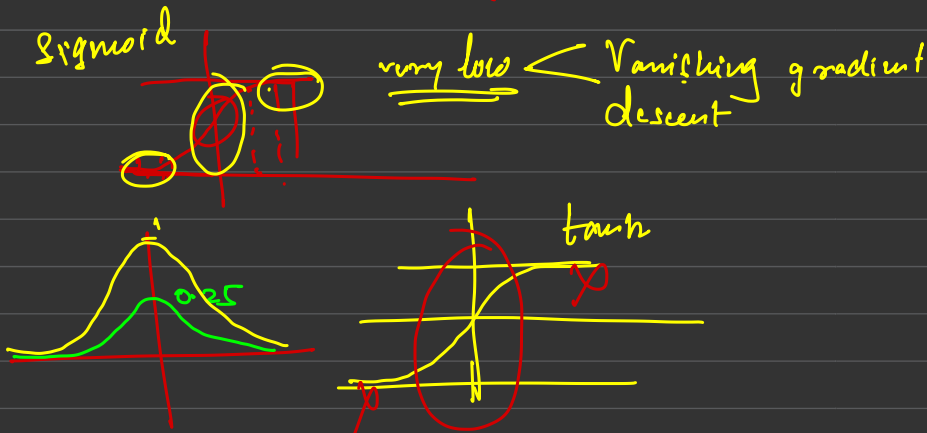


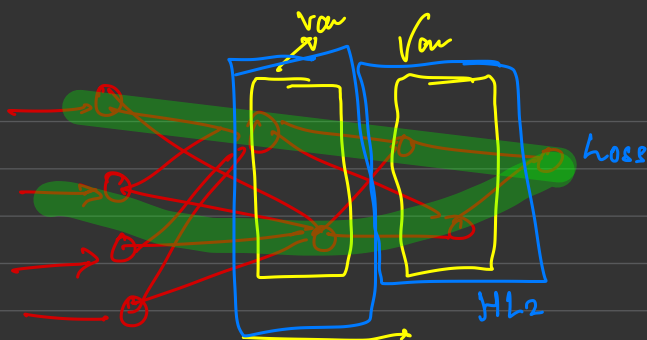
Weight Initialization

- 1) Define your own architecture, input-dim, $HL \rightarrow X$ neurons, o/L, activation, loss, metrics
- 2) weights, and bias in these \rightarrow randomly initialized
- 3) Training \rightarrow FPE BP, the weights get trained

Zero Initialization:
 \hookrightarrow the entire network becomes symmetrical
 $\rightarrow X$ Random Initialization

\hookrightarrow generate weights which are very high
 \hookrightarrow activations high





forward flowing signal

$$\text{var} \left(\frac{dh}{dH_{L2}} \right) = \text{var} \left(\frac{dh}{dH_{L1}} \right)$$

to maintain that the neural network does not enter into a Vanishing or Exploding gradient problem.

Xavier initialization
He initialization

Xavier initialization (tanh)

$$W^{[i]} \sim N(0, 1) \times \sqrt{\frac{1}{n_i}}$$



$$W^{[i]} \sim N(0, 1) \times \sqrt{\frac{2}{n_{i+1} + n_i}}$$

Normalized Xavier initialization

For ReLU activation

He initialization:

$$N(0, 1)$$

$$W^{(L)} \sim N(n_{i+1}, n_i) \times \sqrt{\frac{2}{n_i}}$$

The diagram illustrates the He initialization formula with arrows indicating the mapping of variables to their matrix dimensions:

- n_{i+1} is the number of nodes in the next layer, shown as the number of columns in the weight matrix.
- n_i is the number of nodes in the current layer, shown as the number of rows in the weight matrix.
- $\sqrt{\frac{2}{n_i}}$ is the scaling factor, shown as a multiplier of the Gaussian distribution.