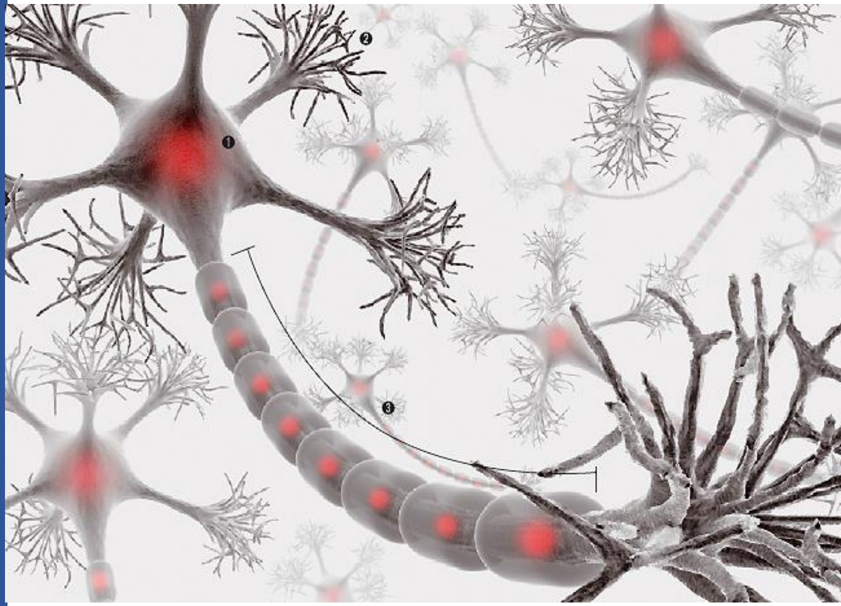
신경망 학습

학습 목표

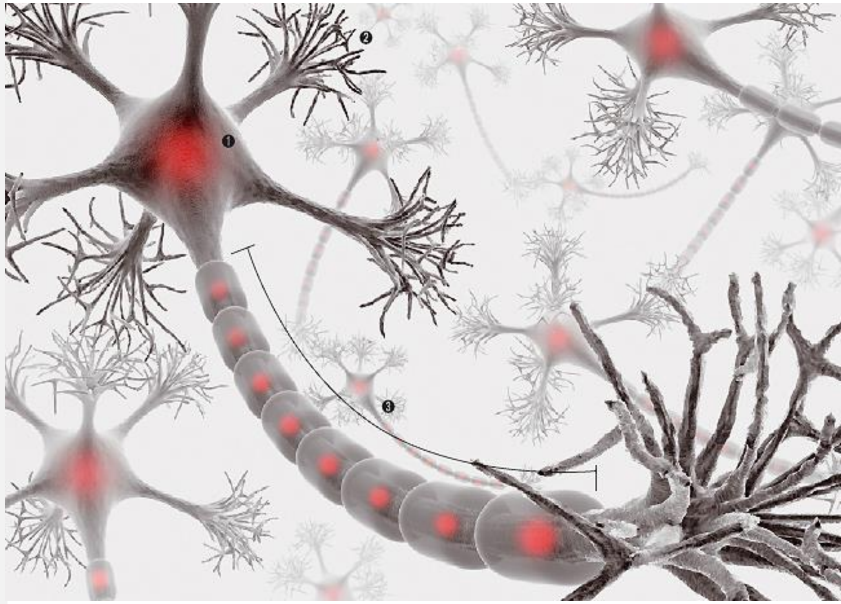
- 신경망 학습과 역전파 알고리즘을 이해한다.

주요 내용

1. 최적화 문제로서의 인공신경망 학습
2. 경사 하강법과 역전파 알고리즘
3. 데이터셋 구성과 훈련 데이터 단위



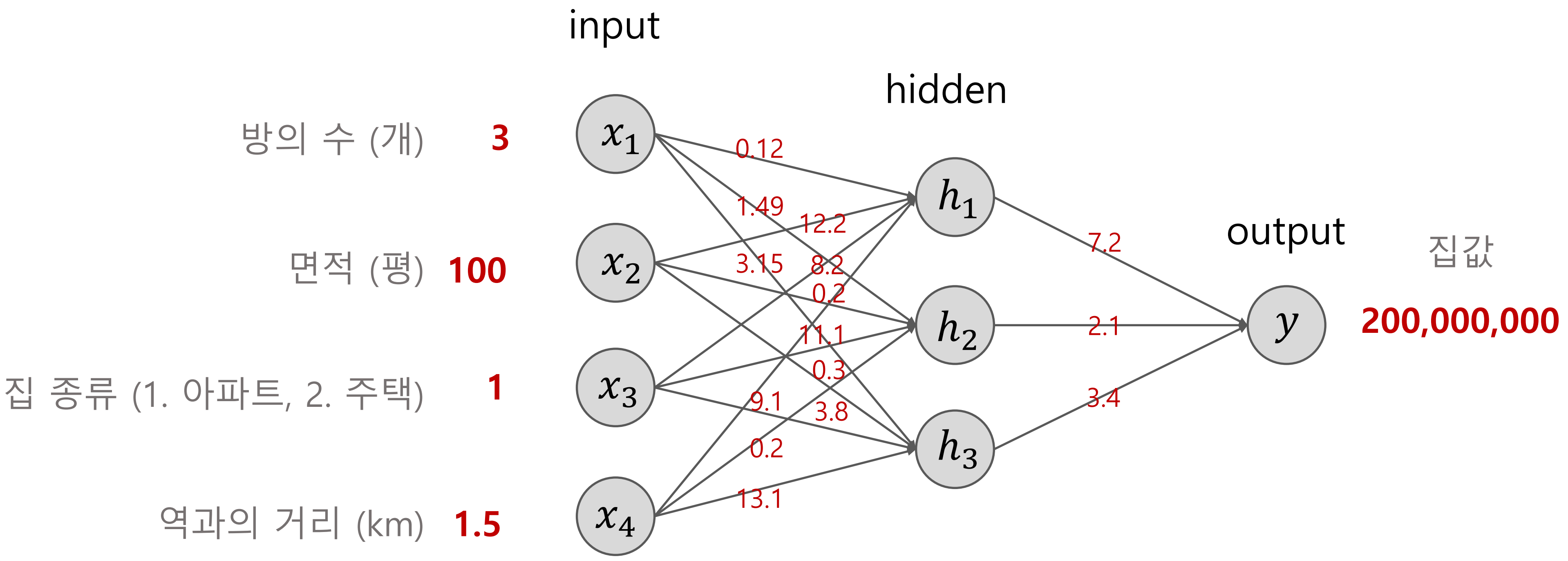
1 최적화 문제로서의 인공신경망 학습



인공 신경망의 학습

주어진 입력과 타깃 데이터를 이용해서 신경망 스스로 함수를 찾아내는 것

집값 예측 신경망



신경망 스스로 파라미터를 찾아 함수를 정의하는 것을 학습이라고 한다!

최적화 기반의 학습 방식

인공 신경망을 학습한다는 것은?

최적해를 향해 반복적으로 수렴하도록 하는 **최적화 문제**를 푸는 것이다!

Loss 계산

- 모델의 출력과 실제 값의 오차를 이용해서 Loss 계산
- 평균 오차 제곱 (Mean Squared Error), 크로스 엔트로피 (Cross Entropy)

최적화

- Loss가 최소화되도록 파라미터를 변경
- 역전파 알고리즘 + Gradient Descent의 변형 알고리즘

최적화란?

어떤 문제에 대해 선택가능한 해가 여러 개 존재할 때
최적해(Optimal Solution) 또는 근사해(Approximation)을 찾는 방식

예 : Supply Chain Optimization

최적화란?

특정 집합 위에서 정의된 함수, 실수, 정수에 대해 그 값이 최대나 최소가 되는 상태를 해석하는 문제

Standard Form

목적 함수 → 제약 조건 →

$$\min_{x \in D} \quad f(x)$$
$$\text{subject to} \quad g_i(x) \le 0, \qquad i = 1, \dots, m$$
$$h_j(x) = 0, \qquad j = 1, \dots, r$$

목적 함수(Objective Function)

최소화 : 비용 함수 (Cost Function), 손실 함수 (Loss Function)

최대화 : 유틸리티 함수 (Utility Function)

최적화 문제

<u>식단 (Diet Problem)</u>

필요한 영양을 만족하는 음식의 조합 중 가장 저렴한 조합을 찾으시오.

$$\min_{x} \quad c^T x$$
$$\text{subject to} \quad Dx \geq d$$
$$x \geq 0$$

식단 → x

$c^T x$ ← 식단의 가격

$Dx \geq d$ ← 최소 영양을 만족하는지 조건

$x \geq 0$ ← 음식의 양

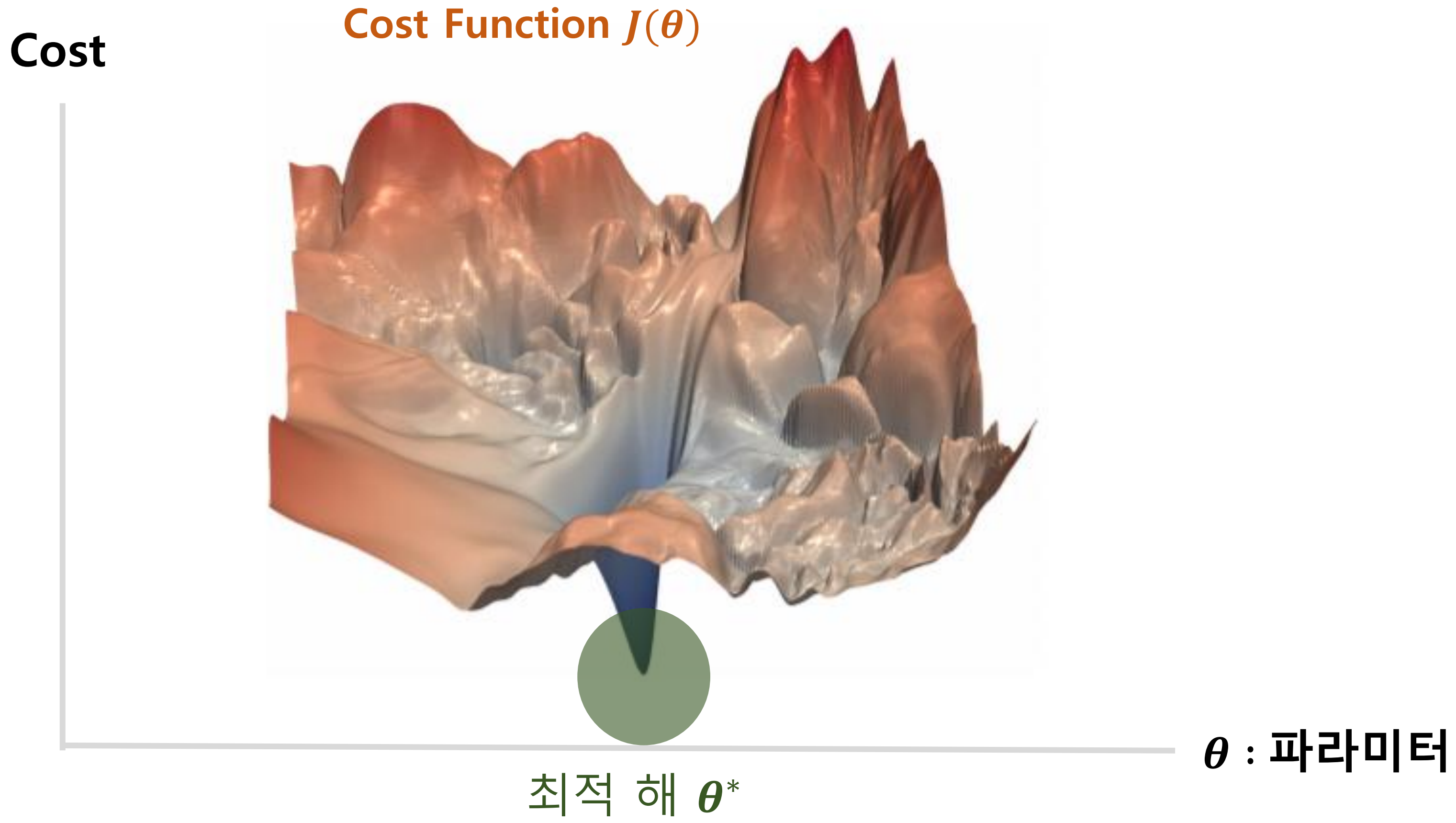
x_j 식단을 구성하는 음식 j의 양

c_j 음식 j의 단위 별 가격

D_{ij} 음식 j의 단위 별 각 영양소 i의 함유량

d_i 영양소 i의 최소 섭취량

최적화란?



비용 함수를 최소화 시키는 파라미터 값을 찾는 것

최적화 문제

<u>회귀 (Regression)</u>

타깃과 인공신경망이 예측한 값의 차이를 최소화하는 파라미터를 찾아라.

파라미터

$$\min_{\theta} \quad \frac{1}{n} \sum \| t - f(x; \theta) \|_2^2$$

타깃 (관측 레이블)

모델의 예측

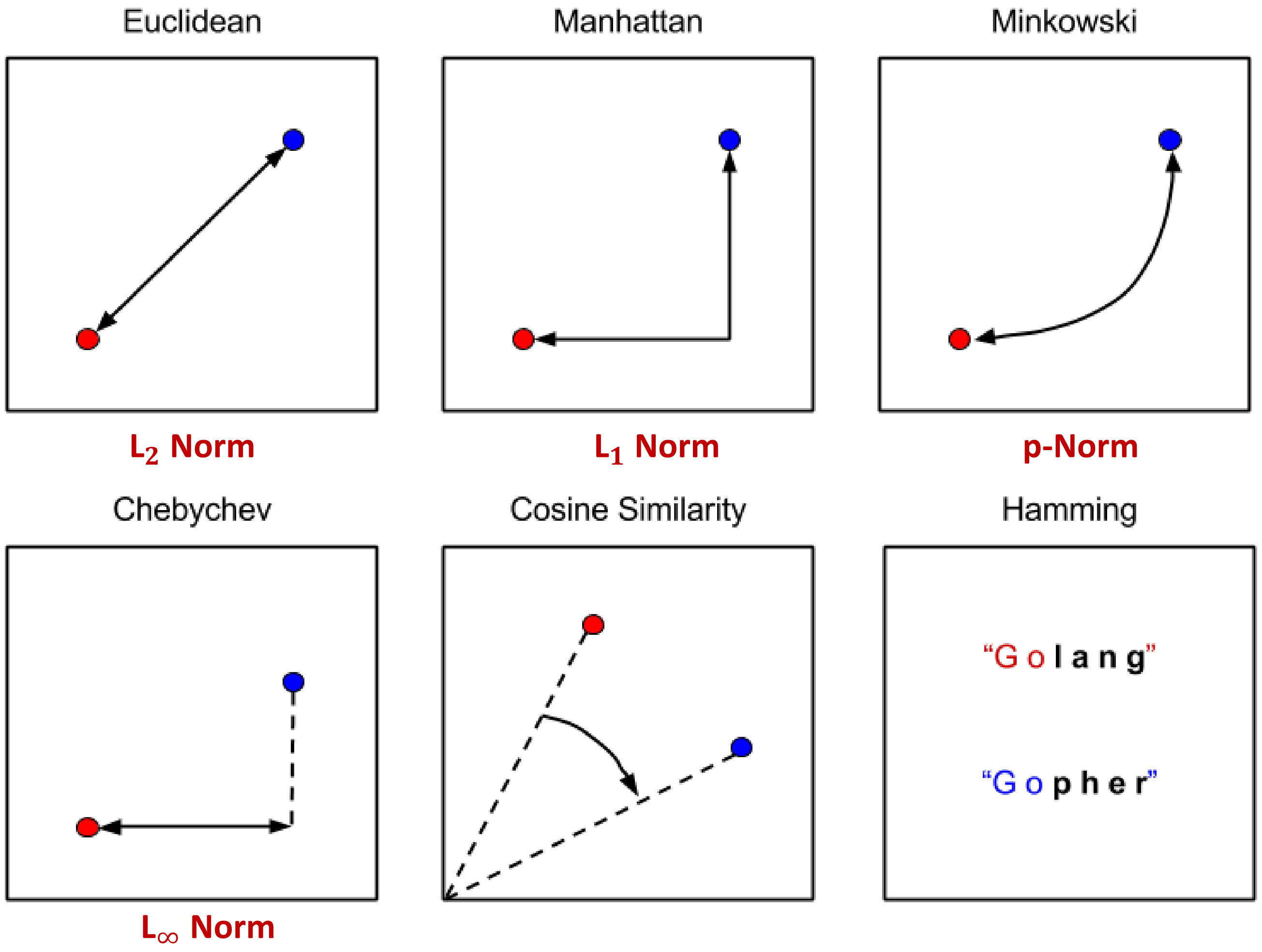
평균 제곱 오차 (Mean Squared Error)

참고 Norm (크기)

p-Norm

$$\| x \|_p = \left(\sum_{i=0}^{n} |x_i|^p \right)^{1/p} \quad \text{for } p \geq 1$$

- $p = 1$: L_1 Norm (Manhattan)
- $p = 2$: L_2 Norm (Euclidean)
- $p = \infty$: L_∞ Norm (Chebychev)



최적화 문제

분류 (Classification)

관측 분포와 인공신경망이 예측한 분포의 차이를 최소화하는 파라미터를 찾아라.

파라미터 →

$$\min_{\theta} \quad -\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K} t_k \cdot \log f(x;\theta)_k$$

K : Class 개수

관측 분포 / 모델이 예측한 분포

크로스 엔트로피 (Cross Entropy)

참고 정보량 (Self-Information)

<u>확률 변수의 정보량은?</u>

- 정보란 **놀라움의 정도**를 의미한다.
- 발생 확률이 낮은 사건일수록 정보가 크다.

$$\frac{1}{p(x)}$$

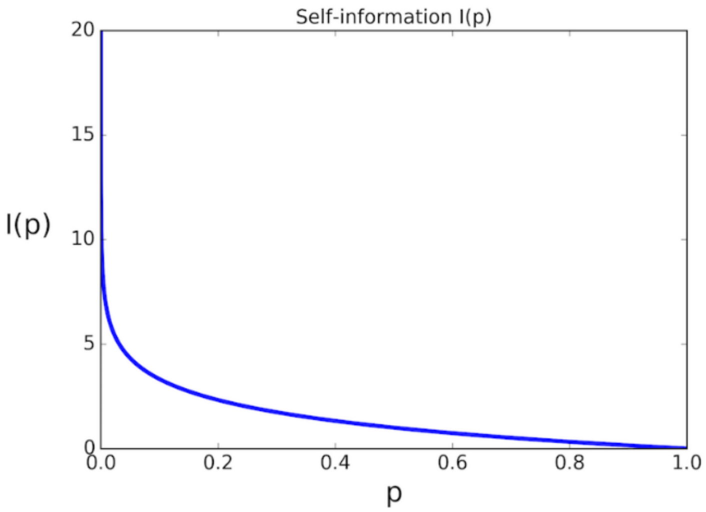
- 정보를 표현하는 Bit 수를 구해보자.

$$\log\frac{1}{p(x)} = -\log p(x)$$

<u>정보량 (Self-Information)</u>

Random Variable의 확률 값을 표현하기 위해 필요한 Bit 수

$$I(x) = -\log p(x)$$

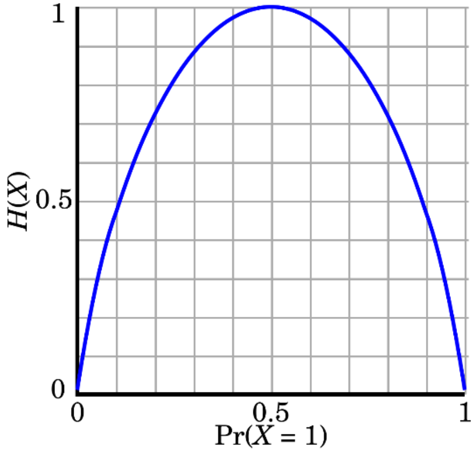


참고 엔트로피 (Entropy)

엔트로피 (Entropy)

- 확률 분포가 얼마나 불확실한지 또는 랜덤한지를 나타냄
- Random Variable의 정보량의 기댓값

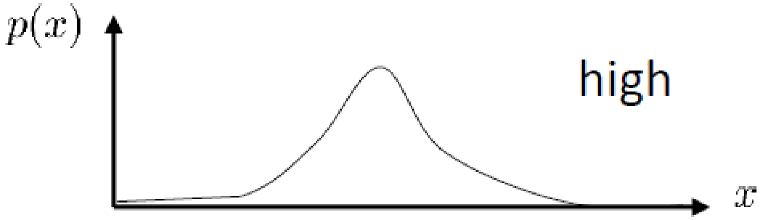
$$\mathcal{H}(p) = -\mathbb{E}_{x \sim p(x)}[\log p(x)] = -\int_x p(x) \log p(x)\ dx$$



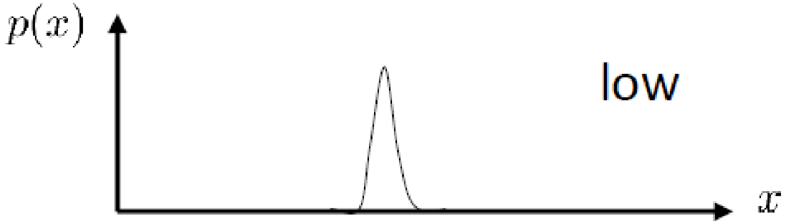
동전을 던졌을 때 앞면이 나올 확률에 대한 엔트로피

Random variable p가 얼마나 random한가?

엔트로피가 높으며 기대 확률은 낮다



엔트로피가 낮으면 기대 확률은 높다



참고 크로스 엔트로피 (Cross Entropy)

크로스 엔트로피 (Cross Entropy)

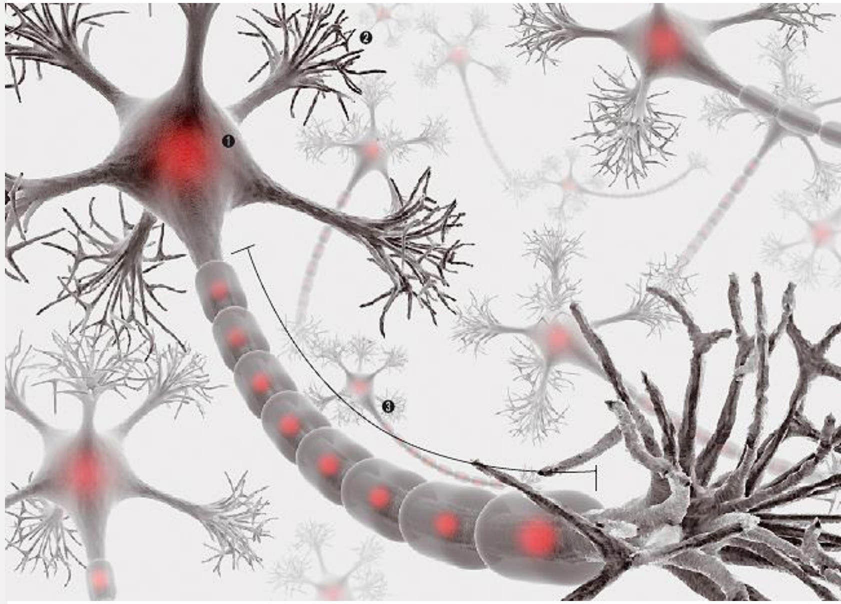
- 두 확률 분포의 차이 또는 유사하지 않은 정도(dissimilarity)를 나타냄
- 추정 확률 q 의 정보량을 확률 p 에 대해 구한 기댓값

$$\mathcal{H}(p,q) = -\mathbb{E}_{x\sim p(x)}[\log q(x)] = -\int_x p(x)\log q(x)\ dx$$

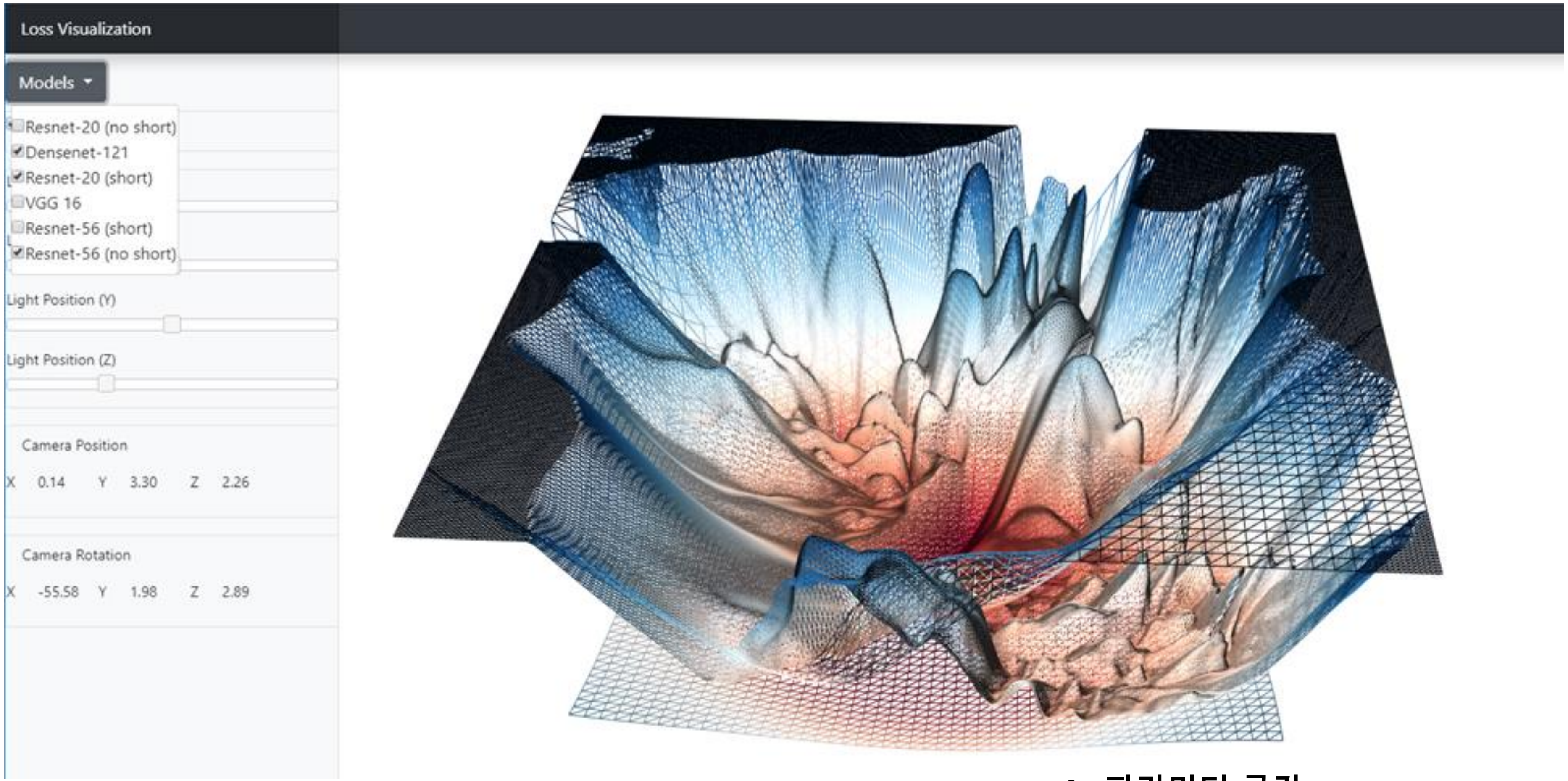
$p(x)$: Random Variable의 확률

$q(x)$: 추정 확률

2 경사 하강법과 역전파 알고리즘



Loss Surface

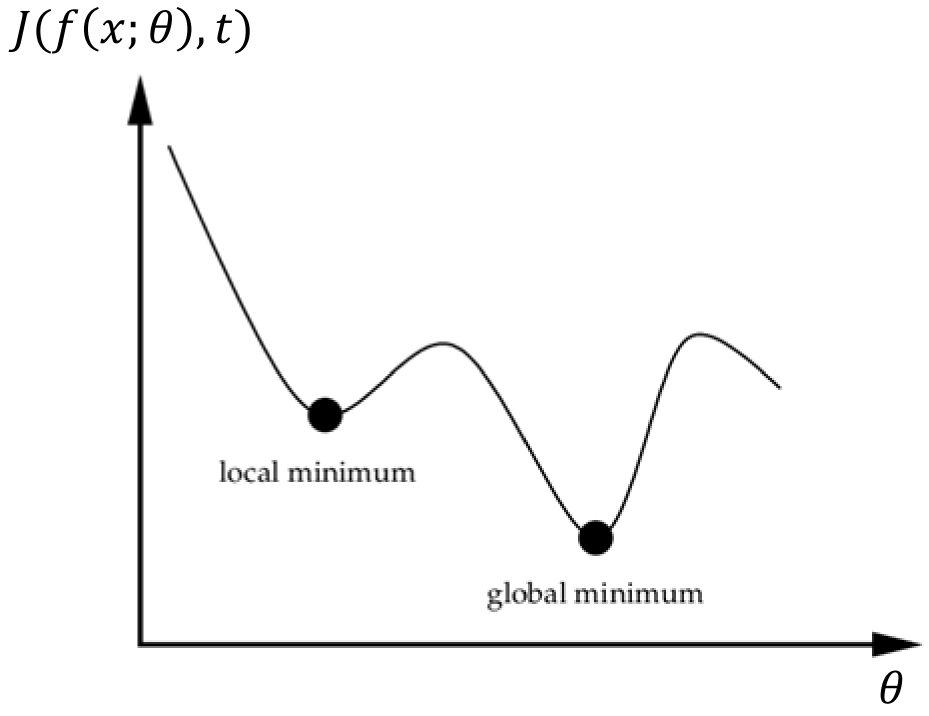


θ : 파라미터 공간

http://www.telesens.co/2019/01/16/neural-network-loss-visualization/

Loss를 최소화 하려면?

Loss Minimization



최적화 알고리즘

1차 미분

- Gradient Descent
- Variants of Gradient Descent :
  : SGD, AdaGrad, Momentum, RMSProp, Adam

Deep Learning에서 주로 사용하는 방법

1.5차 미분

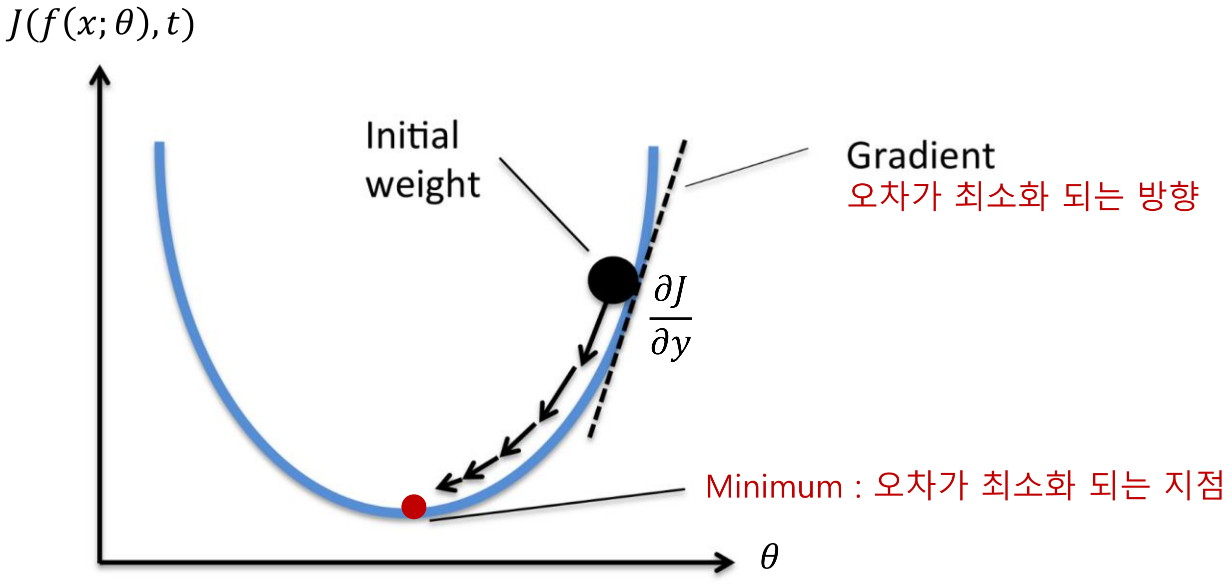
- Quasi-Newton Method
- Conjugate Gradient Descent
- Levenberg-Marquardt Method

2차 미분

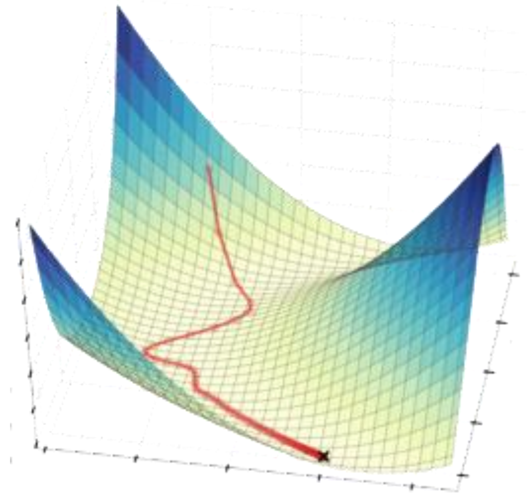
- Newton Method
- Interior Point Method

Gradient Descent

Gradient Descent



3D View



Parameter Update

$$\theta^{+} = \theta - \alpha \frac{\partial J}{\partial \theta}$$

Step Size

Gradient

참고 연속 함수의 미분

Real Valued Function

$$\boldsymbol{f}\colon \mathbb{R}^{\boldsymbol{n}} \longrightarrow \mathbb{R}$$

Gradient

$$\nabla f = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{pmatrix}$$

$$f(x) = f(x_1, x_2, \dots, x_n)$$

Hessian

$$\nabla^2 f = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{pmatrix}$$

Vector Function

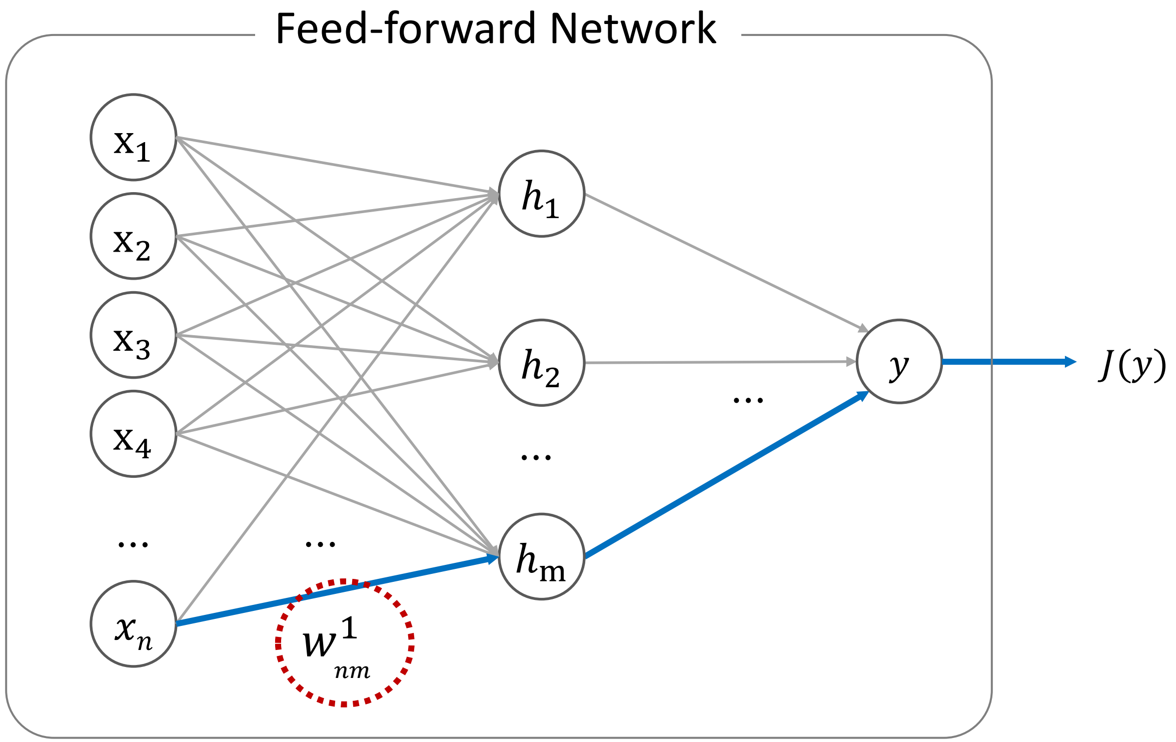
$$\boldsymbol{f}\colon \mathbb{R}^{\boldsymbol{n}} \longrightarrow \mathbb{R}^{\boldsymbol{m}}$$

Jacobian

$$\boldsymbol{J}f = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & & \frac{\partial f_2}{\partial x_n} \\ \vdots & & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{pmatrix}$$

$$f(x) = (f_1(x), f_2(x), \dots, f_m(x))$$
$$x = (x_1, x_2, \dots, x_n)$$

Gradient Descent



Parameter Update

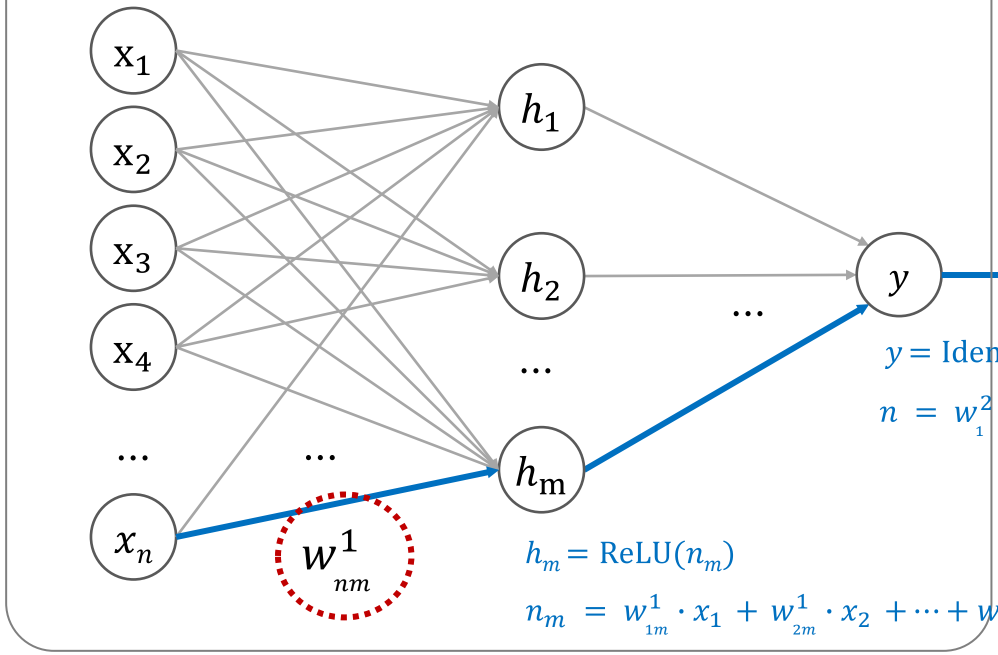
$$w^{1\,+}_{nm} = w^1_{nm} - \alpha \frac{\partial J}{\partial w^1_{nm}}$$

Step Size

Gradient

Gradient Descent

Feed-forward Network



$y = \mathrm{Identity}(n)$

$n = w^2_1 \cdot h_1 + w^2_2 \cdot h_2 + \cdots + w^2_m \cdot h_m$

$h_m = \mathrm{ReLU}(n_m)$

$n_m = w^1_{1m} \cdot x_1 + w^1_{2m} \cdot x_2 + \cdots + w^1_{nm} \cdot x_n$

$J(y) = \frac{1}{\mathrm{N}} \sum_{i=1}^{\mathrm{N}} (y - \mathrm{t})^2$

<u>Gradient of Parameter</u>

$$w^{1\,+}_{nm} = w^1_{nm} - \alpha \frac{\partial J}{\partial w^1_{nm}}$$

Step Size — Gradient

"가중치는 Loss Function의 간접 파라미터이므로 직접 미분이 안됨"

참고 합성 함수의 미분

$\boldsymbol{z = t^2}$
$\boldsymbol{t = 2x + y}$ 과 같은 식이 있을 때 미분 $\boldsymbol{\frac{\partial z}{\partial x}}$ 를 구해보자.

$\Leftrightarrow \quad z = (2x + y)^2$

$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial t} \cdot \frac{\partial t}{\partial x}$$

연쇄 법칙(Chain Rule)을 사용

$$\frac{\partial z}{\partial x} = 2(2x + y).2 = 8x + 4y$$

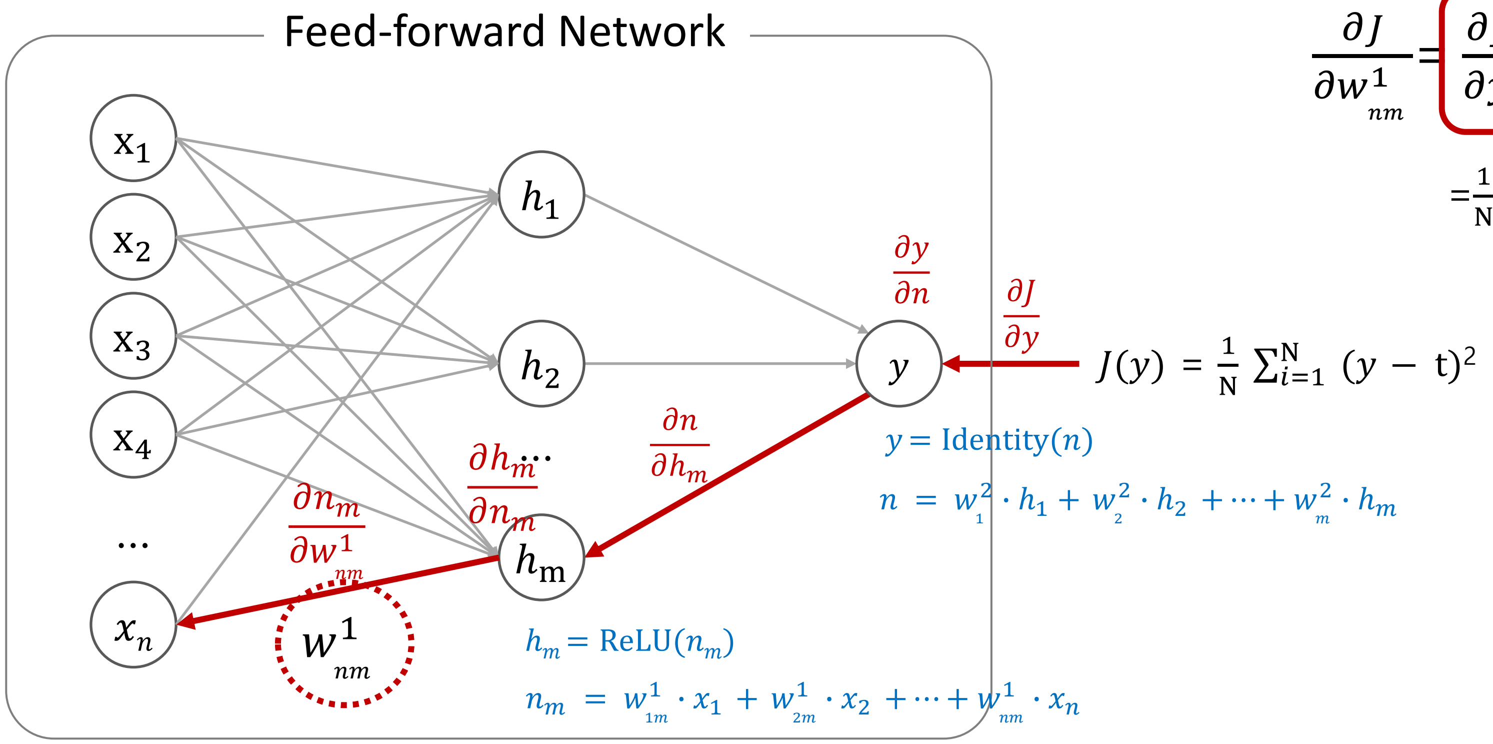
$$\frac{\partial z}{\partial t} = 2t$$

$$\frac{\partial t}{\partial x} = 2$$

각 식에 대한 미분을 구함

$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial t} \cdot \frac{\partial t}{\partial x} = 2t.2 = 4(2x + y) = 8x + 4y$$

Backpropagation



Gradient of Parameter

$$\frac{\partial J}{\partial w^1_{nm}} = \boxed{\frac{\partial J}{\partial y} \cdot \frac{\partial y}{\partial n} \cdot \frac{\partial n}{\partial h_m} \cdot \frac{\partial h_m}{\partial n_m} \cdot \frac{\partial n_m}{\partial w^1_{nm}}}$$

연쇄 법칙 (Chain Rule) 사용

$$= \frac{1}{N}\sum_{i=1}^{N} 2(y - t) . \mathrm{Identity}'(n) . {w^2}_m . \mathrm{ReLU}'(n_m) . x_n$$

$$\left\{ \begin{aligned} \frac{\partial J}{\partial y} &= \frac{1}{N}\sum_{i=1}^{N} 2(y - t) \\ \frac{\partial y}{\partial n} &= \mathrm{Identity}'(n) \\ \frac{\partial n}{\partial h_m} &= {w^2}_m \\ \frac{\partial h_m}{\partial n_m} &= \mathrm{ReLU}'(n_m) \\ \frac{\partial nm}{\partial w^1_{nm}} &= x_n \end{aligned} \right.$$

Backpropagation

같은 계층의 파라미터의 미분은 활성함수의 미분까지 공통 부분을 갖는다.

같은 계층의 파라미터의 미분

공통 부분

$$\frac{\partial J}{\partial w_{nm}^{1}} = \boxed{\frac{\partial J}{\partial y} \cdot \frac{\partial y}{\partial n} \cdot \frac{\partial n}{\partial h_m} \cdot \frac{\partial h_m}{\partial n_m}} \cdot \frac{\partial n_m}{\partial w_{nm}^{1}}$$

$$\frac{\partial J}{\partial w_{n-1m}^{1}} = \boxed{\frac{\partial J}{\partial y} \cdot \frac{\partial y}{\partial n} \cdot \frac{\partial n}{\partial h_m} \cdot \frac{\partial h_m}{\partial n_m}} \cdot \frac{\partial n_m}{\partial w_{n-1m}^{1}}$$

미분의 공통 부분 분리해서 표시

공통 부분

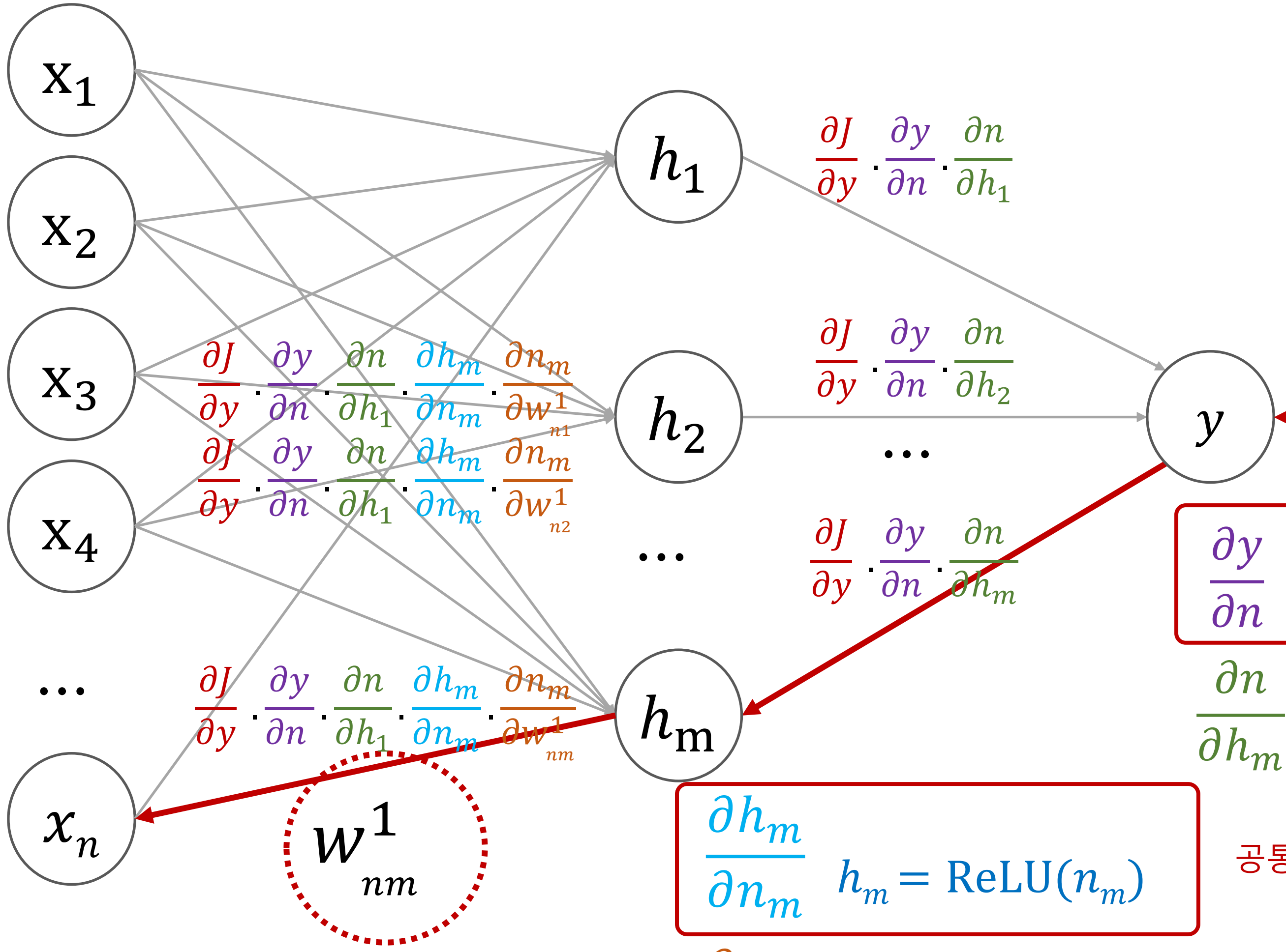
$$\boxed{\frac{\partial J}{\partial n_m} = \frac{\partial J}{\partial y} \cdot \frac{\partial y}{\partial n} \cdot \frac{\partial n}{\partial h_m} \cdot \frac{\partial h_m}{\partial n_m}}$$

$$\frac{\partial J}{\partial w_{nm}^{1}} = \frac{\partial J}{\partial n_m} \cdot \frac{\partial n_m}{\partial w_{nm}^{1}}$$

$$\frac{\partial J}{\partial w_{n-1m}^{1}} = \frac{\partial J}{\partial n_m} \cdot \frac{\partial n_m}{\partial w_{n-1m}^{1}}$$

공통 부분은 한번만 계산하자!

Backpropagation



$\frac{\partial J}{\partial y}$

$J(y) = \frac{1}{N}\sum_{i=1}^{N}(y - t)^2$

$\frac{\partial y}{\partial n}$ $\quad y = \mathrm{Identity}(n)$

공통

$\frac{\partial n}{\partial h_m}$ $\quad n = w^2_1 \cdot h_1 + w^2_2 \cdot h_2 + \cdots + w^2_m \cdot h_m$

가중치 별로 별도 계산

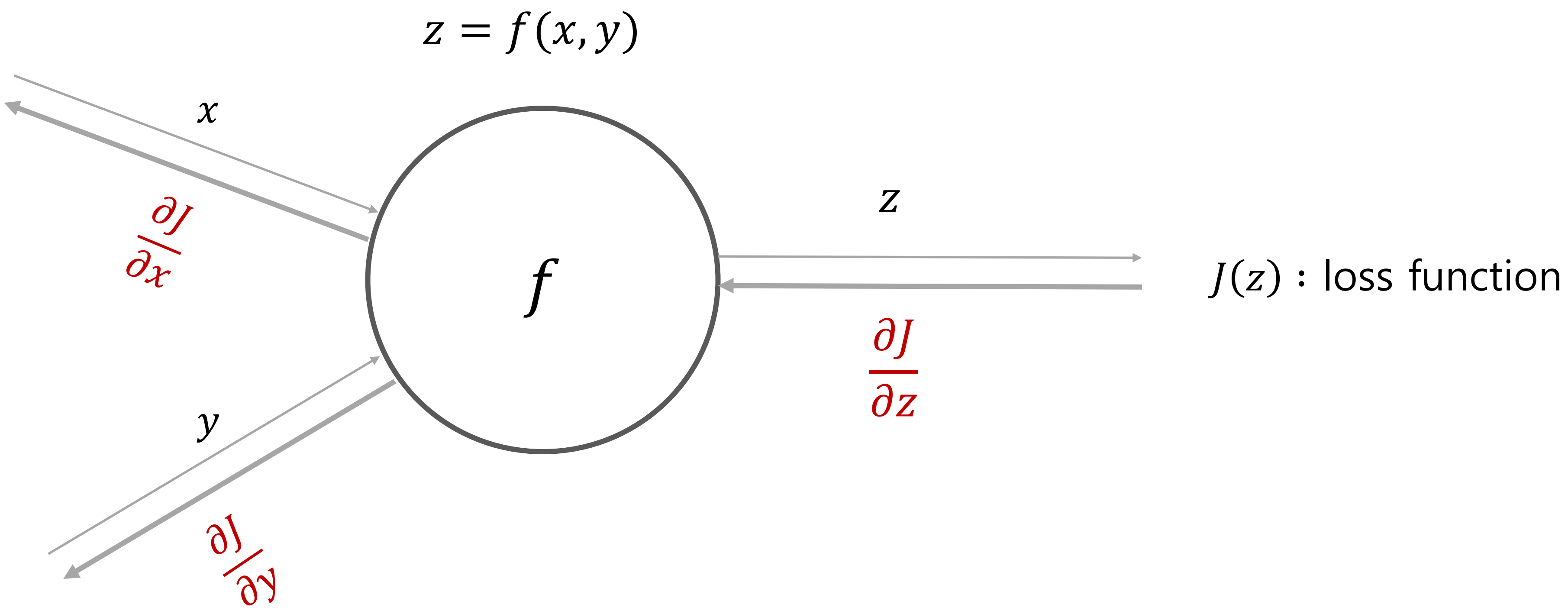
$\frac{\partial h_m}{\partial n_m}$ $\quad h_m = \mathrm{ReLU}(n_m)$

공통

$\frac{\partial n_m}{\partial w^1_{nm}}$ $\quad n_m = w^1_{1m} \cdot x_1 + w^1_{2m} \cdot x_2 + \cdots + w^1_{nm} \cdot x_n$

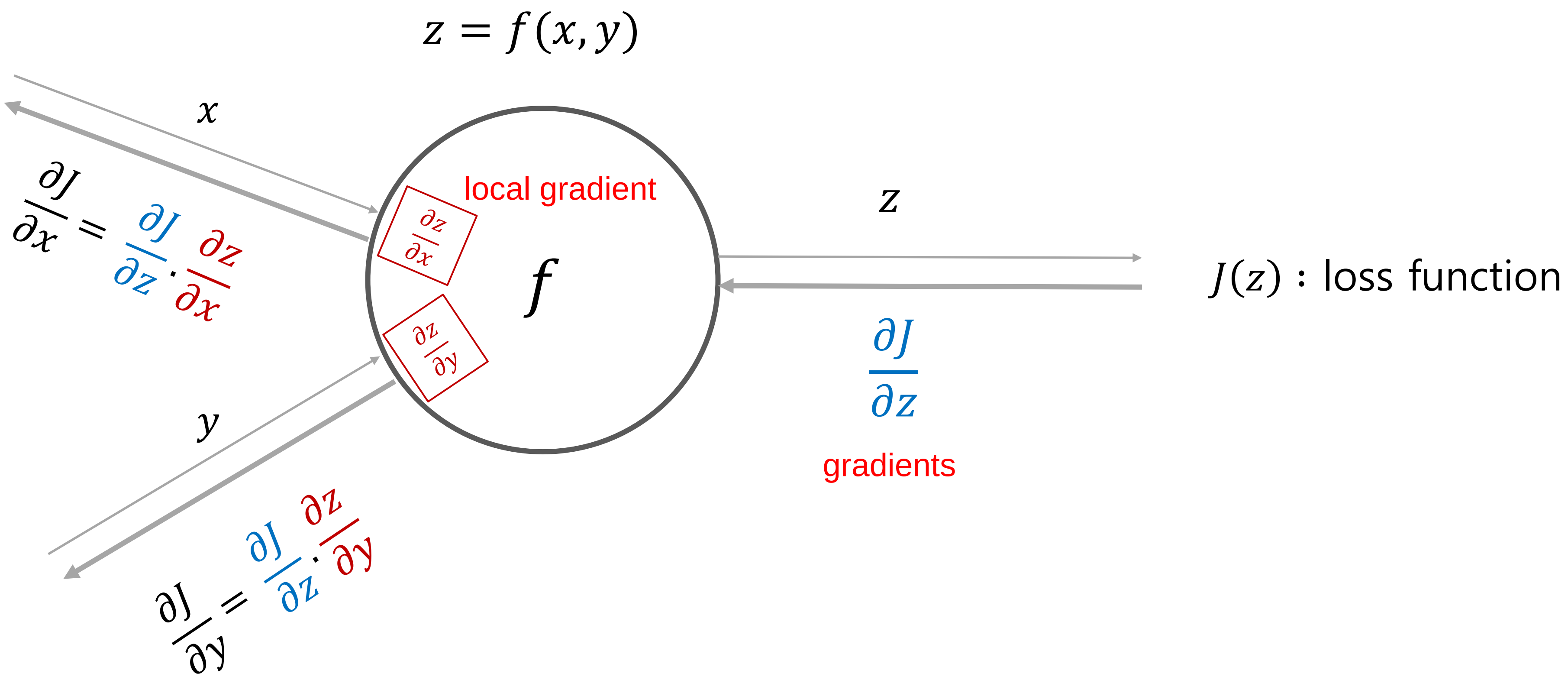
가중치 별로 별도 계산

Backpropagation



Loss $J(z)$에 대해 x, y, z의 미분을 구하라!

Backpropagation

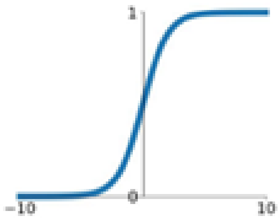


각 노드에서 Local Gradient를 구한 후 전달 받은 Gradient와 곱해서 이전 노드에 전달

주요 Activation Function의 미분

Sigmoid

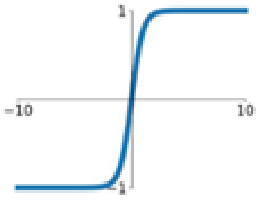
$$f(x) = \frac{1}{1+e^{-x}}$$



$$f'(x) = f(x)(1 - f(x))$$

Tanh

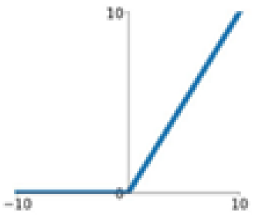
$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$



$$f'(x) = 1 - f(x)^2$$

ReLU

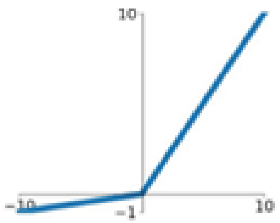
$$f(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$$



$$f'(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$$

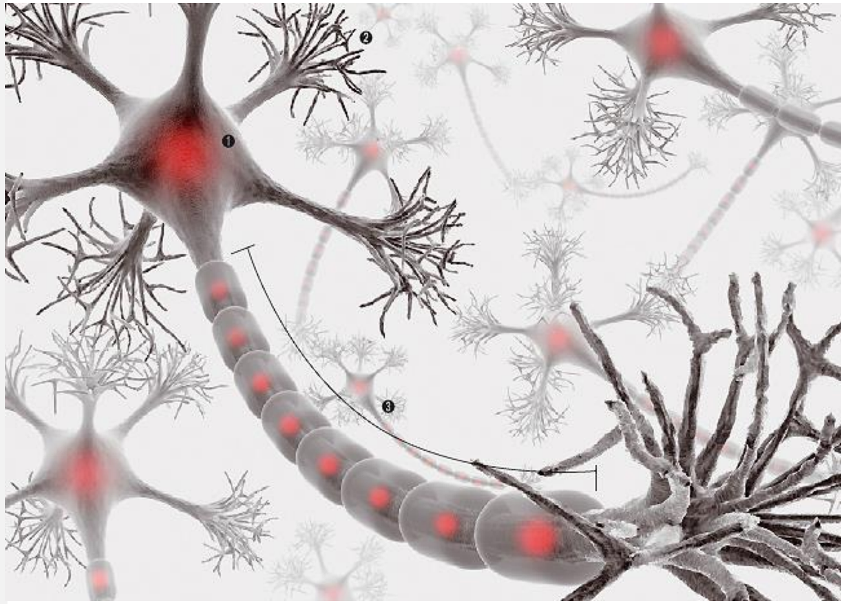
Leaky ReLU

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ 0.01x & \text{otherwise} \end{cases}$$



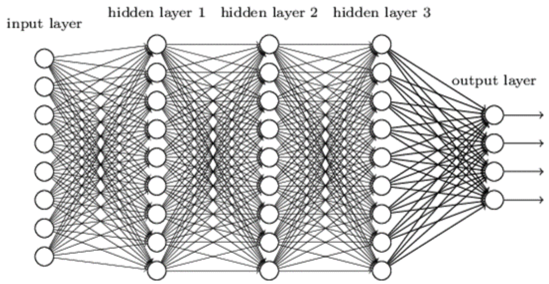
$$f'(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0.01 & \text{otherwise} \end{cases}$$

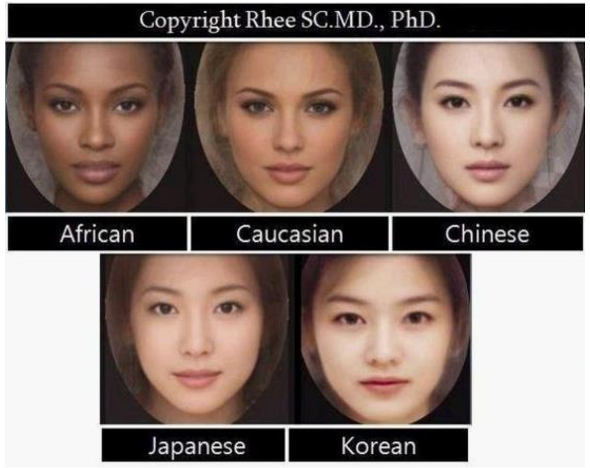
3 데이터셋 구성과 훈련 데이터 단위



관측 데이터 vs. 훈련 데이터

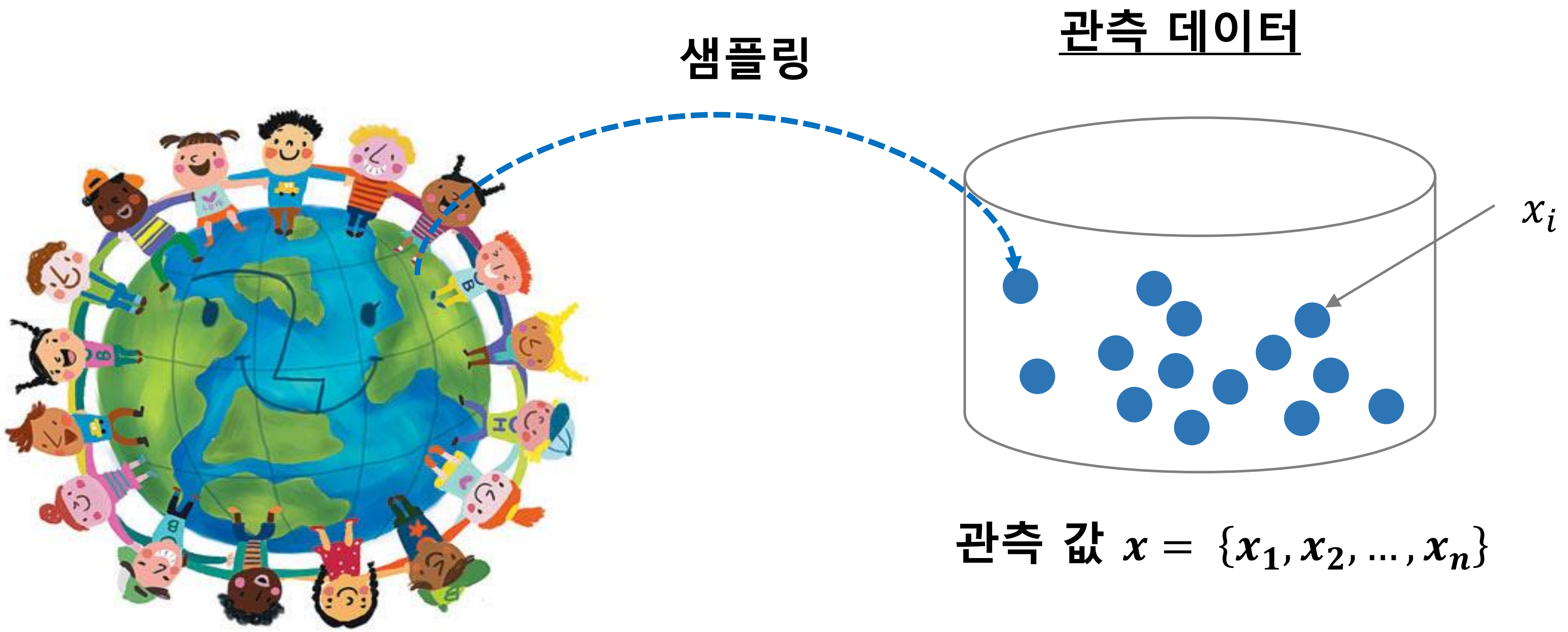
예 : 인종 분류 문제





Real World에서 데이터 전체를 구할 수 있을까?

관측 데이터 vs. 훈련 데이터

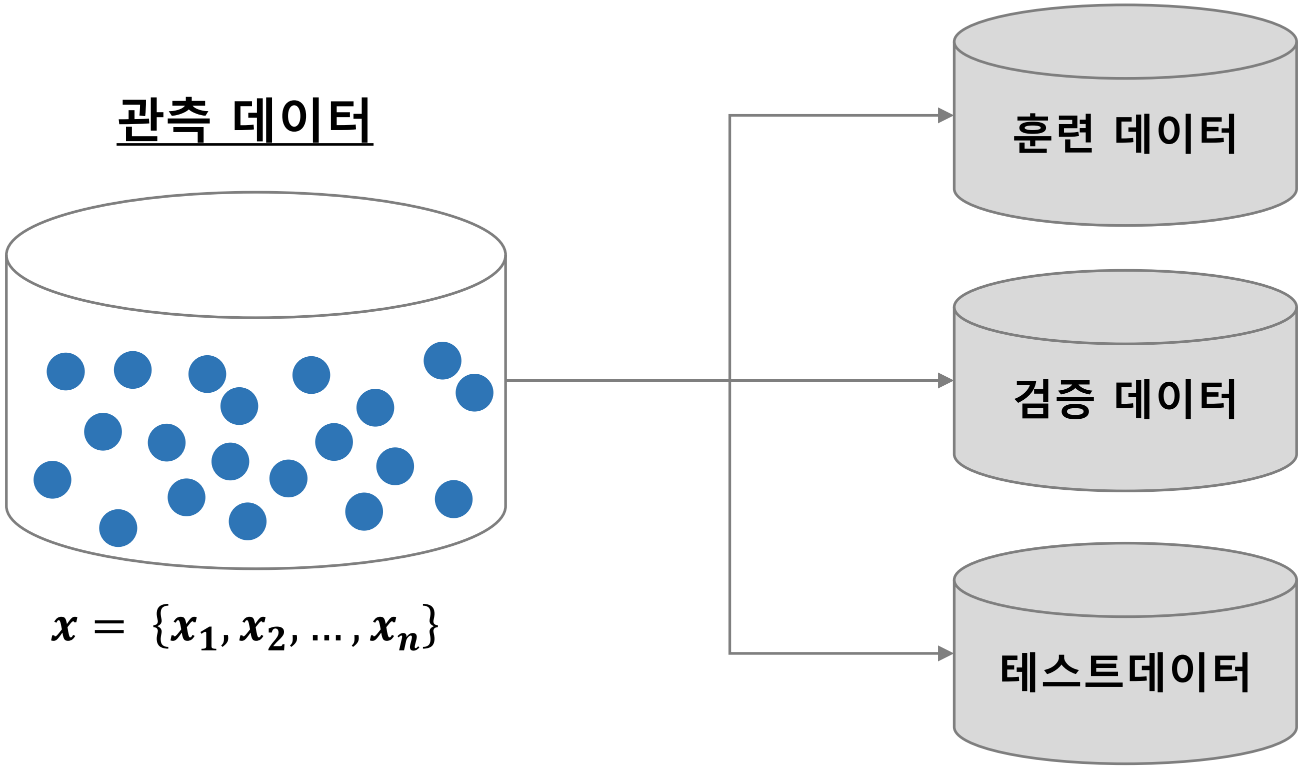


관측 값 $\boldsymbol{x} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n\}$

데이터를 샘플링을 통해 관측 데이터를 만듬

관측 데이터 vs. 훈련 데이터

관측 데이터를 세 종류의 데이터로 분리

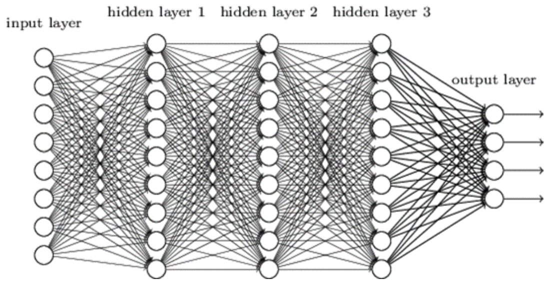


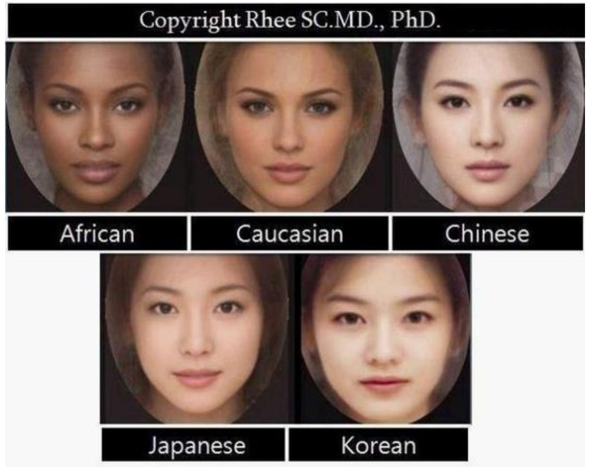
$$x = \{x_1, x_2, \dots, x_n\}$$

관측 데이터 vs. 훈련 데이터

<u>예 : 인종 분류 문제</u>

훈련 데이터





훈련 단위

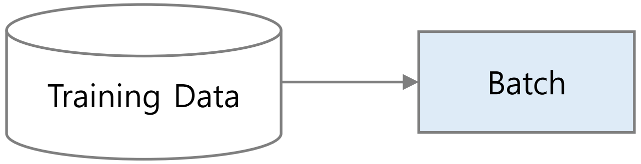
훈련 데이터를 한꺼번에 모델에 넣어서 훈련시킬 수 있을까?

- 메모리/CPU/GPU 용량이 허용되면 가능! 하지만 훈련이 속도가 매우 느림
- 훈련 데이터가 점진적으로 계속 늘어나는 경우엔 불가능 (Online Training)

훈련 단위

Batch

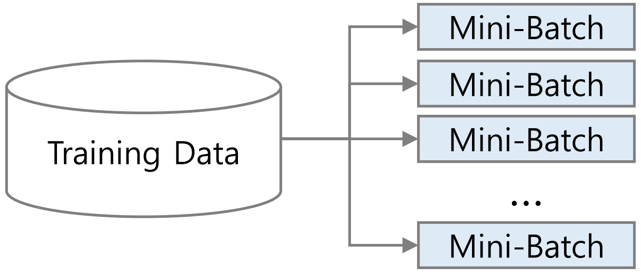
전체 훈련 데이터를 하나의 배치로
만들어 훈련



훈련 집합이 너무 크면 불가능!

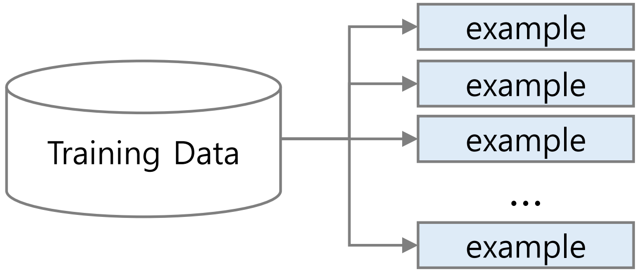
Mini-Batch

n개 샘플을 묶은 미니배치 단위로 훈련



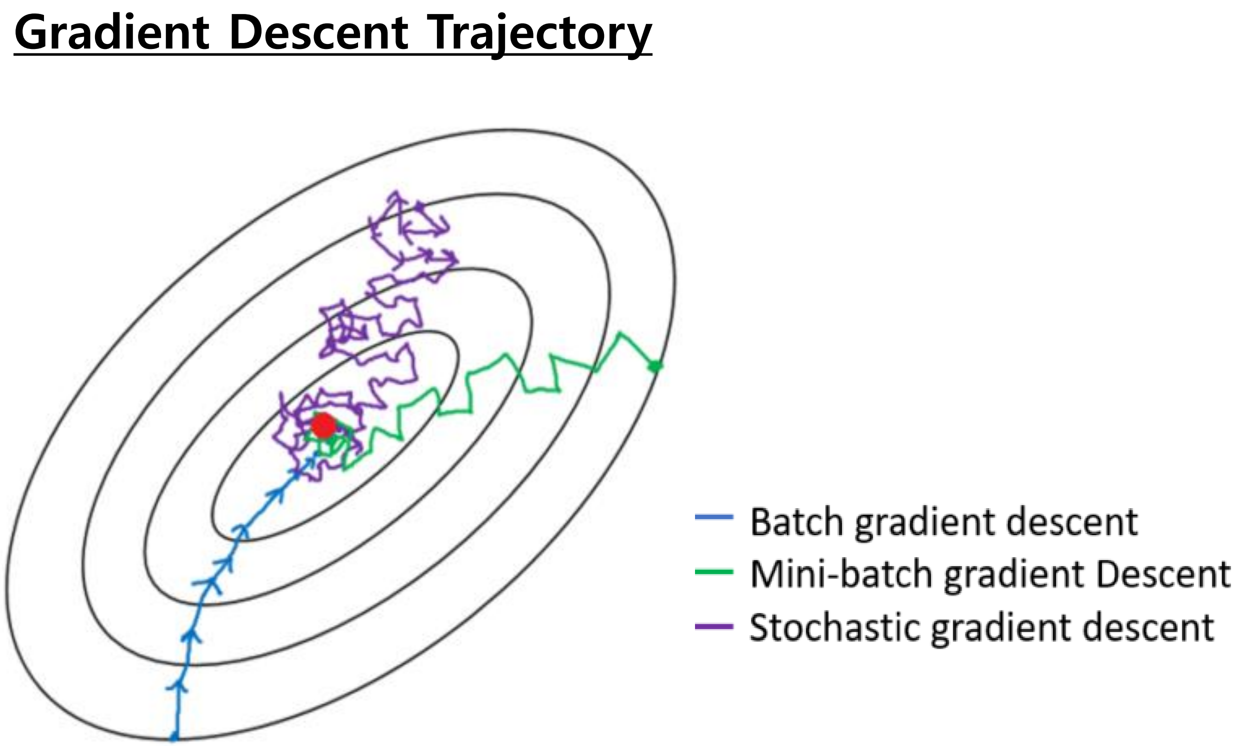
Stochastic

각 example 단위로 훈련



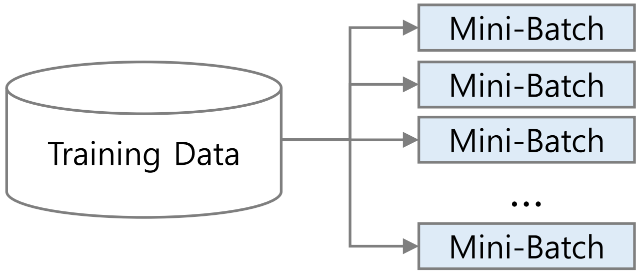
Too Noisy!

훈련 단위



[그림] https://towardsdatascience.com/gradient-descent-algorithm-and-its-variants-10f652806a3

미니 배치 크기



모집단의 표준 편차 : σ

표본 평균의 표준 편차 : $\frac{\sigma}{\sqrt{n}}$ n : 샘플 수

Mini-Bach 크기	100개	10,000개
계산 시간		100배 증가
표준 편차		10배 감소

대부분의 최적화 알고리즘은 천천히 정확하게 Gradient를 계산하는 것보다
빠르게 Gradient 근사치를 계산할 때 수렴 속도가 빠름

Thank you!