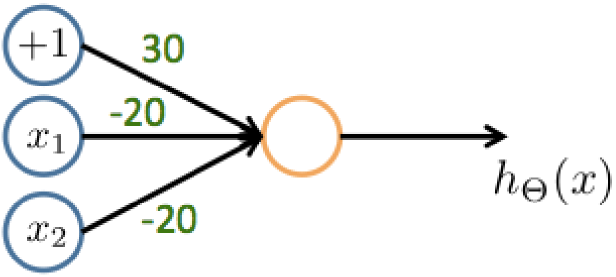
# Feedback — VIII. Neural Networks: Representation

Help Center

You submitted this quiz on **Tue 7 Apr 2015 10:24 AM CEST**. You got a score of **5.00** out of **5.00**.

## Question 1

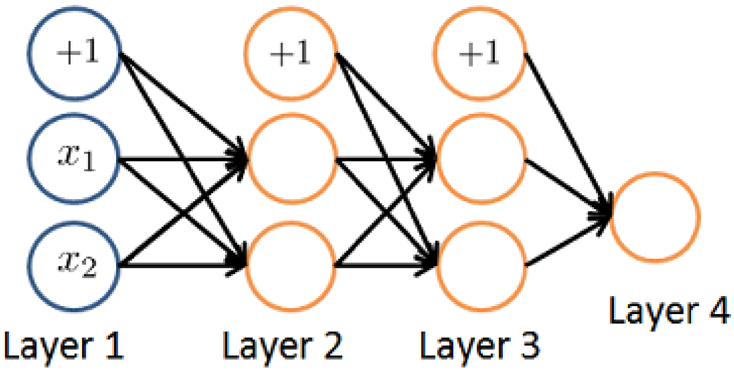
Consider the following neural network which takes two binary-valued inputs $x_1, x_2 \in \{0, 1\}$ and outputs $h_\Theta(x)$. Which of the following logical functions does it (approximately) compute?




| Your Answer | | Score | Explanation |
|---|---|---|---|
| XOR (exclusive OR) | | | |
| AND | | | |
| OR | | | |
| NAND (meaning "NOT AND") | ✔ | 1.00 | This network outputs approximately 1 as long as one of the two inputs is 0. |
| Total | | 1.00 / 1.00 | |

## Question 2

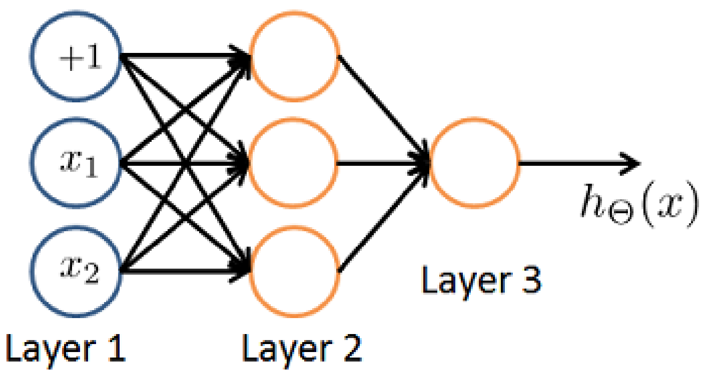
Consider the neural network given below. Which of the following equations correctly computes the activation $a_1^{(3)}$? Note: $g(z)$ is the sigmoid activation function.




| Your Answer | | Score | Explanation |
|---|---|---|---|
| $a_1^{(3)} = g(\Theta_{1,0}^{(1)} a_0^{(2)} + \Theta_{1,1}^{(1)} a_1^{(2)} + \Theta_{1,2}^{(1)} a_2^{(2)})$ | | | |
| $a_1^{(3)} = g(\Theta_{1,0}^{(2)} a_0^{(2)} + \Theta_{1,1}^{(2)} a_1^{(2)} + \Theta_{1,2}^{(2)} a_2^{(2)})$ | ✔ | 1.00 | This correctly uses the first row of $\Theta^{(2)}$ and includes the "+1" term of $a_0^{(2)}$. |
| The activation $a_1^{(3)}$ is not present in this network. | | | |
| $a_1^{(3)} = g(\Theta_{1,0}^{(1)} a_0^{(1)} + \Theta_{1,1}^{(1)} a_1^{(1)} + \Theta_{1,2}^{(1)} a_2^{(1)})$ | | | |
| Total | | 1.00 / 1.00 | |

## Question 3

You have the following neural network:




You'd like to compute the activations of the hidden layer $a^{(2)} \in \mathbb{R}^3$. One way to do so is the following Octave code:

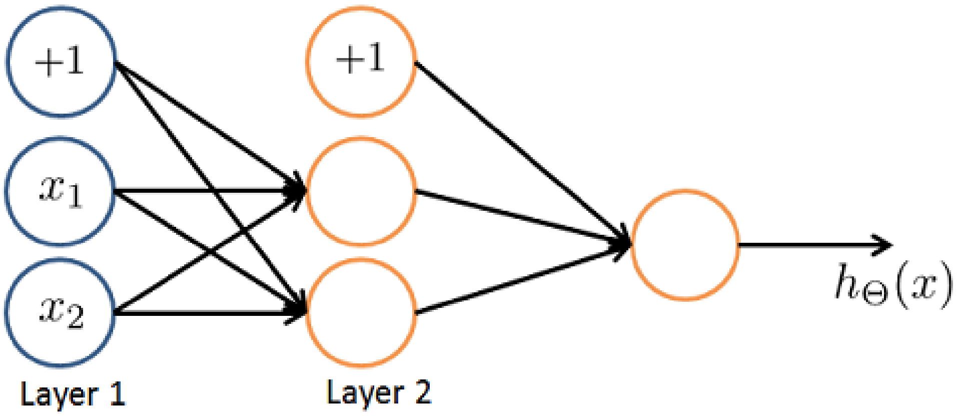
```
% Theta1 is Theta with superscript "(1)" from lecture
% ie, the matrix of parameters for the mapping from layer 1 (input) to laye
r 2
% Theta1 has size 3x3
% Assume 'sigmoid' is a built-in function to compute 1 / (1 + exp(-z))

a2 = zeros (3, 1);
for i = 1:3
  for j = 1:3
    a2(i) = a2(i) + x(j) * Theta1(i, j);
  end
  a2(i) = sigmoid (a2(i));
end
```

You want to have a vectorized implementation of this (i.e., one that does not use for loops). Which of the following implementations correctly compute $a^{(2)}$? Check all that apply.

| Your Answer | | Score | Explanation |
|---|---|---|---|
| ☐ a2 = sigmoid (Theta2 * x); | ✔ | 0.25 | $\Theta^{(2)}$ specifies the parameters from the second to third layers, not first to second. |
| ☐ a2 = sigmoid (x * Theta1); | ✔ | 0.25 | The order of the multiplication is important, this will not work as x is a vector of size $3 \times 1$ while Theta1 is a matrix of size $3x3$. |
| ☐ z = sigmoid(x); a2 = sigmoid (Theta1 * z); | ✔ | 0.25 | You do not need to apply the sigmoid function to the inputs. |
| ☑ z = Theta1 * x; a2 = sigmoid (z); | ✔ | 0.25 | This version computes $a^{(2)} = g(\Theta^{(1)}x)$ correctly in two steps, first the multiplication and then the sigmoid activation. |
| Total | | 1.00 / 1.00 | |

# Question 4

You are using the neural network pictured below and have learned the parameters $\Theta^{(1)} = \begin{bmatrix} 1 & 1 & 2.4 \\ 1 & 1.7 & 3.2 \end{bmatrix}$ (used to compute $a^{(2)}$) and $\Theta^{(2)} = \begin{bmatrix} 1 & 0.3 & -1.2 \end{bmatrix}$ (used to compute $a^{(3)}$} as a function of $a^{(2)}$). Suppose you swap the parameters for the first hidden layer between its two units so $\Theta^{(1)} = \begin{bmatrix} 1 & 1.7 & 3.2 \\ 1 & 1 & 2.4 \end{bmatrix}$ and also swap the output layer so $\Theta^{(2)} = \begin{bmatrix} 1 & -1.2 & 0.3 \end{bmatrix}$. How will this change the value of the output $h_\Theta(x)$?




| Your Answer | Score | Explanation |
| --- | --- | --- |
| It will stay the same. ✔ | 1.00 | Swapping $\Theta^{(1)}$ swaps the hidden layers output $a^{(2)}$. But the swap of $\Theta^{(2)}$ cancels out the change, so the output will remain unchanged. |
| Insufficient information to tell: it may increase or decrease. | | |
| It will increase. | | |
| It will decrease | | |
| Total | 1.00 / 1.00 | |

## Question 5

Which of the following statements are true? Check all that apply.

| Your Answer | Score | Explanation |
| --- | --- | --- |
| Suppose you have a multi-class classification problem with three classes, trained with a 3 layer network. Let $a_1^{(3)} = (h_\Theta(x))_1$ be the activation of the first output unit, and ✔ | 0.25 | The outputs of a neural network are not probabilities, so their sum need not be 1. |

| | | | |
|---|---|---|---|
| similarly $a_2^{(3)} = (h_\Theta(x))_2$ and $a_3^{(3)} = (h_\Theta(x))_3$. Then for any input $x$, it must be the case that $a_1^{(3)} + a_2^{(3)} + a_3^{(3)} = 1$. | | | |
| ☐ A two layer (one input layer, one output layer; no hidden layer) neural network can represent the XOR function. | ✔ | 0.25 | We must compose multiple logical operations by using a hidden layer to represent the XOR function. |
| ☑ Any logical function over binary-valued (0 or 1) inputs $x_1$ and $x_2$ can be (approximately) represented using some neural network. | ✔ | 0.25 | Since we can build the basic AND, OR, and NOT functions with a two layer network, we can (approximately) represent any logical function by composing these basic functions over multiple layers. |
| ☑ If a neural network is overfitting the data, one solution would be to increase the regularization parameter $\lambda$. | ✔ | 0.25 | A larger value of $\lambda$ will shrink the magnitude of the parameters $\Theta$, thereby reducing the chance of overfitting the data. |
| Total | | 1.00 / 1.00 | |