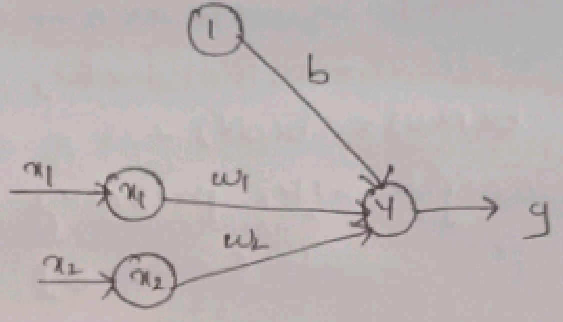
8a. Find the weights using the perceptron network for ANDNOT function when all the inputs are presented only once. Use bipolar inputs and targets.

→

x_1	x_2	t
1	1	-1
1	-1	1
-1	1	-1
-1	-1	-1



Network for ANDNOT function.

Let $\alpha = 1$, $\theta = 0$.

(1, 1, -1)

$y_{in} = b + x_1 w_1 + x_2 w_2 = 0 + 1 \times 0 + 1 \times 0 = 0.$

Applying activation function over the net input.

$$y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} > \theta \\ 0 & \text{if } -\theta \le y_{in} \le \theta. \\ -1 & \text{if } y_{in} < -\theta \end{cases}$$

$w_1(\text{new}) = w_1(\text{old}) + \alpha t x_1 = 0 + 1 \times -1 \times 1 = -1$

$w_2(\text{new}) = w_2(\text{old}) + \alpha t x_2 = 0 + 1 \times -1 \times 1 = -1.$

$b(\text{new}) = b(\text{old}) + \alpha t = 0 + 1 \times -1 = -1.$

weight after presenting the first sample.

$w = [-1 \ -1 \ -1].$

(1, -1, 1)

$y_{in} = b + x_1 w_1 + x_2 w_2.$

2026.02.20 09:12

$= -1 + 1 \times -1 + (-1 \times -1) = -1 - 1 + 1 = \underline{-1}.$

the output $y = f(y_{in})$

$y = -1, \quad t = 1. \quad y \neq t.$

$w_1(new) = w_1(old) + \alpha t x_1 = -1 + 1 \times 1 \times 1 = 0.$

$w_2(new) = w_2(old) + \alpha t x_2 = -1 + 1 \times 1 \times -1 = -2.$

$b(new) = b(old) + \alpha t =$ ~~0+1×1 = 1~~ $-1 + 1 \times 1 = \underline{0}.$

weights after presenting the second sample are

$w = [0 \ -2 \ 0].$

~~8o~~

$\underline{(-1, 1, -1)}$

$t = -1.$

$y_{in} = b + x_1 w_1 + x_2 w_2 = 0 + -1 \times 0 + 1 \times -2 = 0 + 0 - 2 = -2.$

$w_1(new) = w_1(old) + \alpha t x_1 = -1 + 1 \times 1 \times 1 = 0.$

$w_2(new) = w_2(old) + \alpha t x_2 = -1 + 1 \times 1 \times -1 = -2.$

$b(new) = b(old) + \alpha t = -1 + 1 \times 1 = 0.$

weights after second sample are $w = [0 \ -2 \ 0].$

~~for third~~ input sample. $x_1 = -1, x_2 = 1, t = -1.$

6

$y_{in} = b + x_1 w_1 + x_2 w_2$

$= 0 + -1 \times 0 + 1 \times -2 = 0 + 0 - 2 = -2.$

$y = f(y_{in}) = -1$

$t = y.$ no weight changes.

weights are $[0, -2 \ 0].$

$(-1, -1, -1)$

$y_{in} = b + x_1 w_1 + x_2 w_2 = 0 + -1 \times 0 + -1 \times -2$

$= 0 + 0 + 2 = 2.$

2026.02.20 09:12

$y = f(y_{in}) = 1$.

$t \neq y$.

$w_1(new) = w_1(old) + \alpha t x_1 = 0 + 1 \times -1 \times -1 = 1$.

$w_2(new) = w_2(old) + \alpha t x_2 = -2 + 1 \times -1 \times -1 = -1$.

$b(new) = b(old) + \alpha t = 0 + 1 \times -1 = -1$.

weights after presenting fourth input Sample are

$w = [1 \ -1 \ -1]$.

One epoch of training for ANDNOT function using.

INPUT			Target	Net i/p	Calculated output	Weights		
x_1	x_2	b.	(t)	(y_{in})	(y)	w_1 (0	w_2 0	b 0).
1	1	1	-1	0	0	-1	-1	-1
1	-1	1	1	-1	-1	0	-2	0
-1	1	1	-1	-2	-1	0	-2	0
-1	-1	1	-1	2	1	1	-1	-1

8b. How is the training algorithm performed in back propogation neural networks?

⇒ <u>step 0</u>: Initialize weights and learning rate.

<u>step 1</u>: Perform steps 2-9 when stopping condition is false.

step 2: Perform step 3-8 for each training pair.

Feed - <u>forward phase</u> (Phase I)

step 3: Each input unit receives input signal x_i and sends it to the hidden unit (i=1 to n).

step 4: Each hidden unit z_j (j=1 to p) sums its

2026.02.20 09:13

weighted input signals to calculate net input:

$$z_{inj} = v_{oj} + \sum_{i=1}^{n} x_i v_{ij}$$

Calculate output of the hidden unit by applying activation function,

$$z_j = f(z_{inj})$$

step 5: For each output unit y_k (k=1 to m), Calculate the net input:

$$y_{ink} = w_{ok} + \sum_{j=1}^{p} z_j w_{jk}.$$

and apply the activation function to compute the output signal:

$$y_k = f(y_{ink})$$

Back Propogation of error (phase II)

step 6: Each output unit y_k (k=1 to m) receives a target pattern corresponding to the input training pattern and computes the error correction term:

$$\delta_k = (t_k - y_k) f'(y_{ink})$$

Then update the weights and bias:

$$\Delta w_{jk} = \alpha \delta_k z_j$$

$$\Delta w_{ok} = \alpha \delta_k.$$

Send δ_k to the hidden layer backwords.

step 7: Each hidden unit z_j (j=1 to p) sums its delta inputs from the output ~~[illegible]~~ units:

$$\delta_{inj} = \sum_{k=1}^{m} \delta_k w_{jk}.$$

2026.02.20 09:13

The term δ_{inj} gets multiplied with the derivative of $f(z_{inj})$ to calculate the error term:

$$\delta_j = \delta_{inj} f'(z_{inj})$$

Then update the weights and bias.

$$\Delta v_{ij} = \alpha \delta_j x_i$$

~~Ao~~ $\Delta v_{0j} = \alpha \delta_j$

Weight and bias updation (phase III)

step 8: Each output unit y_k (k=1 to m) updates the bias and weights.

$$w_{jk}(new) = w_{jk}(old) + \Delta w_{jk}.$$

$$w_{0k}(new) = w_{0k}(old) + \Delta w_{0k}.$$

Each output unit z_j (j=1 to p) updates the bias and weights.

$$v_{ij}(new) = v_{ij}(old) + \Delta v_{ij}$$

$v_{0j}(new) =$ ~~vo~~ $v_{0j}(old) + \Delta v_{0j}$

step 9: check for the stopping condition may be certain number of epochs reached or when the actual output equals to target output.

2026.02.20 09:13