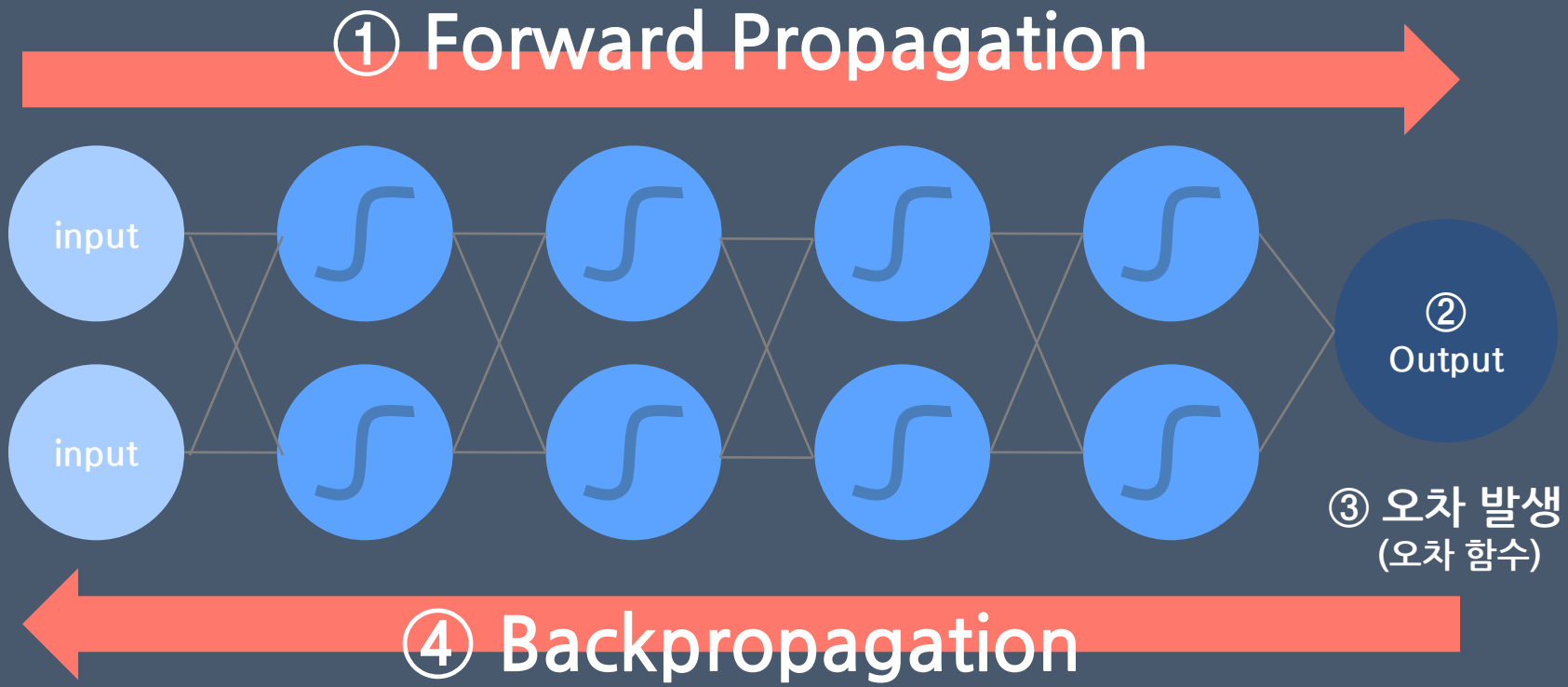
뉴럴넷의 학습 방법부터 알아보자

Backpropagation

Backpropagation

뉴럴넷의 학습 방법

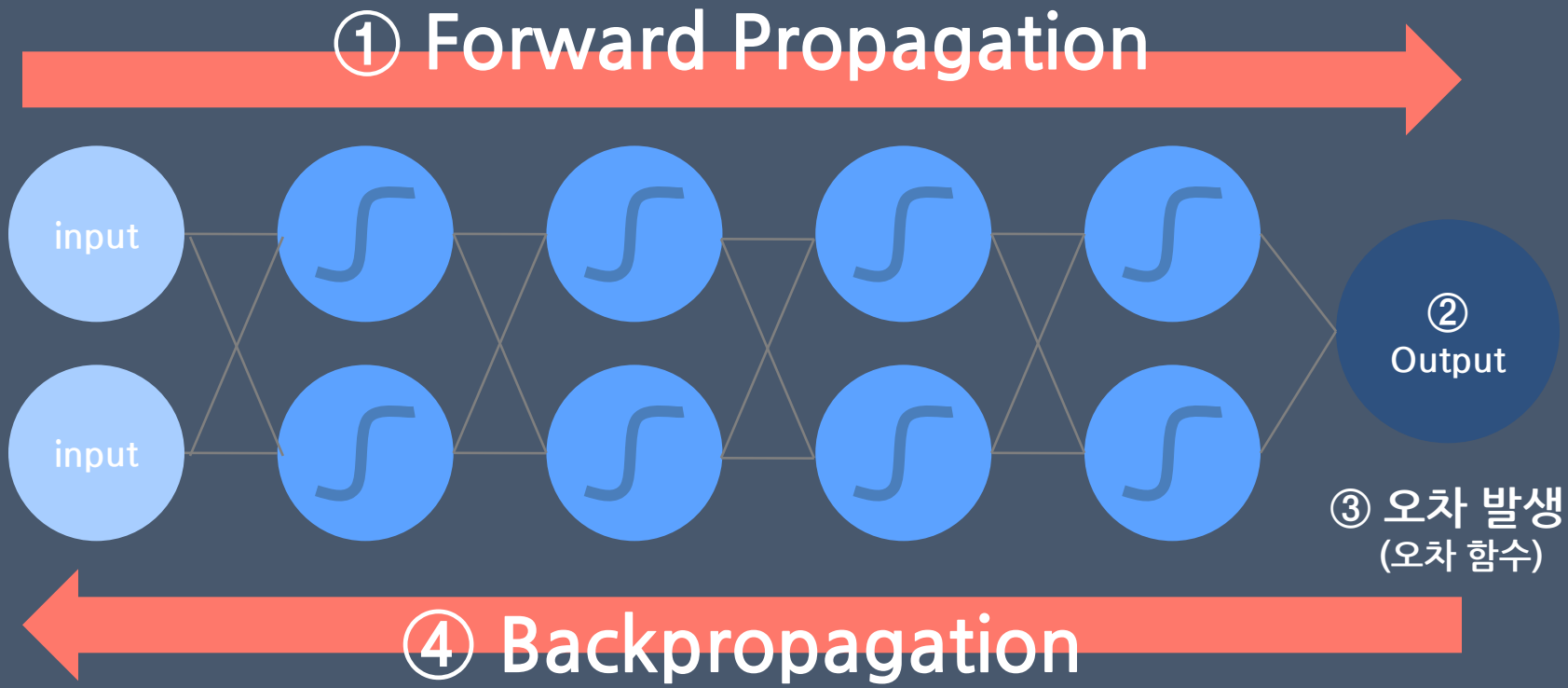
키워드 #backpropagation



Backpropagation

뉴럴넷의 학습 방법

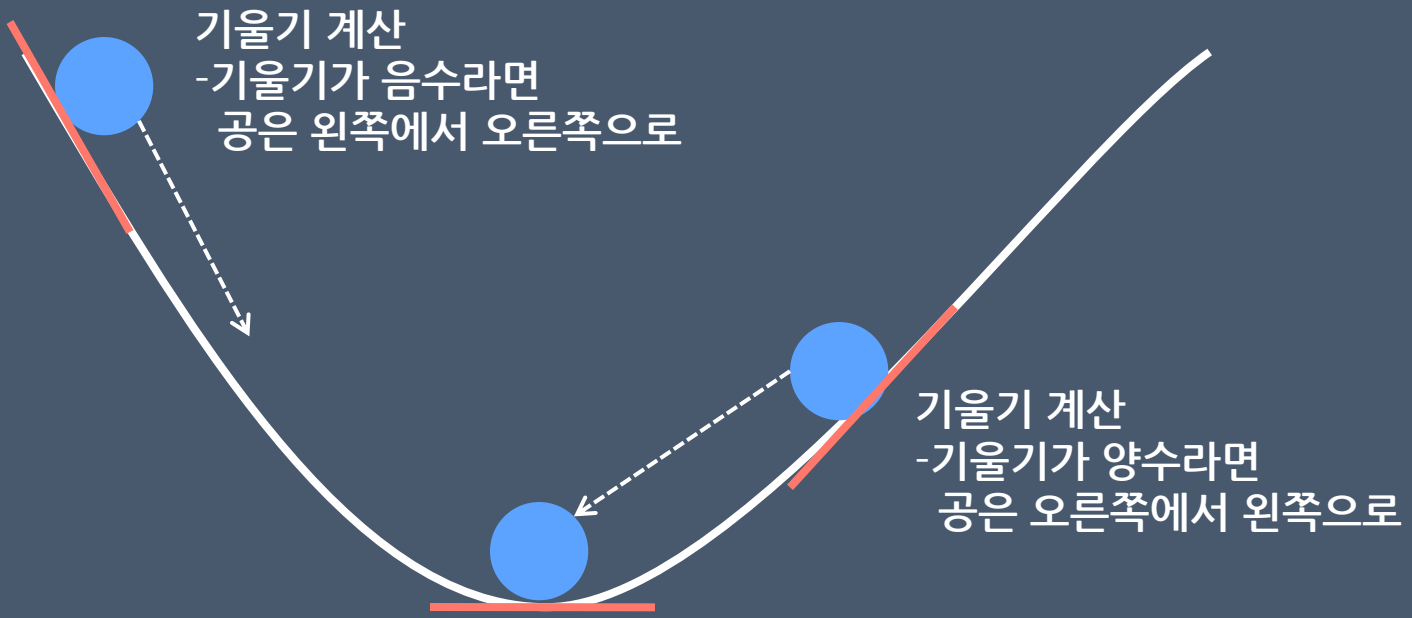
키워드 #backpropagation



Backpropagation

‘틀린 정도의 기울기’?

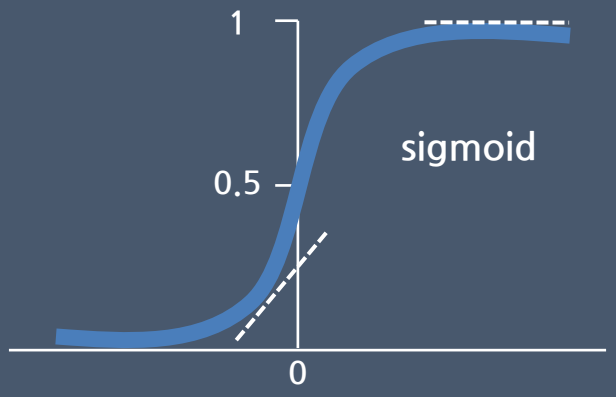
키워드 #backpropagation



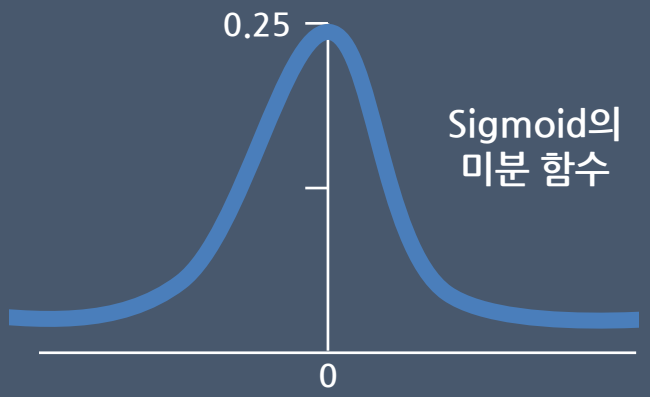
3가지 한계점

Vanishing Gradient란?

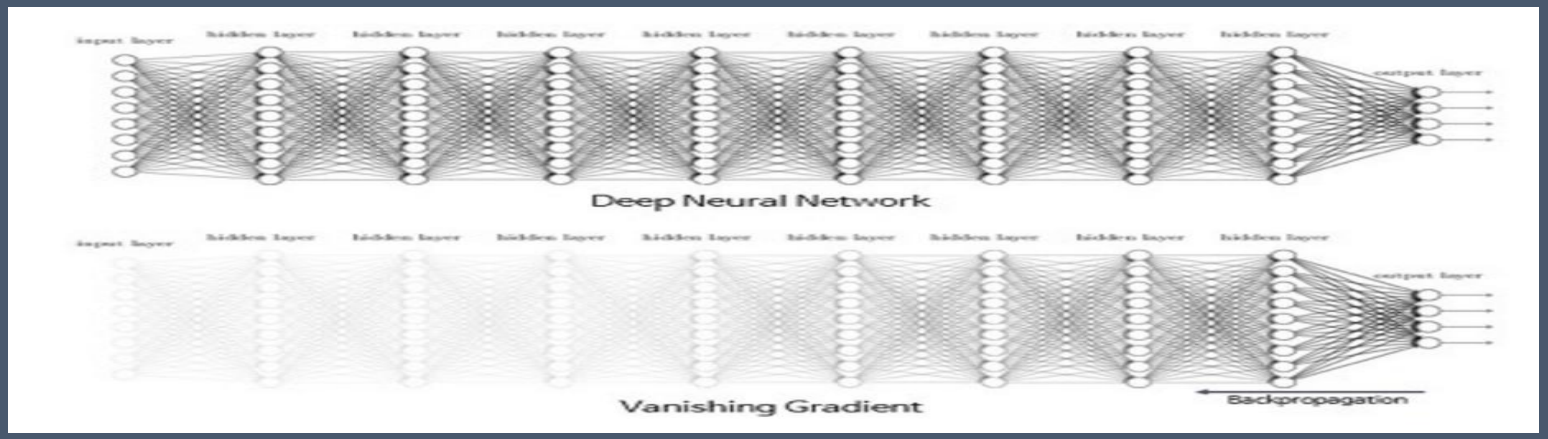
'틀린 정도의 기울기' = 미분값



새로운가중치 = f(기존가중치, 미분값)



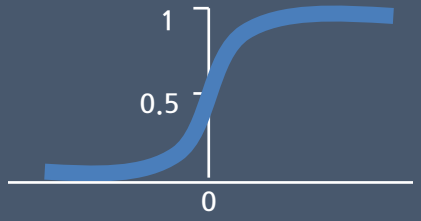
Vanishing gradient로 인해 학습되는 양이 0에 가까워져,
학습이 더디게 진행되다 오차를 더 줄이지 못하고 그 상태로 수렴하게 됨



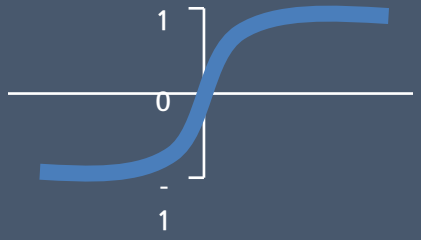
3가지 한계점

Vanishing Gradient 해결방안

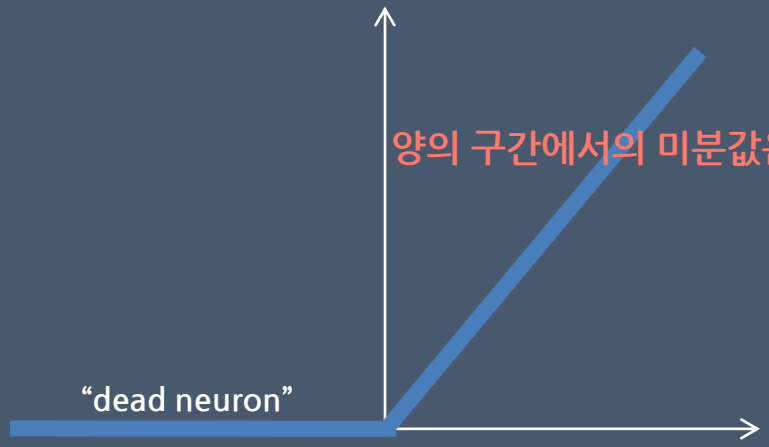
Sigmoid



Tanh



ReLU(Rectified Linear Unit)



$$h(x) = \begin{cases} x & (x > 0) \\ 0 & (x \leq 0) \end{cases}$$

수렴속도가 시그모이드류 함수 대비 6배나 빠르다!

1. Overfitting 과하거나

2. Vanishing Gradient 덜하거나

3. Too slow 느리거나

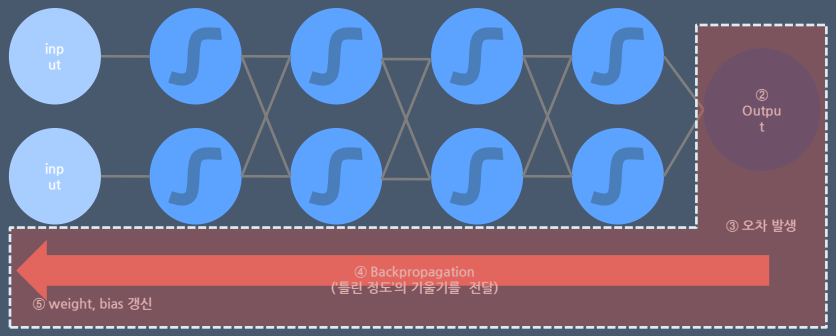
3가지 한계점

너무 느리다

키워드	#cost_function	#gradient_descent
	#learning_rate	

우리의 목적은 최적화된 bias, weight를 찾는 것

어떻게?



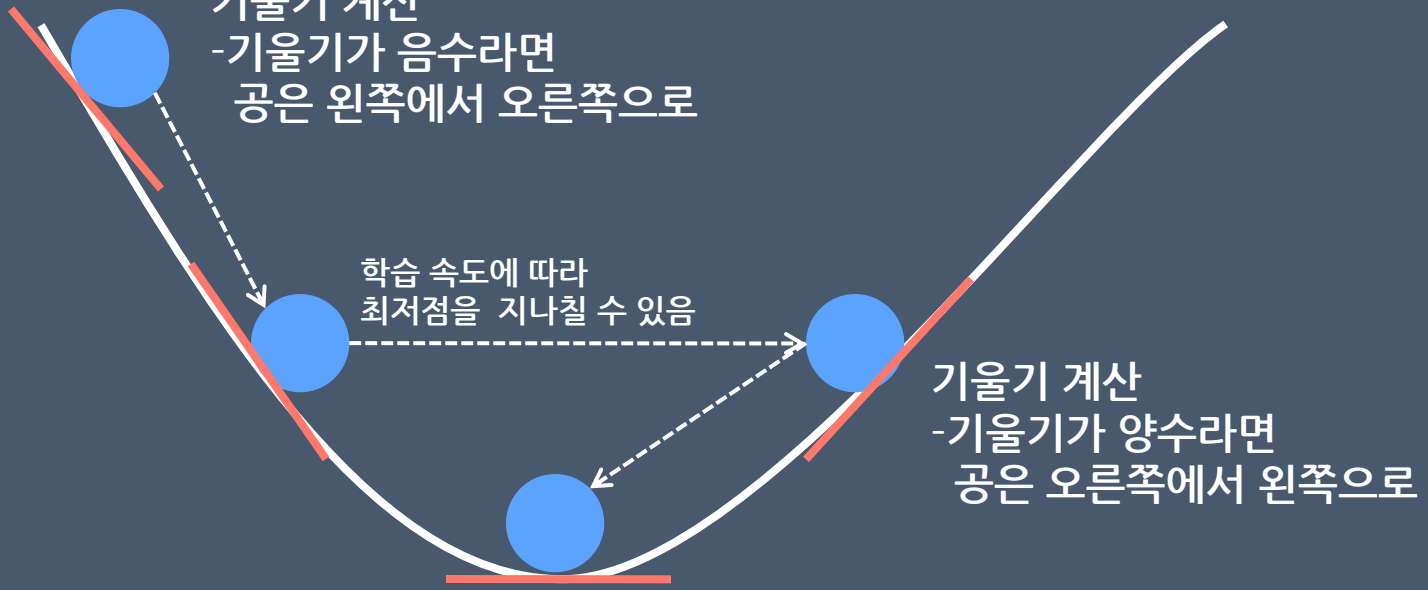
현 가중치로 산출된 cost function의 기울기(gradient)를 구해서 cost를 낮추는 방향(descent)으로 업데이트하고 이는 일정 속도(learning rate)를 기준으로 한다.

$$W = W - \alpha \nabla J(W,b)$$

3가지 한계점

너무 느리다

키워드	#cost_function	#gradient_descent
	#learning_rate	



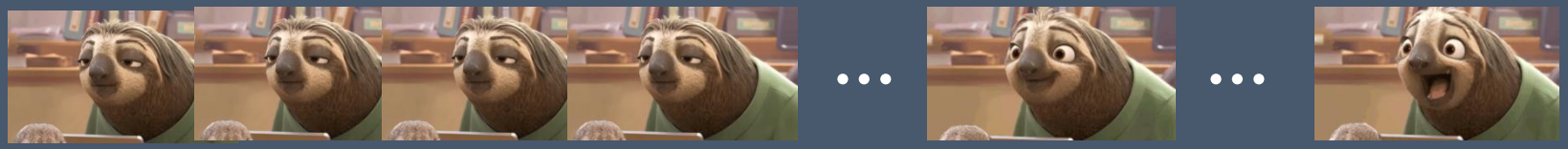
기울기 = 0의 구간에서 멈춤

3가지 한계점

너무 느리다

키워드 | #cost_function | #gradient_descent | #learning_rate

시작!



Step 1. 초기 parameters에 train set 전체 데이터 넣고 에러 산출
Step 2. 현재 위치를 미분하여 에러를 줄일 수 있는 방향 찾고
Step 3. 일정 학습 속도만큼 움직이자
Step 4. 초기값 에러를 반영한 두 번째 parameter값 산출
Step 5. 새로운 parameter를 가지고 다시 train set 전체 데이터를 넣고 에러 산출
Step 6. 반복

진짜 최적화된 bias와 weight는 언제 찾지?

SGD(Stochastic Gradient Descent)

-확률적 경사 하강법

‘확률적으로 무작위로 골라낸 데이터’에 대해

수행하는 경사 하강법

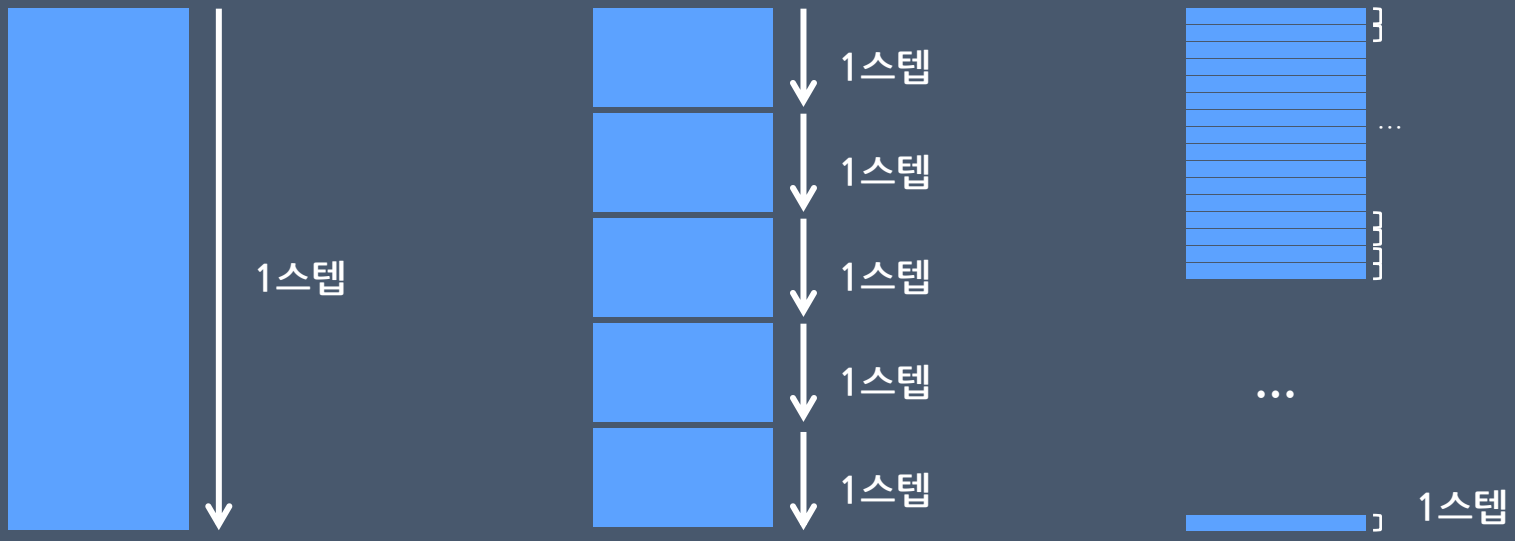
https://www.coursera.org/learn/machine-learning/lecture/DoRHJ/stochastic-gradient-descent

3가지 한계점

Stochastic Gradient Descent

키워드 | #SGD

Gradient Descent	Mini-batch Gradient Descent	Stochastic Gradient Descent
$W = W-\alpha\nabla J(W,b)$	$W = W-\alpha\nabla J(W,b,x^{(z:z+bs)},y^{(z:z+bs)})$	$W = W-\alpha\nabla J(W,b,x^{(z)},y^{(z)})$



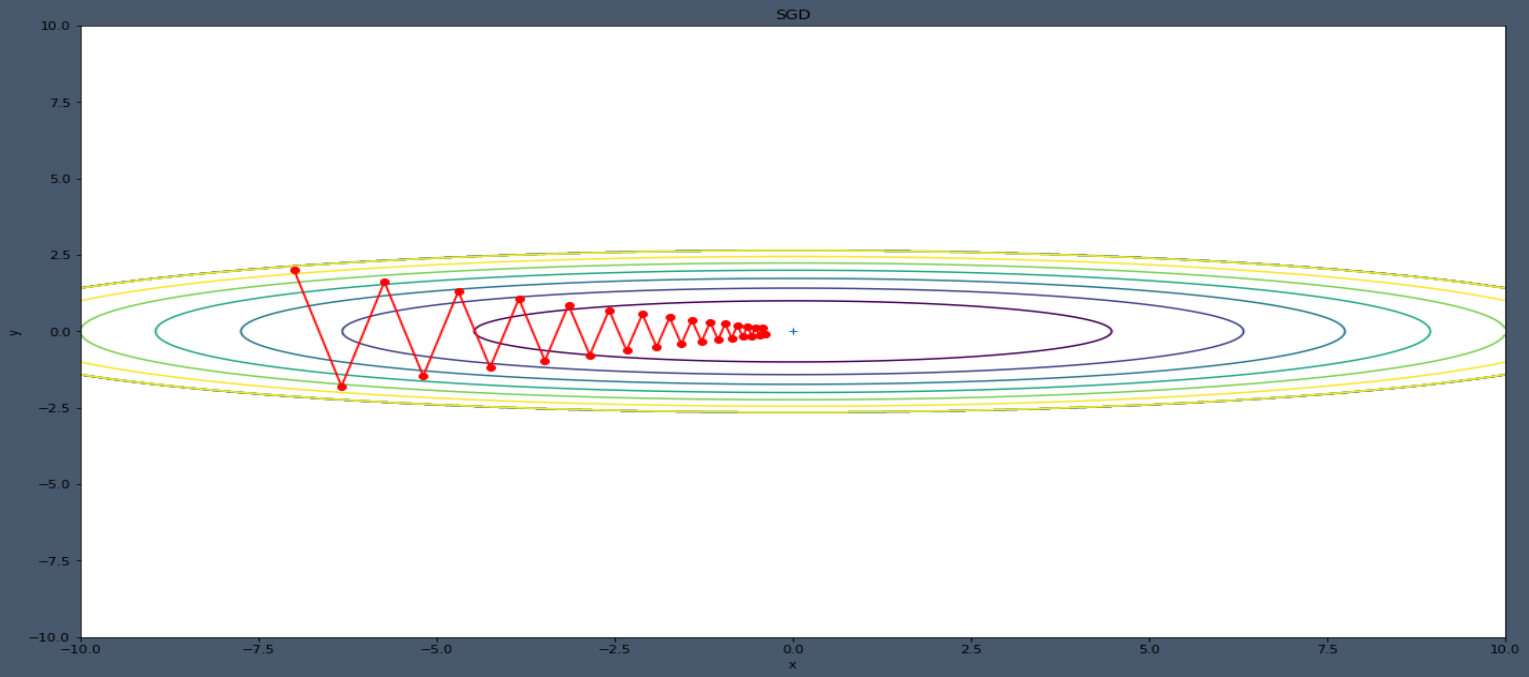
3가지 한계점

SGD의 문제점

키워드	#SGD

SGD처럼 학습을 한다면

당연히 왔다리 갔다리 하는 문제가 발생



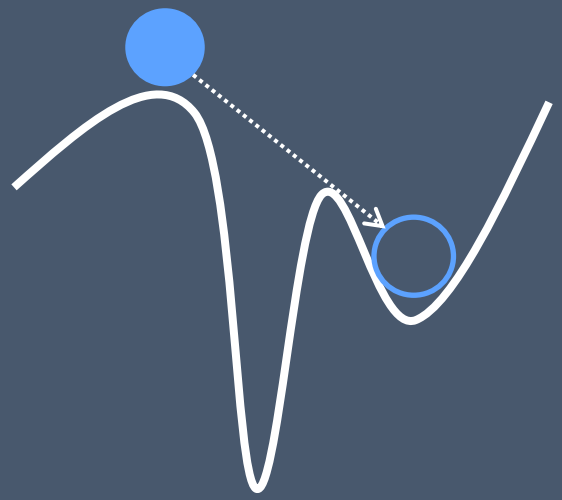
3가지 한계점

SGD의 대안

키워드	#learning_rate	#momentum

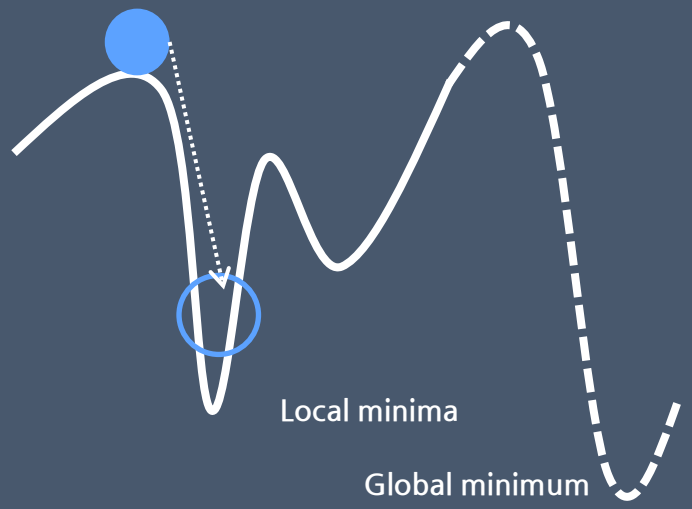
그렇다면,

학습속도를 조절해보고



너무 빨리 지나가서 최저점을 지나치게 된다

운동량을 고려해보자

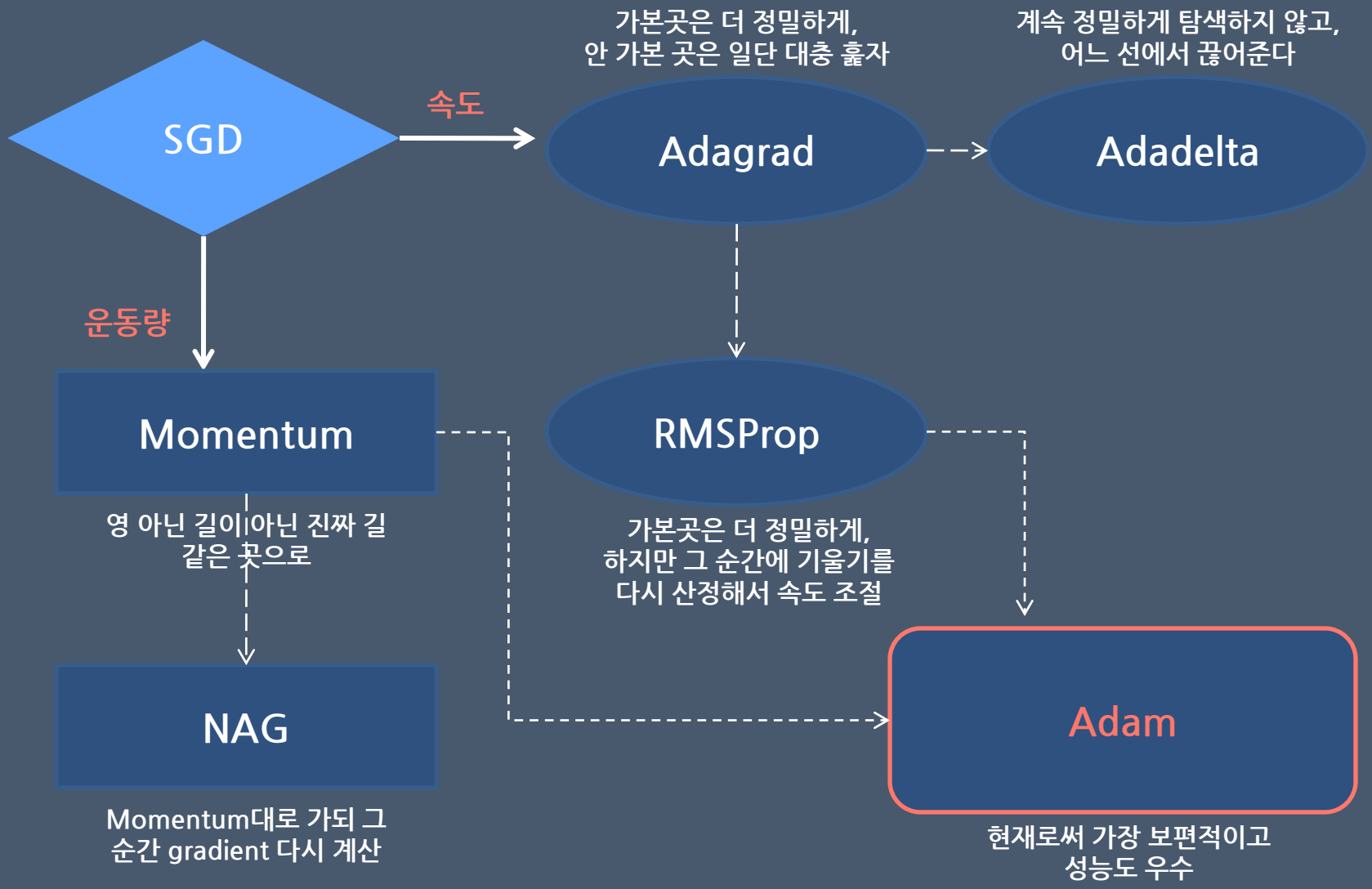


Local minimum에 갇히게 된다

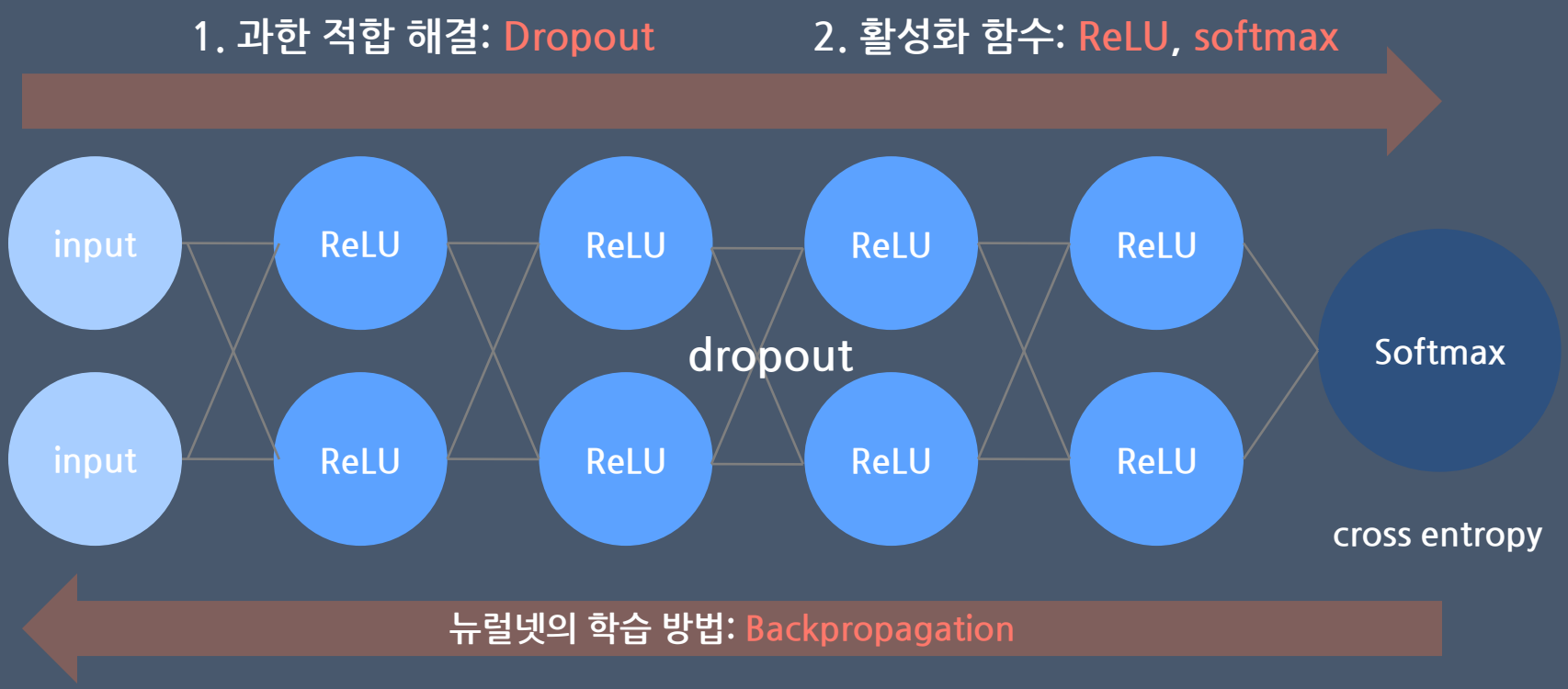
학습속도와 운동량을 고려한 Optimizer 등장

3가지 한계점

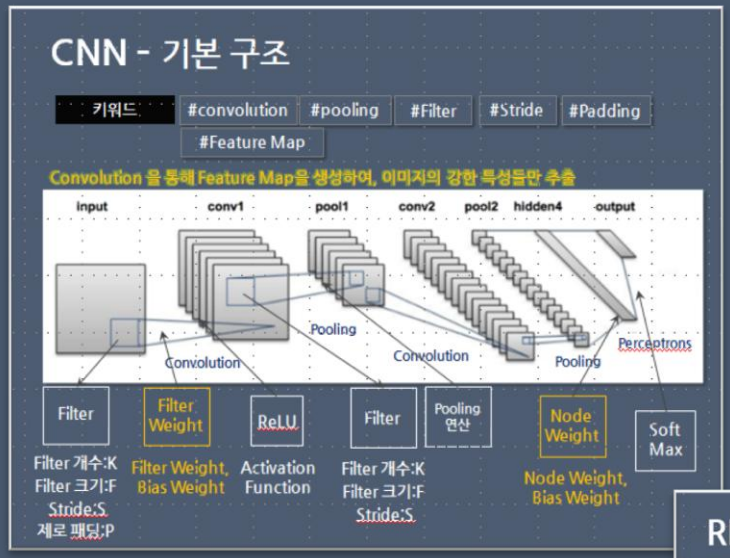
Optimizer 종류별 요약

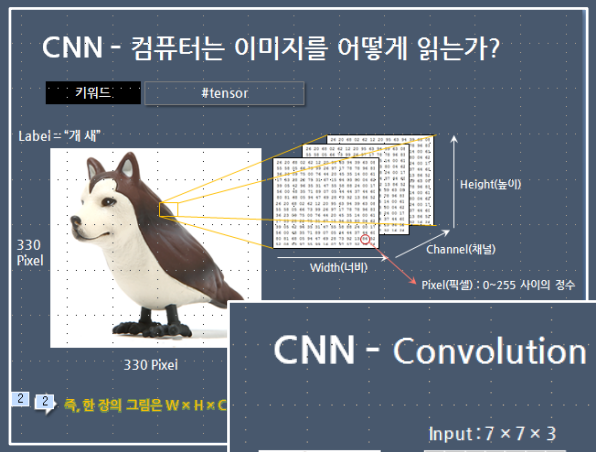


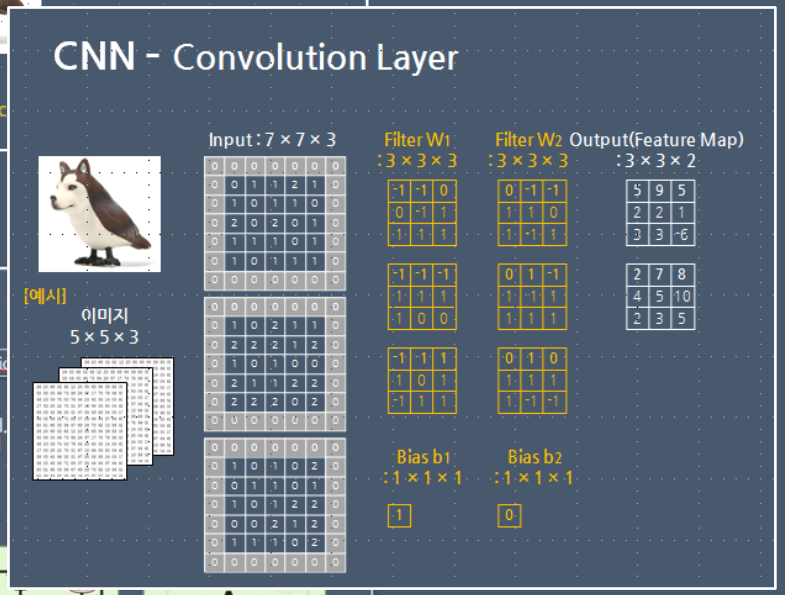
정리해봅시다

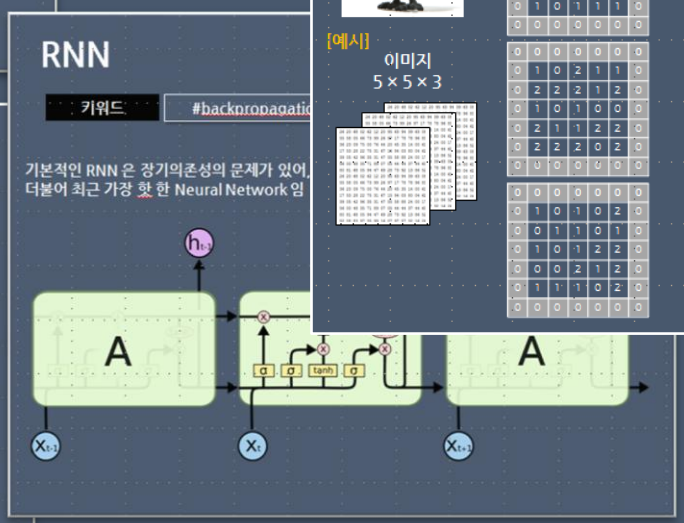

1. 과한 적합 해결: Dropout
2. 활성화 함수: ReLU, softmax
input
input
ReLU
ReLU
ReLU
ReLU
ReLU
ReLU
ReLU
ReLU
ReLU
dropout
Softmax
cross entropy
뉴럴넷의 학습 방법: Backpropagation
4. 느린 적합 해결: Adam
3. 덜한 적합 해결: ReLU

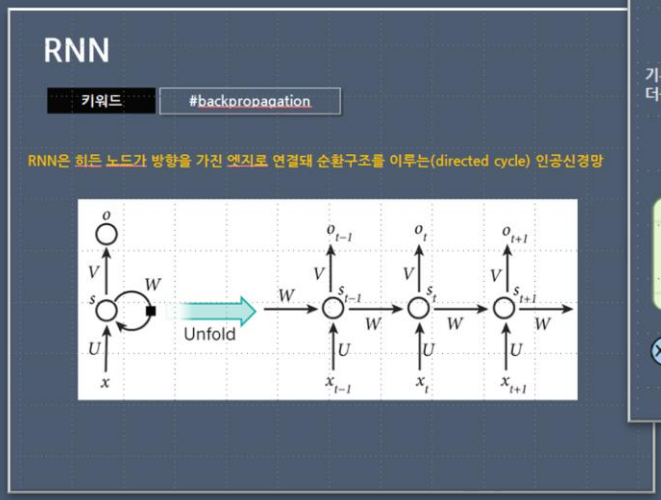
Coming Soon


CNN - 기본 구조
키워드
#convolution
#pooling
#Filter
#Stride
#Padding
#Feature Map
Convolution 을 통해 Feature Map을 생성하여, 이미지의 강한 특성들만 추출
input
conv1
pool1
conv2
pool2
hidden4
output
Convolution
Pooling
Convolution
Pooling
Perceptrons
Filter
Filter Weight
ReLU
Filter
Pooling 연산
Node Weight
Soft Max
Filter 개수:K
Filter 크기:F
Stride:S
제로 패딩:P
Filter Weight, Bias Weight
Activation Function
Filter 개수:K
Filter 크기:F
Stride:S
Node Weight, Bias Weight


CNN - 컴퓨터는 이미지를 어떻게 읽는가?
키워드
#tensor
Label = "개 새"
330 Pixel
330 Pixel
Height(높이)
Channel(채널)
Width(너비)
Pixel(픽셀) : 0~255 사이의 정수
2
2
즉, 한 장의 그림은 W × H × C


CNN - Convolution Layer
[예시]
이미지
5 × 5 × 3
Input : 7 × 7 × 3
Filter W1
: 3 × 3 × 3
Filter W2
: 3 × 3 × 3
Output(Feature Map)
: 3 × 3 × 2
Bias b1
: 1 × 1 × 1
Bias b2
: 1 × 1 × 1


RNN
키워드
#backpropagatio
기본적인 RNN 은 장기의존성의 문제가 있어,
더불어 최근 가장 핫 한 Neural Network 임
A
A


RNN
키워드
#backpropagation
RNN은 히든 노드가 방향을 가진 엣지로 연결돼 순환구조를 이루는(directed cycle) 인공신경망
Unfold

감사합니다.