VIT Vidyalankar Institute of Technology
(Accredited A+ by NAAC)
(Autonomous Institute Affiliated to University of Mumbai)

Mid Semester Examination

MSE-I

Branch	Date	Sem.	Roll No. / Exam Seat No.	Subject	Student's Signature	Junior Supervisor's Name and Sign
		VIII		Deep Learning		

Question No.	A	B	C	D	E	F	G	H	Total	Total out of (20 /30 / 40)
1										
2										
3										
4										

Examiners Signature	Student's Sign (After receiving the assessed answer sheet)

Q1(a) Inputs x_1, x_2

Weights $w_1 = 1, w_2 = 1$

Threshold $\theta = 2$

Activation function

$$y = \begin{cases} 1 & \text{if } (w_1 x_1 + w_2 x_2) \geq \theta \\ 0 & \text{otherwise} \end{cases}$$

Simulation

x_1	x_2	$w_1 x_1 + w_2 x_2$	θ	o/p of y
0	0	0	2	0
0	1	1	2	0
1	0	1	2	0
1	1	2	2	1

Only when both $x_1 = 1$ & $x_2 = 1$ the

VIT Vidyalankar Institute of Technology
(Accredited A+ by NAAC)
(Autonomous Institute Affiliated to University of Mumbai)

Mid Semester Examination

Branch	Date	Sem.	Roll No. / Exam Seat No.	Subject	Student's Signature	Junior Supervisor's Name and Sign

Question No.	A	B	C	D	E	F	G	H	Total	Total out of (20 /30 / 40)
1										
2										
3										
4										

Examiners Signature	Student's Sign (After receiving the assessed answer sheet)

Q1 (B) $\eta = 0.1$ $w_1, w_2 = [0.5, -0.5]$

$x = [1, 2]$ $B = 0.1$ $d = 0.8$

y = net = Linear

Iteration 1

$y_1 = (0.5 \times 1) + (-0.5 \times 2) + 0.5 = -0.4$

Error $(d - y_1) = 0.8 - (-0.4) = 1.2$

$\Delta w_1 = 0.1 \times 1.2 \times 1 = 0.12$

$\Delta w_2 = 0.1 \times 1.2 \times 2 = 0.24$

new weight

$w_1 = 0.5 + 0.12 = 0.62$

$w_2 = -0.5 + 0.24 = -0.26$

Update bia = $\Delta b = 0.1 \times 1.2 = 0.12$

$b = 0.1 + 0.12 = 0.22$

Iteration 2 =

$y = (0.62 \times 1) + (-0.26 \times 2) + 0.22$

$= 0.32$

$E = 0.8 - 0.32 = 0.48$

~~Update~~ update weight

$\Delta w_1 = 0.1 \times 0.48 \cdot 1 = 0.048$

$\Delta w_2 = 0.1 \times 0.48 \cdot 2 = 0.0096$

New weight

$w_1 = 0.62 + 0.048 = 0.668$

$w_2 = -0.26 + 0.096 = -0.164$

Update bias

$\Delta b = 0.1 \times 0.48 = 0.048$

$b_{new} = 0.22 + 0.048 = 0.268$

Q2a)

Given $x_1 = 1 \quad x_2 = -1$

$$w = \begin{bmatrix} 1 & -2 \\ -1 & 1 \end{bmatrix} \qquad b = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$f(x) = \frac{1}{1+e^{-x}}$$

$$\sigma\left(\begin{bmatrix} 4 \\ -2 \end{bmatrix}\right) = \begin{bmatrix} 0.982 \\ 0.119 \end{bmatrix}$$

2) tan H

$$f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$$

$$\sigma\left(\begin{bmatrix} 4 \\ -2 \end{bmatrix}\right) = \begin{bmatrix} 0.9994 \\ -0.964 \end{bmatrix}$$

3) ReLU

$$f(x) = 0 \quad \text{for } x < 0$$

$$x \quad x \geq 0$$

$$\sigma\left\{\begin{bmatrix} 4 \\ -2 \end{bmatrix}\right\} = \begin{bmatrix} 4 \\ 0 \end{bmatrix}$$

(4) Arc Tan

$$f(x) = \tan^{-1}(x)$$

$$\sigma\left(\begin{bmatrix} 4 \\ -2 \end{bmatrix}\right) = \begin{bmatrix} 1.3258 \\ -1.107 \end{bmatrix}$$

(5) Leaky Relu

$$f(x) = ax \quad \text{for } x < 0$$

$$= x \quad x \geq 0$$

$$a = 0.1$$

$$\therefore \sigma\left(\begin{bmatrix} 4 \\ -2 \end{bmatrix}\right) = \begin{bmatrix} 4 \\ -0.2 \end{bmatrix}$$

Q2(b) $y_1 = 0.9 \times 0.786 + 0.2 \times 0.808$

$= 0.869$

$t^1 = 0.7$

$y_2^1 = 0.578$

$h_{11} = 0.3 \times 0.1 + 0.5 \times 0.3 + 0.5 = 0.68$

$h_{12} = 0.3 \times 0.2 + 0.5 \times 0.4 + 0.5 = 0.76$

$h_{21} = 0.786$

$h_{22} = 0.808$

$y_1^1 = 0.704$

$y_2^1 = 0.58$

$h_{11} = \frac{1}{1+e^{-0.68}} = 0.66$

$h_{12} = \frac{1}{1+e^{-0.76}} = 0.68$

$MSE = \frac{1}{2}(0.704-0)^2 + (0.58-1)^2$

$= 0.336$

Q3(a)

Mathematical Formulation of ReLU

The Rectified Linear Unit (ReLU) activation function is mathematically expressed as

$$f(x) = \max(0, x)$$

Where $f(x)$ is the o/p of the activation function
x is the input to the neuron.

ReLU returns x if $x > 0$ & 0 if $x \leq 0$

Relu is commonly used in Deep Learning because

1) Simplicity and Efficient
2) Sparsity
3) Gradient Efficiency
4) ReLU mimics biological neural activation by being inactive for certain inputs and Linearly active for others

Dying ReLU problem

It occurs when a neuron consistently outputs Zero during training, effectively becoming inactive and ceasing to learn. This can happen when the inputs to the ReLU is negative or the weights updates in such a way that the neuron never activates again.

Solution to mitigate the dying ReLU problem is

(1) Leaky ReLU

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha x & \text{if } x \leq 0 \end{cases}$$

α is a small positive constant (e.g 0.01)

(2) Parametric ReLU

Similar to Leaky ReLU, but α is Learned during training

(3) Randomized ReLU

Introduces randomness by sampling α from a uniform distribution during training, providing variability to reduce the chance of neurons dying.