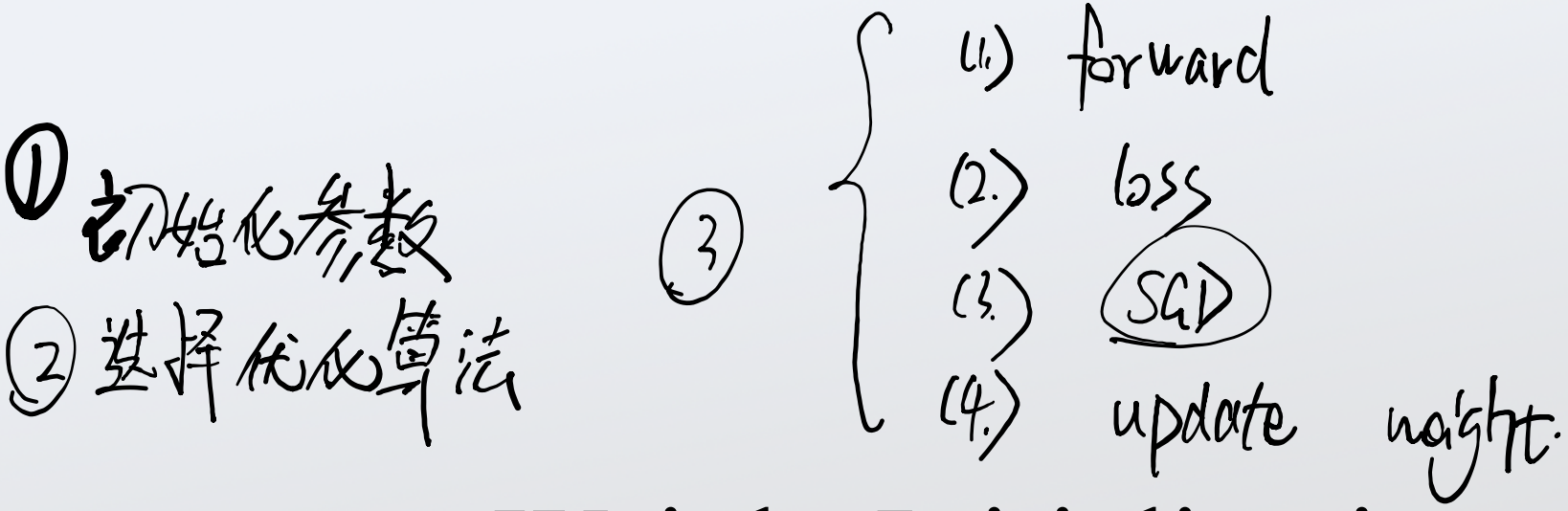
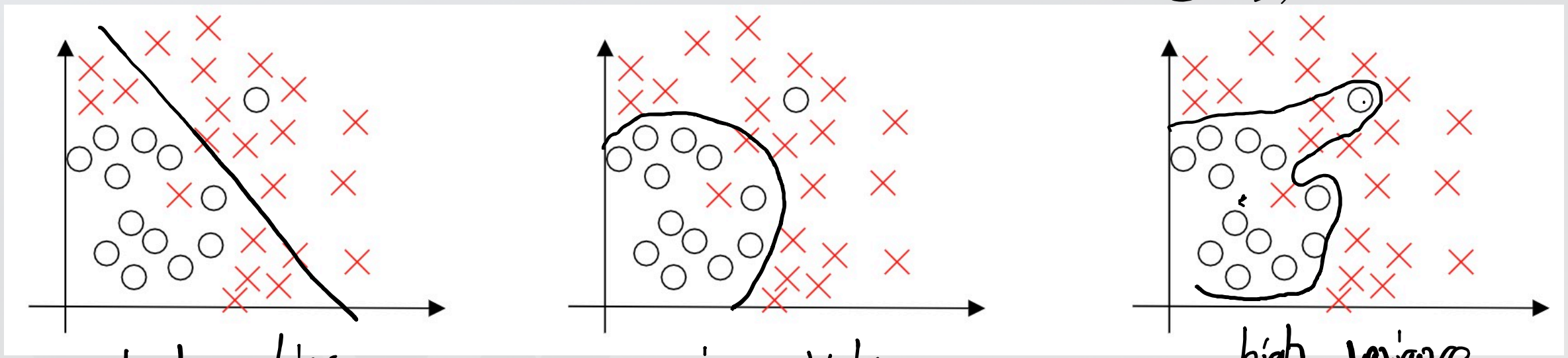


Weight Initialization

Bias and Variance



high bias ⇒ Bigger network
longer/depth mIU
training data error

predicted
A B ← grand truth
union
A + B
IU mIU

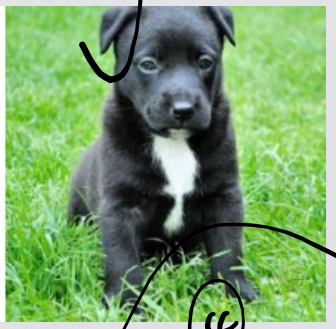
Bias and Variance

high var ⇒ more data
regularization
Test/validation dat error.

Cat classification

Cat

Dog



	overfitting	underfitting	③	④
Train data error :	1%	15%	15%	0.5%
Test(validation/dev) data error:	15%	16%	30%	1%
			high var	low bias
			high bias	low var.

平衡偏差 $\hat{y} \to y$ (w)

Regularization

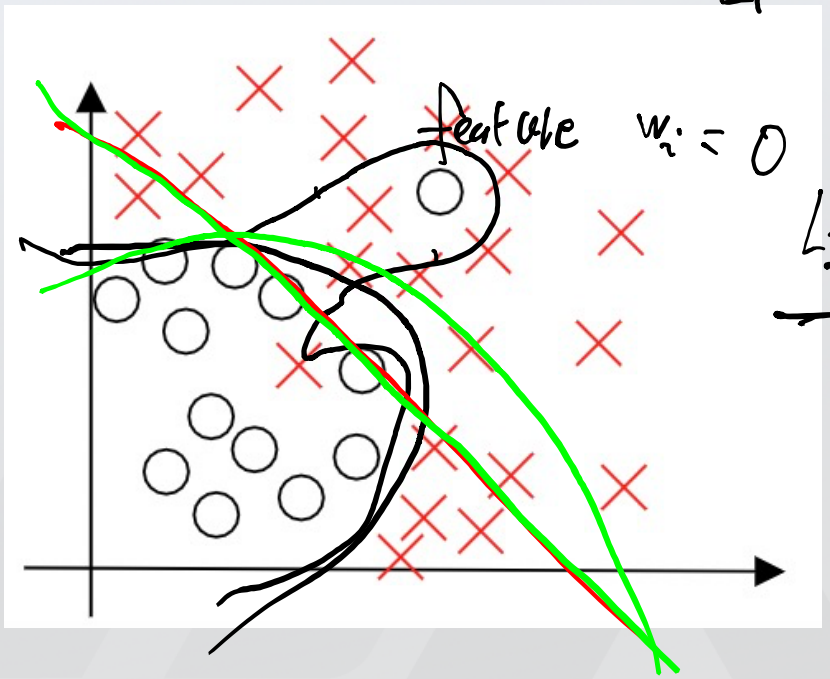
$$J(w,b) = \left[\frac{1}{m}\sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)})\right] + \frac{\lambda}{2m}\Omega(w)$$

hyper parameter

real number of sample

$$L_1: \frac{\lambda}{2m}\sum_{i=1}^{n_x}(|w_i|) = \frac{1}{2m}\|w\|_1 \qquad + \frac{\lambda}{2m} b^2 \quad b \text{ real number}$$

$$L_2 = \sum_{i=1}^{n_x} w_i^2 = w^T w$$



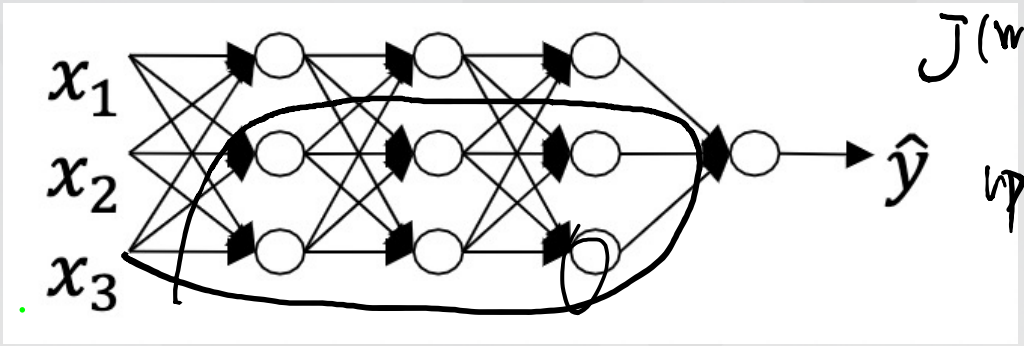
w_1, w_2

$$J(w,b) = G + \frac{\lambda}{2m}(|w_1| + |w_2|)$$

update w_1: $w_1 := w_1 - \alpha dw_1 = w_1 - \frac{\alpha\lambda}{2m}\text{sign}(w_1) - \frac{\partial G}{\partial w_1}$

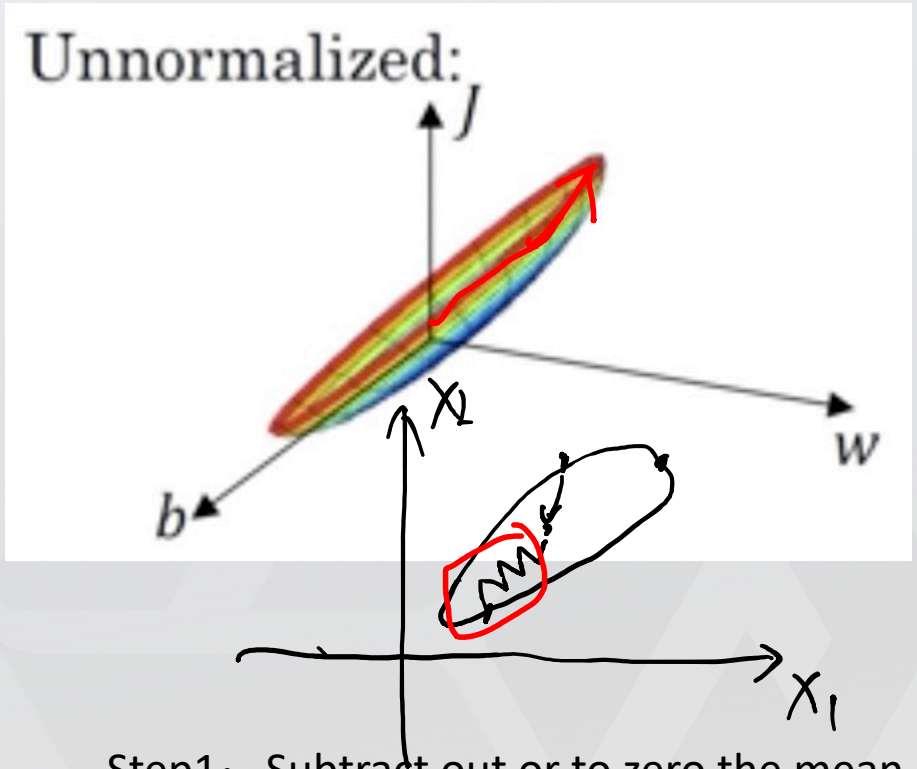
$$J(w,b) = G + \frac{\lambda}{2m}(w_1^2 + w_2^2)$$

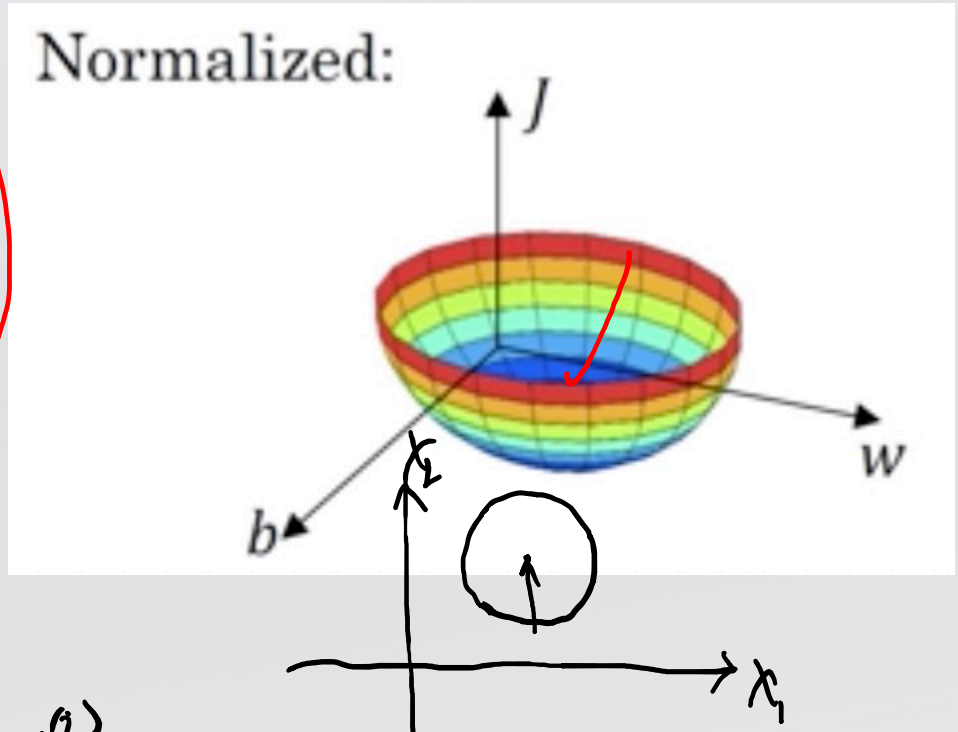
update $w_1 := (1 - \frac{\alpha\lambda}{m})w_1 - \frac{\partial G}{\partial w_1}$ ⇒ 这种情况叫做 weight decay



Normalizing inputs

加快收敛速度.





Step1： Subtract out or to zero the mean

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}$$

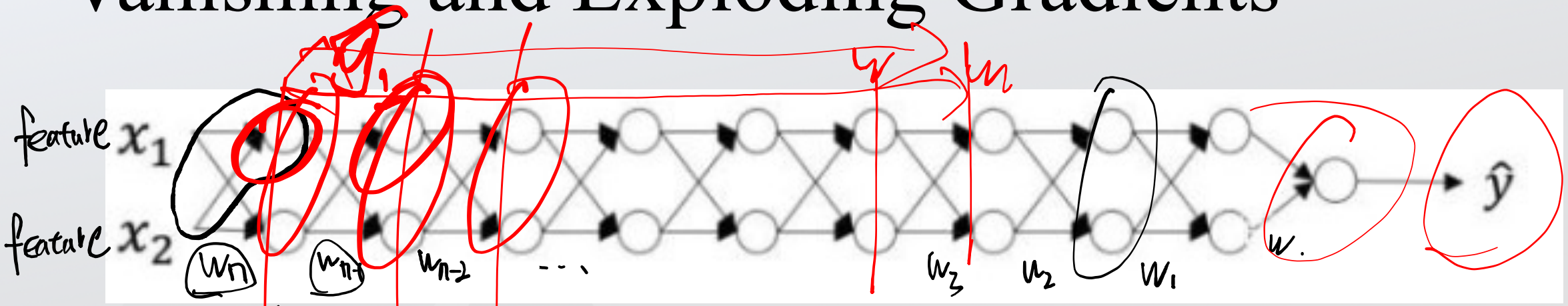
$$x := x - \mu$$

Step2： Normalize the variances

$$\sigma^2 = \frac{1}{m}\sum_{i=1}^{m} x^{(i)2}$$

$$x_i := \frac{x}{\sigma^2}$$

Vanishing and Exploding Gradients



active function $g(z) = z$ bias $b = 0$

$\hat{y} = w^{(n)} \cdot w^{(n-1)} w^{(n-2)} \cdots w^{(3)} \cdot w^{(2)} \cdot w^{(1)} \cdot x$ $+b = 0$ $= [w]^n \cdot x$

$\begin{bmatrix} 1.5 & 0 \\ 0 & 1.5 \end{bmatrix} \begin{bmatrix} 1.5 & 0 \\ 0 & 1.5 \end{bmatrix} \cdots \times$ 内存爆掉

$\begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} \longrightarrow 0$

$z^{(1)} = w^{(1)} \cdot x$

$a^{(1)} = g(z^{(1)}) = z^{(1)}$

$a^{(1)} = g(z)^{(1)}) = g(w^{(1)} \cdot a^{(1)}) = w^{(1)} \cdot w^{(1)} \cdot x$

Vanishing and Exploding Gradients

To prevent this problem：

$$E[a^{[l-1]}] = E[a^{[l]}]$$
$$Var(a^{[l-1]}) = Var(a^{[l]})$$

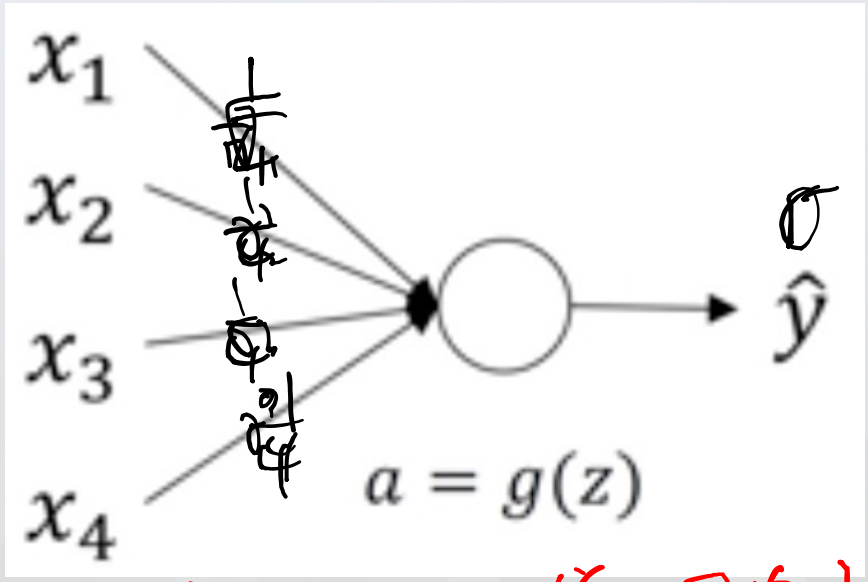
$\hat{y} = w \cdot x$

$a = L(\hat{y})$

$\frac{\partial L}{\partial \hat{y}}$

$\frac{\partial L}{\partial w}$

Weight initialization for deep networks



$\hat{y} = w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 x_4 + \cdots w_n x_n$

n

$n \nearrow$ $w \searrow$

$\frac{1}{4}x1 + \frac{1}{4}x1 + \frac{1}{4}x1 + \frac{1}{4}x1 = 1$

$\frac{2}{4}x0 + \frac{2}{4}x0 + \frac{2}{4}x1 + \frac{2}{4}x1 = 1$

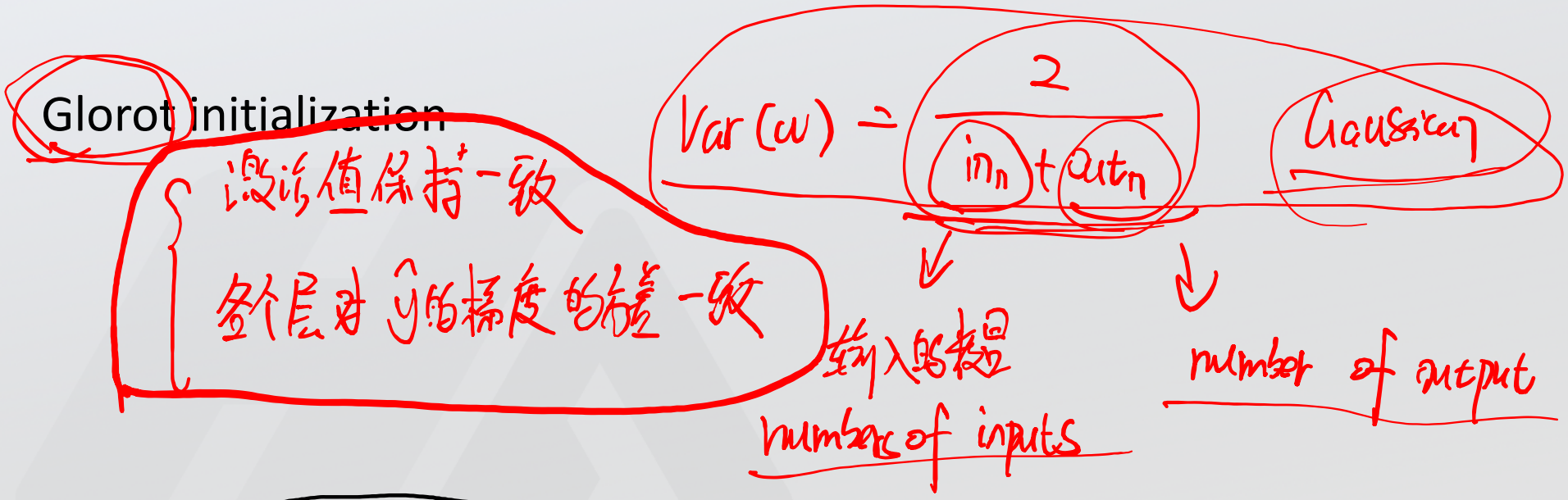
每层的 output 的值与网络深度有关

每个 $\hat{y}$ 的梯度值与网络层有关

多个层的 w 的梯度值与层数有关

Weight initialization for deep networks

Glorot initialization



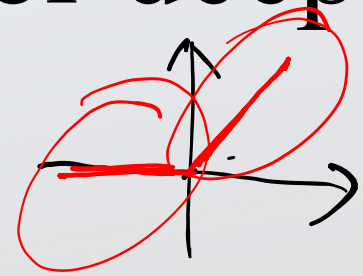
Keras: lecun_normal(seed=**None**), glorot_uniform(seed=**None**)

Link: http://proceedings.mlr.press/v9/glorot10a/glorot10a.pdf

Weight initialization for deep networks

He initialization

relu



$$Var(w) = \frac{2}{n}$$

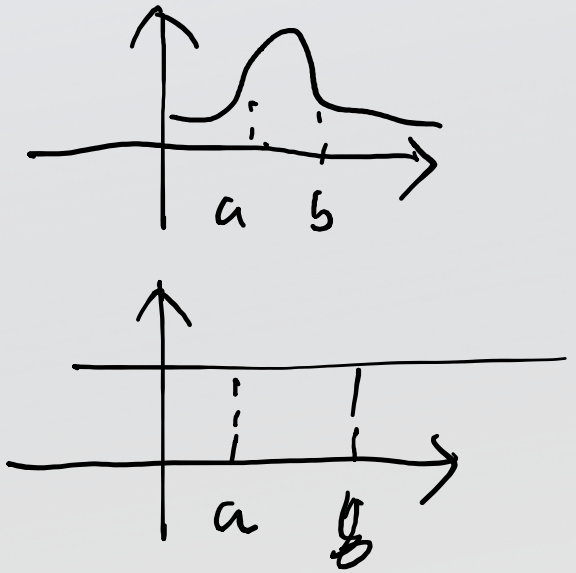
Keras：he_normal(seed=**None**)， he_uniform(seed=**None**)
Link：https://arxiv.org/pdf/1502.01852.pdf

Weight initialization for deep networks

Lechun initialization

$y = wx + b$

$Var(w) = \frac{1}{n}$



Keras：lecun_normal(seed=**None**)，lecun_uniform(seed=**None**)

Link：http://yann.lecun.com/exdb/publis/pdf/lecun-98b.pdf

一所专注前沿互联网技术领域的创新实战大学