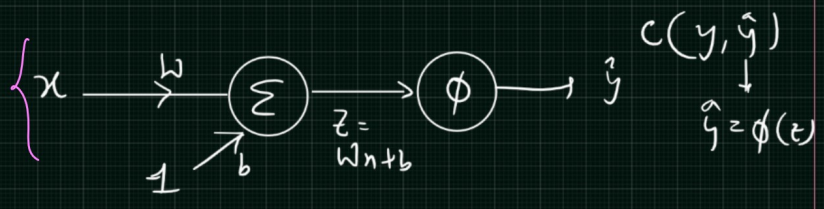
Activation Functions



forward pass ✓

$$\hat{y} = \phi(z)$$

where $z = wx + b$

domain → $(-\infty, \infty)$

range → $(0, 1)$ sigmoid

for forward pass :-

use $\phi(z)$

consider its domain & range

$\phi \rightarrow$ activation fn

Backward pass

$$\frac{\partial C}{\partial w} = \frac{\partial C}{\partial a} \cdot \frac{\partial a}{\partial z} \cdot \frac{\partial z}{\partial w}$$

$a' = \phi'(z) = \frac{\partial \phi}{\partial z}$ ✓

domain → $(-\infty, \infty)$ ←

range → $(0, 0.25)$ ←

sigmoid

$\sigma'(z) = \sigma(z)\{1 - \sigma(z)\}$

at $z = 0$ $\sigma'(z=0) = 0.25$

$z = +\infty$ or $-\infty$ $\sigma'(z) \simeq 0$

for backward pass

use $\phi'(z)$

Consider its domain & range.

1) Sigmoid activation fn

$$\sigma(z) = \frac{1}{1+e^{-z}}$$

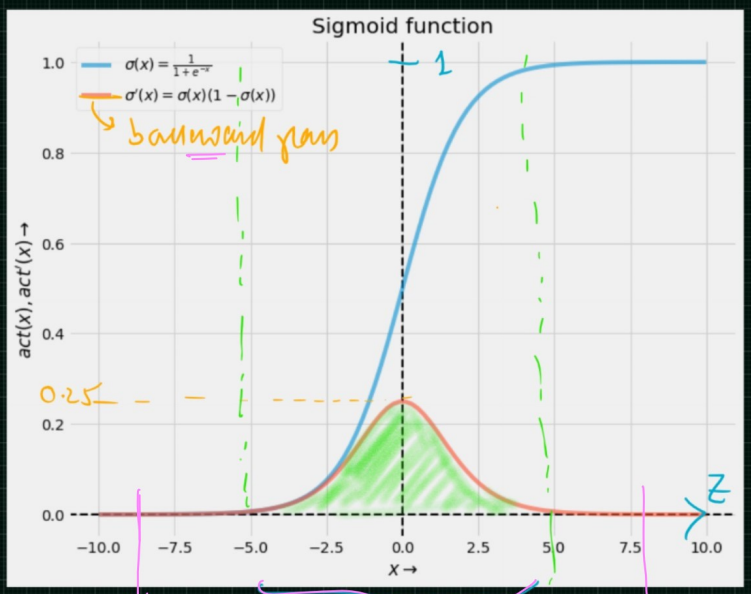
range $\sigma(z) \in (0, 1)$

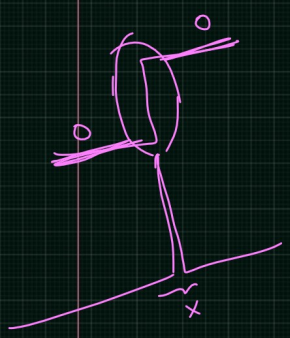
domain $z \in (-\infty, \infty)$

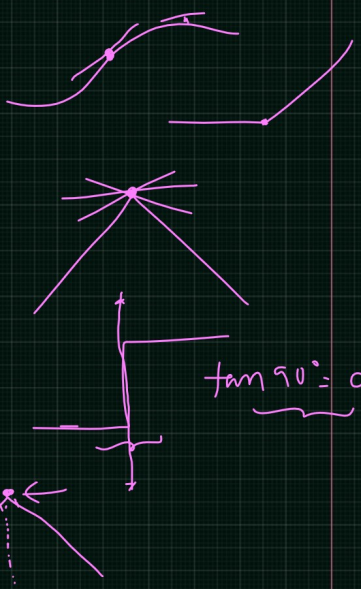
$$\sigma'(z) = \sigma(z)\{1-\sigma(z)\}$$

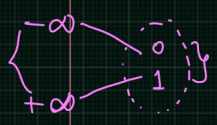
range $\sigma'(z) \in (0, 0.25)$

domain $z \in (-\infty, \infty)$









Advantages :-

1) It has smooth gradient => prevents "jumps" in output.
   Continuous f^n => derivable everywhere.
2) output value or Range $\sigma(z)$ is between 0 & 1.
   => normalizing the o/p of each neuron
   => help us to reduce the solution space.

Disadvantages :-

1) Prone to gradient vanishing
   $\sigma'(z) \in (0, 0.25)$ $\quad 0.25 < 1$
2) Power operation or exponential terms makes calculation time consuming
   ⇓
   latency ↑

$\left\{\frac{1}{1+e^{-z}}\right\}$

2. Hyperbolic Tangent activation function

$$\tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$$

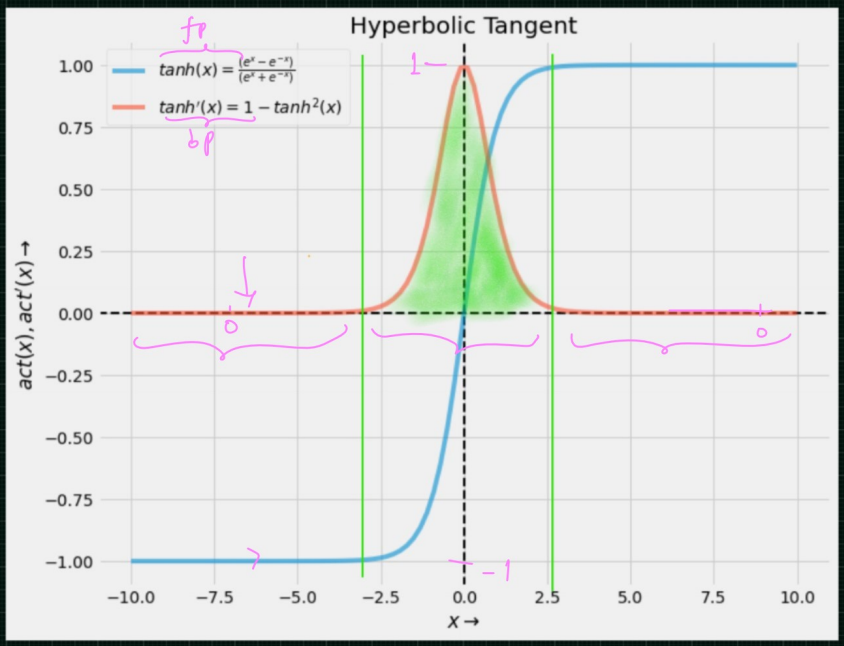
range → $\tanh(z) \in (-1, 1)$

domain → $z \in (-\infty, \infty)$

$$\tanh'(z) = 1 - \tanh^2(z)$$

range → $\tanh'(z) \in (0, 1)$

domain → $z \in (-\infty, \infty)$



(0, 1)

Advantages:-

1) Smooth gradients.
2) Symmetric to origin ⇒ Zero mean
3) Solves vanishing gradient issue to some extent provided there is a proper Wt. initialization and also because its range for $\tanh'(x)$ is between (0, 1)
4) Suitable function for hidden layers

Disadvantage.

1) It also has saturation region
2) It introduces latency because of exponential terms.

3> Rectified linear unit (ReLU) activation fn.

$$ReLU(z) = max(z, 0)$$

— or —

$$ReLU(z) = \begin{cases} z & z \geq 0 \\ 0 & z < 0 \end{cases}$$ if else

domain $z \in (-\infty, \infty)$

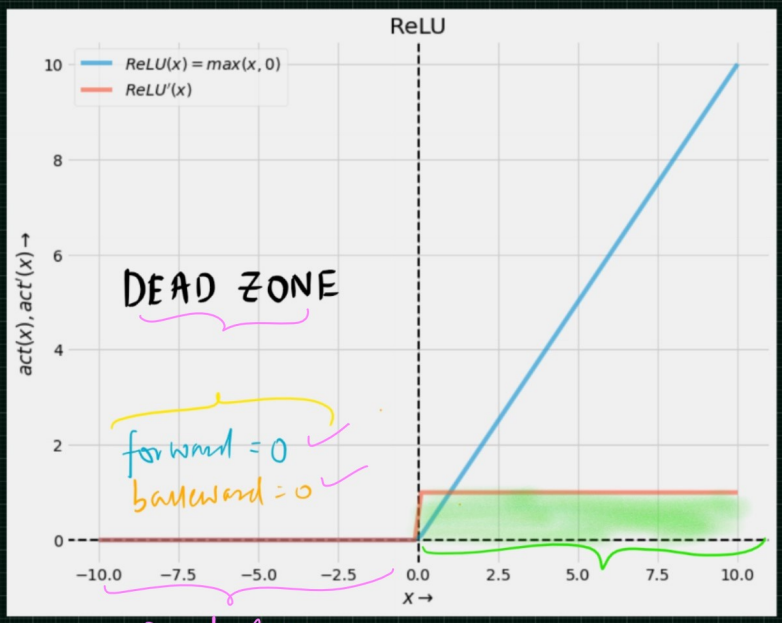
range $ReLU(z) \in [0, \infty)$

$$ReLU'(z) = \begin{cases} 1 & z > 0 \\ 0 & z < 0 \end{cases}$$ if else

derivative is not defined at $z = 0$

∴ domain $z \in (-\infty, \infty)$ except $z = 0$

range $ReLU'(z) \in 0$ and 1.



Dead ReLU

Advantages :-

1> If input is +ve => There is no gradient saturation problem

2> Calculation is much faster {forward or backward} as compared to tanh & sigmoid.

Disadvantage :-

1> for -ve inputs ReLU is completely inactive

⇓

Dying ReLU problem.

2> Not a zero centric function.

3> At zero its derivative is not defined => jump

4) Leaky ReLU function

$\frac{d(\alpha z)}{dz} = \alpha(1) = \alpha$

Leaky ReLU$(z) = \max(z, \alpha z)$

— or —

$$\text{ReLU}(z) = \begin{cases} z & z \geq 0 \\ \alpha z & z < 0 \end{cases}$$

domain $z \in (-\infty, \infty)$

Range of LeakyReLU$(z) \in (-\infty, \infty)$

$\alpha = 0.01$ (0.01z)

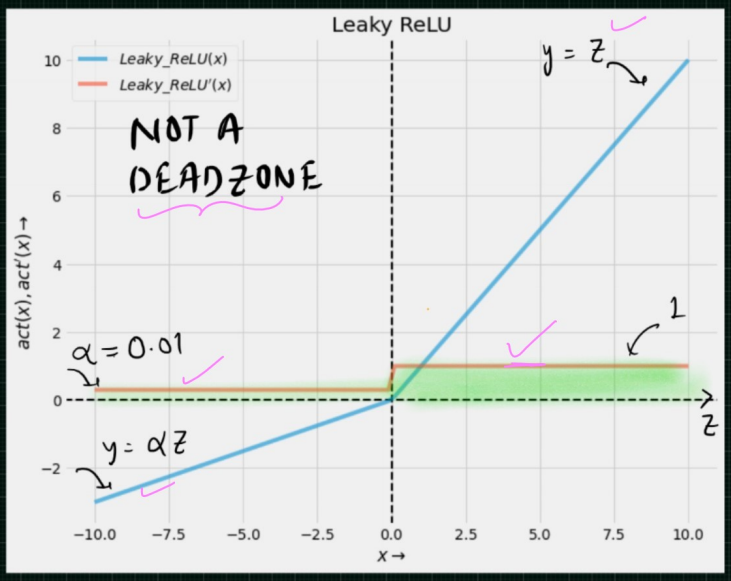
$$\text{Leaky ReLU}'(z) = \begin{cases} 1 & z > 0 \\ \alpha & z < 0 \end{cases}$$

derivative is not defined at $z = 0$

∴ domain $z \in (-\infty, \infty)$ except $z = 0$

range: 0.01 or 1

Leaky ReLU$'(z) \in \alpha$ or 1.



Advantages:-

1) ~~If input is +ve~~ for all input ⇒ There is no gradient saturation problem

2) Calculation is much faster {forward or backward} as compared to tanh & sigmoid.

Disadvantage:-

1) ~~for -ve inputs ReLU is completely inactive~~

⇓

~~Dying ReLU problem.~~

2) Not a zero centric function.

3) At zero its derivative is not defined ⇒ jump

5> PRelu (Parametric ReLU)

$$PRelu(z) = \begin{cases} z & z \geq 0 \\ \alpha z & z < 0 \end{cases}$$

α → learnable parameter / trainable

$$\omega = \omega - \eta \frac{\partial e}{\partial \omega}$$

$$\alpha = \alpha - \eta \frac{\partial e}{\partial \alpha}$$

if $\alpha = 0$ ⇒ ReLU

if $\alpha > 0$ ⇒ Leaky ReLU

if α is learnable parameter ⇒ PRelU

Advantage:

1> α is now intelligently updated / learnt during the training

⇓

No manual tweaking required to set α

⇓

α will be adaptable as per the data

Disadvantage:

1> You have extra parameter α to train

⇓

Training time ↑