Forward vs Backward Propagation

Forward Propagation

- Forward Propagation is the way to move from the Input layer (left) to the Output layer (right) in the neural network.
- Each layer consists of neurons that perform a weighted sum of inputs followed by an activation function.
- Forward propagation calculates the activations of neurons layer by layer until the final output is produced.

Formulas:

1. **Weighted Sum (Z):**

$$Z^{[l]} = W^{[l]}A^{[l-1]} + b^{[l]}$$

Where:

- $Z^{[l]}$ is the weighted sum at layer l.
- $W^{[l]}$ is the weight matrix for layer l.
- $A^{[l-1]}$ is the activation from the previous layer $l-1$.
- $b^{[l]}$ is the bias vector for layer l.

2. **Activation (A):**

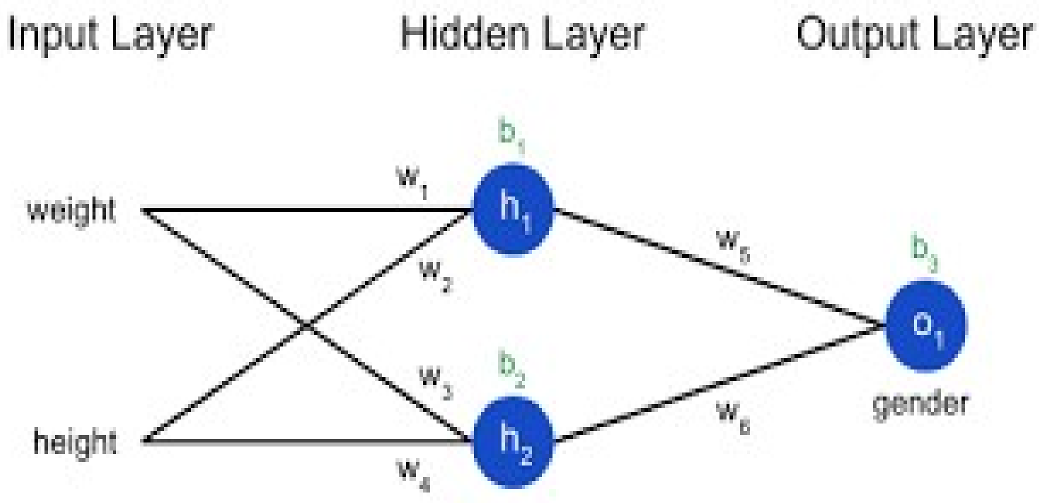
$$A^{[l]} = g^{[l]}(Z^{[l]})$$

Where $g^{[l]}$ is the activation function of layer l.

Example:

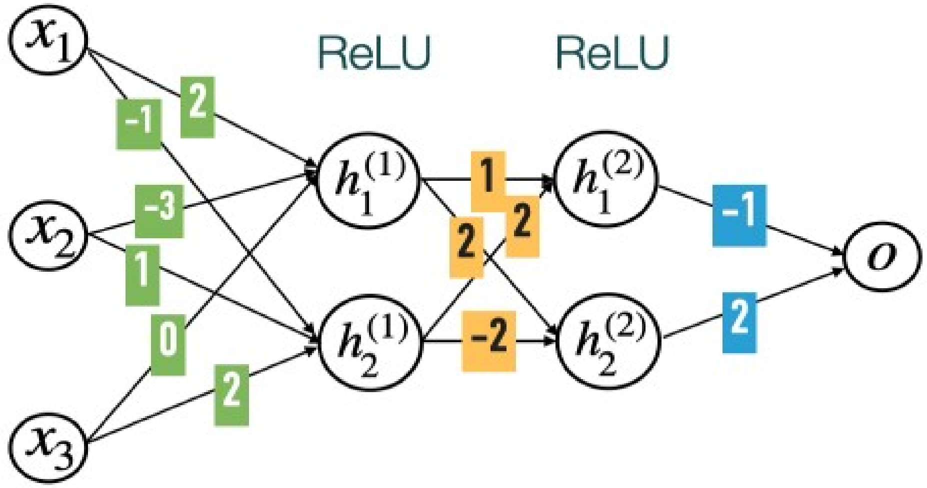
- Suppose we have a simple neural network with 1 input layer, 1 hidden layer with 2 neurons, and 1 output layer with one neuron.

- Forward propagation would involve calculating the activations step by step from the input to the output layer using the defined formulas.



11.1.1 One Output Node

We start with the network in Figure 11.1 as an example. The network receives three inputs x_1, x_2, x_3 and has a first hidden layer with two nodes $h_1^{(1)}, h_2^{(1)}$, a second hidden layer with two nodes $h_1^{(2)}, h_2^{(2)}$, and a final output layer with one node o. We assign the *ReLU* activation function to the hidden units, and define weights as shown in Figure 11.1.



The two hidden nodes in the first hidden layer are characterized by the following equations:

$$
\begin{aligned}
h_1^{(1)} &= ReLU(2x_1 - 3x_2) \\
h_2^{(1)} &= ReLU(-x_1 + x_2 + 2x_3)
\end{aligned}
\tag{11.1}
$$

and the two hidden nodes in the second hidden layer are characterized by the following equations:

$$
\begin{aligned}
h_1^{(2)} &= ReLU(h_1^{(1)} + 2h_2^{(1)}) \\
h_2^{(2)} &= ReLU(2h_1^{(1)} - 2h_2^{(1)})
\end{aligned}
\tag{11.2}
$$

and the output node is characterized by the following equation:

$$o = -h_1^{(2)} + 2h_2^{(2)}$$

Therefore, if we know the input values x_1, x_2, x_3, we can first calculate the values $h_1^{(1)}, h_2^{(1)}$, then using these values, calculate $h_1^{(2)}, h_2^{(2)}$, and finally using these values, we can calculate the output o of the network.

Backward Propagation

- process of moving from the right to left i.e backward from the Output to the Input layer.
- uses the chain rule to compute the gradient efficiently.
- adjusts the neural network's weights to minimize the difference b/w predicted and actual output (i.e., the loss).
- After performing forward propagation and obtaining the output, backpropagation computes gradients layer by layer, starting from the output layer back to the input layer.
- These gradients are then used in optimization algorithms like gradient descent to update the weights and biases iteratively.

Formulas:

1. **Output Layer Error (dZ):**

$$dZ^{[L]} = A^{[L]} - Y$$

Where $A^{[L]}$ is the activation of the output layer and Y is the actual output (target).

2. **Gradient of Weights and Biases (dW, db):**

$$dW^{[l]} = \frac{1}{m} dZ^{[l]} A^{[l-1]T}$$

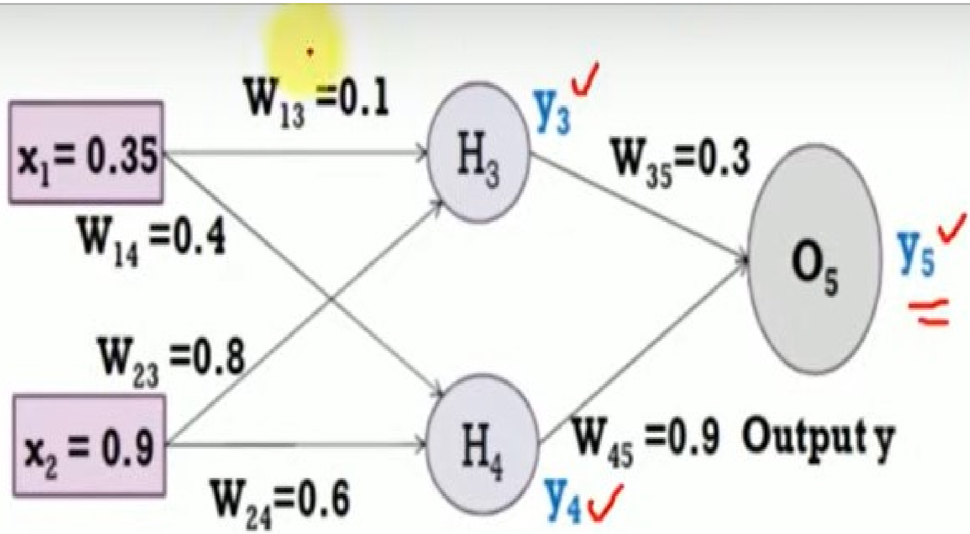
$$db^{[l]} = \frac{1}{m} \sum_{i=1}^{m} dZ^{[l](i)}$$

Where m is the number of examples in the training set.

3. **Backpropagation through Layers:**

$$dZ^{[l-1]} = W^{[l]T} dZ^{[l]} * g'^{[l-1]}(Z^{[l-1]})$$

Where $g'^{[l-1]}$ is the derivative of the activation function in layer $l-1$.



- Forward Pass: Compute output for y3, y4 and y5.

$$a_j = \sum_j (w_{i,j} * x_i) \qquad yj = F(aj) = \frac{1}{1+e^{-a_j}}$$

$$a_1 = (w_{13} * x_1) + (w_{23} * x_2)$$
$$= (0.1 * 0.35) + (0.8 * 0.9) = 0.755$$
$$y_3 = f(a_1) = 1/\ (1 + e^{-0.755}) = 0.68$$

$$a_2 = (w_{14} * x_1) + (w_{24} * x_2)$$
$$= (0.4 * 0.35) + (0.6 * 0.9) = 0.68$$
$$y_4 = f(a_2) = 1/\ (1 + e^{-0.68}) = 0.6637$$

$$a_3 = (w_{35} * y_3) + (w_{45} * y_4)$$
$$= (0.3 * 0.68) + (0.9 * 0.6637) = 0.801$$
$$y_5 = f(a_3) = 1/\ (1 + e^{-0.801}) = \mathbf{0.69} \text{ (Network Output)}$$

Error $= y_{target} - y_5 = -0.19$

0.5 - 0.69

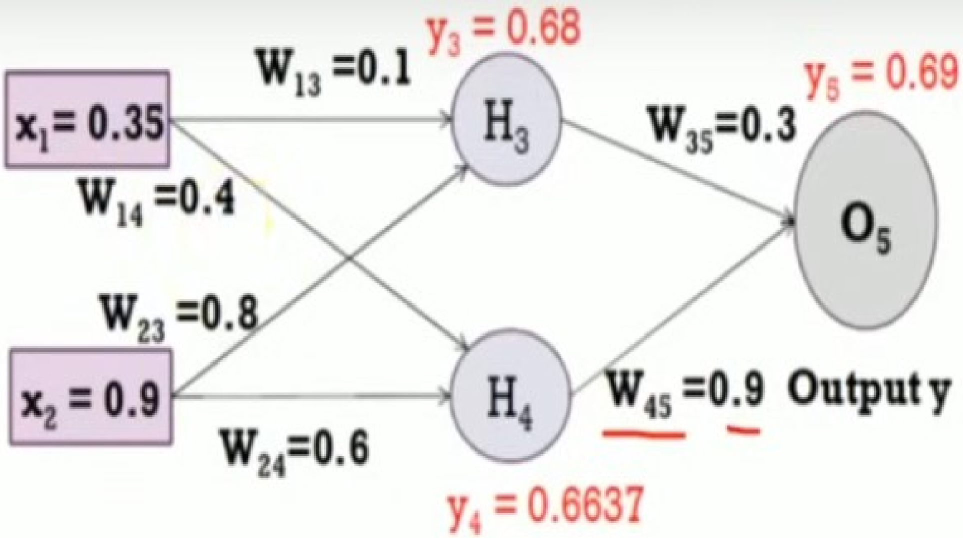
- Each weight changed by:

$$\Delta w_{ji} = \eta \delta_j o_i$$

$$\delta_j = o_j(1 - o_j)(t_j - o_j) \qquad \text{if } j \text{ is an output unit}$$

$$\delta_j = o_j(1 - o_j)\sum_k \delta_k w_{kj} \qquad \text{if } j \text{ is a hidden unit}$$

- where η is a constant called the learning rate
- tj is the correct teacher output for unit j
- δj is the error measure for unit j



- Backward Pass: Compute δ3, δ4 and δ5.

For output unit:

$$\delta_5 = y(1-y)\,(y_{target} - y)$$
$$= 0.69*(1-0.69)*(0.5-0.69) = -0.0406$$

For hidden unit:

$$\delta_3 = y_3(1-y_3)\,w_{35} * \delta_5$$
$$= 0.68*(1-0.68)*(0.3 * -0.0406) = -0.00265$$

Compute new weights

$$\Delta w_{ji} = \eta \delta_j o_i$$

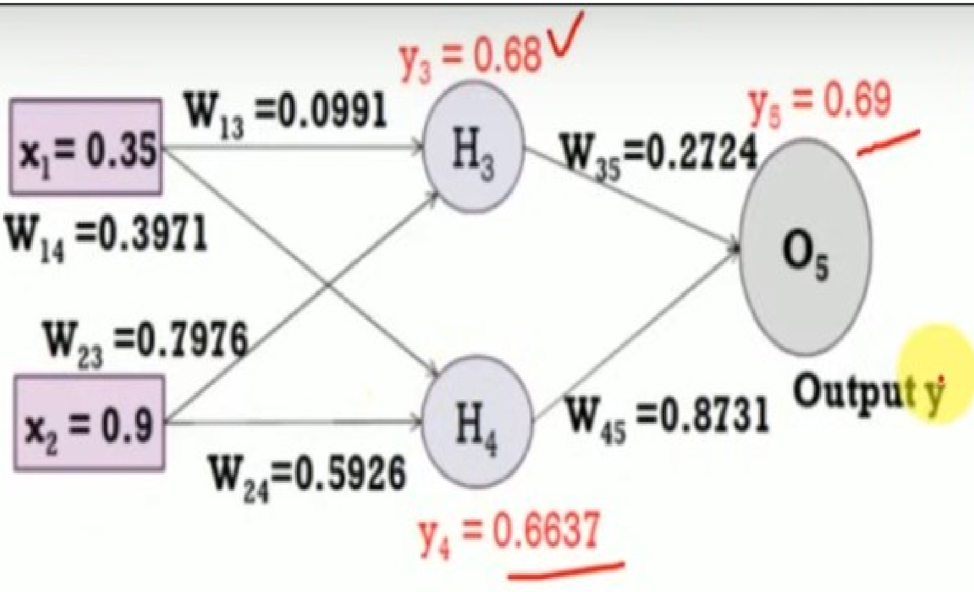
$$\Delta w_{45} = \eta\, \delta_5\, y_4 = 1 * -0.0406 * 0.6637 = \underline{-0.0269}$$
$$w_{45}\,(\text{new}) = \Delta w_{45} + w_{45}(\text{old}) = \underline{-0.0269 + (0.9)} = \underline{0.8731}$$

$$\delta_4 = y_4(1-y_4)w_{45} * \delta_5$$
$$= 0.6637*(1 - 0.6637)*(0.9 * -0.0406) = \underline{-0.0082}$$

$$\Delta w_{14} = \eta\, \delta_4\, x_1 = 1 * -0.0082 * 0.35 = \underline{-0.00287}$$
$$w_{14}\,(\text{new}) = \Delta w_{14} + w_{14}(\text{old}) = -0.00287 + 0.4 = 0.3971$$

- Similarly, update all other weights

i	j	w_{ij}	δ_j	x_i	η	Updated w_{ij}
1	3	0.1	-0.00265	0.35	1	0.0991
2	3	0.8	-0.00265	0.9	1	0.7976
1	4	0.4	-0.0082	0.35	1	0.3971
2	4	0.6	-0.0082	0.9	1	0.5926
3	5	0.3	-0.0406	0.68	1	0.2724
4	5	0.9	-0.0406	0.6637	1	0.8731



- Forward Pass: Compute output for y3, y4 and y5.

$$a_j = \sum_j (w_{i,j} * x_i) \qquad yj = F(aj) = \frac{1}{1+e^{-a_j}}$$

$$a_1 = (w_{13} * x_1) + (w_{23} * x_2)$$
$$= (0.0991 * 0.35) + (0.7976 * 0.9) = 0.7525$$
$$y_3 = f(a_1) = 1/\ (1 + e^{-0.7525}) = 0.6797$$

$$a_2 = (w_{14} * x_1) + (w_{24} * x_2)$$
$$= (0.3971 * 0.35) + (0.5926 * 0.9) = 0.6723$$
$$y_4 = f(a_2) = 1/\ (1 + e^{-0.6723}) = 0.6620$$

$$a_3 = (w_{35} * y_3) + (w_{45} * y_4)$$
$$= (0.2724 * 0.6797) + (0.8731 * 0.6620) = 0.7631$$
$$y_5 = f(a_3) = 1/\ (1 + e^{-0.7631}) = \mathbf{0.6820} \text{ (Network Output)}$$

$$\text{Error} = y_{target} - y_5 = -0.182$$