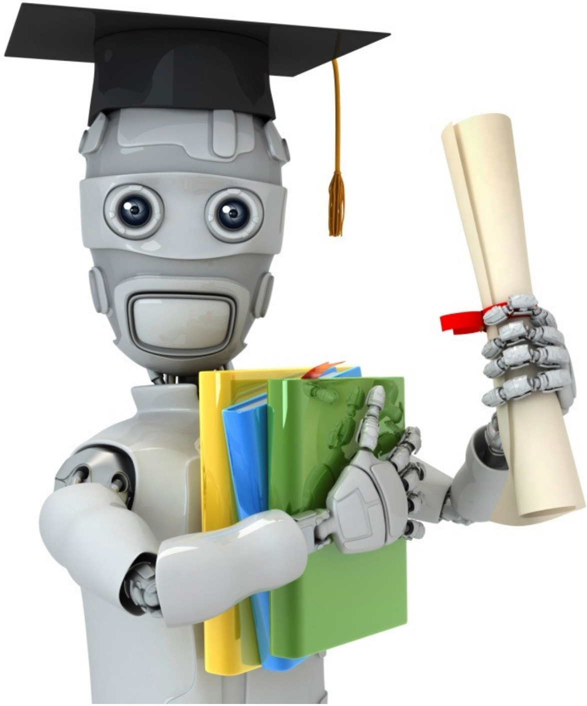
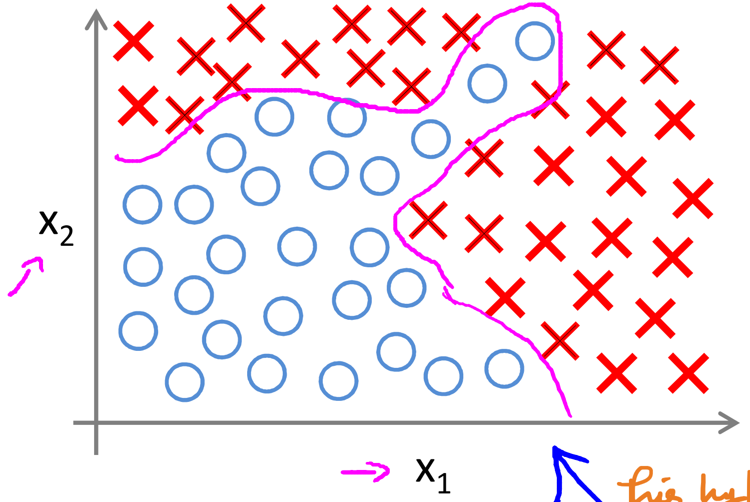


Machine Learning

Neural Networks: Representation

Non-linear hypotheses

Non-linear Classification



sigmoid function

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

this hypothesis separates +ve and -ve examples

$x_1 =$ size
$x_2 =$ # bedrooms
$x_3 =$ # floors
$x_4 =$ age
$\dots$
x_{100}

$n = 100$ features

$\rightarrow x_1^2, x_1x_2, x_1x_3, x_1x_4 \dots x_1x_{100}$
$x_2^2, x_2x_3 \dots$

≈ 5000 feature $\quad O(n^2) \quad \approx \frac{n^2}{2}$

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

$\rightarrow x_1x_2x_3, x_1^2x_2, x_{10}x_{11}x_{17}, \dots$

$O(n^3)$ $\quad$ 170,000

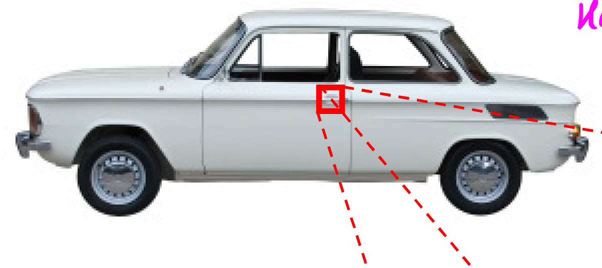
With these features we want to find whether our house will be sold / Not

Andrew Ng

What is this?

Normal people will this image as a car and various parts of it

You see this:



How Computer see this car?

Computer perspective

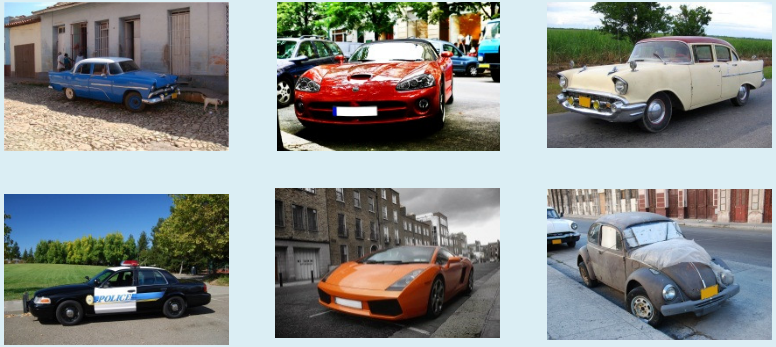
Computer sees the door handle

these are the matrix of pixel intensity values to represent the door handle.

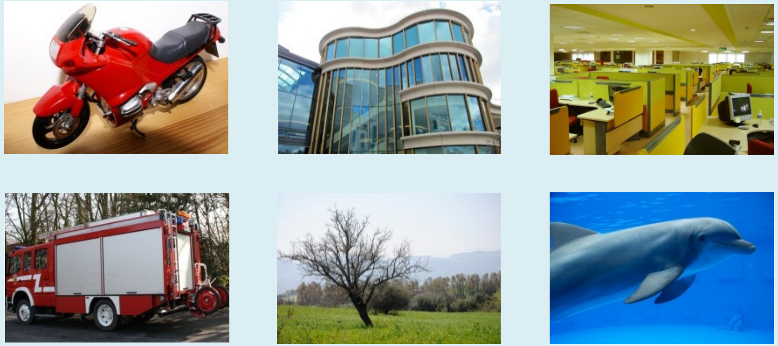
But the camera sees this:

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



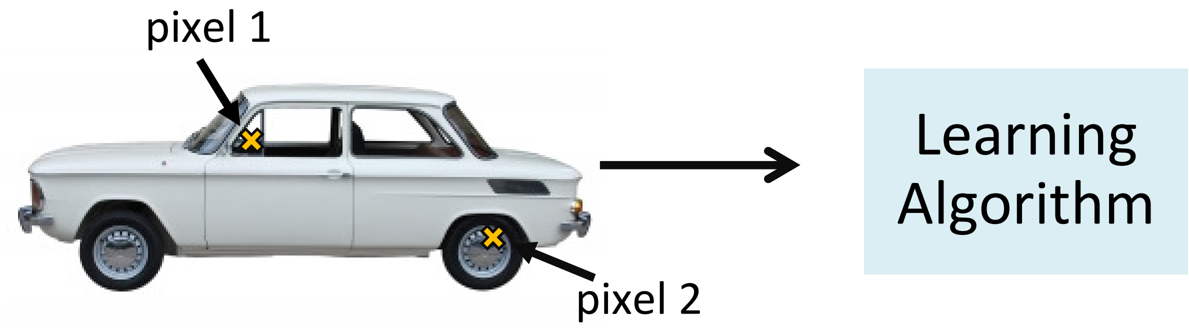
Not a car

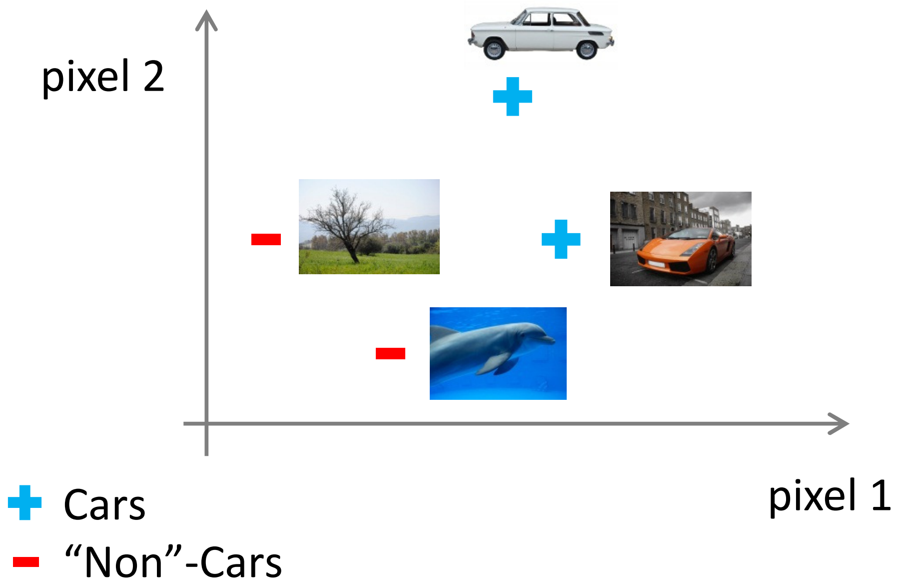
1) Data-set ⇒ labelled training set

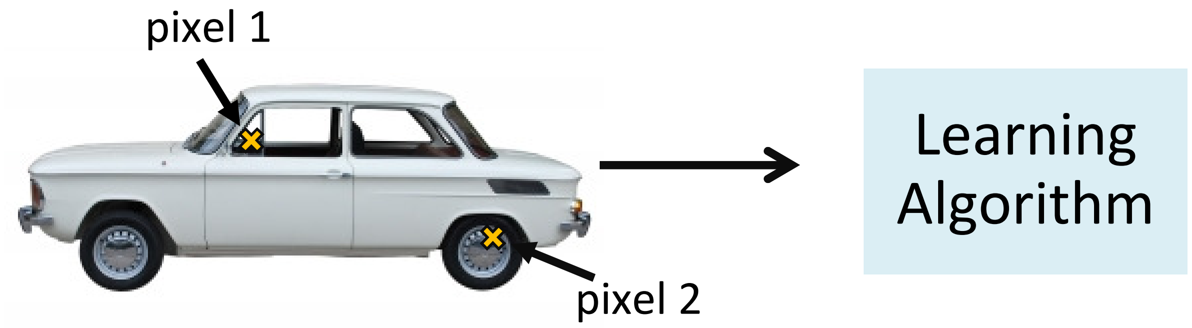
2) Fit the data into Model.

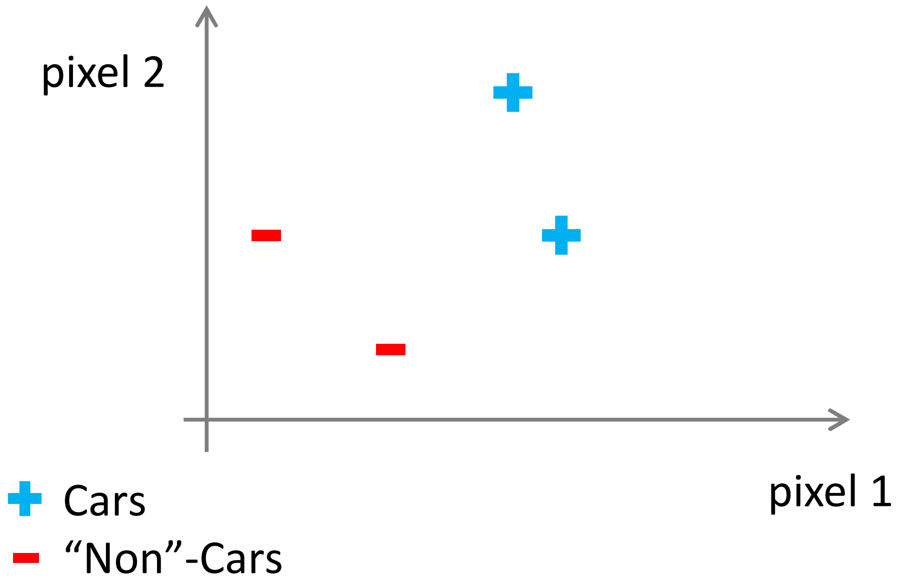
3) Testing:

4) What is this? The Model should predict whether this is a car | Not


pixel 1
pixel 2
Learning Algorithm

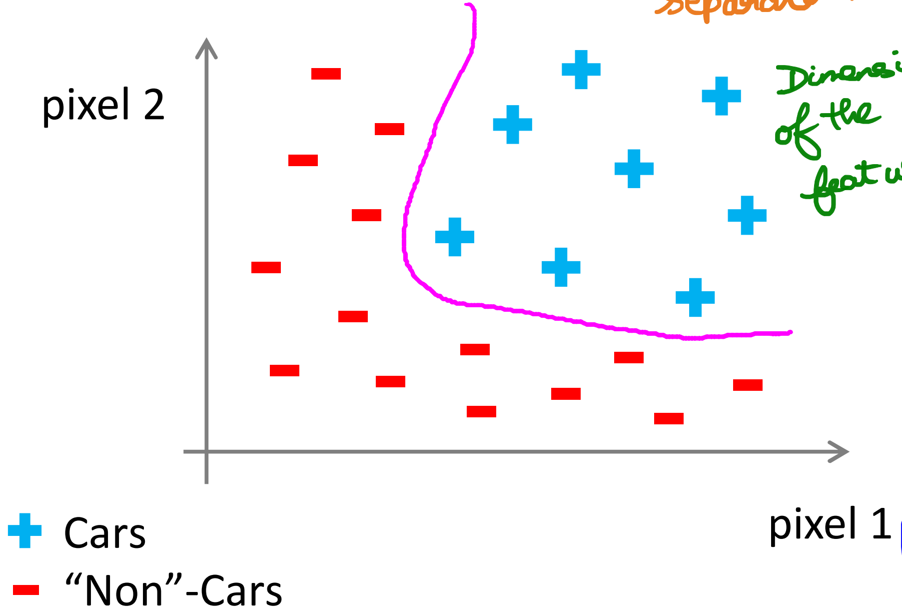

pixel 2
pixel 1
Cars
“Non”-Cars


pixel 1
pixel 2
Learning Algorithm


pixel 2
pixel 1
Cars
"Non"-Cars

→ **Learning Algorithm**

★ Non-Linear hypothesis to separate + and -



50 x 50 pixel images → 2500 pixels

Dimension of the feature $\boxed{n = 2500}$ gray scale (7500 if RGB) ← Dimension of the feature (RGB)

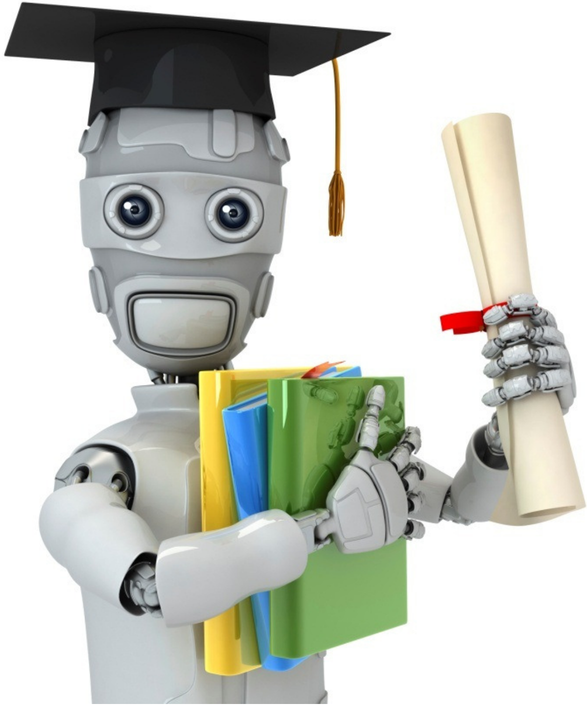
0-255

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

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

If we include all 2500 pixel intensity

Andrew Ng



Machine Learning

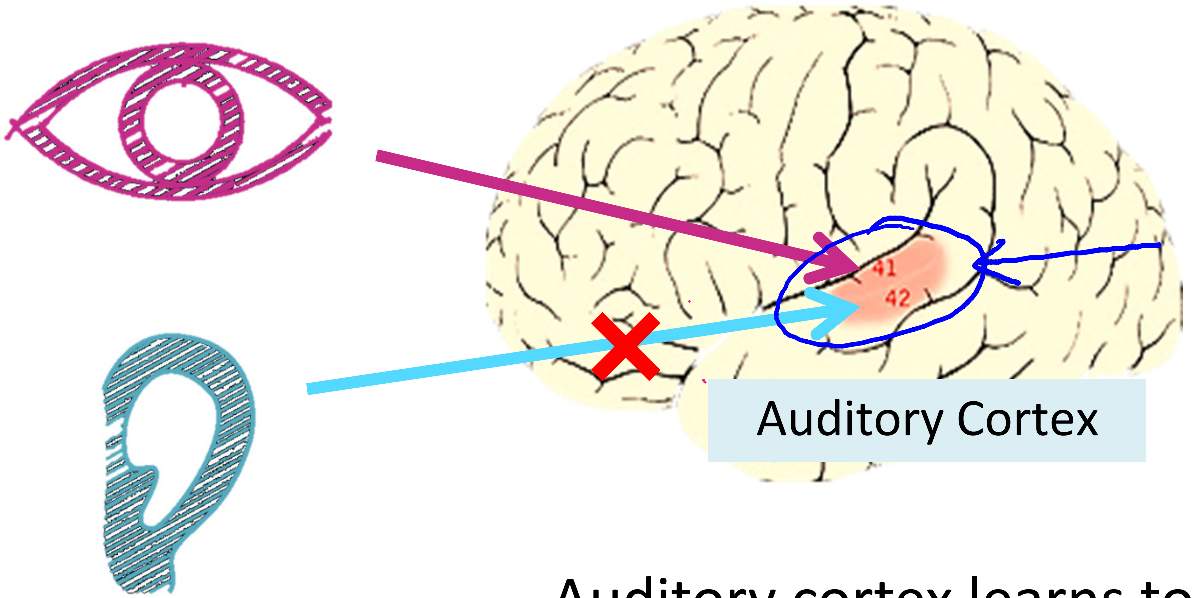
Neural Networks: Representation

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



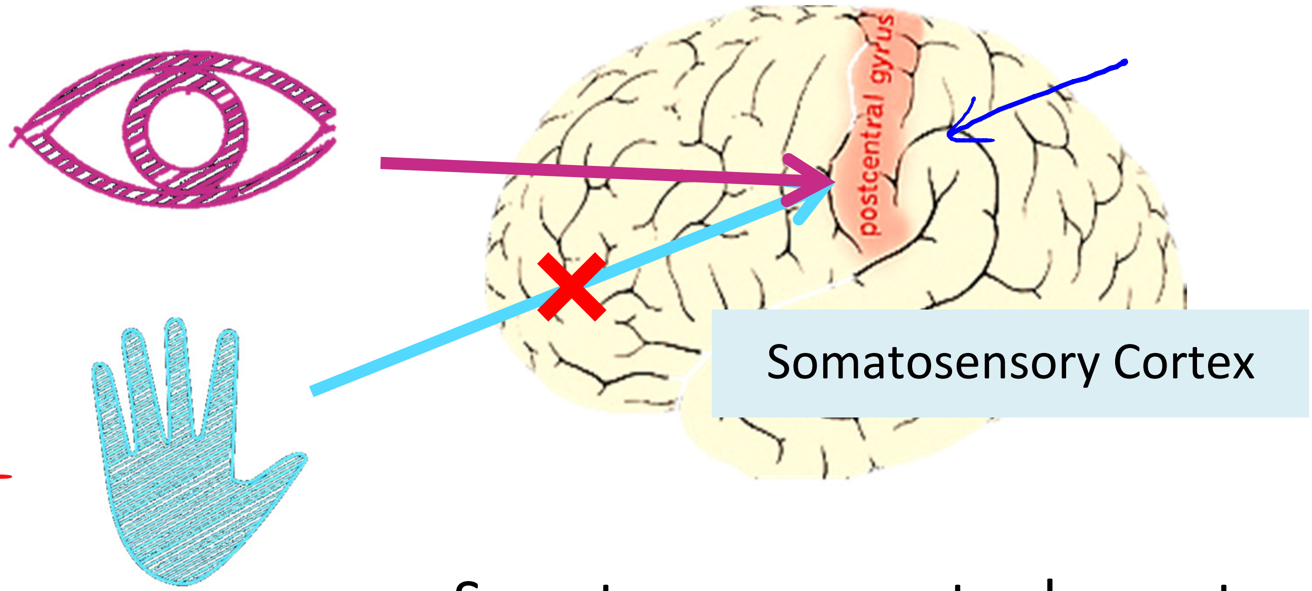
Auditory cortex learns to see

[Roe et al., 1992]

The “one learning algorithm” hypothesis

Now able to hear

Feeling the touch

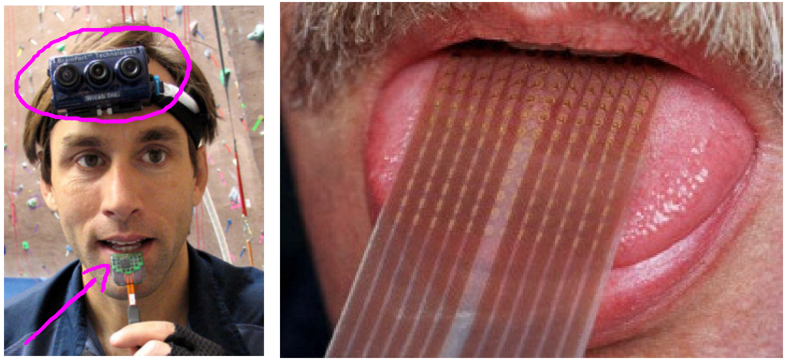


Somatosensory cortex learns to see

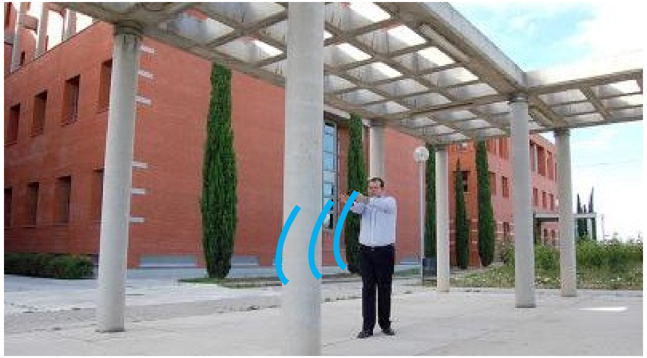
(for sensory organs)

[Metin & Frost, 1989]

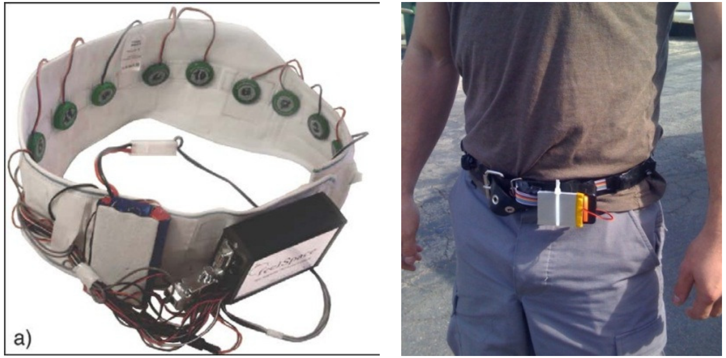
Sensor representations in the brain



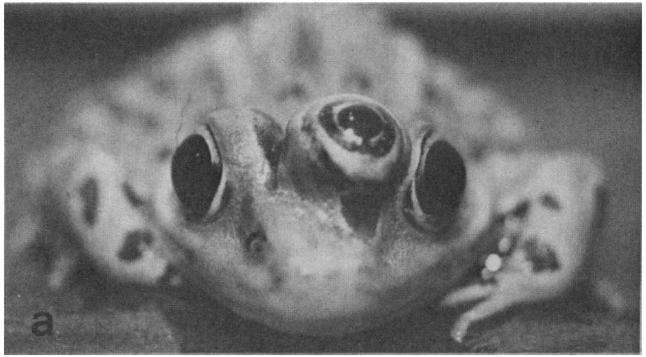
Seeing with your tongue



Human echolocation (sonar)

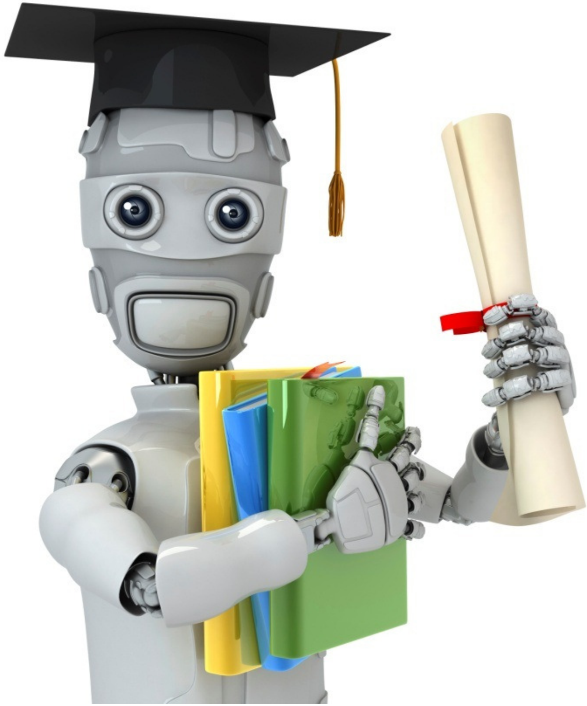


Haptic belt: Direction sense



Implanting a 3rd eye

[BrainPort; Welsh & Blasch, 1997; Nagel et al., 2005; Constantine-Paton & Law, 2009]



Machine Learning

Neural Networks: Representation

Model representation I

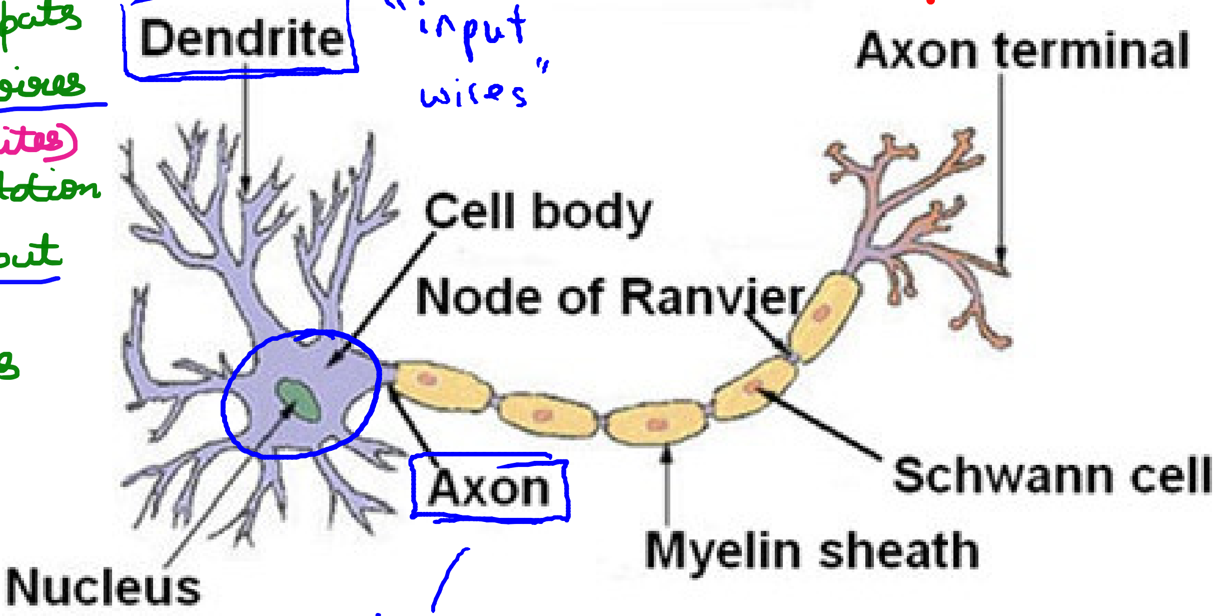
Neuron in the brain

Neural networks were developed as simulating neurons or networks of neurons in the brain

Computational unit that gets a number of inputs through input wires (via Dendrites) does some computation and sends output (via Axon) to other Neurons

Receive input from other neurons

"input wires"

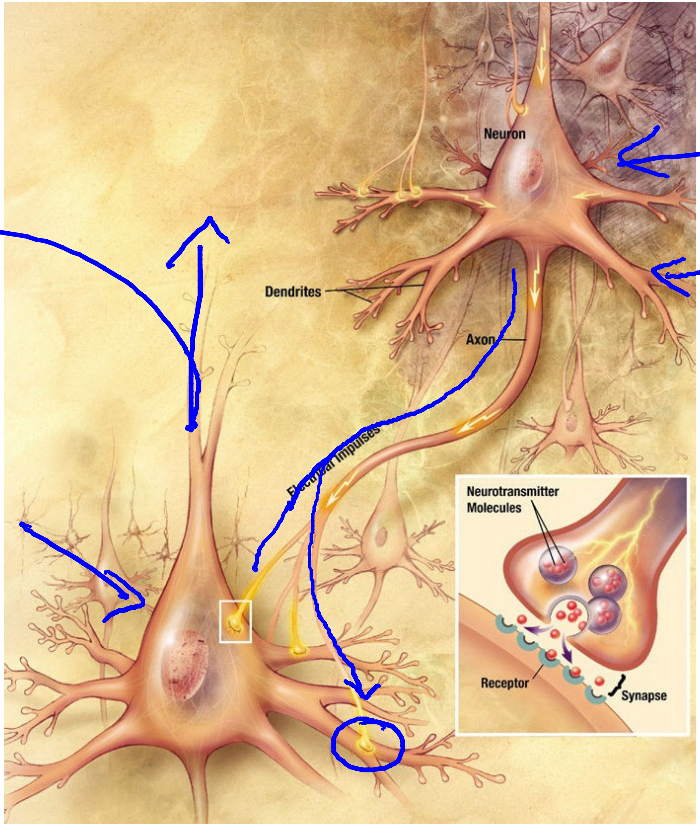


"output wire"

Sends the signal to other neurons

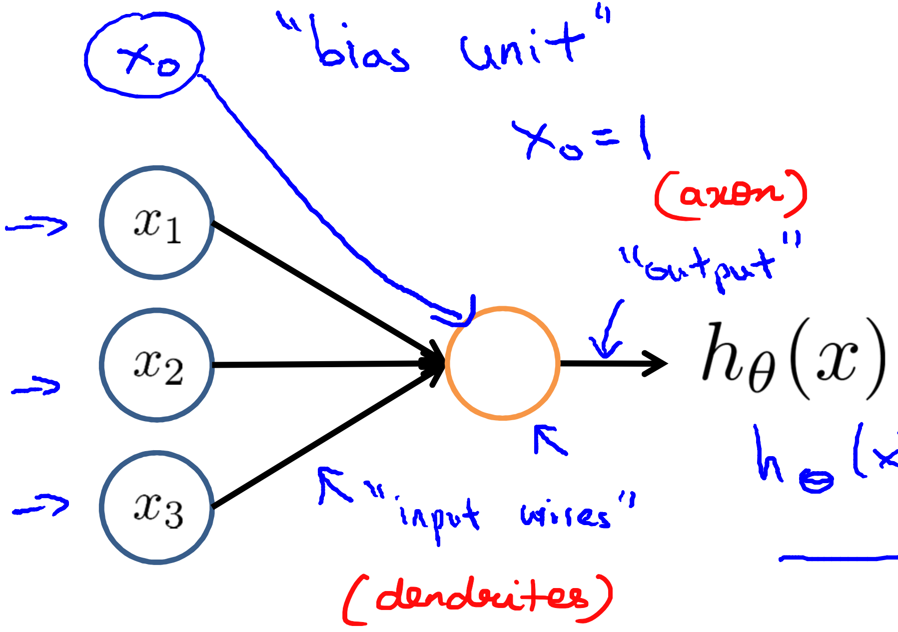
Neurons in the brain

muscle



[Credit: US National Institutes of Health, National Institute on Aging]

Neuron model: Logistic unit



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

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

"weights"

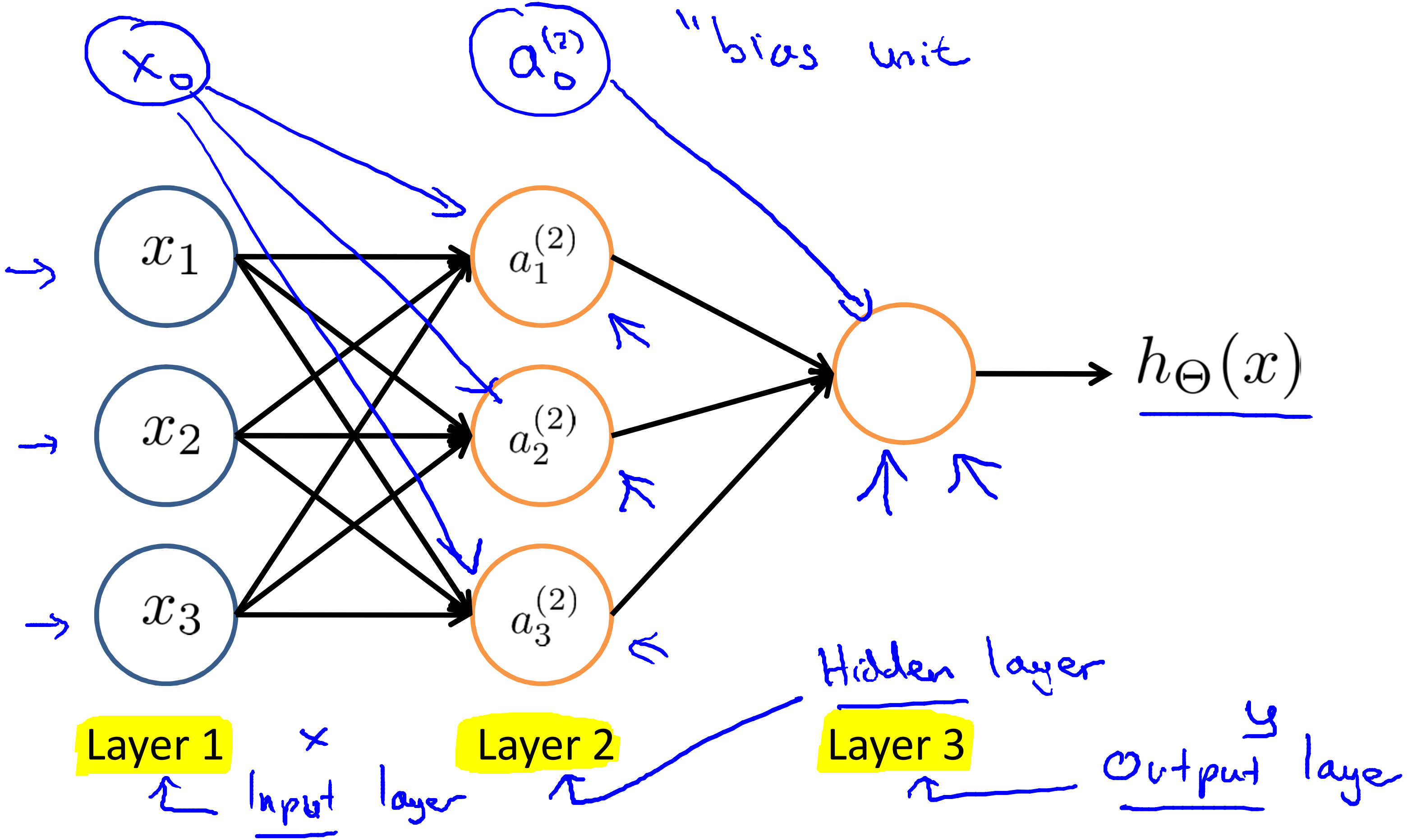
(parameters)

Sigmoid (logistic) activation function.

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

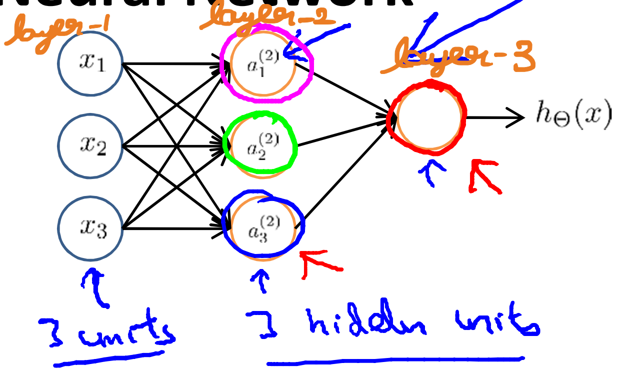
(Group of different neuron strong together)

Neural Network



$x_0 \Rightarrow$ bias

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 $j+1$

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

$h_\Theta(x)$

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

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

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

layer 1 to layer 2 ($j=1$)

$\Theta^{(2)}$

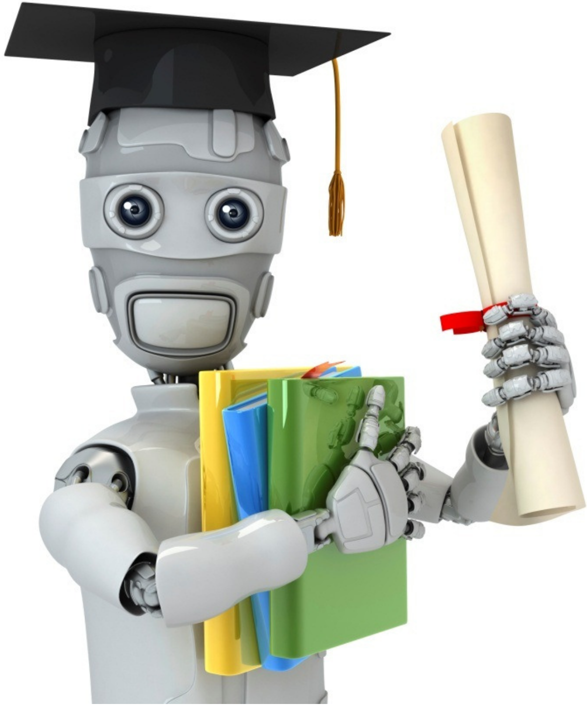
$$h_\Theta(x) = a_1^{(3)} = g(\Theta_{10}^{(2)}a_0^{(2)} + \Theta_{11}^{(2)}a_1^{(2)} + \Theta_{12}^{(2)}a_2^{(2)} + \Theta_{13}^{(2)}a_3^{(2)})$$

($j=2$) layer-2 to layer-3

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

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

Andrew Ng

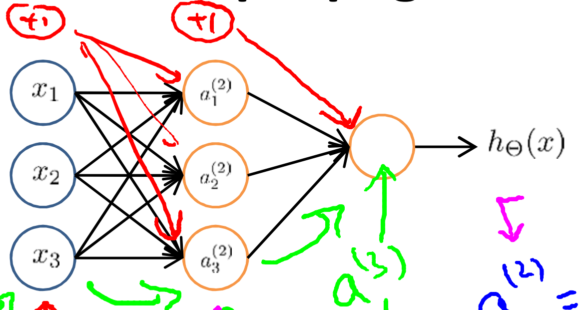


Neural Networks: Representation

Model representation II

Machine Learning

Forward propagation: Vectorized implementation



$a^{(1)} = x$

$a_1^{(2)} = g(z_1^{(2)})$

$z_1^{(2)}$

$$a_1^{(2)} = g(\underbrace{\Theta_{10}^{(1)}x_0 + \Theta_{11}^{(1)}x_1 + \Theta_{12}^{(1)}x_2 + \Theta_{13}^{(1)}x_3}_{z_1^{(2)}})$$

$$a_2^{(2)} = g(\underbrace{\Theta_{20}^{(1)}x_0 + \Theta_{21}^{(1)}x_1 + \Theta_{22}^{(1)}x_2 + \Theta_{23}^{(1)}x_3}_{z_2^{(2)}})$$

$$a_3^{(2)} = g(\underbrace{\Theta_{30}^{(1)}x_0 + \Theta_{31}^{(1)}x_1 + \Theta_{32}^{(1)}x_2 + \Theta_{33}^{(1)}x_3}_{z_3^{(2)}})$$

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

$\Theta^{(1)}$ x

$a_3^{(2)} = g(z_3^{(2)})$

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

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

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

$\mathbb{R}^3$ $\mathbb{R}^3$

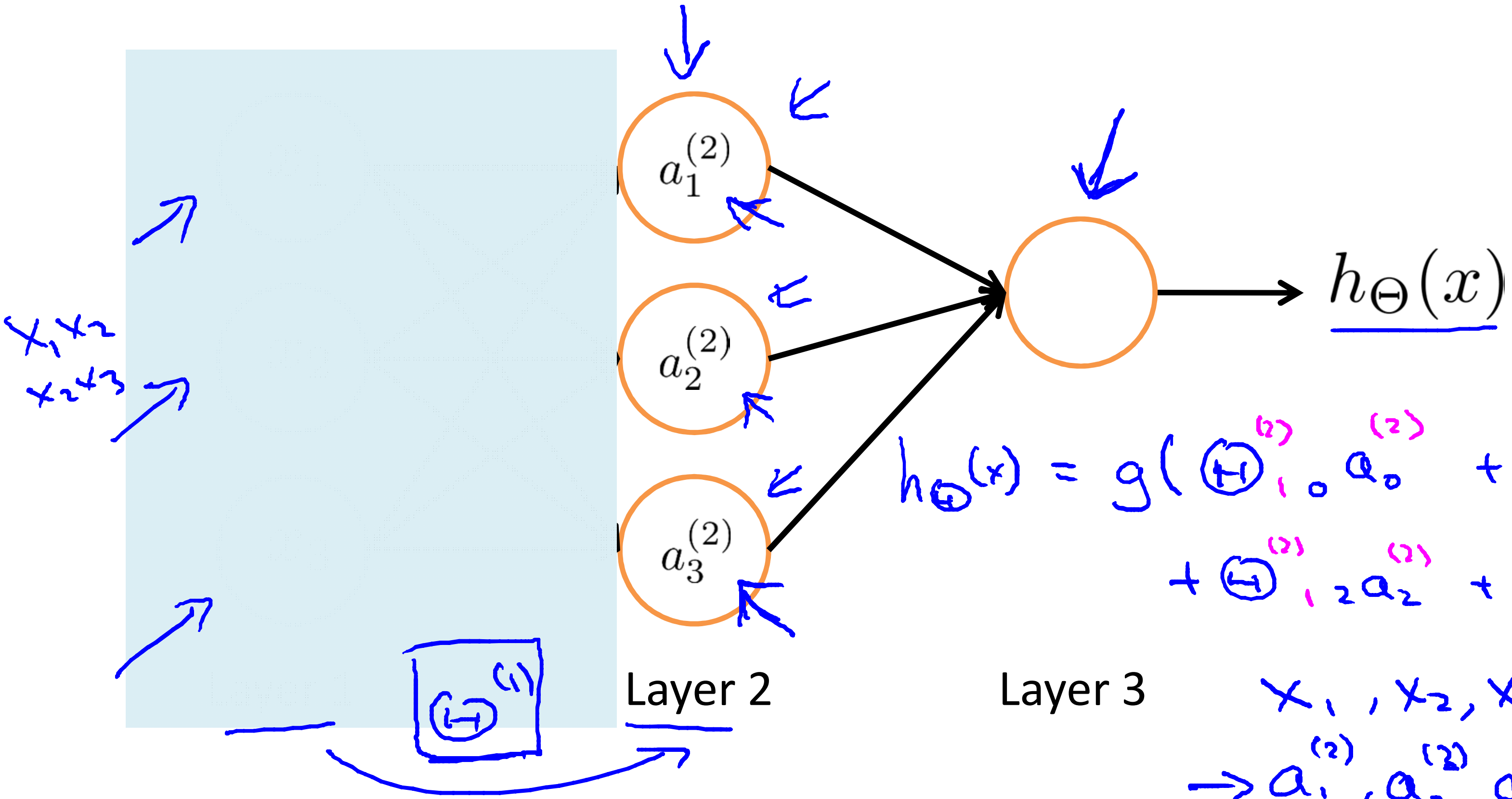
$a_1^{(2)}$, $a_2^{(2)}$, $a_3^{(2)}$

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

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

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

Neural Network learning its own features

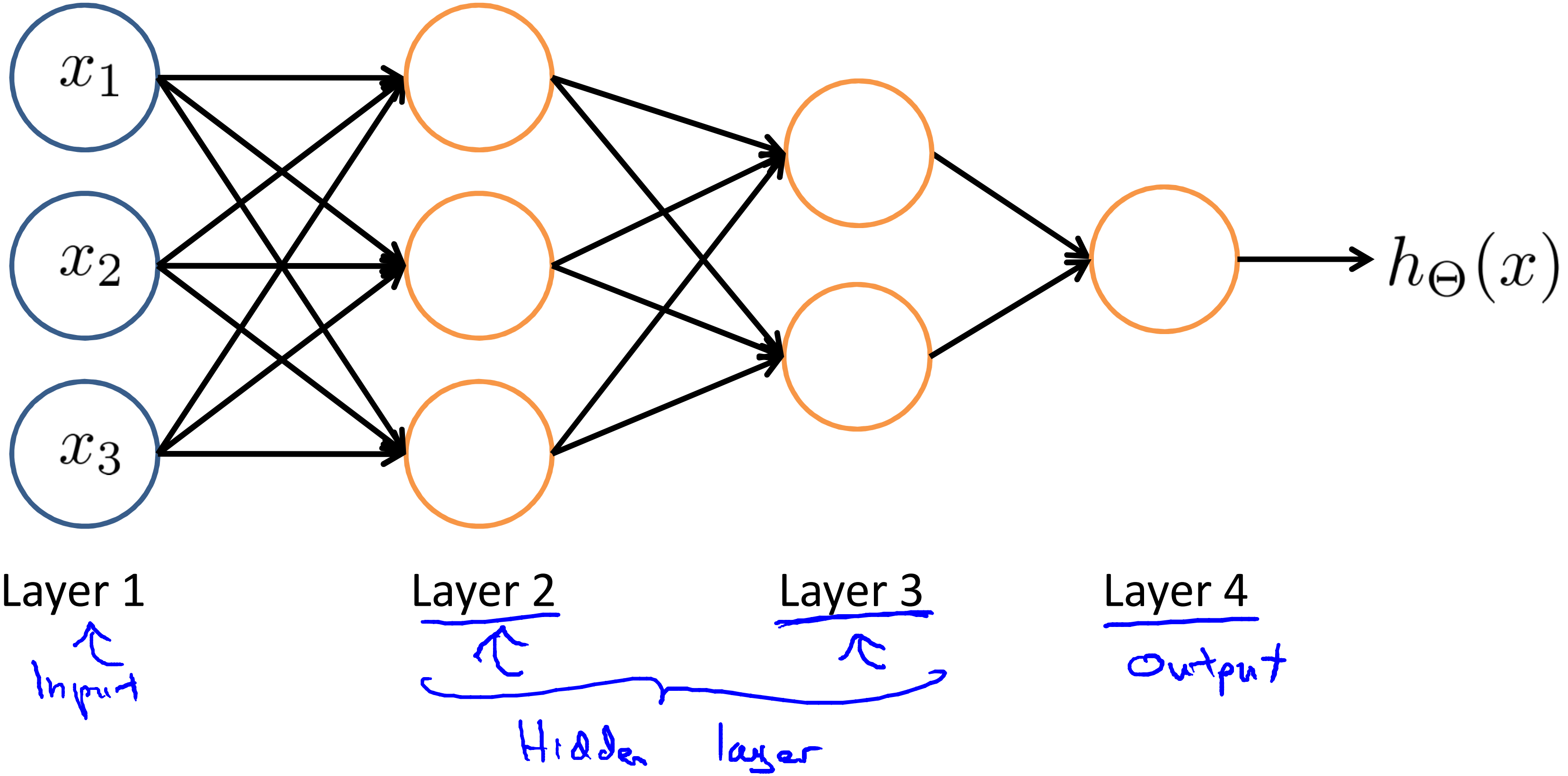


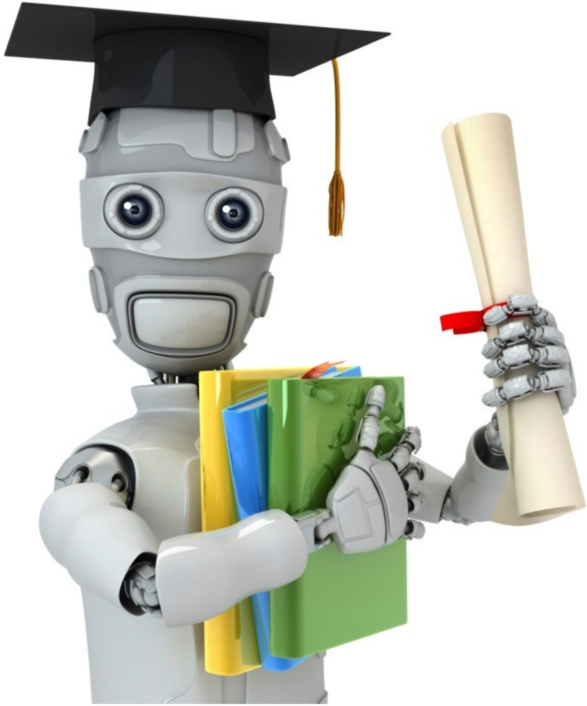
$$h_\Theta(x) = g\left(\Theta^{(2)}_{10} a_0^{(2)} + \Theta^{(1)}_{11} a_1^{(2)} + \Theta^{(2)}_{12} a_2^{(2)} + \Theta^{(2)}_{13} a_3^{(2)}\right)$$

x_1, x_2, x_3

$\rightarrow a_1^{(2)}, a_2^{(2)}, a_3^{(2)}$

Other network architectures





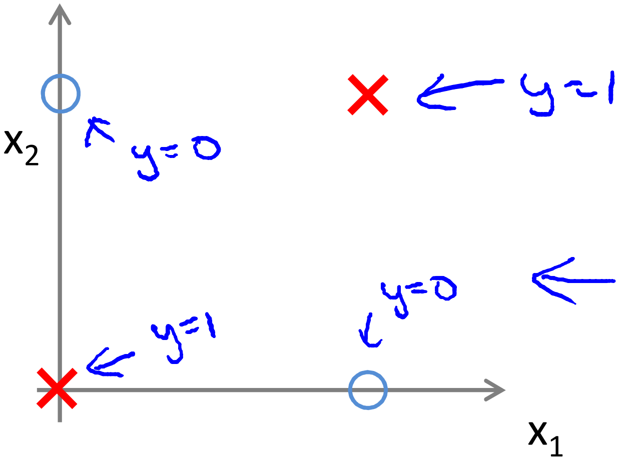
Machine Learning

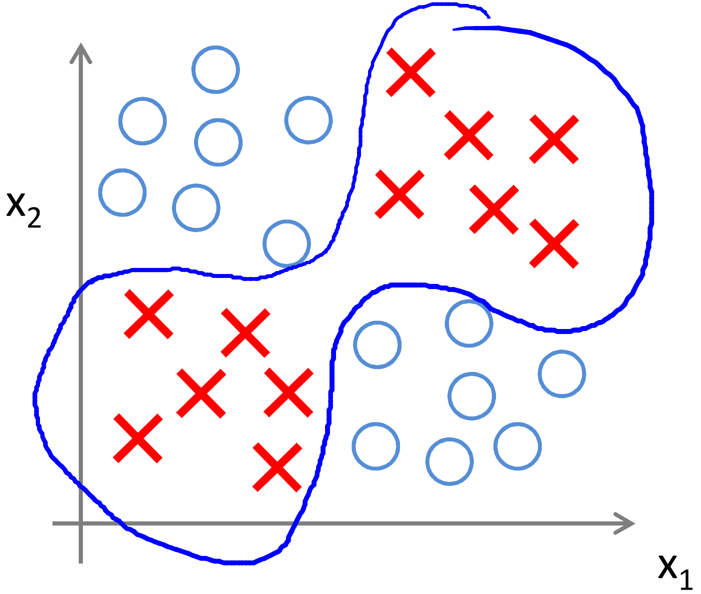
Neural Networks: Representation

Examples and intuitions I

Non-linear classification example: XOR/XNOR

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





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

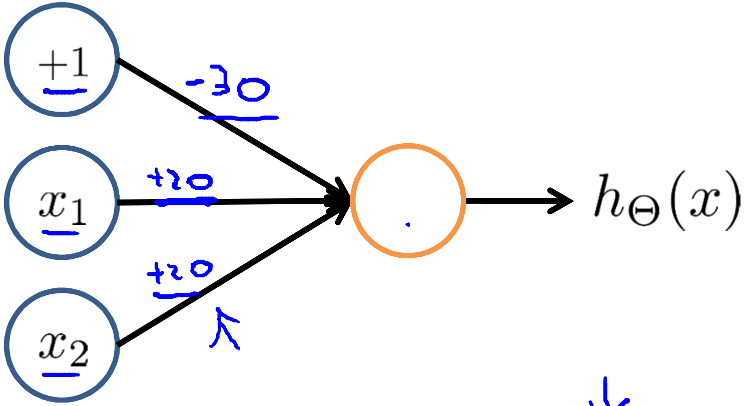
$$x_1 \text{ XNOR } x_2$$

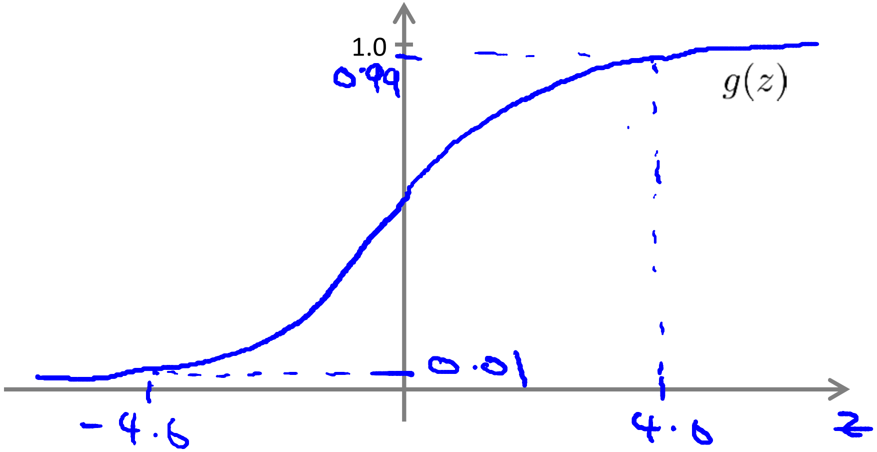
$$\text{NOT } (x_1 \text{ XOR } x_2)$$

Simple example: AND

→ $x_1, x_2 \in \{0, 1\}$

→ $y = x_1 \text{ AND } x_2$



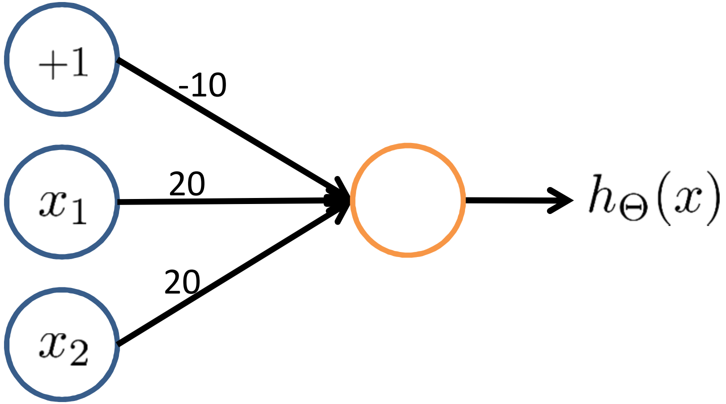


→ $h_\Theta(x) = g(\underbrace{-30}_{\Theta_{10}^{(1)}} + \underbrace{20}_{\Theta_{11}^{(1)}} x_1 + \underbrace{20}_{\Theta_{12}^{(1)}} x_2)$

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	$g(10) \approx 1$

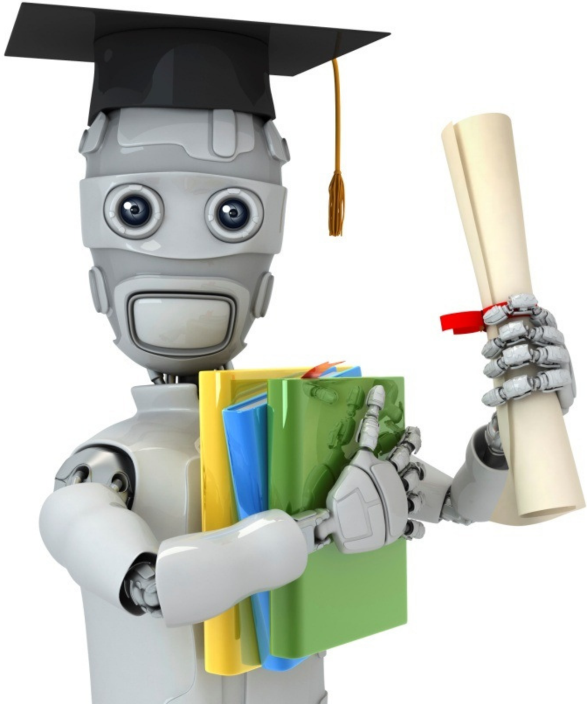
$h_\Theta(x) \approx x_1 \text{ AND } x_2$

Example: OR function



$g(-10 + 20x_1 + 20x_2)$

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



Machine Learning

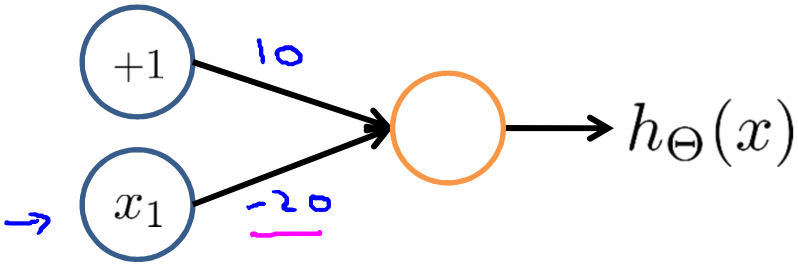
Neural Networks: Representation

Examples and intuitions II

$\rightarrow x_1 \text{ AND } x_2$ $\rightarrow x_1 \text{ OR } x_2$ $\{0,1\}.$

Negation:

NOT x_1



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

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

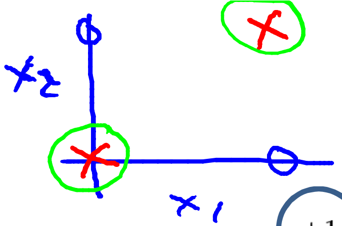
$\rightarrow$ (NOT x_1) AND (NOT x_2)

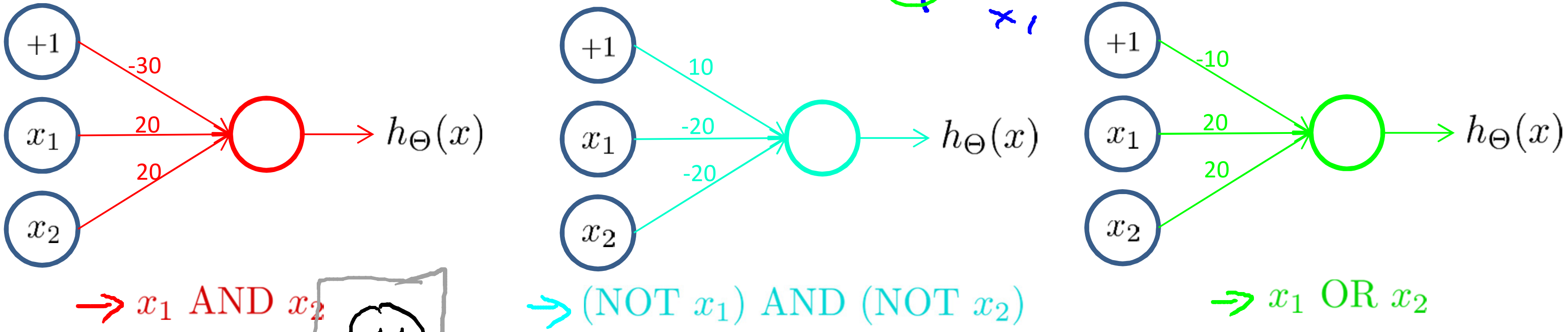
$(=1$ if and only if

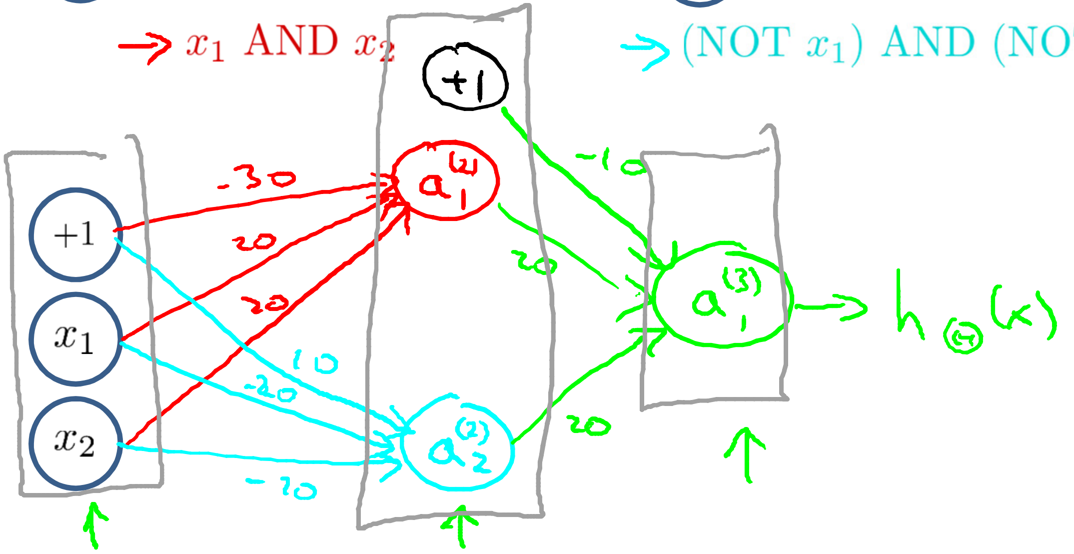
$\rightarrow x_1 = x_2 = 0$

Andrew Ng

Putting it together: $x_1 \text{ XNOR } x_2$

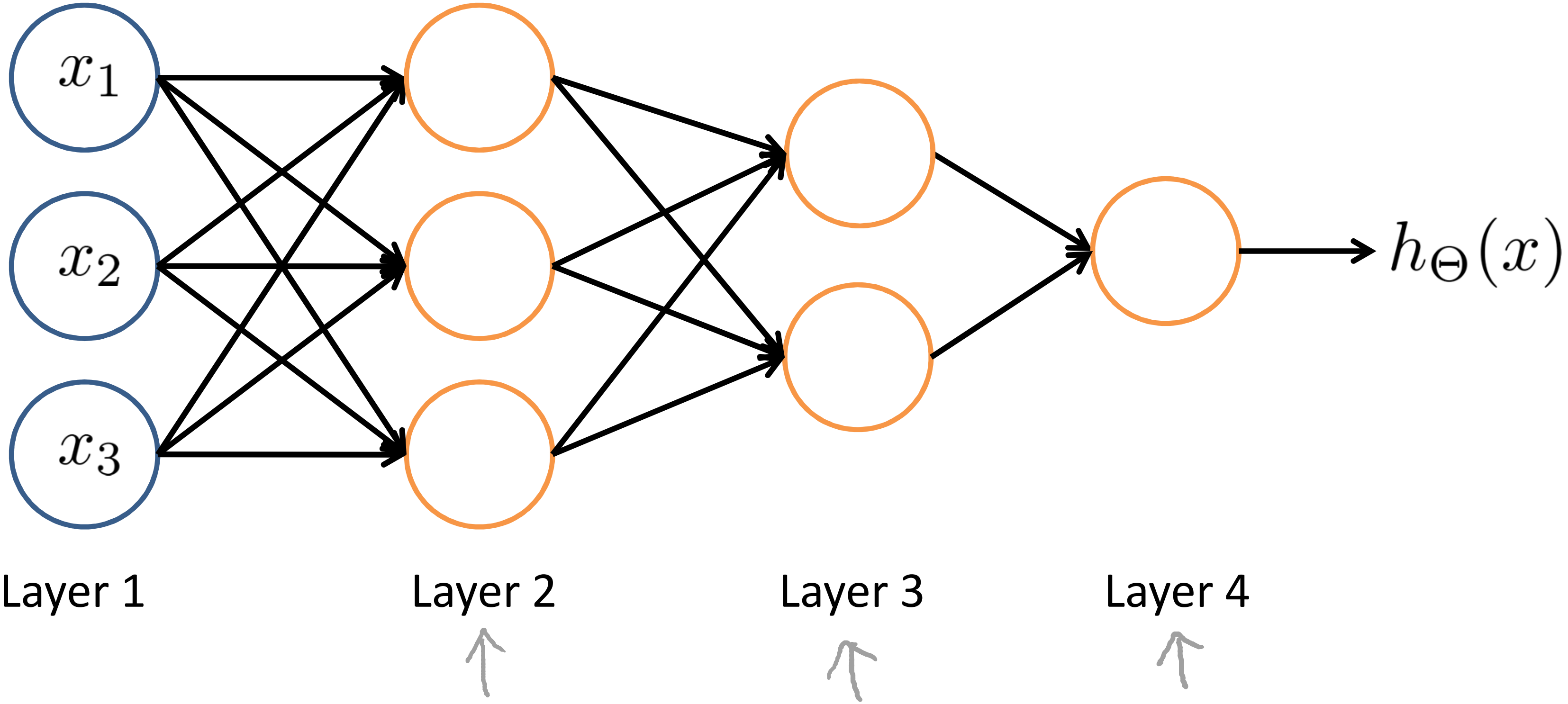




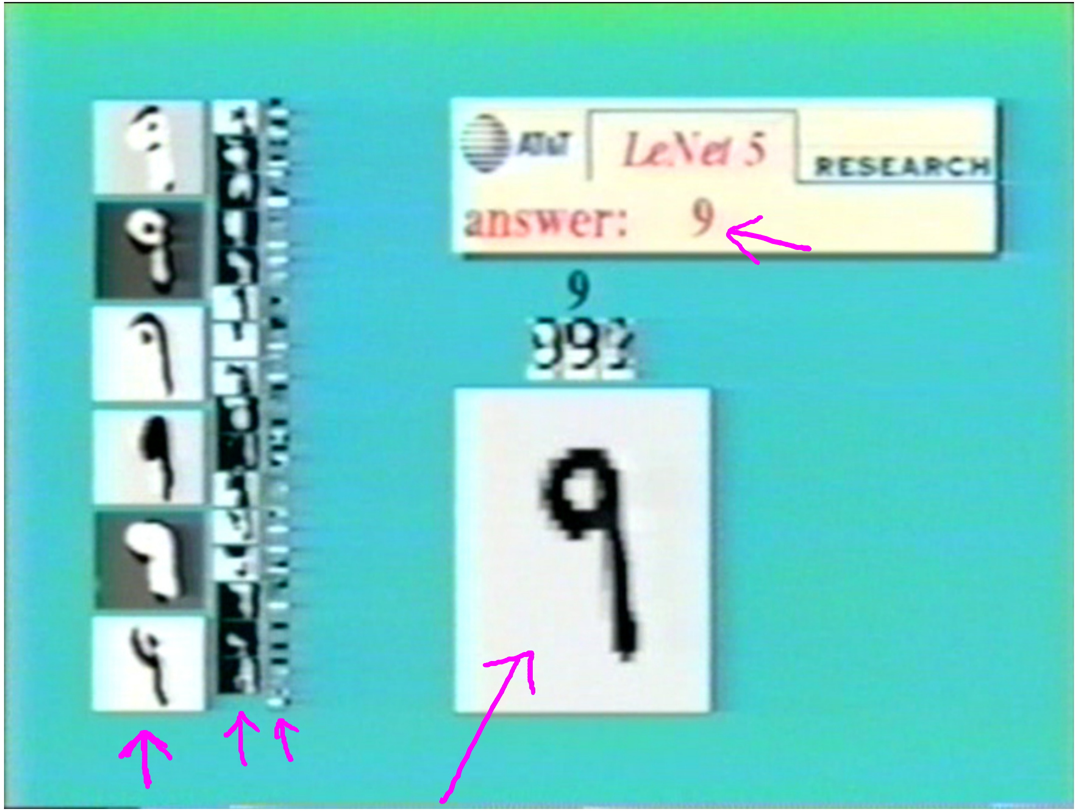


x_1	x_2	$a_1^{(2)}$	$a_2^{(2)}$	$h_\Theta(x)$
0	0	0	1	1
0	1	0	0	0
1	0	0	0	0
1	1	1	0	1

Neural Network intuition

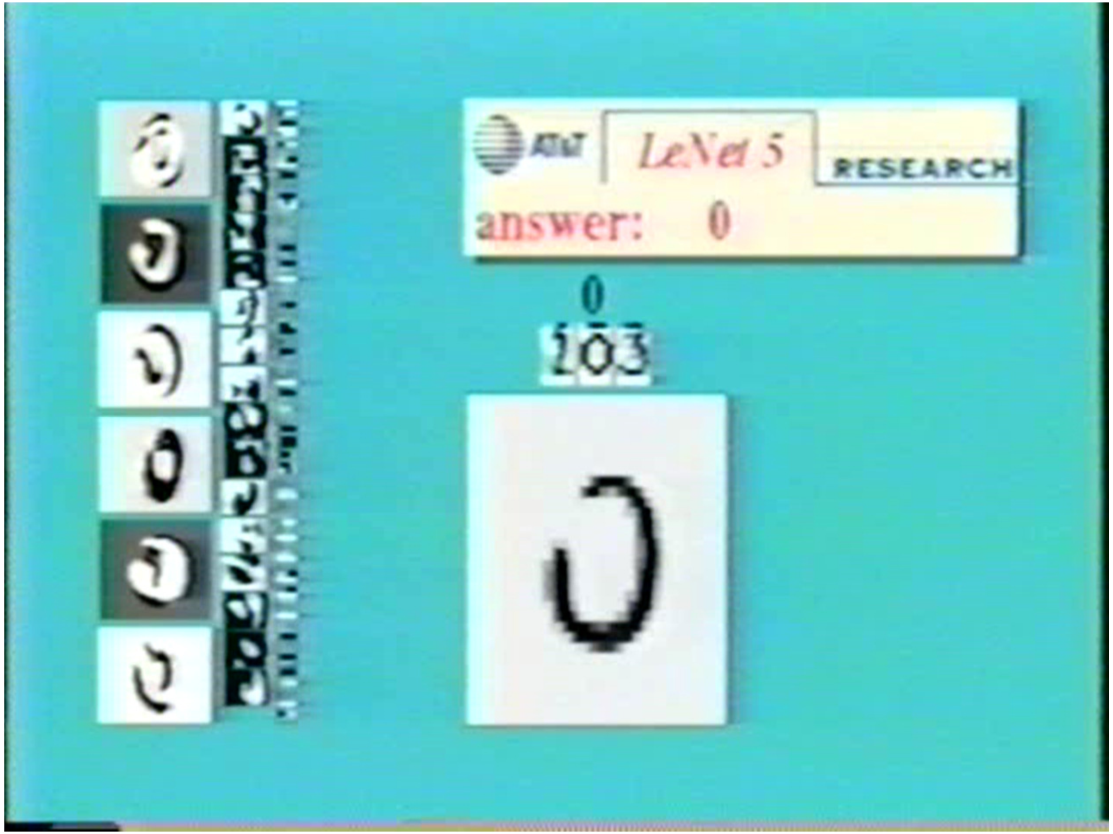


Handwritten digit classification

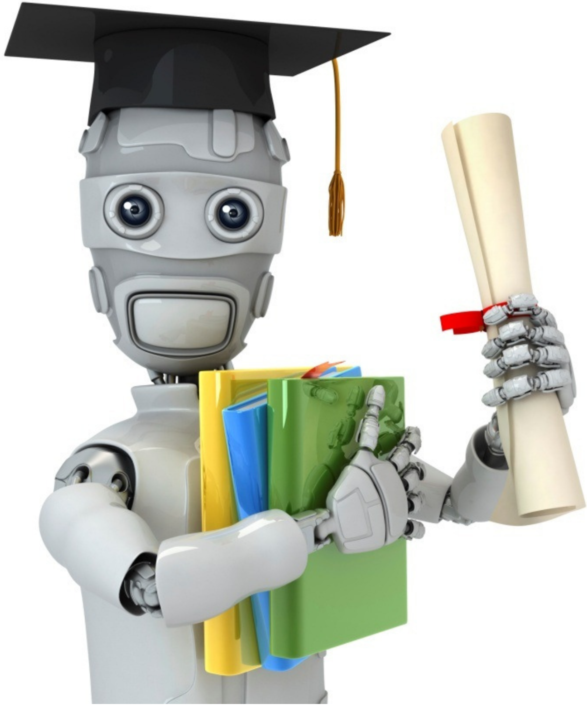


[Courtesy of Yann LeCun]

Handwritten digit classification



[Courtesy of Yann LeCun]



Machine Learning

Neural Networks: Representation

Multi-class classification

Andrew Ng

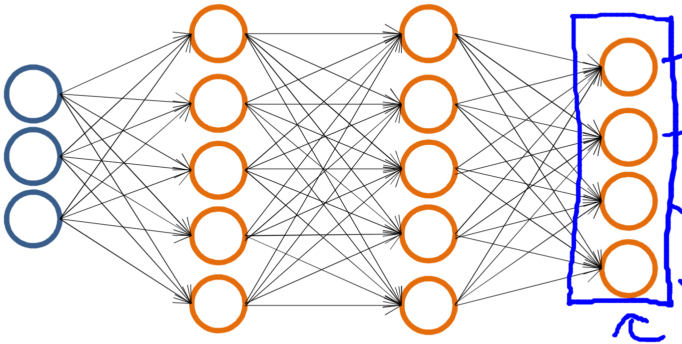
Multiple output units: One-vs-all.


Pedestrian


Car


Motorcycle


Truck



pedestrian?

car?

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

motorcycle?

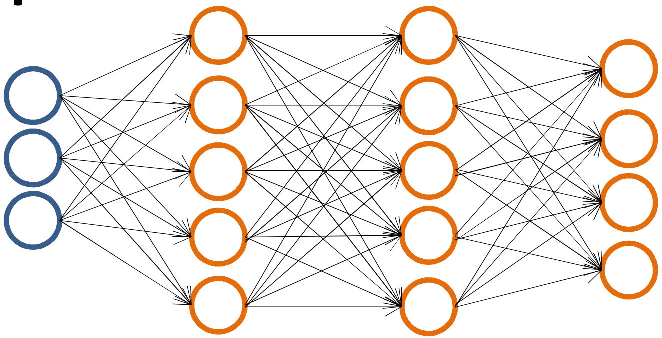
truck?

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

Andrew Ng

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 $\begin{array}{cccc} \begin{bmatrix}1\\0\\0\\0\end{bmatrix}, & \begin{bmatrix}0\\1\\0\\0\end{bmatrix}, & \begin{bmatrix}0\\0\\1\\0\end{bmatrix}, & \begin{bmatrix}0\\0\\0\\1\end{bmatrix} \\ \text{pedestrian} & \text{car} & \text{motorcycle} & \text{truck} \end{array}$

$(x^{(i)}, y^{(i)})$

~~Previously~~ $\cancel{y \in \{1,2,3,4\}}$

$\underbrace{h_\Theta(x^{(i)})}_{\mathbb{R}^4} \approx y^{(i)}$