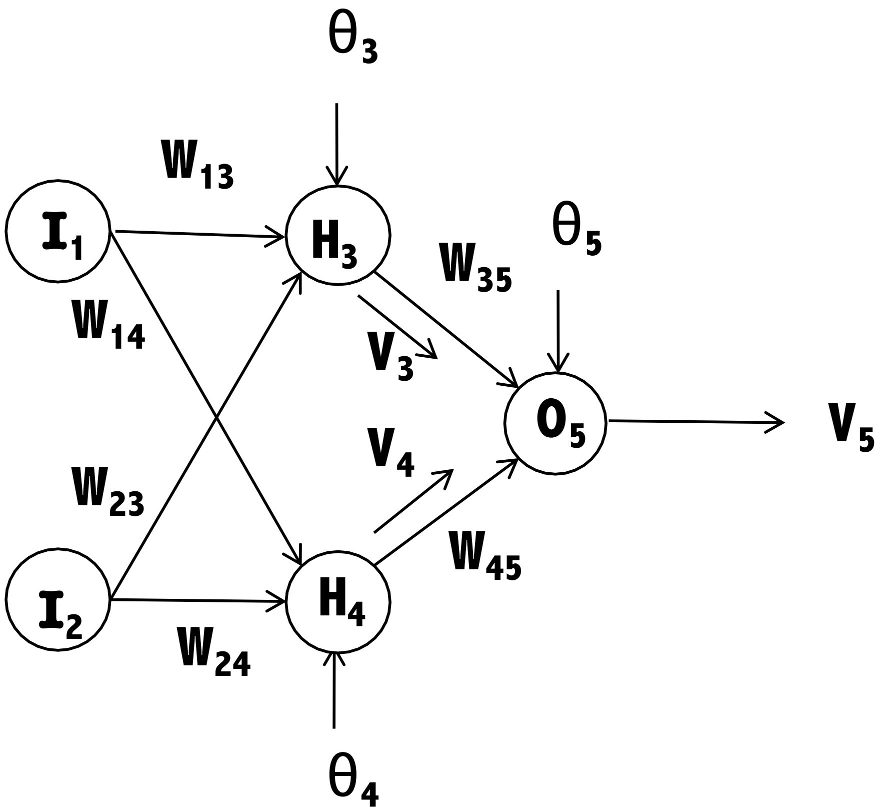
An XOR architecture

Apply the perceptron delta rule to determine the weights for the XOR neural network



For the output node

$\delta_5 = (t_5 - V_5)$

$\Delta W_{i5} = \varepsilon \bullet \delta_5 \bullet V_i$

$\Delta \theta_5 = \varepsilon \bullet \delta_5 \bullet 1$

for i = 3, 4 and ε = learning rate

For hidden nodes (fabricated):

$\delta_i = \varepsilon^* \bullet \delta_5$

$\Delta \theta_i = \varepsilon^* \bullet \delta_i \bullet 1$ (?)

$\Delta W_{ij} = \varepsilon^* \bullet \delta_i \bullet \mathbf{I}_j$

for i = 1, 2; j = 3, 4

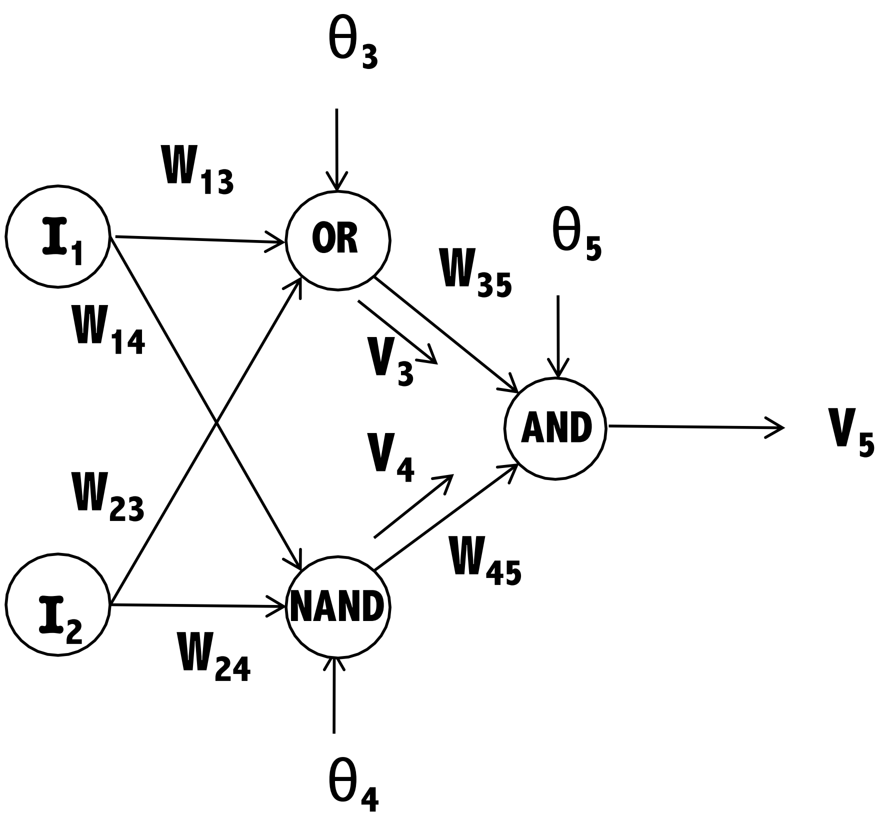
Run the problem with the learning rate, α = 0.25, 0.50, 0.75

You can chose ε^*.

Verify the XOR network function

- Fill out an XOR table showing the initial weights and the final weights upon convergence (i.e., if it does!).
  - This XOR architecture is a three layer perceptron network
  - Besides the inputs I_1 and I_2, the number of weight have increase to W_{13}, W_{14}, W_{23}, W_{24}, W_{35}, and W_{45}, and biases θ_3, θ_4, and θ_5
  - Initialize the weights to zero and randomly with values between 0 and 1, excluding 0 and 1
  - Apply the "delta rule" at each level of the XOR architecture
  - Stopping when there is no further changes in delta

An XOR architecture



Form the XOR function using an OR, NAND and AND function.

Separately, train:

1) The OR function to determine W_{13} and W_{23}
2) The NAND function to determine W_{14} and W_{24}
3) The AND function to determine W_{35} and W_{45}

Then use these weights as the initial weights for the XOR function and apply the fabricated delta rule.