

To maintain a balance within a network so that things don't explode/vanish.

Weight Initialization

Zero → ~~not~~ derivative w.r.t loss is same for every w .

Constant → ~~not~~ weights are updated with the same value.

$w \sim \text{uniform}[-a, a]$ → a small +ve number

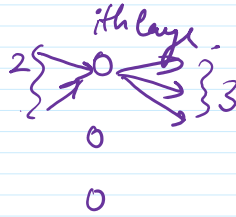
2010: Xavier Initialization

$$\text{uniform} \left(\frac{-\sqrt{6}}{\sqrt{f_{in} + f_{out}}}, \frac{\sqrt{6}}{\sqrt{f_{in} + f_{out}}} \right)$$

i^{th} layer

f_{in} → no. of inputs into a neuron.

f_{out} → no. of outputs going out of a neuron.



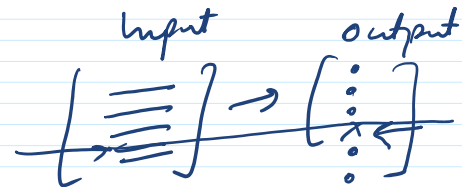
Dataints :-

$$image \sim [R, a, B]$$

↓ ↓ ↓

$$[0, 255] \dots$$

Normalization → scaling inputs into a range.



Standardization :-

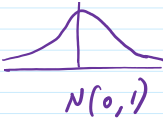
$$y = \frac{x - \mu}{\sigma}$$

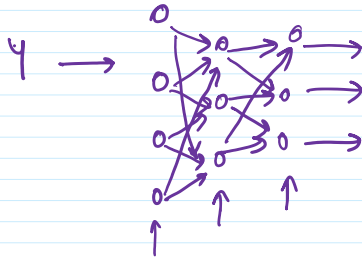
↙ ↘

normalized data.

$$y = \frac{x - \mu}{\sigma}$$

mean = 0
variance = 1





At every epoch, weights are updated and different data is being processed everytime.

Batch Normalization :-

Minibatch GD

Batch 1 \rightarrow $\begin{bmatrix} | & | & | & | & \dots & | \\ 1 & 2 & \dots & m \end{bmatrix}$
m samples.

$$\mu_B = \frac{1}{m} \sum_{i=1}^m x_i$$

$$\sigma_B = \frac{1}{m} \sum_{i=1}^m (x_i - \mu_B)^2$$

$$y_i = \frac{x_i - \mu_B}{\sigma_B}$$

Batch after normalization