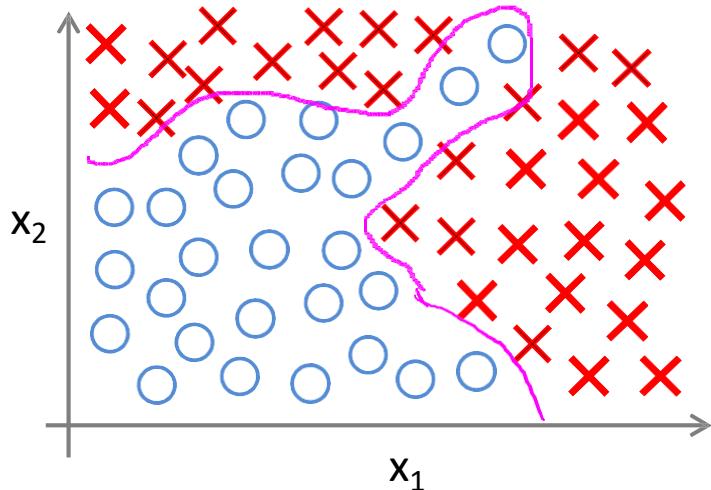


Machine Learning

# Neural Networks: Representation Non-linear hypotheses

# Non-linear Classification



$x_1$  = size

$x_2$  = # bedrooms

$x_3$  = # floors

$x_4$  = age

...

$x_{100}$

$\{ h=100 \}$

$$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 x_2 + \theta_5 x_1^3 x_2 + \theta_6 x_1 x_2^2 + \dots)$$

$$\rightarrow \underline{x_1^2}, \underline{x_1 x_2}, \underline{x_1 x_3}, \underline{x_1 x_4} \dots \underline{x_1 x_{100}} \\ \underline{x_2^2}, \underline{x_2 x_3} \dots$$

$\approx 5000$  feature

$\mathcal{O}(n^2)$

$$\rightarrow \underline{x_1^2}, \underline{x_2^2}, \underline{x_3^2}, \dots, \underline{x_{100}^2}$$

$\frac{n^2}{2}$

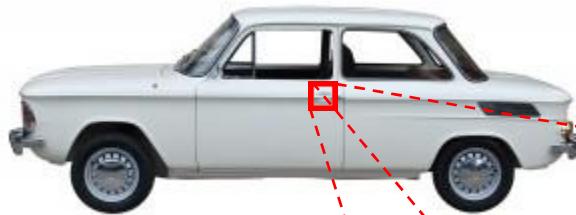
$$\rightarrow \underline{x_1 x_2 x_3}, \underline{x_1^2 x_2}, \underline{x_{10} x_{11} x_{12}}, \dots$$

$\mathcal{O}(n^3)$

170,000

# What is this?

You see this:



But the camera sees this:

194	210	201	212	199	213	215	195	178	158	182	209
180	189	190	221	209	205	191	167	147	115	129	163
114	126	140	188	176	165	152	140	170	106	78	88
87	103	115	154	143	142	149	153	173	101	57	57
102	112	106	131	122	138	152	147	128	84	58	66
94	95	79	104	105	124	129	113	107	87	69	67
68	71	69	98	89	92	98	95	89	88	76	67
41	56	68	99	63	45	60	82	58	76	75	65
20	43	69	75	56	41	51	73	55	70	63	44
50	50	57	69	75	75	73	74	53	68	59	37
72	59	53	66	84	92	84	74	57	72	63	42
67	61	58	65	75	78	76	73	59	75	69	50

# Computer Vision: Car detection



Cars

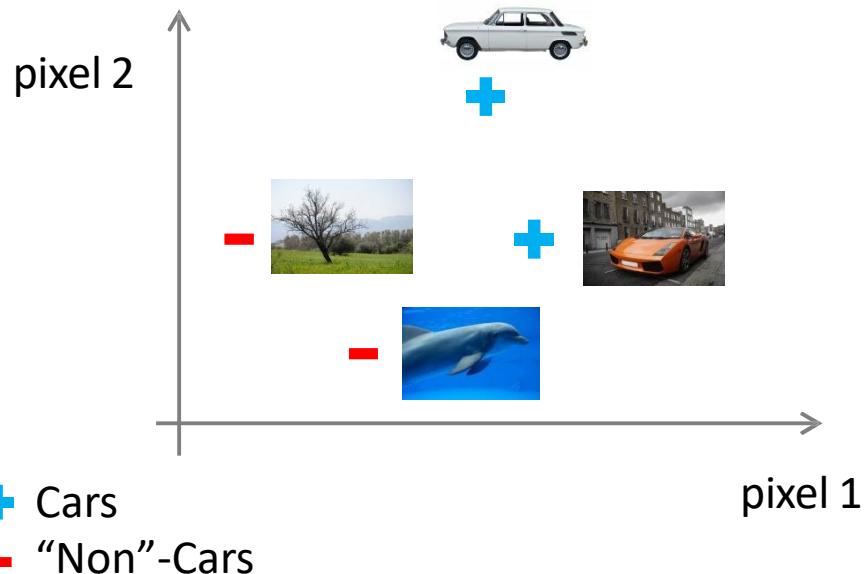
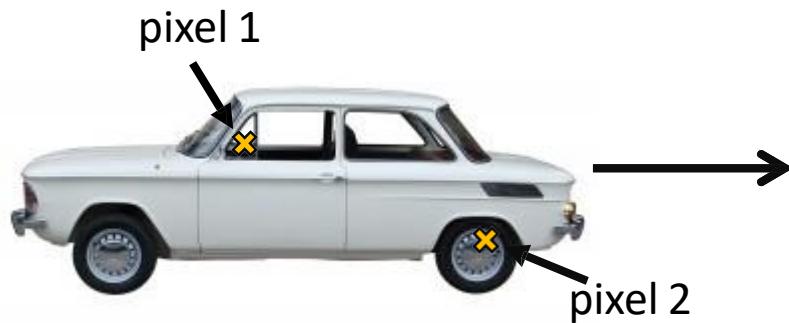


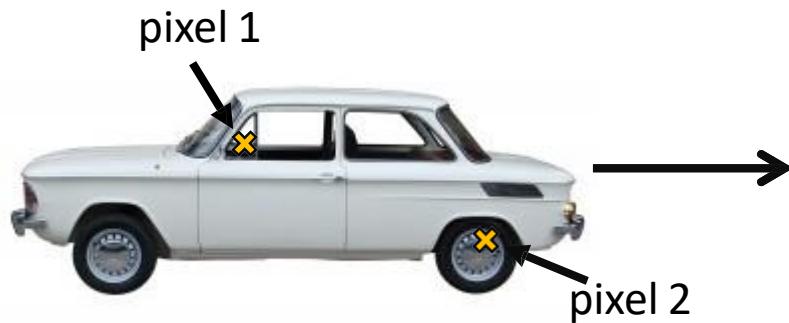
Not a car

Testing:

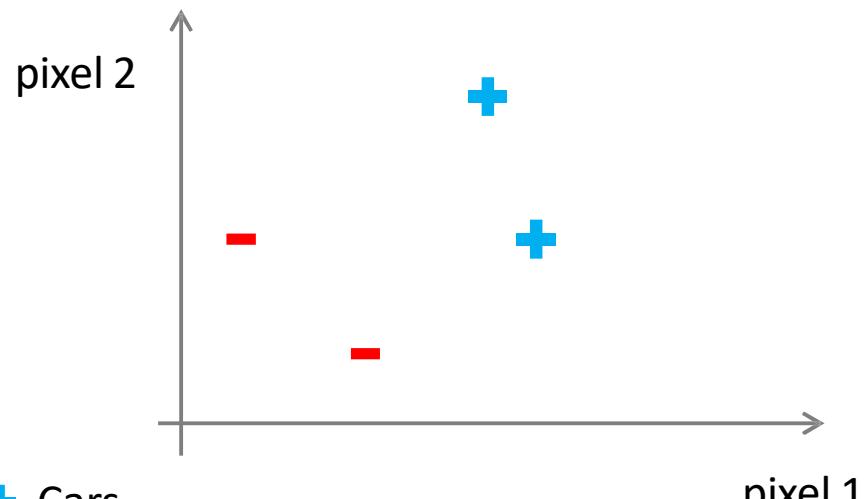


What is this?





Learning  
Algorithm

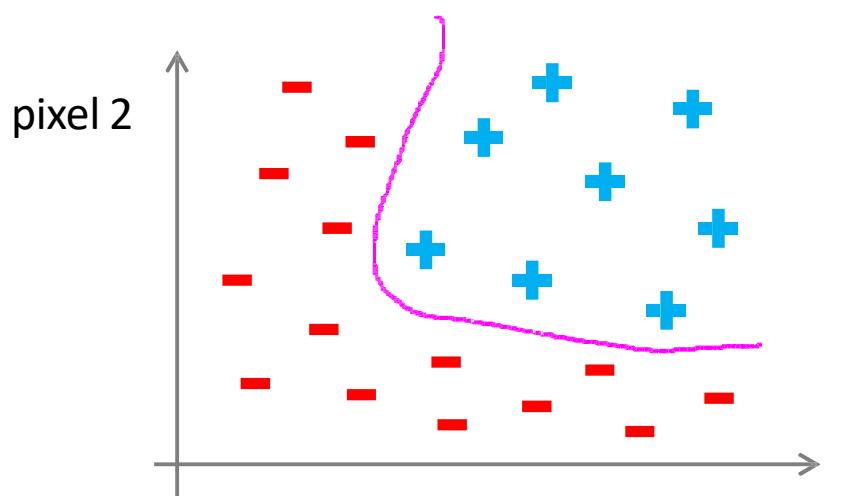
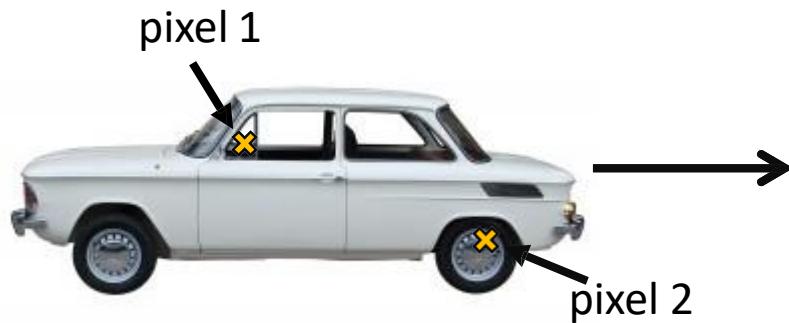


+

Cars

-

"Non"-Cars



+ Cars  
- "Non"-Cars

$50 \times 50$  pixel images  $\rightarrow$  2500 pixels  
 $n = 2500$     (7500 if RGB)

$$x = \begin{bmatrix} \text{pixel 1 intensity} \\ \text{pixel 2 intensity} \\ \vdots \\ \text{pixel 2500 intensity} \end{bmatrix}$$

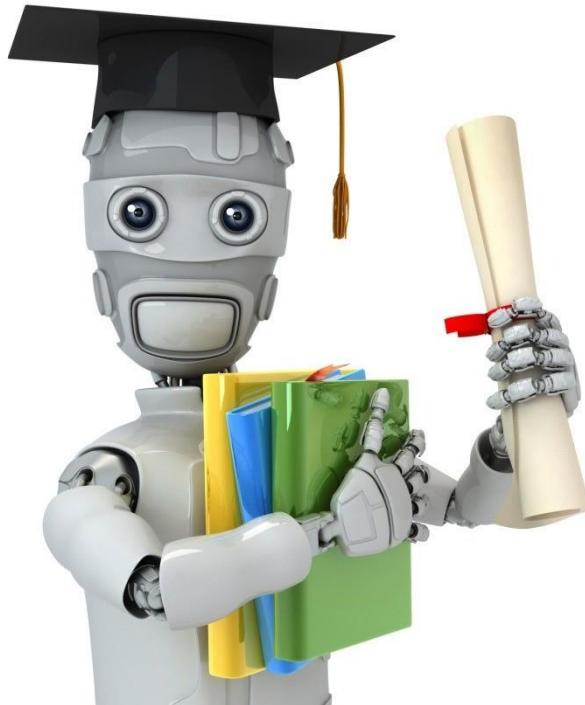
← 0-255

Quadratic features ( $x_i \times x_j$ ):  $\approx 3$  million features

# Quiz

Suppose you are learning to recognize cars from  $100 \times 100$  pixel images (grayscale, not RGB). Let the features be pixel intensity values. If you train logistic regression including all the quadratic terms ( $x_i x_j$ ) as features, about how many features will you have?

- 5,000
- 100,000
- 50 million ( $5 \times 10^7$ )
- 5 billion ( $5 \times 10^9$ )



Machine Learning

# Neural Networks: Representation

## Neurons and the brain

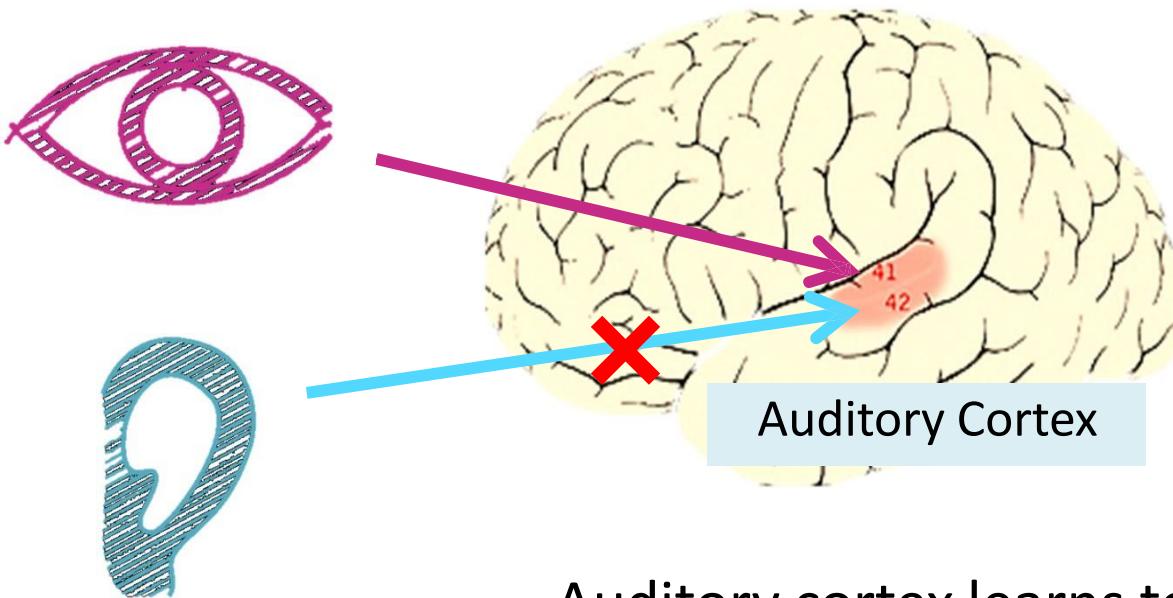
# Neural Networks

Origins: Algorithms that try to mimic the brain.

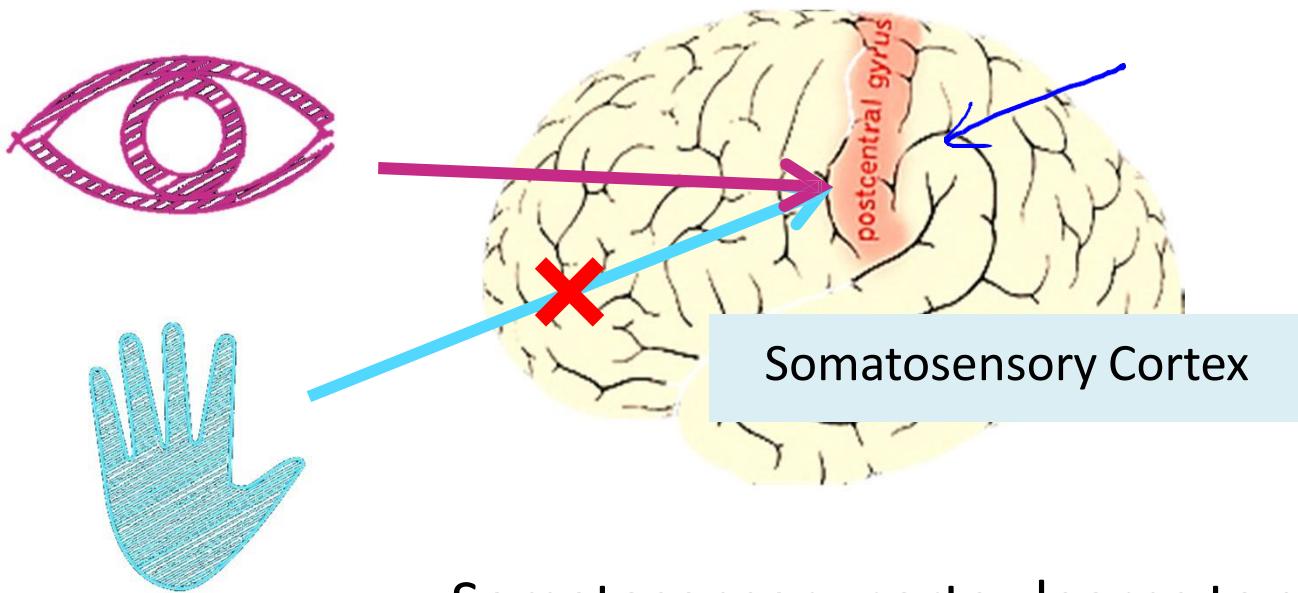
Was very widely used in 80s and early 90s; popularity diminished in late 90s.

Recent resurgence: State-of-the-art technique for many applications

# The “one learning algorithm” hypothesis



# The “one learning algorithm” hypothesis



Somatosensory cortex learns to see

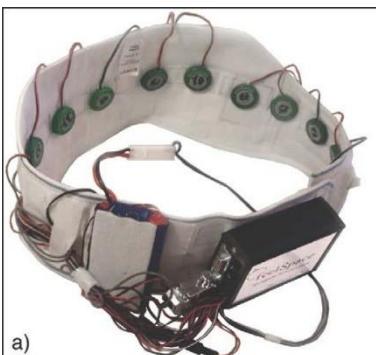
# Sensor representations in the brain



Seeing with your tongue



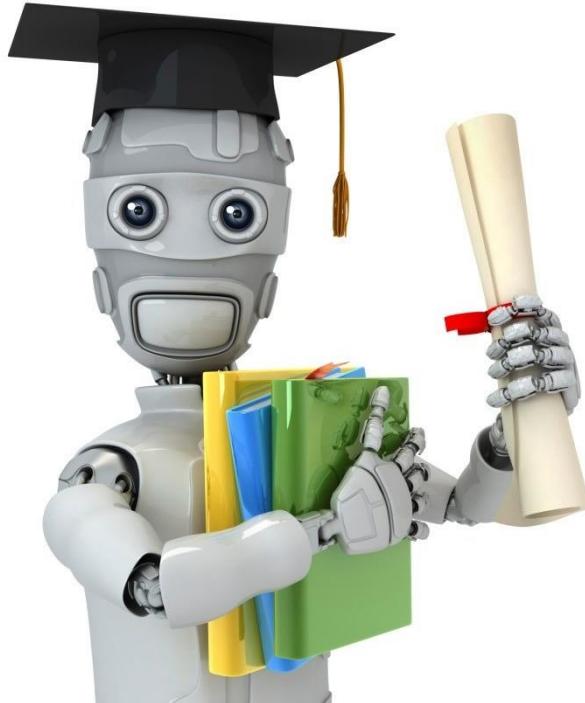
Human echolocation (sonar)



Haptic belt: Direction sense



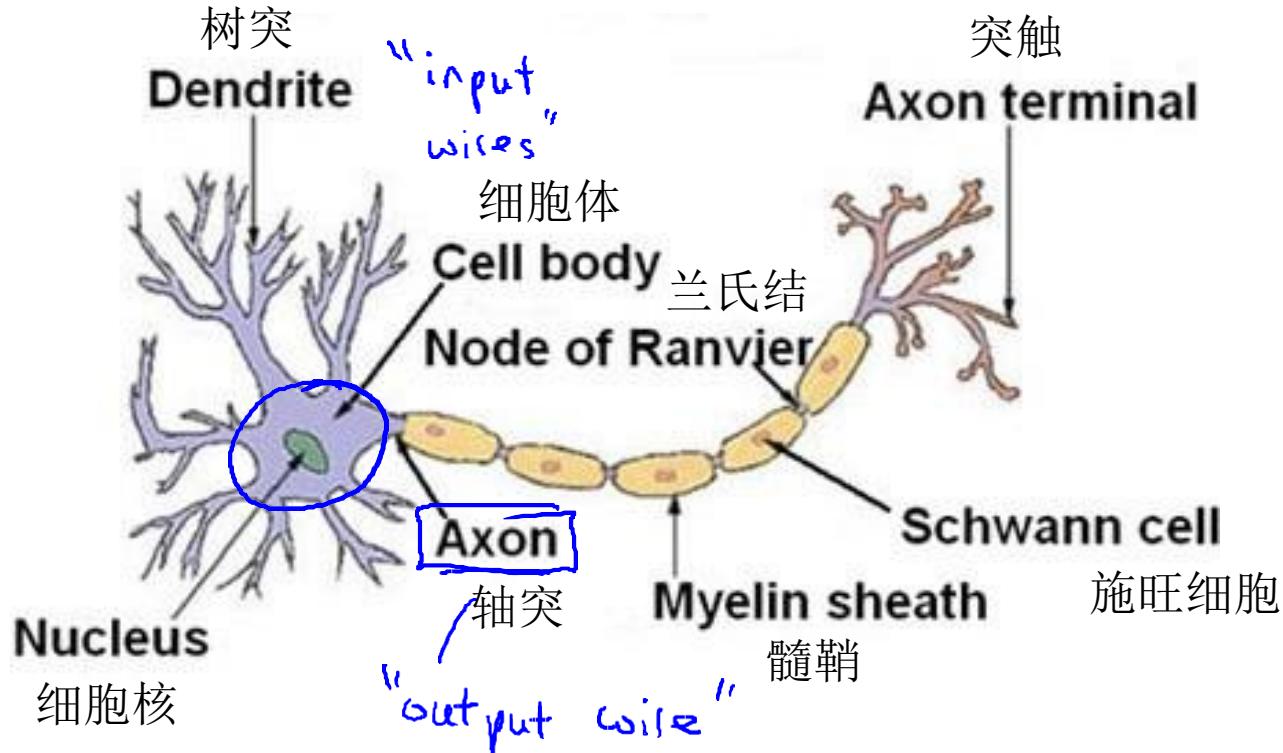
Implanting a 3<sup>rd</sup> eye



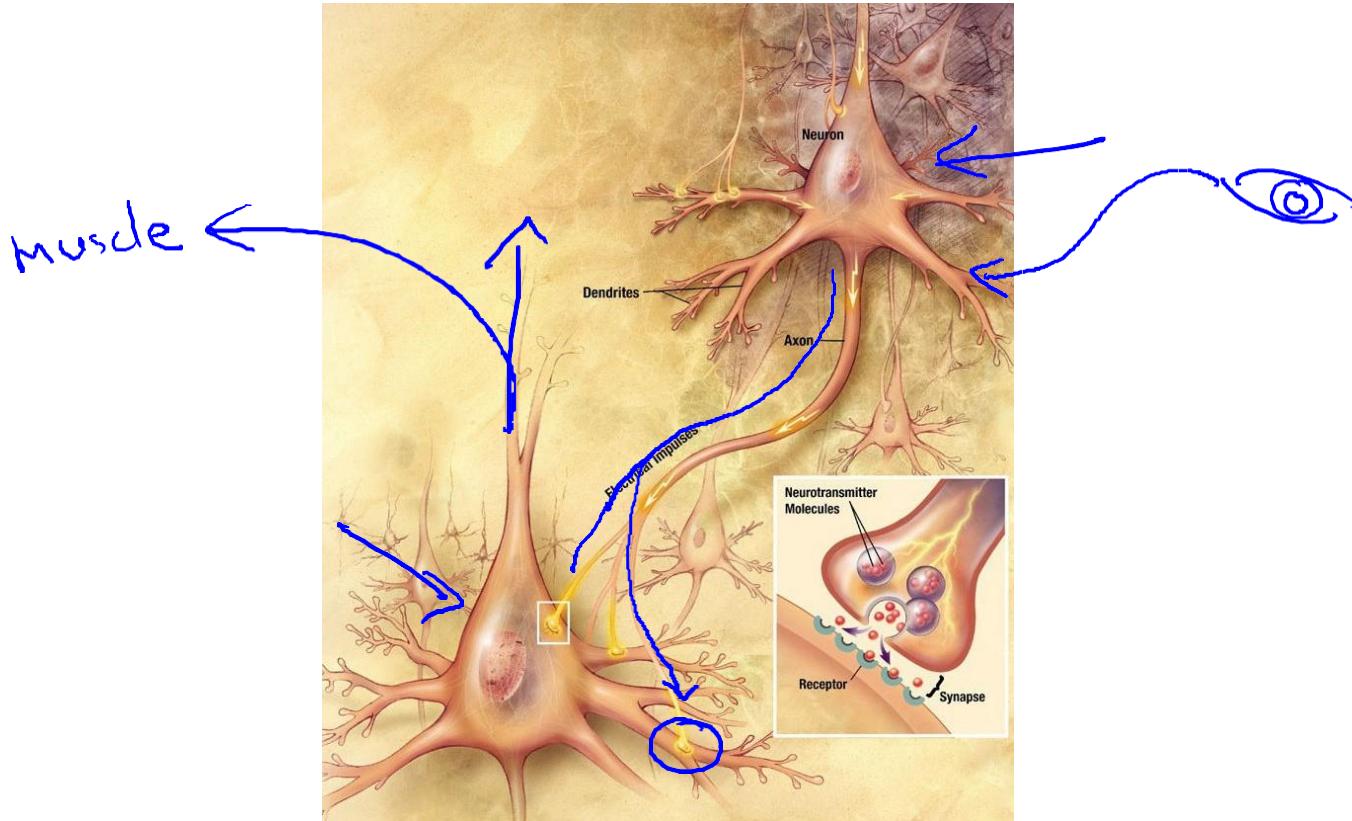
Machine Learning

# Neural Networks: Representation Model representation I

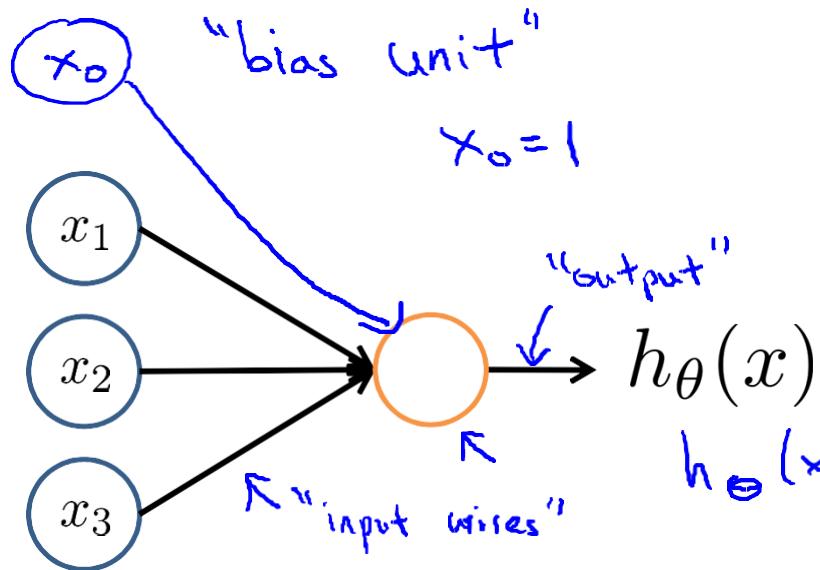
# Neuron in the brain



# Neurons in the brain



## Neuron model: Logistic unit



$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}$$

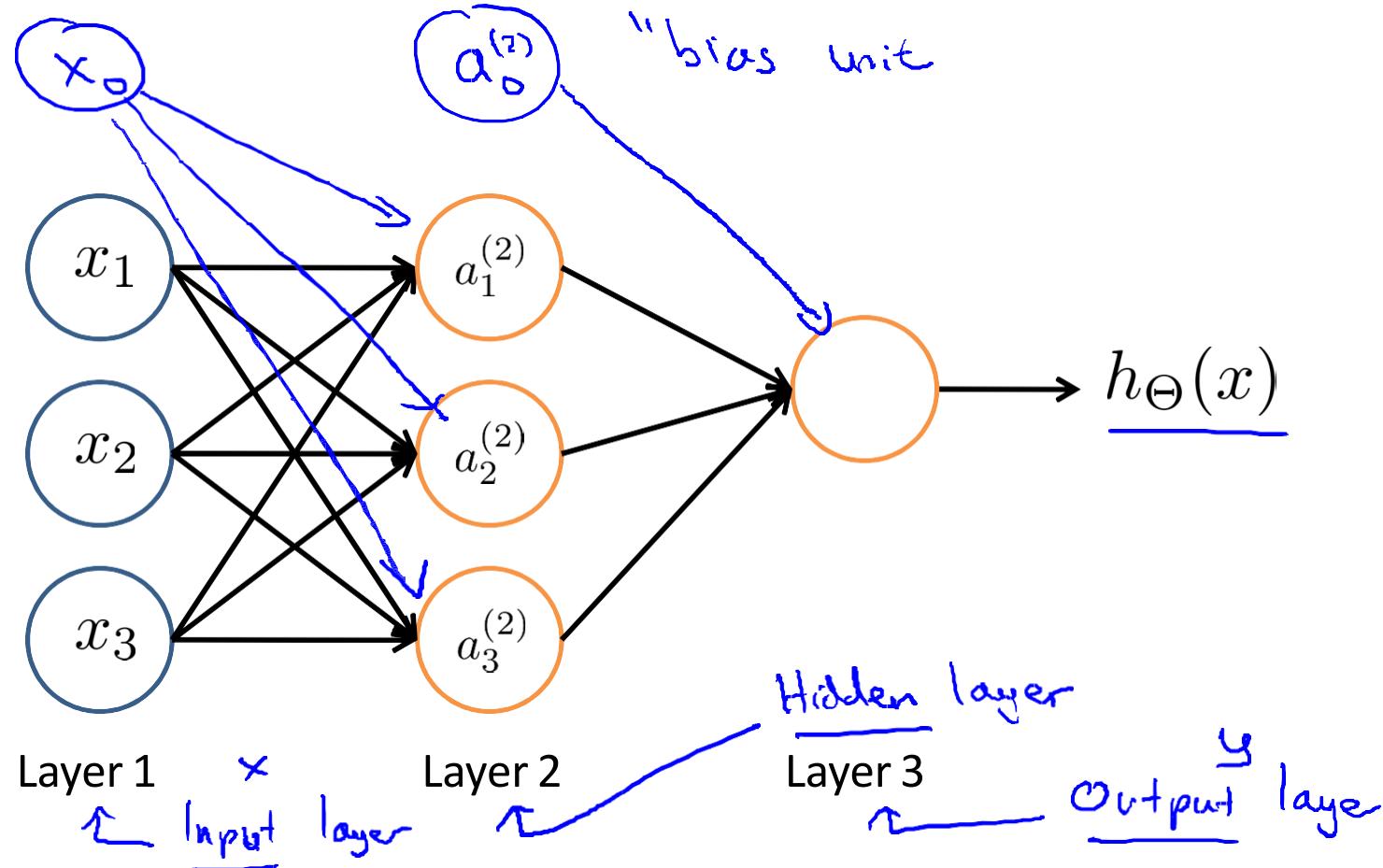
$h_{\theta}(x) = \frac{1}{1+e^{-\theta^T x}}$

↑  
"weights" ←  
(parameters ←)

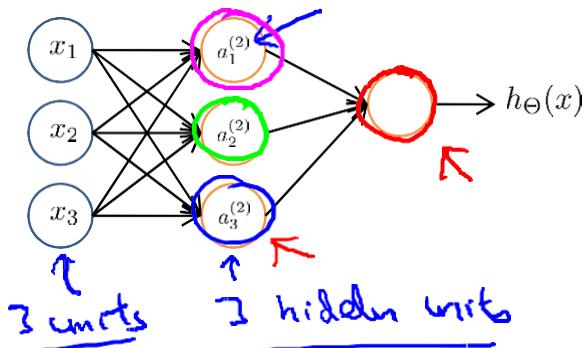
Sigmoid (logistic) activation function.

$$\underline{g(z)} = \frac{1}{1+e^{-z}}$$

# Neural Network



# Neural Network



→  $a_i^{(j)}$  = “activation” of unit  $i$  in layer  $j$

→  $\Theta^{(j)}$  = matrix of weights controlling function mapping from layer  $j$  to layer  $j + 1$

$$\Theta^{(1)} \in \mathbb{R}^{3 \times 4}$$

$$h_{\Theta}(x)$$

$$\rightarrow a_1^{(2)} = g(\underline{\Theta_{10}^{(1)} x_0} + \underline{\Theta_{11}^{(1)} x_1} + \underline{\Theta_{12}^{(1)} x_2} + \underline{\Theta_{13}^{(1)} x_3})$$

$$\rightarrow a_2^{(2)} = g(\underline{\Theta_{20}^{(1)} x_0} + \underline{\Theta_{21}^{(1)} x_1} + \underline{\Theta_{22}^{(1)} x_2} + \underline{\Theta_{23}^{(1)} x_3})$$

$$\rightarrow a_3^{(2)} = g(\underline{\Theta_{30}^{(1)} x_0} + \underline{\Theta_{31}^{(1)} x_1} + \underline{\Theta_{32}^{(1)} x_2} + \underline{\Theta_{33}^{(1)} x_3})$$

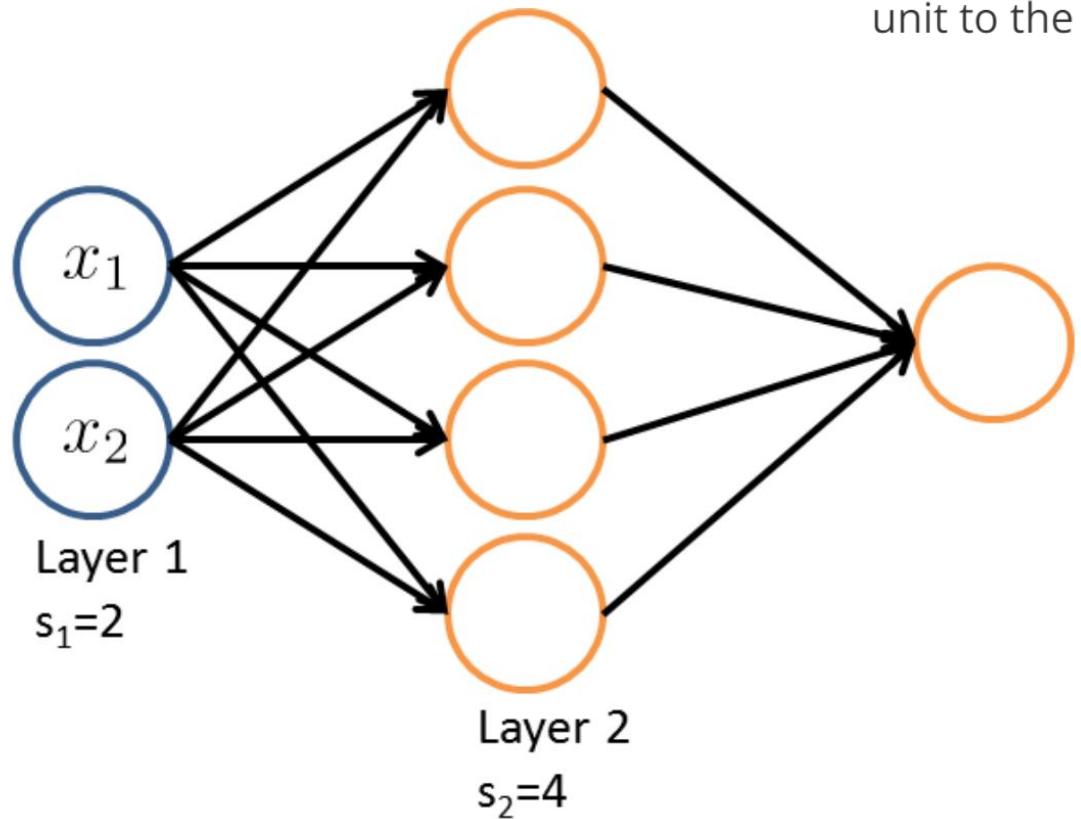
$$\rightarrow h_{\Theta}(x) = \underline{a_1^{(3)}} = g(\underline{\Theta_{10}^{(2)} a_0^{(2)}} + \underline{\Theta_{11}^{(2)} a_1^{(2)}} + \underline{\Theta_{12}^{(2)} a_2^{(2)}} + \underline{\Theta_{13}^{(2)} a_3^{(2)}})$$

If network has  $s_j$  units in layer  $j$ ,  $s_{j+1}$  units in layer  $j + 1$ , then  $\underline{\Theta^{(j)}}$  will be of dimension  $\underline{s_{j+1}} \times (\underline{s_j} + 1)$ .

$$s_{j+1} \times (s_j + 1)$$

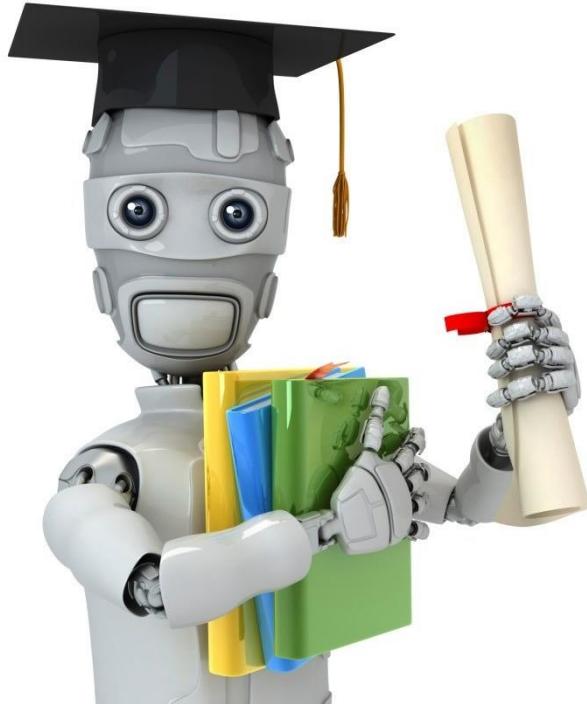
# Quiz

Consider the following neural network:



What is the dimension of  $\Theta^{(1)}$  (Hint: add a bias unit to the input and hidden layers)?

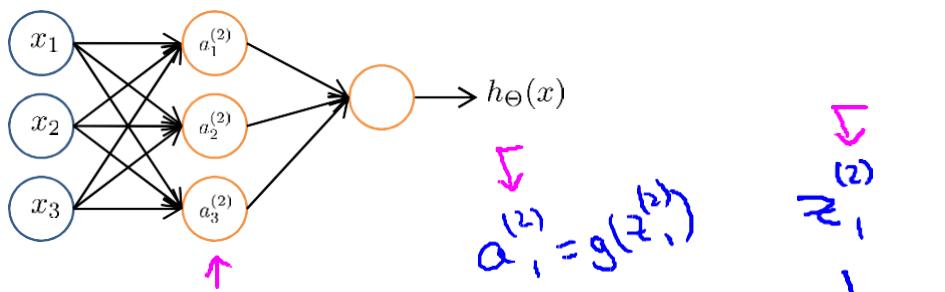
- $2 \times 4$
- $4 \times 2$
- $3 \times 4$
- $4 \times 3$



Machine Learning

# Neural Networks: Representation Model representation II

# Forward propagation: Vectorized implementation



$$a_1^{(2)} = g(\Theta_{10}^{(1)} x_0 + \Theta_{11}^{(1)} x_1 + \Theta_{12}^{(1)} x_2 + \Theta_{13}^{(1)} x_3)$$

$$a_2^{(2)} = g(\Theta_{20}^{(1)} x_0 + \Theta_{21}^{(1)} x_1 + \Theta_{22}^{(1)} x_2 + \Theta_{23}^{(1)} x_3)$$

$$a_3^{(2)} = g(\Theta_{30}^{(1)} x_0 + \Theta_{31}^{(1)} x_1 + \Theta_{32}^{(1)} x_2 + \Theta_{33}^{(1)} x_3)$$

$$h_\Theta(x) = g(\Theta_{10}^{(2)} a_0^{(2)} + \Theta_{11}^{(2)} a_1^{(2)} + \Theta_{12}^{(2)} a_2^{(2)} + \Theta_{13}^{(2)} a_3^{(2)})$$

$$a_0^{(2)} = g(z_0^{(2)})$$

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad z^{(2)} = \underbrace{\begin{bmatrix} z_1^{(2)} \\ z_2^{(2)} \\ z_3^{(2)} \end{bmatrix}}_{\uparrow}$$

$$z^{(2)} = \Theta^{(1)} x$$

$$a^{(2)} = g(z^{(2)})$$

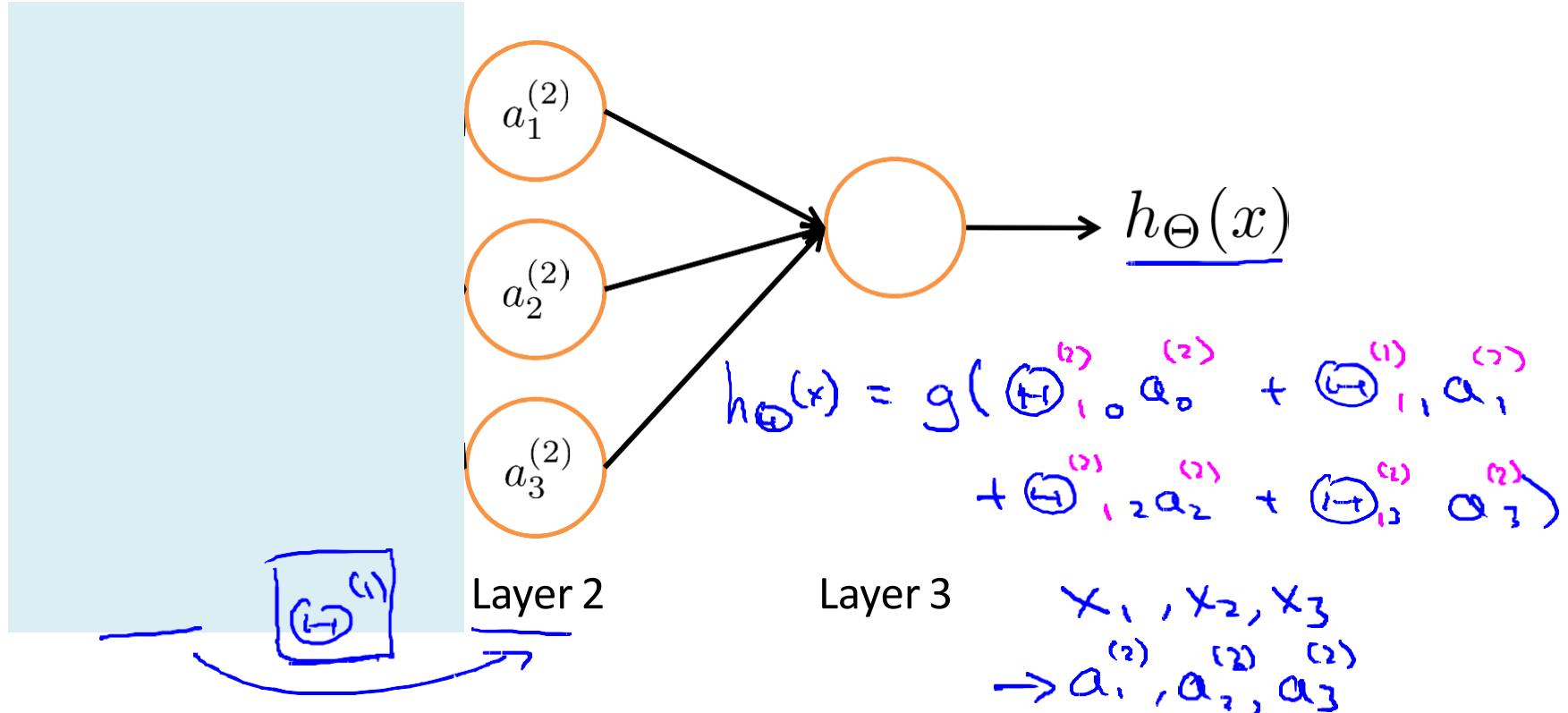
Add  $\underline{a_0^{(2)}} = 1.$   $\rightarrow a^{(2)} \in \mathbb{R}^4$

$$z^{(3)} = \Theta^{(2)} a^{(2)}$$

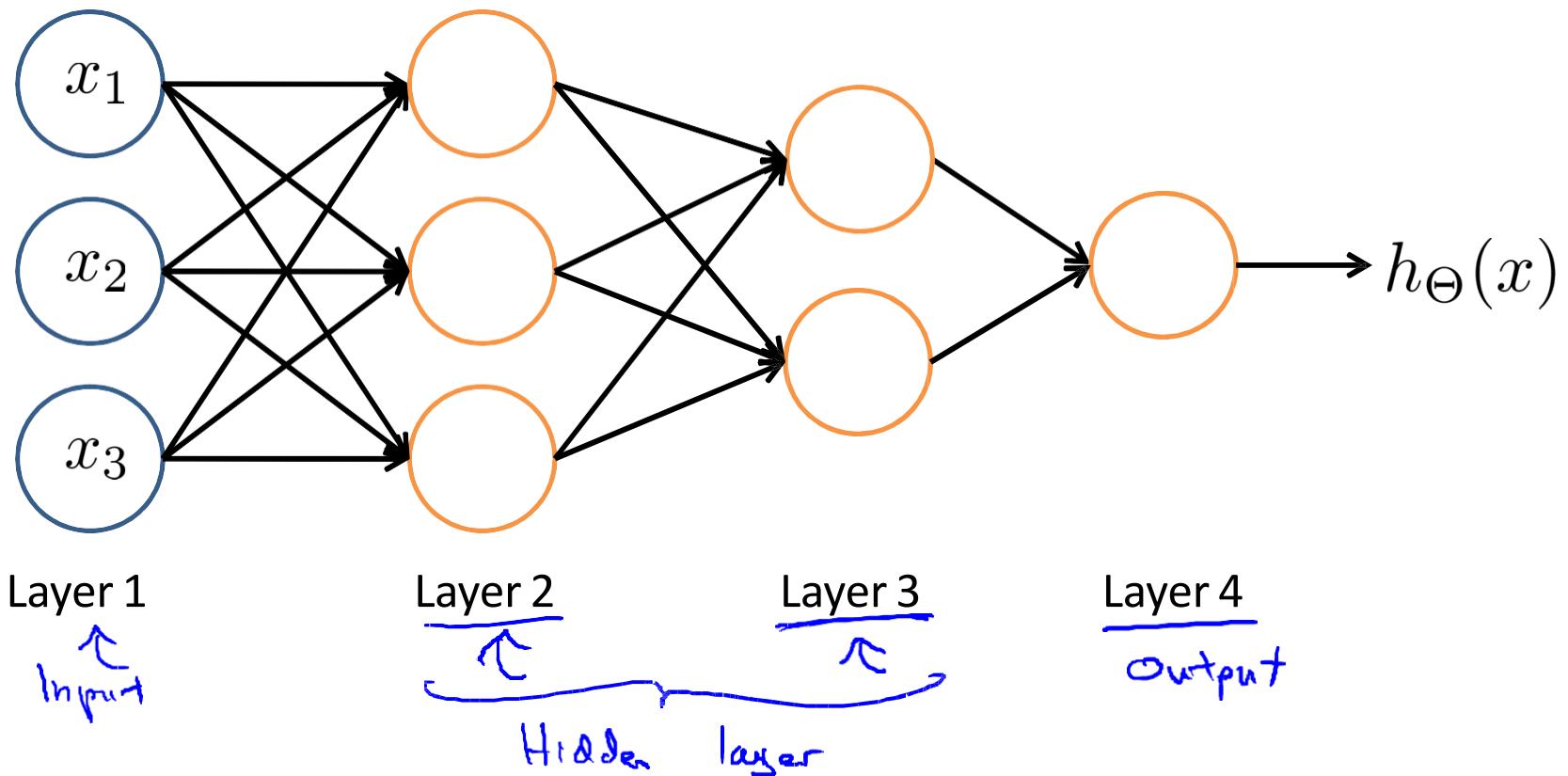
$$h_\Theta(x) = a^{(3)} = g(z^{(3)})$$

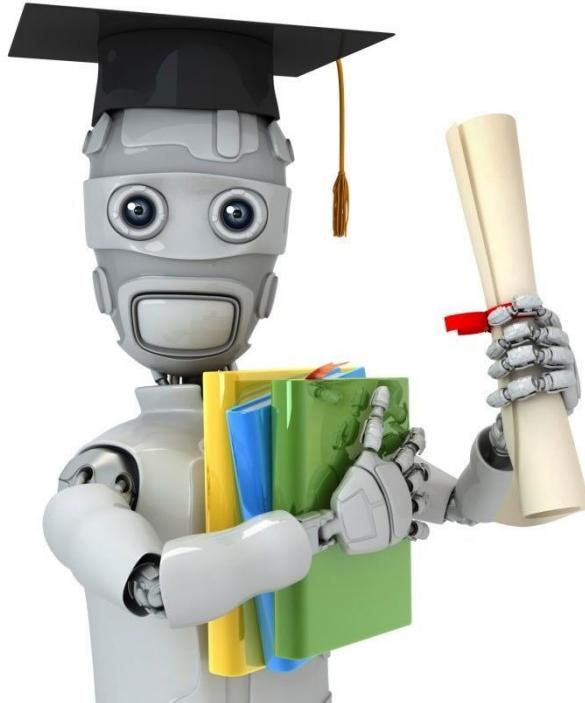
$$\begin{bmatrix} a_1^{(2)} \\ a_2^{(2)} \\ a_3^{(2)} \end{bmatrix}$$

# Neural Network learning its own features



## Other network architectures





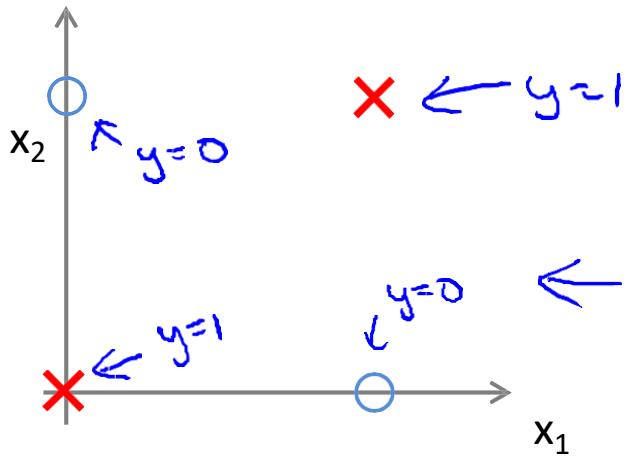
Machine Learning

# Neural Networks: Representation

## Examples and intuitions I

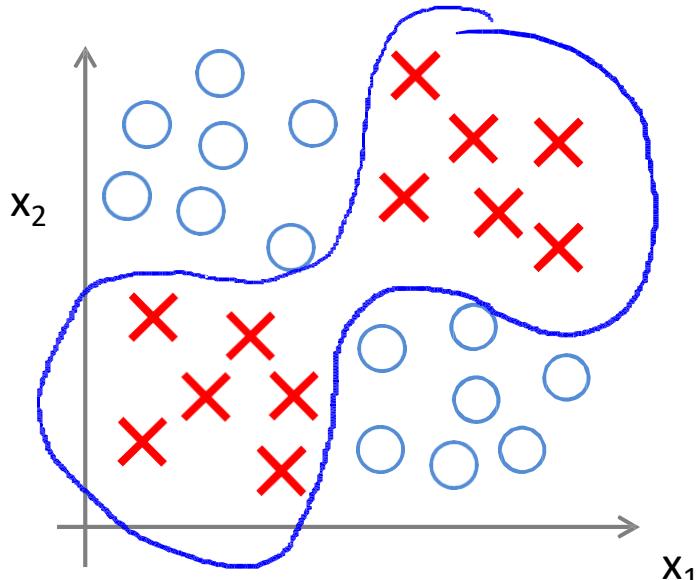
## Non-linear classification example: XOR/XNOR

$x_1, x_2$  are binary (0 or 1).



$$y = \underline{x_1 \text{ XOR } x_2}$$

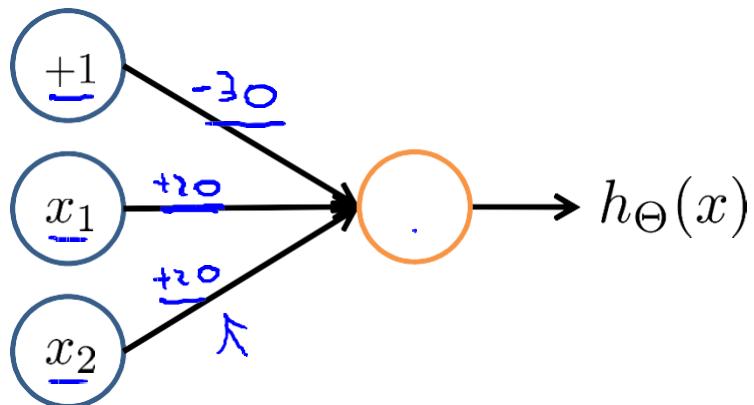
→  $\underline{x_1 \text{ XNOR } x_2}$  ←  
→ NOT (x<sub>1</sub> XOR x<sub>2</sub>)



## Simple example: AND

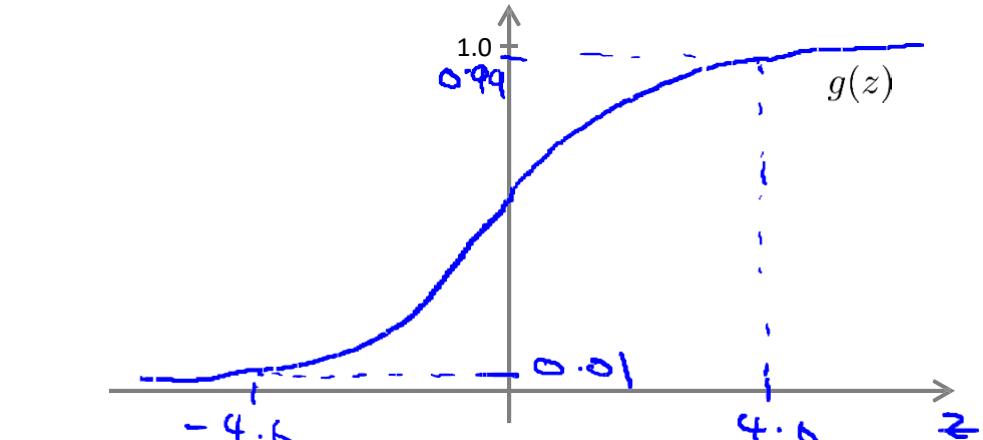
$$x_1, x_2 \in \{0, 1\}$$

$$y = x_1 \text{ AND } x_2$$



$$\rightarrow h_{\Theta}(x) = g\left(\frac{-30}{\pi} + \frac{20}{\pi}x_1 + \frac{20}{\pi}x_2\right)$$

$\Theta_{0,0}$        $\Theta_{0,1}$        $\Theta_{1,2}$

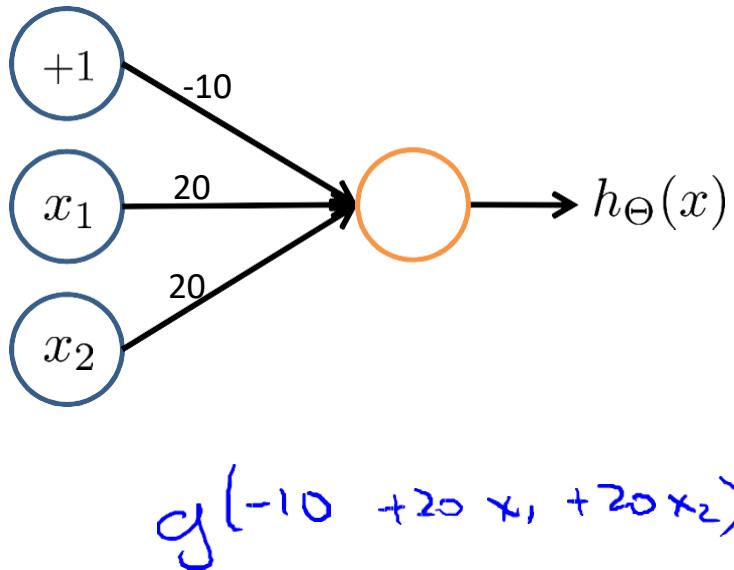


A table showing the output of the function  $h_{\Theta}(x)$  for all combinations of inputs  $x_1$  and  $x_2$ . The table is highlighted with a pink border. The output column is labeled  $h_{\Theta}(x)$  and approximated by  $g(z)$ .

$x_1$	$x_2$	$h_{\Theta}(x)$
0	0	$g(-30) \approx 0$
0	1	$g(-10) \approx 0$
1	0	$g(-10) \approx 0$
1	1	$g(10) \approx 1$

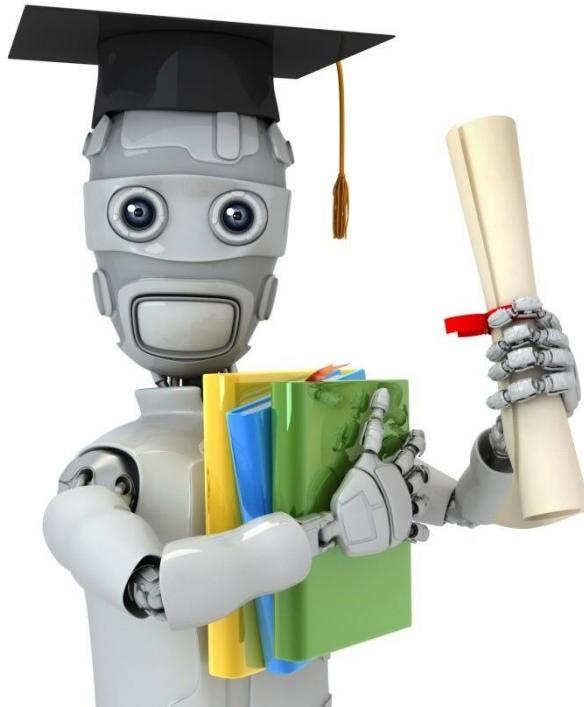
$h_{\Theta}(x) \approx x_1 \text{ AND } x_2$

## Example: OR function



$x_1$	$x_2$	$h_{\Theta}(x)$
0	0	$g(-10) \approx 0$
0	1	$g(10) \approx 1$
1	0	$\approx 1$
1	1	$\approx 1$

The table shows the output of the function  $h_{\Theta}(x)$  for all combinations of  $x_1$  and  $x_2$ . The values are approximated by handwritten blue text. A pink hand-drawn bracket highlights the boundary where the output is approximately 1, which corresponds to the region where either  $x_1$  or  $x_2$  is 1.



Machine Learning

# Neural Networks: Representation

## Examples and intuitions II

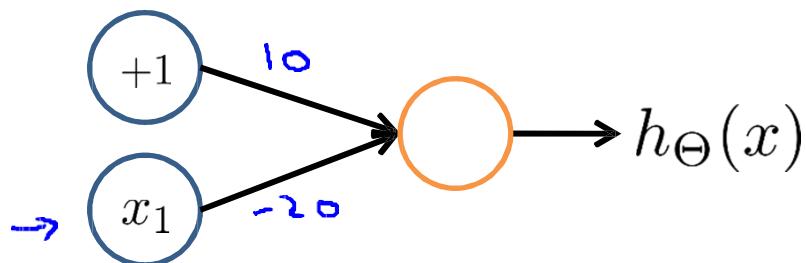
$x_1$  AND  $x_2$

$x_1$  OR  $x_2$

$\{0, 1\}$ .

Negation:

NOT  $x_1$

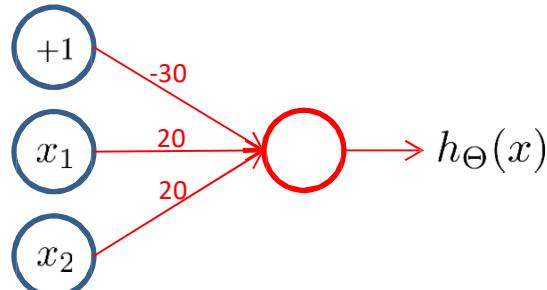


$x_1$	$h_{\Theta}(x)$
0	$g(10) \approx 1$
1	$g(-20) \approx 0$

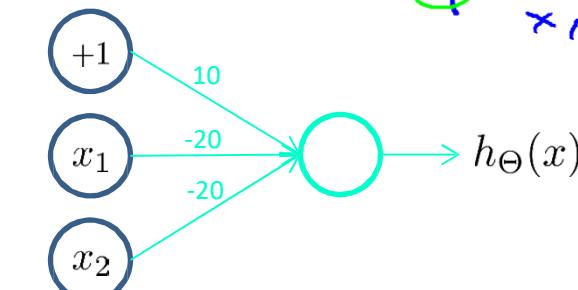
$$h_{\Theta}(x) = g(\underline{10 - 20x_1})$$

(NOT  $x_1$ ) AND (NOT  $x_2$ )  
 $\Leftrightarrow 1$  if and only if  
 $\rightarrow x_1 = x_2 = 0$

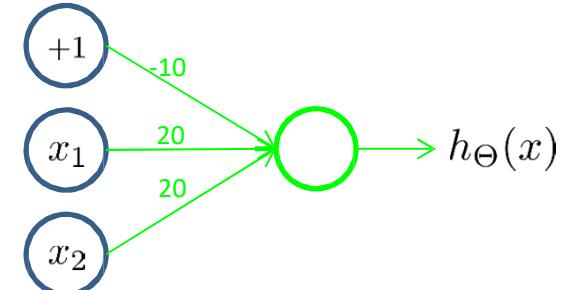
Putting it together:  $x_1$  XNOR  $x_2$



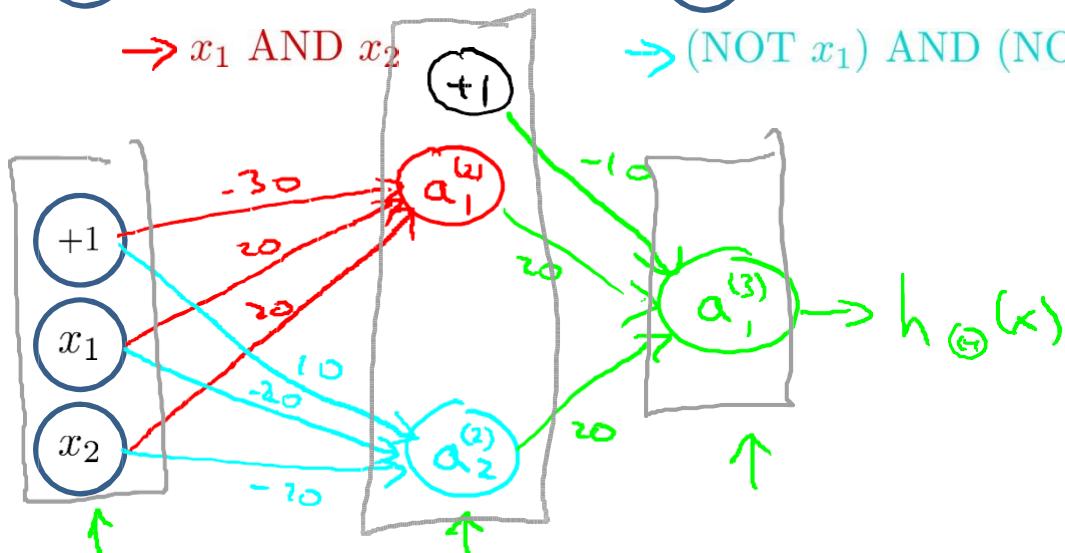
$\rightarrow x_1 \text{ AND } x_2$



$\rightarrow (\text{NOT } x_1) \text{ AND } (\text{NOT } x_2)$

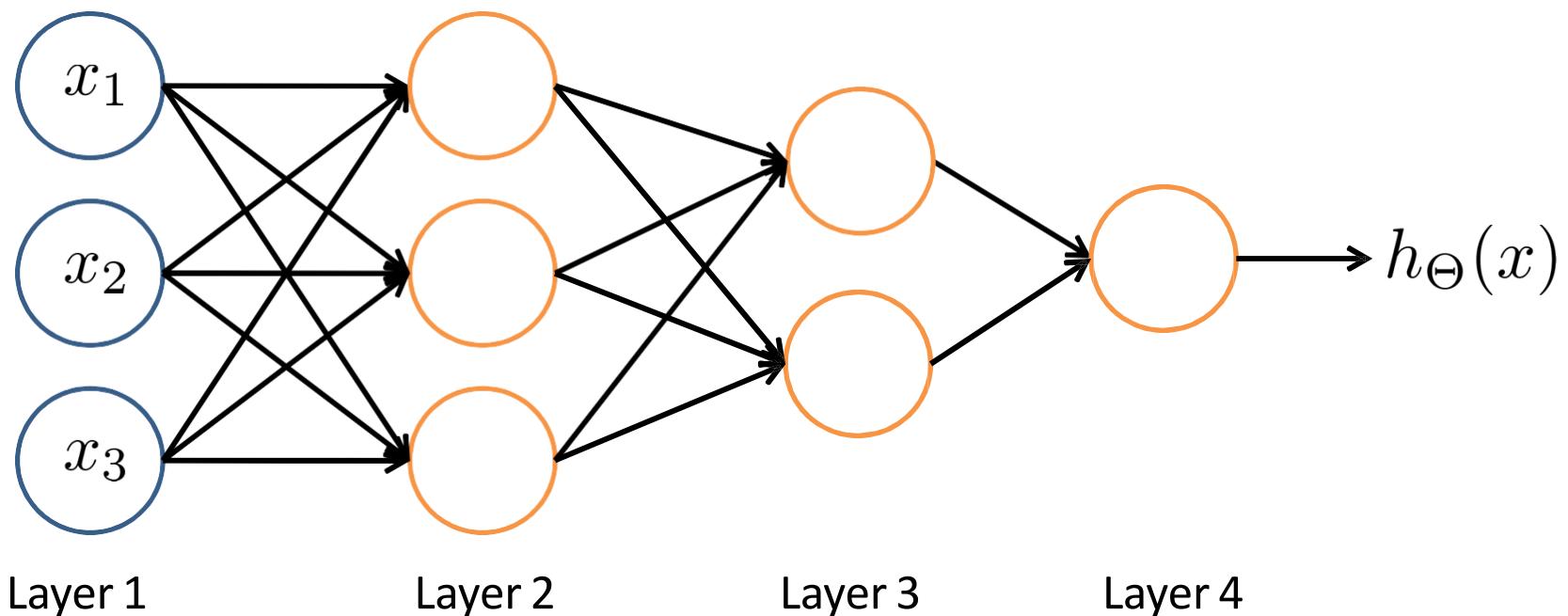


$\rightarrow x_1 \text{ OR } x_2$

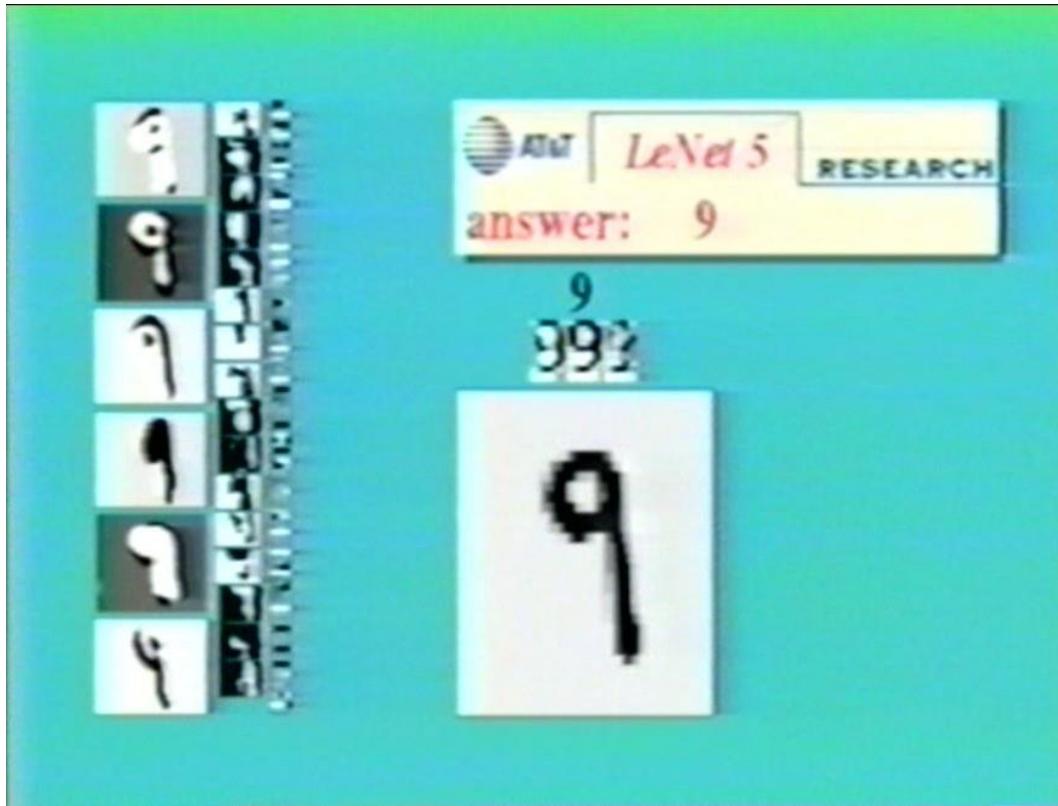


$x_1$	$x_2$	$a_1^{(2)}$	$a_2^{(2)}$	$h_{\Theta}(x)$
0	0	1	1	1 ←
0	1	1	0	0
1	0	0	1	0
1	1	0	0	1 ←

# Neural Network intuition



# Handwritten digit classification





Machine Learning

# Neural Networks: Representation

## Multi-class classification

# Multiple output units: One-vs-all.



Pedestrian



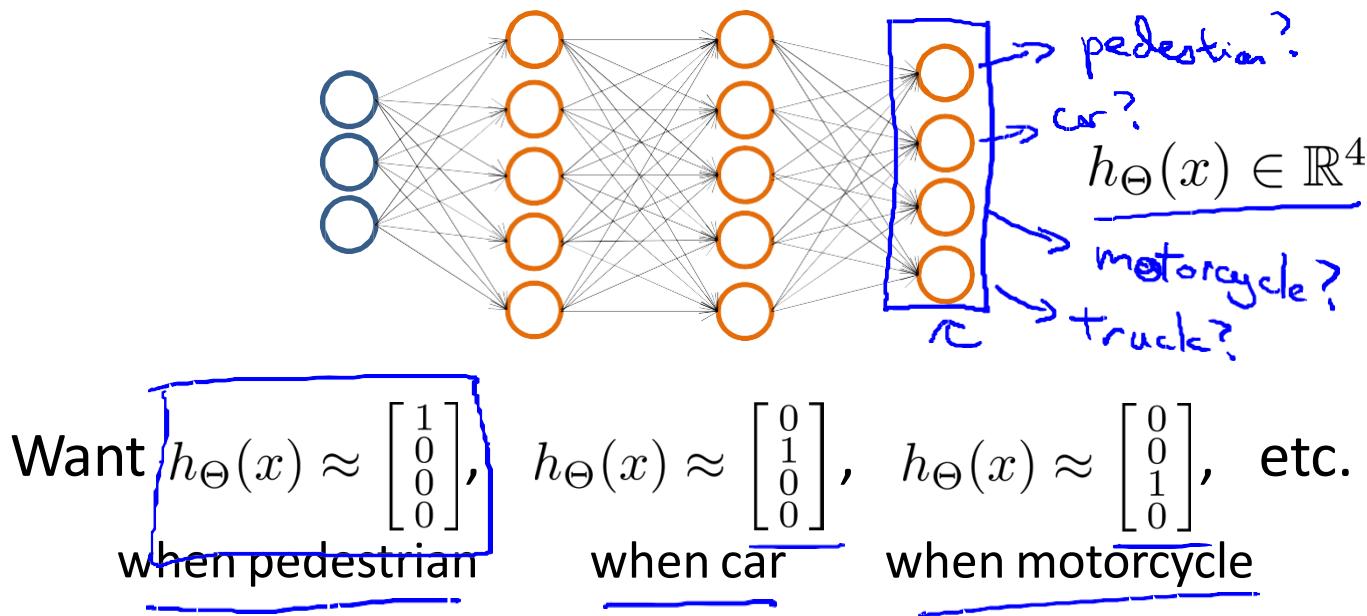
Car



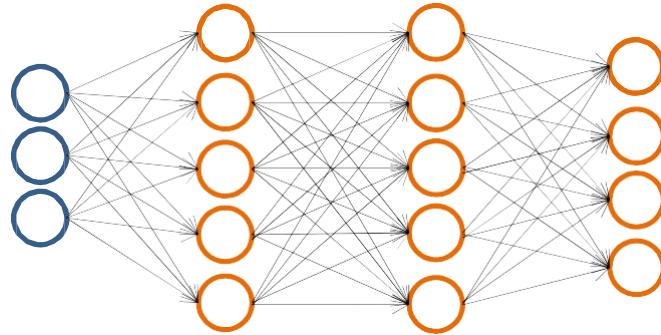
Motorcycle



Truck



## Multiple output units: One-vs-all.



$$h_{\Theta}(x) \in \mathbb{R}^4$$

Want  $h_{\Theta}(x) \approx \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ ,  $h_{\Theta}(x) \approx \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$ ,  $h_{\Theta}(x) \approx \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$ , etc.  
 when pedestrian      when car      when motorcycle

Training set:  $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \dots, (x^{(m)}, y^{(m)})$

$y^{(i)}$  one of  
 $\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ ,  $\begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$ ,  $\begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$ ,  $\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$   
 pedestrian      car      motorcycle      truck

~~Previously~~  
 $y \in \{1, 2, 3, 4\}$   
 $h_{\Theta}(x^{(i)}) \approx y^{(i)}$   
 $\in \mathbb{R}^4$