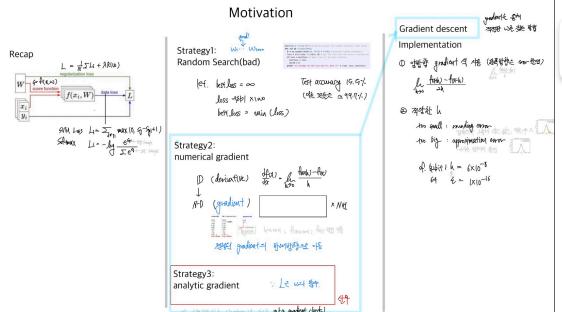


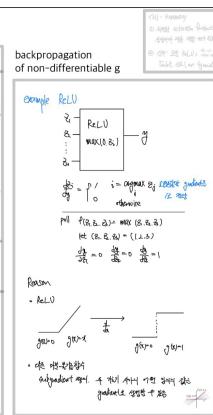
Lec4. back propagation



Motivation

Revisiting activation functions

Many Activation Functions			Revisiting Activation Functions		
Sigmoid	$\frac{1}{1+e^{-x}}$	$\frac{\partial}{\partial x} = e^{-x} / (1+e^{-x})^2$	ReLU	$\max(0, x)$	$\frac{\partial}{\partial x} = 1_{x>0}$
Tanh	$\frac{e^x - e^{-x}}{e^x + e^{-x}}$	$\frac{\partial}{\partial x} = 1 - \tanh^2(x)$	Leaky ReLU	$\max(0.01x, x)$	$\frac{\partial}{\partial x} = 0.01_{x>0} + 1_{x<0}$
ReLU: Rectified Linear Unit	$\max(0, x)$	$\frac{\partial}{\partial x} = 1_{x>0} + 0_{x<0}$	Parametric ReLU	$\max(\alpha x, x)$	$\frac{\partial}{\partial x} = \alpha_{x>0} + 1_{x<0}$
Leaky ReLU	$\max(0.01x, x)$	$\frac{\partial}{\partial x} = 0.01_{x>0} + 1_{x<0}$	ELU : Exponential Linear Units	$\alpha(e^{- x } - 1)$	$\frac{\partial}{\partial x} = 0.01_{x>0} + \alpha(e^{- x } - 1)_{x<0}$
Parametric ReLU	$\max(\alpha x, x)$	$\frac{\partial}{\partial x} = \alpha_{x>0} + 1_{x<0}$	Maxout	$\max(W_1x+b_1, W_2x+b_2)$	$\frac{\partial}{\partial x} = \text{softmax}(W_1, W_2) \cdot \frac{\partial}{\partial x}$



Calculating Gradients

Derivatives

on scalar argument (1D)

on vector argument (2D)

Matrix

Hessian

$$\nabla^2 f(x) = \nabla(\nabla f(x)) = \nabla \left(\begin{bmatrix} \frac{\partial f}{\partial x_1} & \dots & \frac{\partial f}{\partial x_n} \end{bmatrix} \right) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \dots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \dots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

Minimization

Algorithm

Initialization:

- $x^0 = \vec{x}$
- $\epsilon = 0$

do

- $\vec{x}^k = \vec{x}^{k-1} - \eta \nabla f(\vec{x}^{k-1})$
- while $|f(\vec{x}^k) - f(\vec{x}^{k-1})| < \epsilon$

let $\vec{x}^k = \vec{x}^{k-1}$

if $f(\vec{x}^k) > f(\vec{x}^{k-1})$

- positive definite ($\nabla^2 f(\vec{x}^k) > 0$)
- negative ($\nabla^2 f(\vec{x}^k) < 0$)
- stop gradient descent

Example

$$f(x_1, x_2, x_3) = x_1(1-x_2) + (x_2-1)^2 - x_3 + x_3^2 + x_1$$

gradient: $\nabla f = \begin{pmatrix} 1 & -x_2 & 1 \end{pmatrix} = \begin{pmatrix} 1 & -1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & -1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & -1 & 1 \end{pmatrix}$

gradient: $\nabla f = \begin{pmatrix} 1 & -x_2 & 1 \end{pmatrix} = \begin{pmatrix} 1 & -1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & -1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & -1 & 1 \end{pmatrix}$

gradient: $\nabla f = \begin{pmatrix} 1 & -x_2 & 1 \end{pmatrix} = \begin{pmatrix} 1 & -1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & -1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & -1 & 1 \end{pmatrix}$

Calculating gradients

Computation graphs

Backprop with Scalars

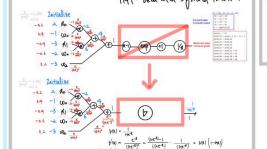
Compute gradients

bad: derive on paper

$$\begin{aligned} L &= \sum_i l_i \\ &= \sum_i \sum_j w_{ij} x_j + b_i \\ &= \sum_i \sum_j w_{ij} x_j + \sum_i b_i \\ &= \sum_i \sum_j w_{ij} x_j + \sum_i b_i \\ &= \sum_i \sum_j w_{ij} x_j + \sum_i b_i \end{aligned}$$

better: computational graphs + back propagation

Computations with Sigmoid Function



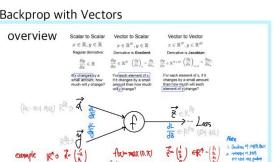
Computations with Vectors

Example

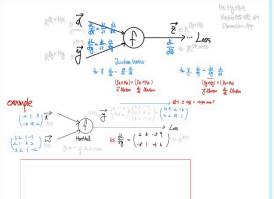
$$\begin{aligned} \text{easy case: } & \text{compute gradients} \\ & \text{grad} = \frac{\partial L}{\partial x} = \frac{\partial L}{\partial x_1} + \frac{\partial L}{\partial x_2} + \frac{\partial L}{\partial x_3} \end{aligned}$$

$$\begin{aligned} \text{hard (note for practice): } & \text{grad} = \frac{\partial L}{\partial x} = \frac{\partial L}{\partial x_1} + \frac{\partial L}{\partial x_2} + \frac{\partial L}{\partial x_3} \end{aligned}$$

Patterns in gradient flow

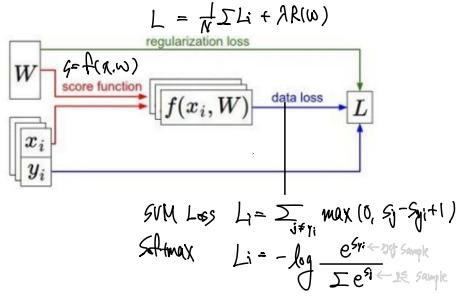


Backprop with Matrices(2D), Tensor(>3D)



Motivation

Recap



Strategy1: Random Search(bad)

$$\text{ref. } \text{best_loss} = \infty$$

$$\text{loss} \rightarrow \text{best_loss} \times \text{num}$$

$$\text{best_loss} = \min(\text{loss})$$

```
bestloss = float("inf") # Python assigns the highest possible float value
for num in range(1000):
    W = np.random.rand(100, 100) # generate random parameters
    loss = L(X_train, Y_train, W) # get the loss over the entire training set
    if loss < bestloss: # keep track of the best solution
        bestW = W
        bestloss = loss
    print "In attempt %d the loss was %f, best %f" % (num, loss, bestloss)
```

Test accuracy 15.5%
 $(\approx 25\% \approx 99.7\%)$

Strategy2: numerical gradient

1D (derivative) $\frac{df(x)}{dx} = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}$

\downarrow

N-D (gradient) $\boxed{\quad} \times N \times 1$

current W		W + h (gradient direction)	
0.34	0.34	0.34	0.34
0.31	0.31	0.31	0.31
0.78	0.78	0.78	0.78
0.12	0.12	0.12	0.12
0.64	0.64	0.64	0.64
0.04	0.04	0.04	0.04
2.91	2.91	2.91	2.91
2.49	2.49	2.49	2.49
3.1	3.1	3.1	3.1
0.53	0.53	0.53	0.53
0.03	0.03	0.03	0.03
1.2345	1.2345	1.2345	1.2345

$h=0.0001 \rightarrow f(x+0.0001), f(x)$ 차이 구하기

gradient의 반복방법으로 이용

Strategy3: analytic gradient

$\therefore L \in w \vdash \text{opt.}$

각 계층에서 Strategy1과 2 (a.k.a. gradient check)

Gradient descent

Implementation

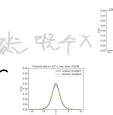
① 수학적 gradient의 사용 (최적 방향은 그라디언트)

$$\frac{f(x+h) - f(x-h)}{2h}$$

적당한 h

too small : rounding error

too big : approximation error



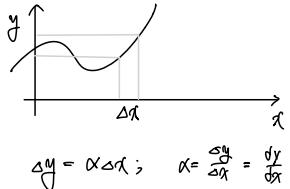
of float1 h = 6×10^{-8}
 $64 \epsilon = 1 \times 10^{-16}$

gradients는 초기
적절한 h를 찾는 방법

Calculating Gradients

Derivatives

on scalar argument (1D)



on vector argument (2D)



$$\Delta y = \alpha \Delta x$$

$$= \left[\begin{array}{c|c|c|c} \frac{\partial y}{\partial x_1} & \frac{\partial y}{\partial x_2} & \dots & \frac{\partial y}{\partial x_n} \\ \hline \end{array} \right] \left(\begin{array}{c} \Delta x_1 \\ \vdots \\ \Delta x_n \end{array} \right)$$

derivative $\nabla_{\vec{x}}$

gradient = $\nabla_{\vec{x}} f^T$ transpose of derivative
함수가 가장 빠르게 증가하는 방향
그 반대방향은 가장 빠르게 감소하는 방향!

Example: $f(x, y) = x^2y + y$

$$\nabla f(x, y) = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

where

$$\frac{\partial f}{\partial x} = \frac{\partial}{\partial x}(x^2y + y) = 2xy$$

(via the power rule and sum rule), and

$$\frac{\partial f}{\partial y} = \frac{\partial}{\partial y}(x^2y + y) = x^2 + 1$$

So, the gradient of the function f is defined as

$$\nabla f(x, y) = \begin{bmatrix} 2xy \\ x^2 + 1 \end{bmatrix}$$

Matrix

Hessian

$$\begin{aligned} \nabla^2 f(\vec{x}) &= \nabla_{\vec{x}} (\nabla_{\vec{x}} f(\vec{x})) \\ &= \nabla_{\vec{x}} \left(\begin{pmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} & \dots & \frac{\partial f}{\partial x_n} \\ \frac{\partial f}{\partial x_2} & \frac{\partial^2 f}{\partial x_2^2} & \dots & \frac{\partial f}{\partial x_2 x_n} \\ \vdots & \ddots & \ddots & \vdots \\ \frac{\partial f}{\partial x_n} & \frac{\partial f}{\partial x_n x_1} & \dots & \frac{\partial^2 f}{\partial x_n^2} \end{pmatrix} \right) \\ &= \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \dots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \dots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \ddots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \dots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix} \end{aligned}$$

Jacobian

차원개기, $y = f(\vec{x})$; 이런 $\vec{y} = f(\vec{x})$

$$f(\vec{x}) = \begin{bmatrix} f_1(\vec{x}) \\ f_2(\vec{x}) \\ \vdots \\ f_m(\vec{x}) \end{bmatrix}$$

$$\nabla_{\vec{x}} f(\vec{x}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \dots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \dots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \ddots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \dots & \frac{\partial f_m}{\partial x_n} \end{bmatrix} f(\vec{x})$$

Minimization

① $f' = 0$ 일 때
② $f'' > 0$ ⇒ local min ←!
 < 0 ⇒ max

① $\nabla f(\vec{x}) = 0$ 일 때
② $\nabla^2 f(\vec{x}) > 0$; eigen-value > 0 MAX
 < 0 min

Hessian matrix positive definite ($\nabla^2 A \vec{x} > 0$)
negative semi-definite

Example

$$f(x_1, x_2, x_3) = (x_1)^2 + x_1(1 - x_2) + (x_2)^2 - x_2x_3 + (x_3)^2 + x_3$$

$$\text{gradient: } \nabla_{\vec{x}} f = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \frac{\partial f}{\partial x_3} \end{pmatrix} = \begin{pmatrix} 2x_1 + (1-x_2) \\ -x_1 + 2x_2 - x_3 \\ -x_2 + 2x_3 + 1 \end{pmatrix} = \begin{pmatrix} 2x_1 + 1 \\ -x_1 + 2x_2 - x_3 \\ -x_2 + 2x_3 + 1 \end{pmatrix}$$

$$\text{find a point } \begin{pmatrix} 2x_1 + 1 \\ -x_1 + 2x_2 - x_3 \\ -x_2 + 2x_3 + 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \xrightarrow{\text{Linear Algebra}} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} -1 \\ 1 \\ 1 \end{pmatrix}$$

$$\begin{aligned} \text{Hessian Matrix} \\ \nabla^2 f = \nabla_{\vec{x}}^2 f(\vec{x}) = \begin{pmatrix} 2x_1 + 1 & & \\ -x_1 + 2x_2 - x_3 & 2x_2 & \\ -x_2 + 2x_3 + 1 & & \end{pmatrix} = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \frac{\partial^2 f}{\partial x_1 \partial x_3} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \frac{\partial^2 f}{\partial x_2 \partial x_3} \\ \frac{\partial^2 f}{\partial x_3 \partial x_1} & \frac{\partial^2 f}{\partial x_3 \partial x_2} & \frac{\partial^2 f}{\partial x_3^2} \end{pmatrix} \\ = \begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{pmatrix} \end{aligned}$$

find eigen of Hessian

$$\begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{pmatrix} \xrightarrow{\text{Linear Algebra}} \begin{pmatrix} 3.414 & & \\ 0.646 & 2 & \\ 0 & 0 & 1 \end{pmatrix} \quad \text{all } \lambda > 0.$$

$\therefore \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}$ is minimum

Algorithm

Initialize:

$$W^0$$

$$k = 0$$

do
 $W^{k+1} = W^k - \eta^k \nabla Loss(W^k)^T$
 $k = k + 1$

general A, f

step-size η (count)

let $x^k, f^k = 0$

if $|f(x^k) - f(x^k)| > \varepsilon$

$x^{k+1} = x^k - \eta^k \nabla f(x^k)^T$

step-size η gradient의 반례방법으로 조정

algorithm의 원리

이전의 point 계산 초기화 ... 경우 Too Much

예제.

local min 설정한 까지 가로 수평면에 초기화

설정까지?

① 차이값이 절대적 차이 $|f(x^k) - f(x^k)| < \varepsilon$

Stop size η

② gradient가 충분히 작을 때

$\| \nabla f(x^k) \| < \delta$

Backprop with Scalars

Compute gradients

bad: derive on paper

$$L = f(x; W) = Wx$$

$$L_i = \sum_{j \neq i} \max(0, s_j - s_i + 1)$$

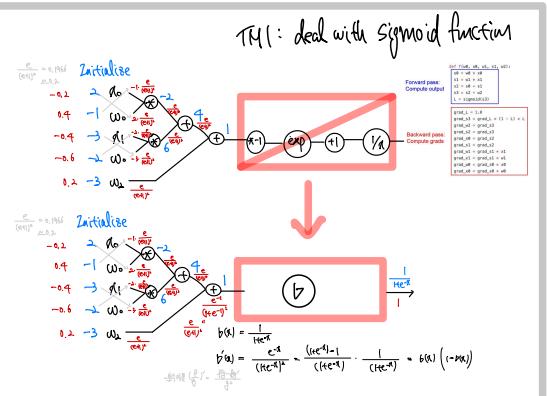
$$= \sum_{j \neq i} \max(0, W_{j,i} \cdot x + W_{g(i),i} \cdot x + 1)$$

$$L = \frac{1}{N} \sum_{i=1}^N L_i + \lambda \sum_k W_k^2$$

$$= \frac{1}{N} \sum_{i=1, j \neq i}^N \max(0, W_{j,i} \cdot x + W_{g(i),i} \cdot x + 1) + \lambda \sum_k W_k^2$$

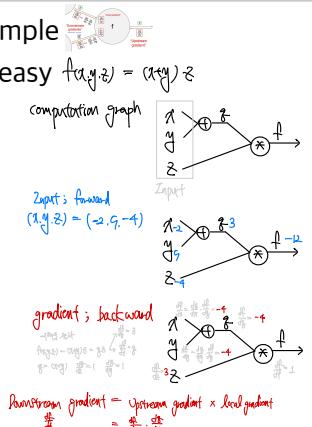
$$\nabla_W L = \nabla_W \left(\frac{1}{N} \sum_{i=1}^N \sum_{j \neq i} \max(0, W_{j,i} \cdot x + W_{g(i),i} \cdot x + 1) + \lambda \sum_k W_k^2 \right)$$

better:
computational graphs
+ back propagation



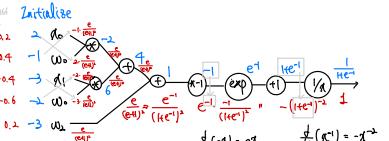
Computation graphs

Example

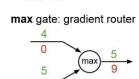
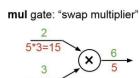
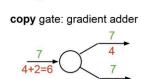
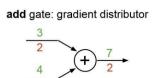


hard(note for practice)

$$f(w, x) = \frac{1}{1 + e^{-(w_0 + w_1 x + b)}}$$



Patterns in gradient flow



Backprop with Vectors

overview

Scalar to Scalar

$x \in \mathbb{R}$, $y \in \mathbb{R}$

Regular derivative:

$$\frac{\partial y}{\partial x} \in \mathbb{R}$$

If x changes by a small amount, how much will y change?

Vector to Scalar

$x \in \mathbb{R}^N$, $y \in \mathbb{R}^M$

Derivative is Gradient:

$$\frac{\partial y}{\partial x} \in \mathbb{R}^{N \times M}$$

For each element of x , if it changes by a small amount then how much will y change?

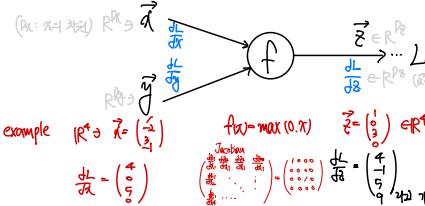
Vector to Vector

$x \in \mathbb{R}^N$, $y \in \mathbb{R}^M$

Derivative is Jacobian:

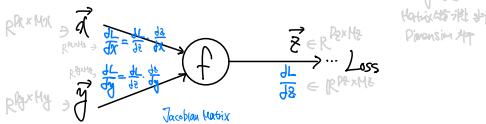
$$\frac{\partial y}{\partial x} \in \mathbb{R}^{M \times N}$$

For each element of x , if it changes by a small amount then how much will each element of y change?

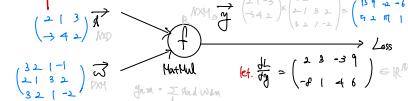


Note:
1. Jacobian of Matrix \vec{x}
2. Matrix \vec{x} is $\vec{x} = \begin{pmatrix} \frac{\partial L}{\partial x_1} \\ \frac{\partial L}{\partial x_2} \end{pmatrix}$
3. Matrix \vec{y} is $\vec{y} = \begin{pmatrix} \frac{\partial L}{\partial x_1} \\ \frac{\partial L}{\partial x_2} \end{pmatrix}$
4. Matrix \vec{f} is $\vec{f} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$

Backprop with Matrices(2D), Tensor(>3D)



example



Matrix \vec{x} is $\vec{x} = \begin{pmatrix} \frac{\partial L}{\partial x_1} \\ \frac{\partial L}{\partial x_2} \end{pmatrix}$

Matrix \vec{y} is $\vec{y} = \begin{pmatrix} \frac{\partial L}{\partial x_1} \\ \frac{\partial L}{\partial x_2} \end{pmatrix}$

Matrix \vec{z} is $\vec{z} = \begin{pmatrix} \frac{\partial L}{\partial x_1} \\ \frac{\partial L}{\partial x_2} \end{pmatrix}$

Many Activation Functions

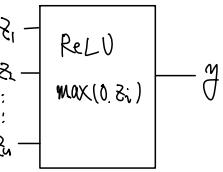
Revisiting Activation Functions

THI: SUMMARY
 ① sigmoid activation function
 성별이나 나이, 성별이나 나이 등
 ② 특성: 1) ReLU: 비선형, 계층 간 연산
 Tanh는 ReLU가 아니라 sigmoid

	function	특징	문제점
Sigmoid	$\frac{1}{1+e^{-x}}$	$x \rightarrow g(x)$ $(-\infty, \infty) \rightarrow (0, 1)$ 활성화율이 매우 높음	① 미지수면 $g' \approx 0$ ② output이 정답이 0.5 (W=0) \rightarrow bias 설정; 예. 활성화율
Tanh	$\frac{2}{1+e^{-2x}} - 1$	$x \rightarrow g(x)$ $(-\infty, \infty) \rightarrow (-1, 1)$ zero-centered	① 미지수면 $g' \approx 0$ \Leftrightarrow saturation kills gradients
ReLU: Rectified Linear Unit	$\text{MAX}(0, x)$	① 계산 부하 ↓ ② g' alive ($x > 0$)	① Not zero-centered ② No learning in $x < 0$ \rightarrow 양수 bias는 $\frac{\partial f}{\partial b} = 0$ (부정적)
Leaky ReLU	$\text{Max}(0.01x, x)$	① 계산 부하 ↓ ② g' alive (in all x) ③ zero-centered가 아님	THI
Parametric ReLU	$\text{MAX}(\alpha x_1, x_1)$	① 계산 부하 ↓ ② g' alive (in all x) ③ $\alpha \leq$ data로부터 학습	
ELU: Exponential Linear Units	$\begin{cases} x & (x > 0) \\ \alpha(e^x - 1) & (x \leq 0) \end{cases}$ <p>default $\alpha = 1$</p>	① 모든 경우에 이용 가능 (여기 예전 모델의 변환 하기) ② Leaky ReLU의 경쟁 모두 가능함 ③ Noise에 강함 \therefore 모델 학습이 강화되며 훈련 속도↑	
Maxout	$\max(a_1^T x + b_1, a_2^T x + b_2)$	① ReLU와 Leaky ReLU의 혼합 ② parameter 개수 증가 방지	

backpropagation
of non-differentiable g

example ReLU



$$\frac{dy}{dz_i} = \begin{cases} 1 & i = \arg\max z_j \\ 0 & \text{otherwise} \end{cases}$$

ReLU gradient은 i 를 제외한 모든 입력에 대해 0이다.

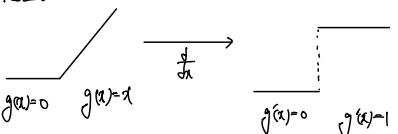
$$\text{pull } f(z_1, z_2, z_3) = \text{MAX}(z_1, z_2, z_3)$$

$$\text{let } (z_1, z_2, z_3) = (1, 2, 3)$$

$$\frac{df}{dz_1} = 0 \quad \frac{df}{dz_2} = 0 \quad \frac{df}{dz_3} = 1$$

Reason

• ReLU



• 다른 미분 불가능함수

subgradient 방식. 두 가지 상황에 따라 gradient를 설정할 수 있다.
 $f(x) = 0$ $f'(x) = 0$



부록