Neural Networks I

Jia-Bin Huang
Virginia Tech

ECE-5424G / CS-5824

Spring 2019

Administrative

- HW 2 released!

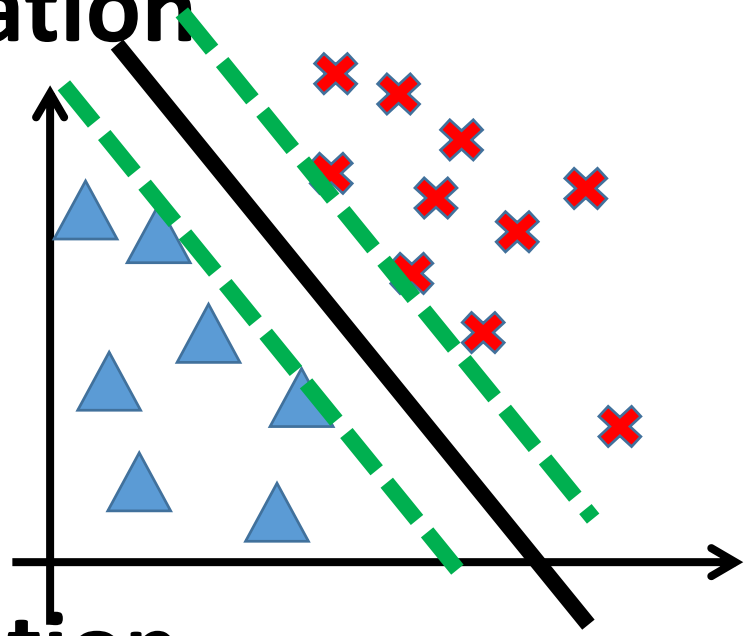
- Jia-Bin out of town next week

- Chen: Deep Neural Networks II
- Shih-Yang: Diagnosing ML systems
- Midterm review
- Midterm: March 6th (Wed)

Hard-margin SVM formulation

$$\min_{\theta} \frac{1}{2}\sum_{j=1}^{n} \theta_j^2$$

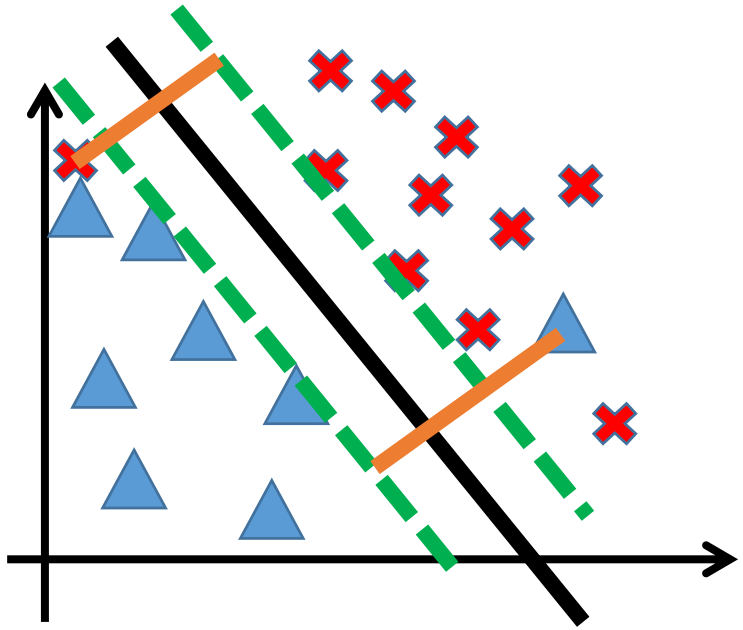
$$\text{s.t.} \quad \theta^\top x^{(i)} \geq 1 \quad \text{if} \quad y^{(i)} = 1$$
$$\theta^\top x^{(i)} \leq -1 \quad \text{if} \quad y^{(i)} = 0$$



Soft-margin SVM formulation

$$\min_{\theta, \{\xi^{(i)}\}} \frac{1}{2}\sum_{j=1}^{n} \theta_j^2 + C\sum_i \xi^{(i)}$$

$$\text{s.t.} \quad \theta^\top x^{(i)} \geq 1 - \xi^{(i)} \quad \text{if} \quad y^{(i)} = 1$$
$$\theta^\top x^{(i)} \leq -1 + \xi^{(i)} \quad \text{if} \quad y^{(i)} = 0$$
$$\xi^{(i)} \geq 0 \qquad \forall\, i$$



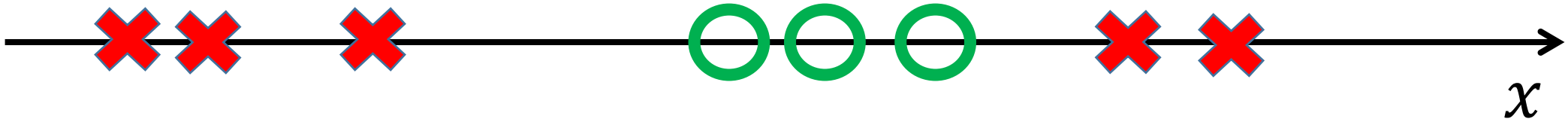
Support Vector Machine

- Cost function
- Large margin classification
- **Kernels**
- Using an SVM

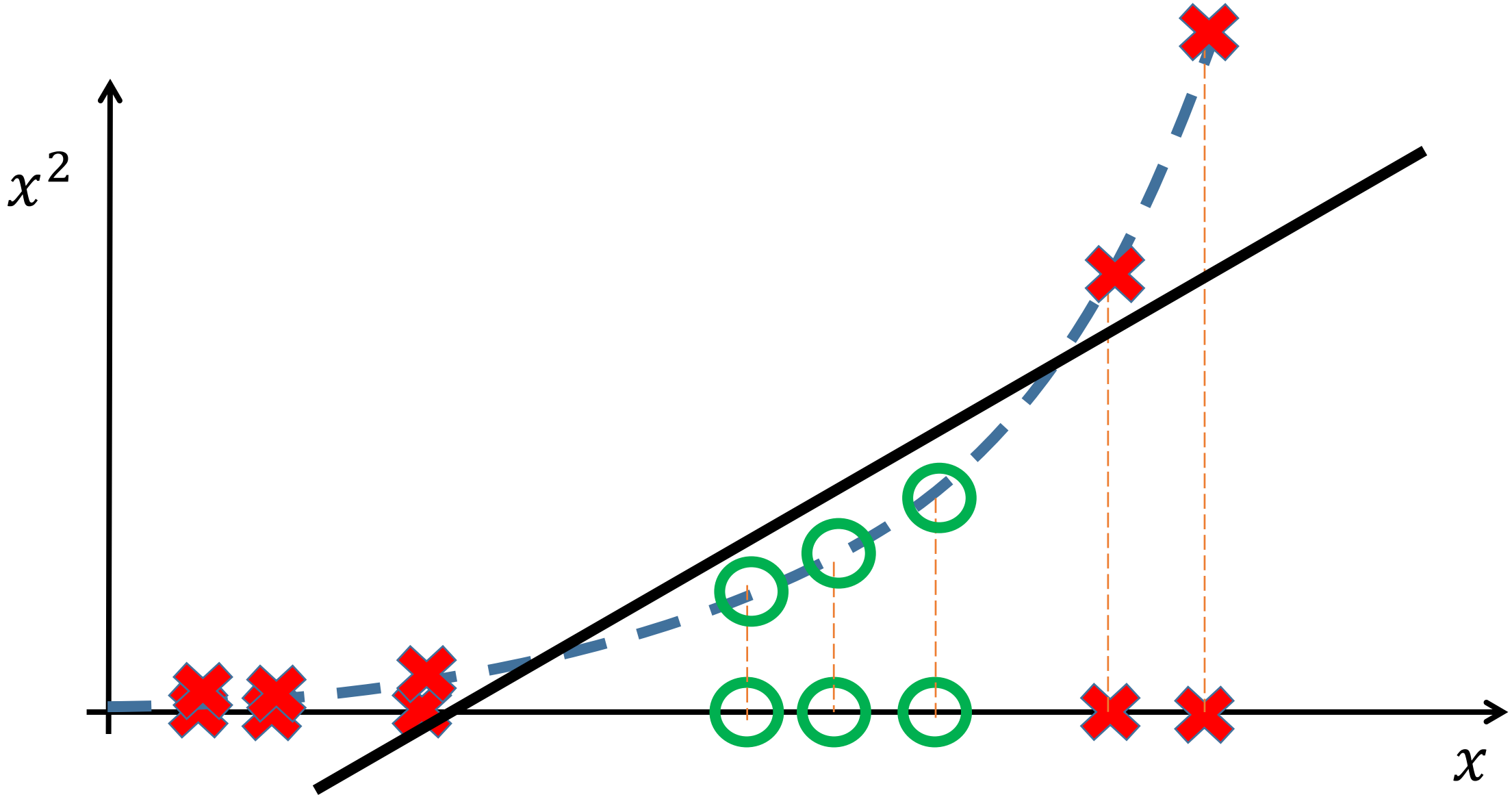
Non-linear classification

- How do we separate the two classes using a hyperplane?

$$h_\theta(x) = \begin{cases} 1 & \text{if } \theta^\top x \geq 0 \\ 0 & \text{if } \theta^\top x < 0 \end{cases}$$



Non-linear classification



Kernel

- $K(\cdot,\cdot)$ a legal definition of inner product:

$\exists \phi: X \to R^N$
s.t. $K(x,z) = \phi(x)^\top \phi(z)$

Why Kernels matter?

- Many algorithms interact with data only via **dot-products**
- Replace $x^\top z$ with $K(x, z) = \phi(x)^\top \phi(z)$
- Act *implicitly* as if data was in the higher-dimensional ϕ-space

Example

$K(x, z) = (x^\top z)^2$ corresponds to

$(x_1, x_2) \to \phi(x) = \left(x_1^2, x_2^2, \sqrt{2}x_1x_2\right)$

$$\phi(x)^\top \phi(z) = \left(x_1^2, x_2^2, \sqrt{2}x_1x_2\right)\left(z_1^2, z_2^2, \sqrt{2}z_1z_2\right)^\top$$

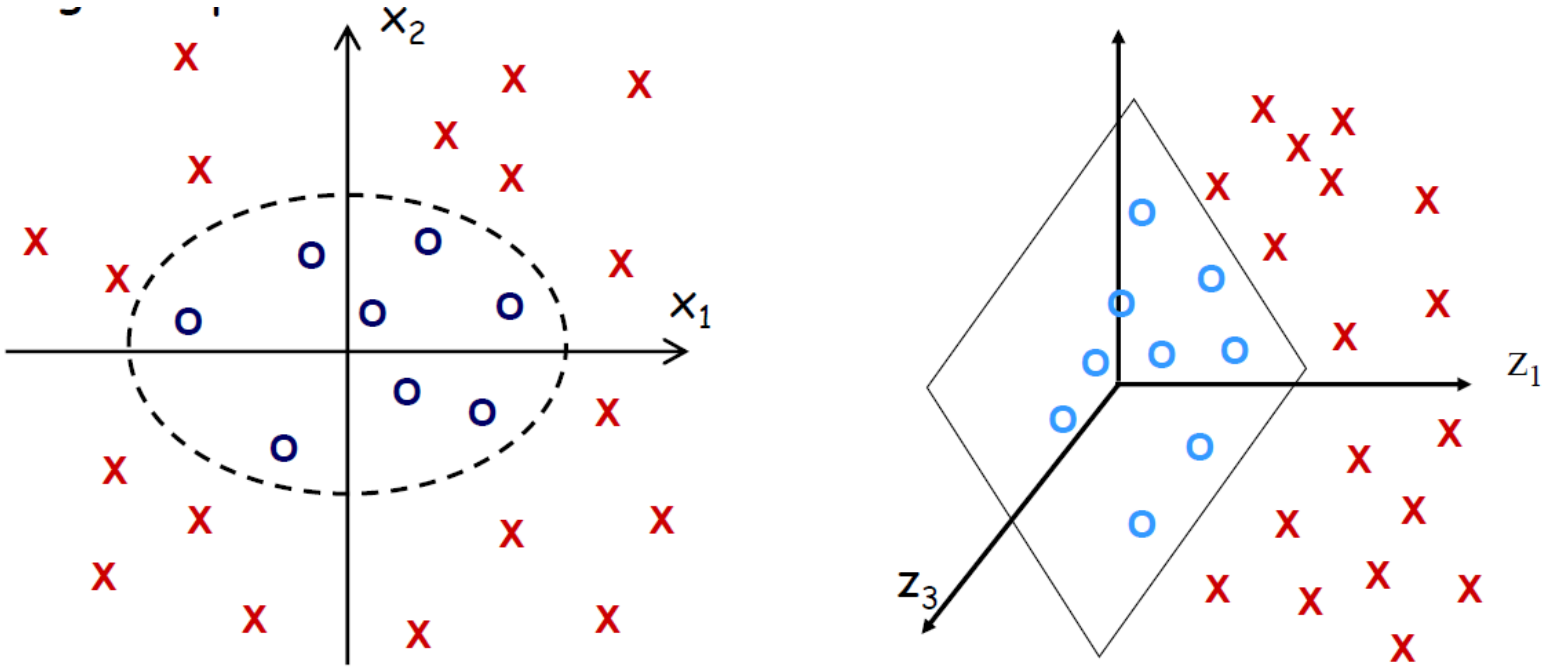
$$= x_1^2z_1^2 + x_2^2z_2^2 + 2\mathrm{x}_1\mathrm{x}_2\mathrm{z}_1\mathrm{z}_2 = (\mathrm{x}_1\mathrm{z}_1 + \mathrm{x}_2\mathrm{z}_2)^2$$

$$= (x^\top z)^2 = K(x, z)$$

Example

$K(x, z) = (x^\top z)^2$ corresponds to

$(x_1, x_2) \rightarrow \phi(x) = \left(x_1^2, x_2^2, \sqrt{2}x_1x_2\right)$



Slide credit: Maria-Florina Balcan

Example kernels

- Linear kernel

$$K(x, z) = x^\top z$$

- Gaussian (Radial basis function) kernel

$$K(x, z) = \exp\left(-\frac{1}{2}(x - z)^\top \Sigma^{-1}(x - z)\right)$$

- Sigmoid kernel

$$K(x, z) = \tanh(a \cdot x^\top z + b)$$

Constructing new kernels

- Positive scaling

$$K(x,z) = cK_1(x,z), c > 0$$

- Exponentiation

$$K(x,z) = \exp\big(K_1(x,z)\big)$$

- Addition

$$K(x,z) = K_1(x,z) + K_2(x,z)$$

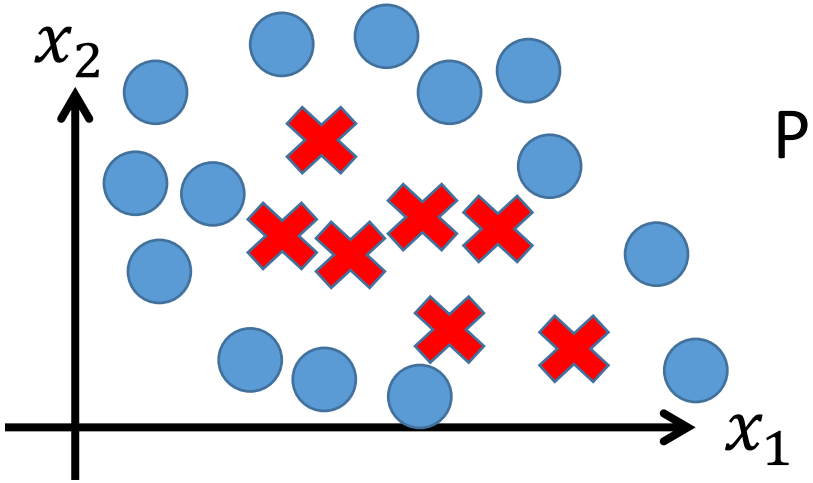
- Multiplication with function

$$K(x,z) = f(x)K_1(x,z)f(z)$$

- Multiplication

$$K(x,z) = K_1(x,z)K_2(x,z)$$

Non-linear decision boundary



Predict $y = 1$ if

$$\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2 + \cdots \geq 0$$

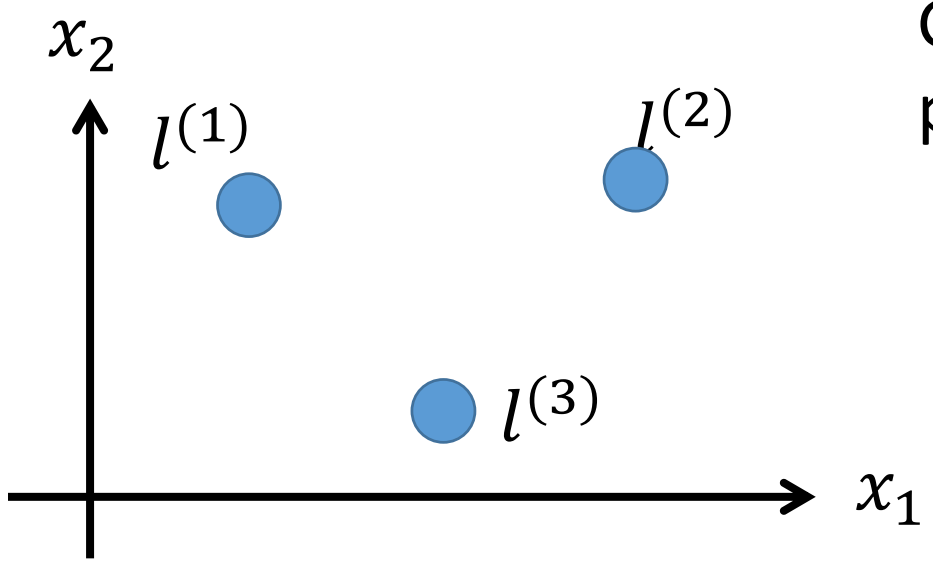
$$\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 + \cdots$$

$$f_1 = x_1, f_2 = x_2, f_3 = x_1 x_2, \cdots$$

Is there a different/better choice of the features $f_1, f_2, f_3, \cdots$?

Slide credit: Andrew Ng

Kernel



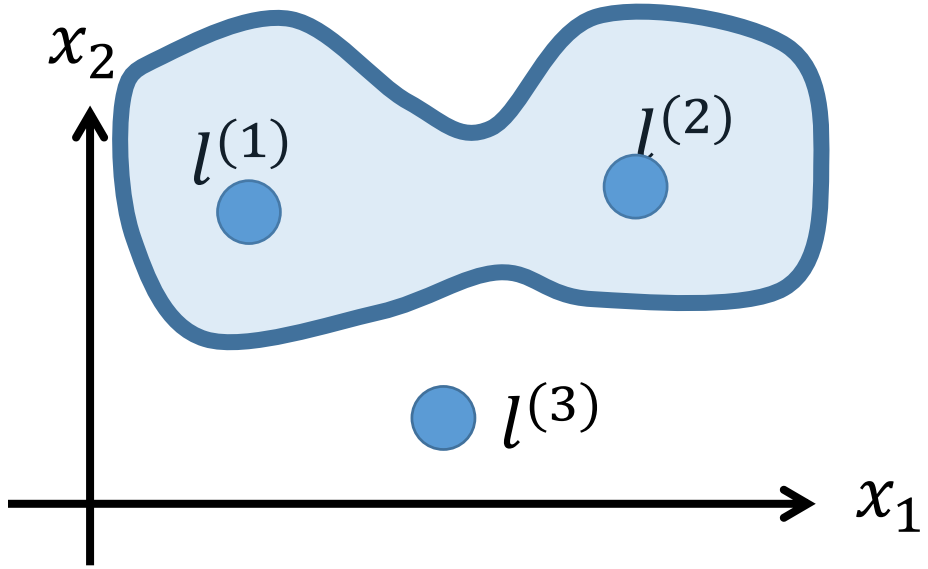
Give x, compute new features depending on proximity to landmarks $l^{(1)}$, $l^{(2)}$, $l^{(3)}$

$$f_1 = \text{similarity}(x, l^{(1)})$$
$$f_2 = \text{similarity}(x, l^{(2)})$$
$$f_3 = \text{similarity}(x, l^{(3)})$$

Gaussian kernel

$$\text{similarity}\left(x, l^{(i)}\right) = \exp\left(-\frac{\left\|x - l^{(i)}\right\|^2}{2\sigma^2}\right)$$

Slide credit: Andrew Ng



Predict $y = 1$ if

$$\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$$

Ex: $\theta_0 = -0.5, \theta_1 = 1, \theta_2 = 1, \theta_3 = 0$

$$f_1 = \text{similarity}(x, l^{(1)})$$
$$f_2 = \text{similarity}(x, l^{(2)})$$
$$f_3 = \text{similarity}(x, l^{(3)})$$

Slide credit: Andrew Ng

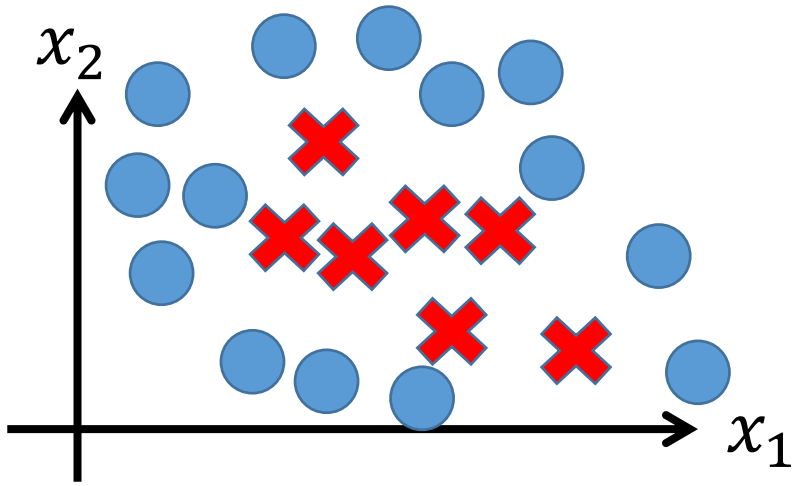
Choosing the landmarks

- Given x

$$f_i = \text{similarity}\left(x, l^{(i)}\right) = \exp\left(-\frac{\left\|x - l^{(i)}\right\|^2}{2\sigma^2}\right)$$

Predict $y = 1$ if $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$

Where to get $l^{(1)}, l^{(2)}, l^{(3)}, \cdots$?



$l^{(1)}$
$l^{(2)}$
$l^{(3)}$
$\vdots$
$l^{(m)}$

Slide credit: Andrew Ng

SVM with kernels

- Given $\left(x^{(1)}, y^{(1)}\right), \left(x^{(2)}, y^{(2)}\right), \cdots, \left(x^{(m)}, y^{(m)}\right)$
- Choose $l^{(1)} = x^{(1)}, l^{(2)} = x^{(2)}, l^{(3)} = x^{(3)}, \cdots, l^{(m)} = x^{(m)}$
- Given example x:
  - $f_1 = \text{similarity}\left(x, l^{(1)}\right)$
  - $f_2 = \text{similarity}\left(x, l^{(2)}\right)$
  - $\cdots$
- For training example $\left(x^{(i)}, y^{(i)}\right)$:
  - $x^{(i)} \rightarrow f^{(i)}$

$$f = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_m \end{bmatrix}$$

Slide credit: Andrew Ng

SVM with kernels

- Hypothesis: Given x, compute features $f \in \mathbb{R}^{m+1}$
  - Predict $y = 1 \quad \text{if} \quad \theta^\top f \geq 0$

- **Training (original)**

$$\min_\theta C\left[\sum_{i=1}^{m} y^{(i)}\ \text{cost}_1\left(\theta^\top x^{(i)}\right) + \left(1 - y^{(i)}\right)\text{cost}_0\left(\theta^\top x^{(i)}\right)\right] + \frac{1}{2}\sum_{j=1}^{n}\theta_j^2$$

- **Training (with kernel)**

$$\min_\theta C\left[\sum_{i=1}^{m} y^{(i)}\ \text{cost}_1\left(\theta^\top f^{(i)}\right) + \left(1 - y^{(i)}\right)\text{cost}_0\left(\theta^\top f^{(i)}\right)\right] + \frac{1}{2}\sum_{j=1}^{m}\theta_j^2$$

Support vector machines (Primal/Dual)

- Primal form

$$\min_{\theta,\xi^{(i)}} \frac{1}{2}\sum_{j=1}^{n}\theta_j^2 + C\sum_i \xi^{(i)}$$

$$\text{s.t.} \quad \begin{aligned} \theta^\top x^{(i)} &\geq 1 - \xi^{(i)} && \text{if} \quad y^{(i)} = 1 \\ \theta^\top x^{(i)} &\leq -1 + \xi^{(i)} && \text{if} \quad y^{(i)} = 0 \\ \xi^{(i)} &\geq 0 \qquad \forall\, i \end{aligned}$$

- Lagrangian dual form

$$\min_{\alpha} \frac{1}{2}\sum_i\sum_j y^{(i)}y^{(j)}\alpha^{(i)}\alpha^{(j)}x^{(i)\top}x^{(j)} - \sum_i \alpha^{(i)}$$

$$\text{s.t.} \quad \begin{aligned} &0 \leq \alpha^{(i)} \leq C_i \\ &\sum_i y^{(i)}\alpha^{(i)} = 0 \end{aligned}$$

SVM (Lagrangian dual)

$$\min_{\alpha} \frac{1}{2} \sum_i \sum_j y^{(i)} y^{(j)} \alpha^{(i)} \alpha^{(j)} \boxed{x^{(i)\top} x^{(j)}} - \sum_i \alpha^{(i)}$$

s. t. $\quad 0 \leq \alpha^{(i)} \leq C_i$

$$\sum_i y^{(i)} \alpha^{(i)} = 0$$

Replace $x^{(i)\top} x^{(j)}$ with $K(x^{(i)}, x^{(j)})$

Classifier: $\theta = \sum_i \alpha^{(i)} y^{(i)} x^{(i)}$

- The points $x^{(i)}$ for which $\alpha^{(i)} \neq 0 \rightarrow$ Support Vectors

SVM parameters

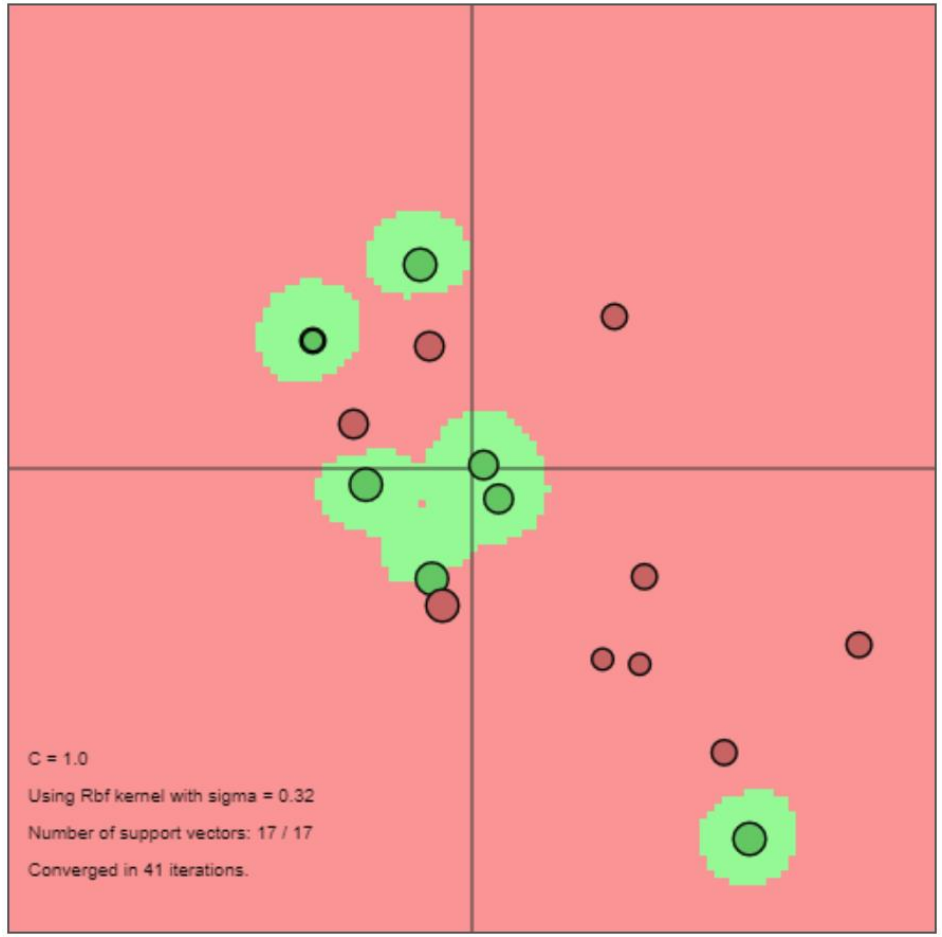
- $C\left(=\frac{1}{\lambda}\right)$
  Large C: Lower bias, high variance.
  Small C: Higher bias, low variance.

- σ^2
- Large σ^2: features f_i vary more smoothly.
  - Higher bias, lower variance
- Small σ^2: features f_i vary less smoothly.
  - Lower bias, higher variance

Slide credit: Andrew Ng

SVM Demo

- https://cs.stanford.edu/people/karpathy/svmjs/demo/



SVM song

- https://www.youtube.com/watch?v=g15bqtyidZs

Video source:

Support Vector Machine

- Cost function
- Large margin classification
- Kernels
- **Using an SVM**

Using SVM

- SVM software package (e.g., liblinear, libsvm) to solve for θ

- Need to specify:
  - Choice of parameter C.
  - Choice of kernel (similarity function):

- Linear kernel: Predict $y = 1 \quad \text{if} \quad \theta^\top x \geq 0$
- Gaussian kernel:
  - $f_i = \exp(-\frac{\|x - l^{(i)}\|^2}{2\sigma^2})$, where $l^{(i)} = x^{(i)}$
  - Need to choose σ^2. Need proper feature scaling

Slide credit: Andrew Ng

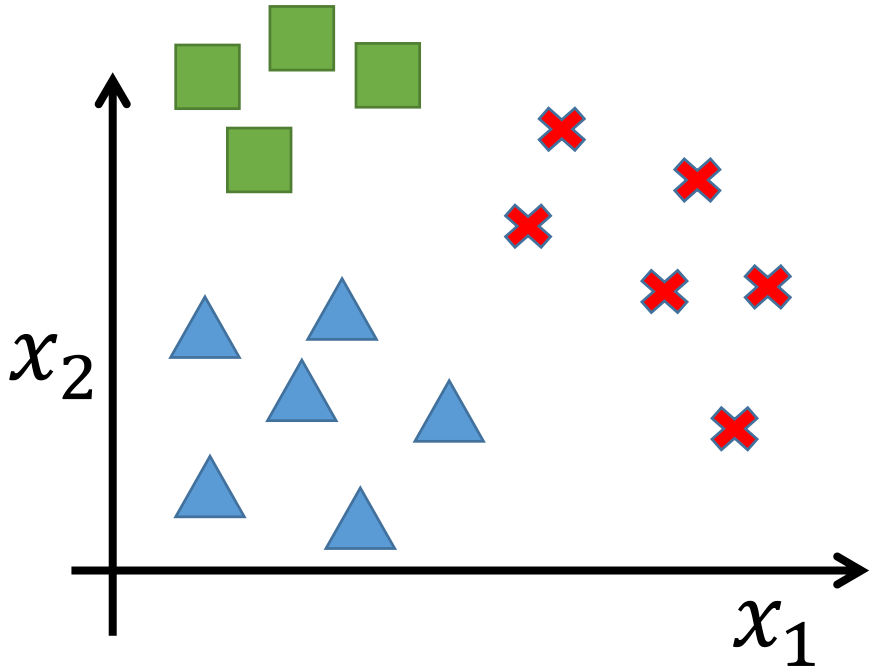
Kernel (similarity) functions

- Note: not all similarity functions make valid kernels.

- Many off-the-shelf kernels available:
  - Polynomial kernel
  - String kernel
  - Chi-square kernel
  - Histogram intersection kernel

Slide credit: Andrew Ng

Multi-class classification



- Use one-vs.-all method. Train K SVMs, one to distinguish $y = i$ from the rest, get $\theta^{(1)}, \theta^{(2)}, \cdots, \theta^{(K)}$

- Pick class i with the largest ${\theta^{(i)}}^{\top} x$

Slide credit: Andrew Ng

Logistic regression vs. SVMs

- n = number of features ($x \in \mathbb{R}^{n+1}$), m = number of training examples

1. **If $\boldsymbol{n}$ is large (relative to $\boldsymbol{m}$)**: $(n = 10{,}000, \mathrm{m} = 10 - 1000)$
   → Use logistic regression or SVM without a kernel ("linear kernel")
2. **If $\boldsymbol{n}$ is small, $\boldsymbol{m}$ is intermediate**: $(n = 1 - 1000, \mathrm{m} = 10 - 10{,}000)$
   → Use SVM with Gaussian kernel
3. **If $\boldsymbol{n}$ is small, $\boldsymbol{m}$ is large**: $(n = 1 - 1000, \mathrm{m} = 50{,}000+)$
   → Create/add more features, then use logistic regression of linear SVM

Neural network likely to work well for most of these case,
but slower to train

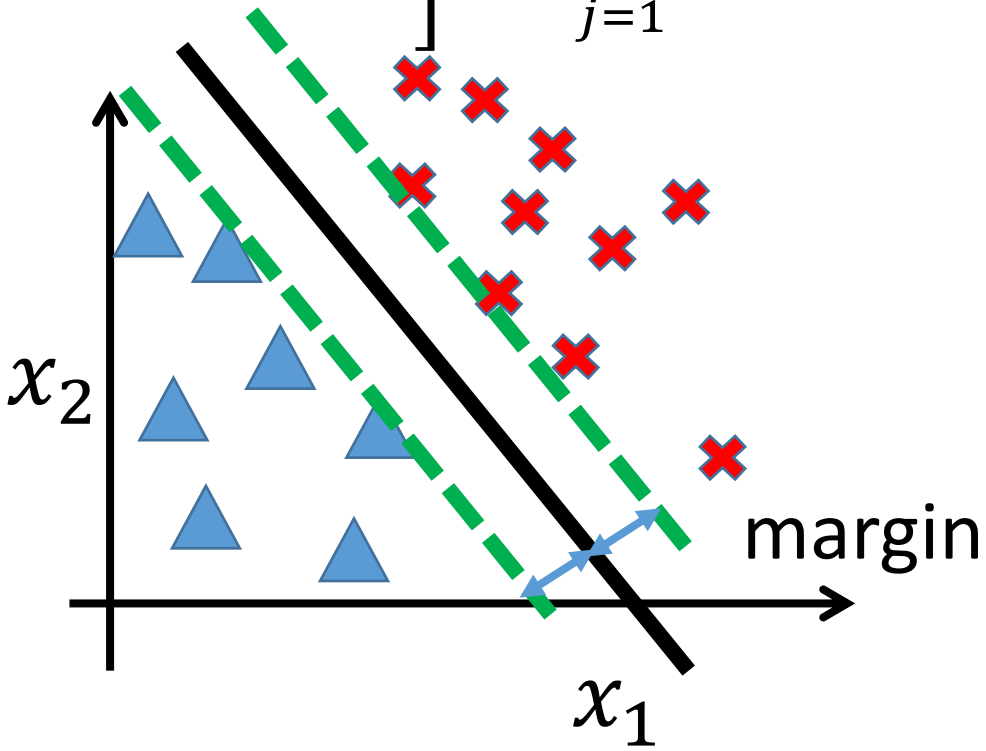
Slide credit: Andrew Ng

Things to remember

- **Cost function**

$$\min_{\theta} C\left[\sum_{i=1}^{m} y^{(i)}\ \mathrm{cost}_1\left(\theta^\top f^{(i)}\right) + \left(1 - y^{(i)}\right) \mathrm{cost}_0\left(\theta^\top f^{(i)}\right)\right] + \frac{1}{2}\sum_{j=1}^{m} \theta_j^2$$

- **Large margin classification**



- **Kernels**

- **Using an SVM**

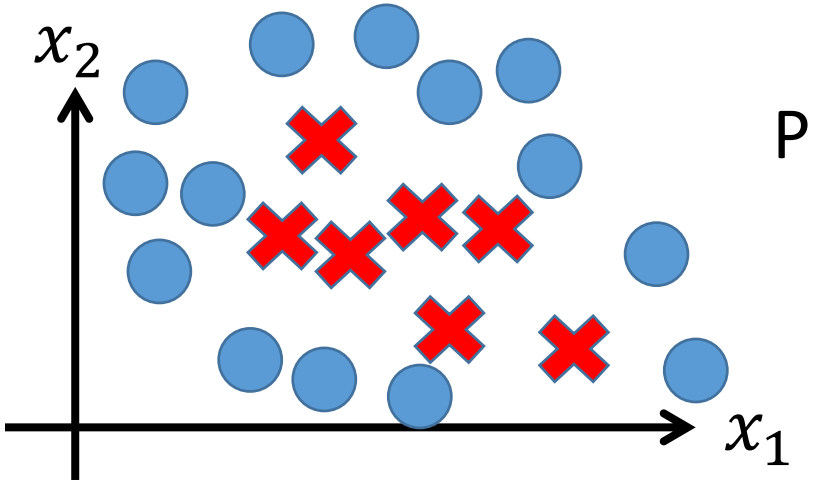
Neural Networks

- Why neural networks?
- Model representation
- Examples and intuitions
- Multi-class classification

Neural Networks

- **Why neural networks?**
- Model representation
- Examples and intuitions
- Multi-class classification

Non-linear classification



Predict $y = 1$ if

$$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2 + \cdots) \geq 0$$

x_1 = size

x_1 = #bedrooms

x_1 = #floors

x_1 = age

…

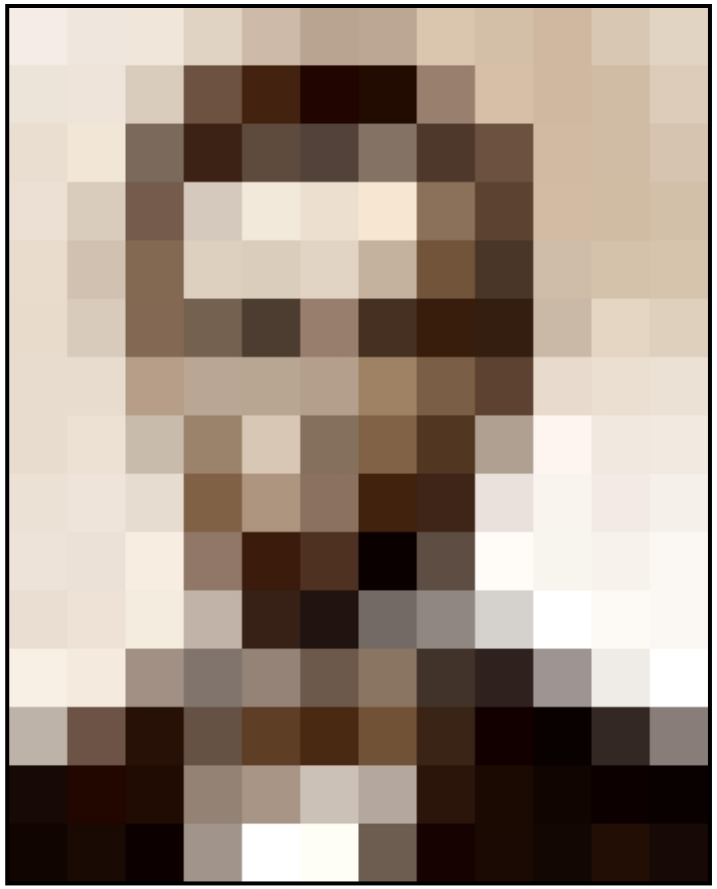
x_{100}

Quadratic features?

Cubic features?

Slide credit: Andrew Ng

What humans see



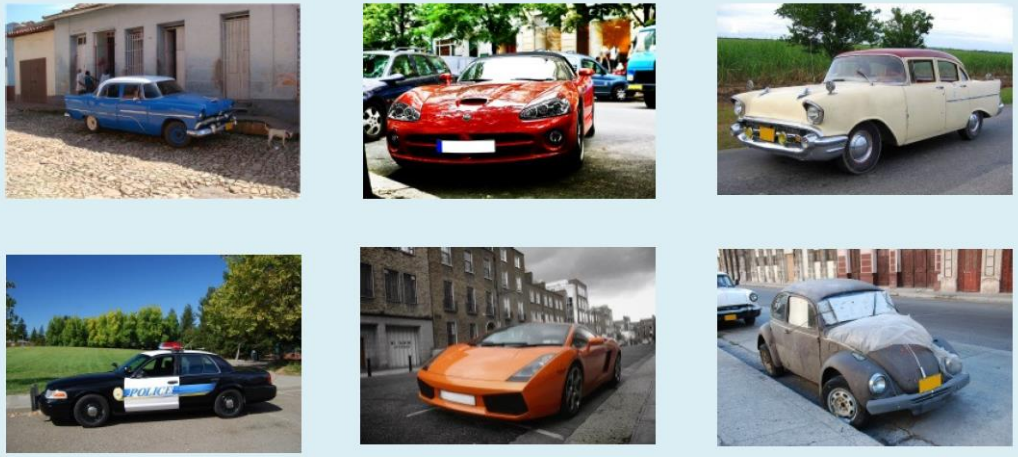
Slide credit: Larry Zitnick

What computers see

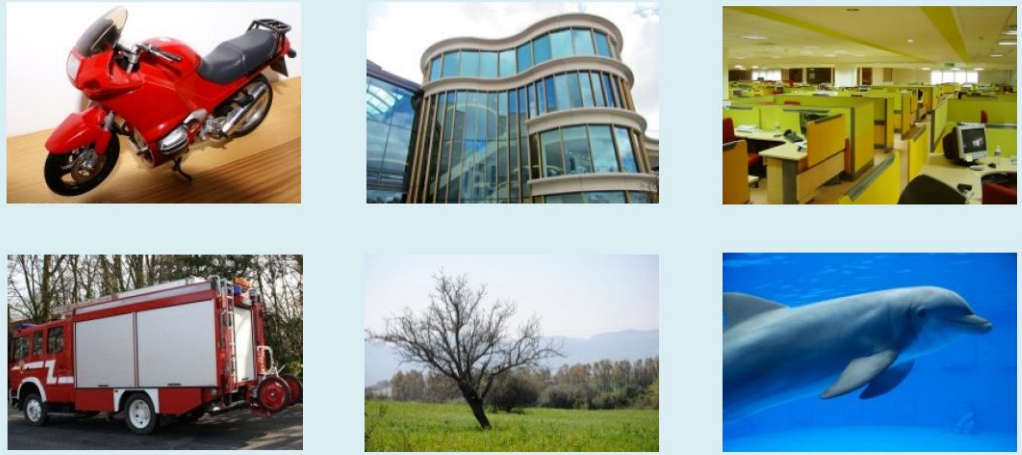
243	239	240	225	206	185	188	218	211	206	216	225
242	239	218	110	67	31	34	152	213	206	208	221
243	242	123	58	94	82	132	77	108	208	208	215
235	217	115	212	243	236	247	139	91	209	208	211
233	208	131	222	219	226	196	114	74	208	213	214
232	217	131	116	77	150	69	56	52	201	228	223
232	232	182	186	184	179	159	123	93	232	235	235
232	236	201	154	216	133	129	81	175	252	241	240
235	238	230	128	172	138	65	63	234	249	241	245
237	236	247	143	59	78	[illegible]	94	255	248	247	251
234	237	245	193	55	33	115	144	213	255	253	251
248	245	161	128	149	109	138	65	47	156	239	255
190	107	39	102	94	73	114	58	17	[illegible]	51	137
23	32	33	148	168	203	179	43	27	17	12	[illegible]
17	26	[illegible]	160	255	255	109	22	26	19	35	24

Slide credit: Larry Zitnick

Computer Vision: Car detection



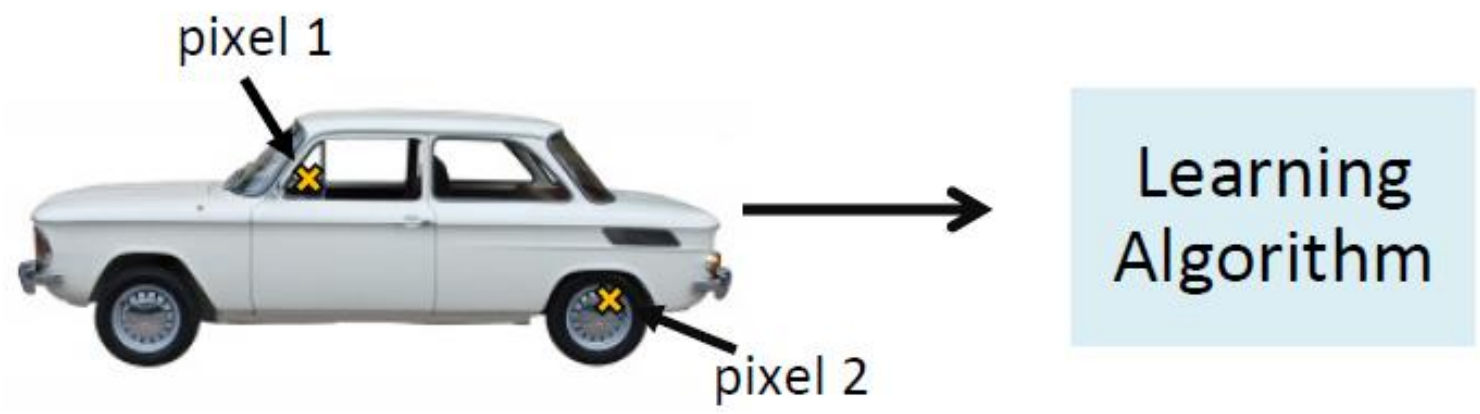
Cars



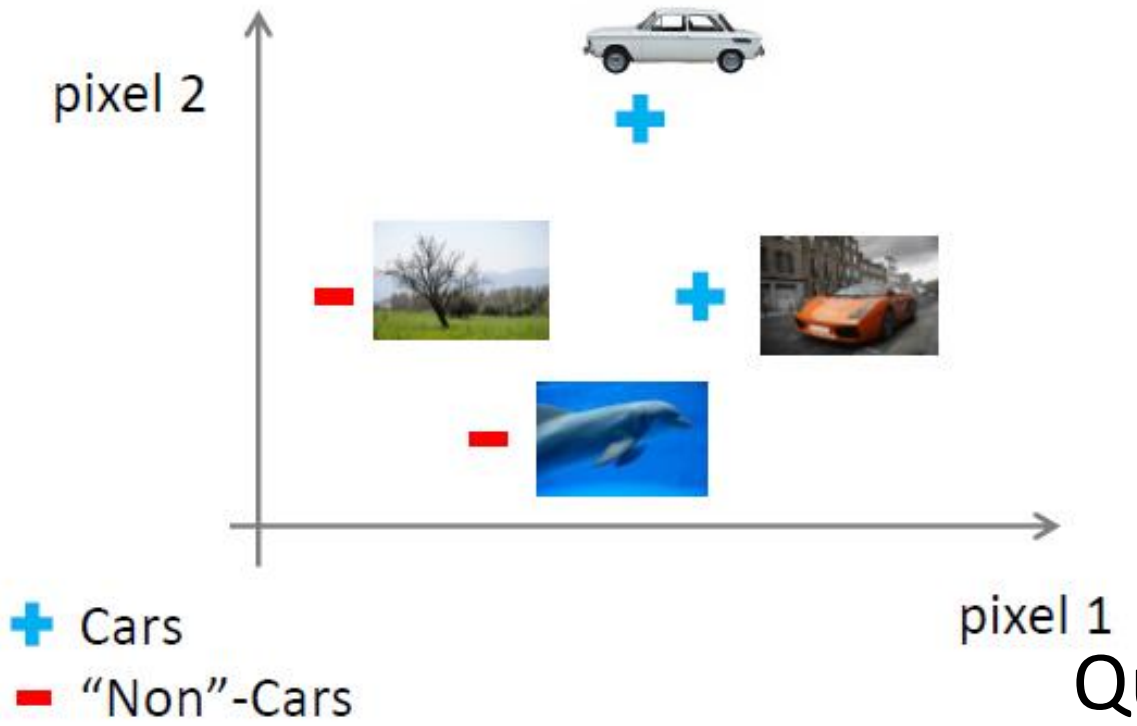
Not a car

Testing:

Slide credit: Andrew Ng



50x50 pixel images -> 2500 pixels



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

Quadratic features $(x_i x_j)$~
3 million features

Slide credit: Andrew Ng

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

Slide credit: Andrew Ng

An AI Timeline

Birth of AI

- Information Theory – digital signals
- Cybernetics – thinking machines
- The Turing Test
- Symbolic reasoning

Focus on Specific 'Intelligence'

- Expert Systems (knowledge)
- Neural networks make a comeback
- Optical character recognition
- Speech recognition

Focus on Specific Problems

- Machine learning
- Deep learning – pattern analysis / classification
  - Big data: large databases
  - Fast processors to crunch data
  - High-speed networks

1950 1960 1970 1980 1990 2000 2010 2020

AI Winter

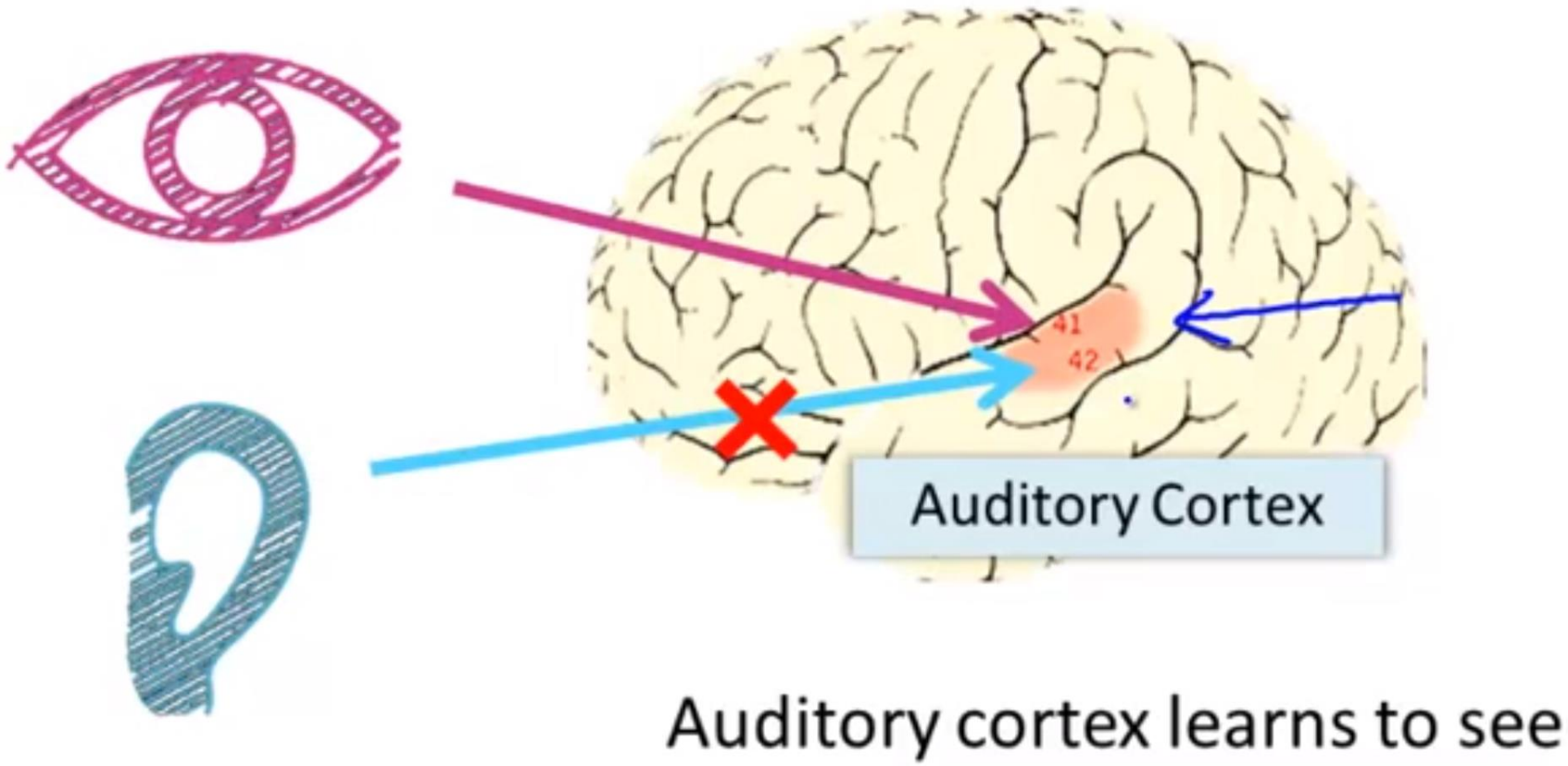
- Limited computer processing power
- Limited database capacity
- Limited networking capabilities
- Real-world problems are complicated
  - Image processing / face recognition
  - Combinatorial explosion

AI Winter II

- Disappointing results
- Collapse of dedicated hardware vendors

https://www.slideshare.net/dlavenda/ai-and-productivity

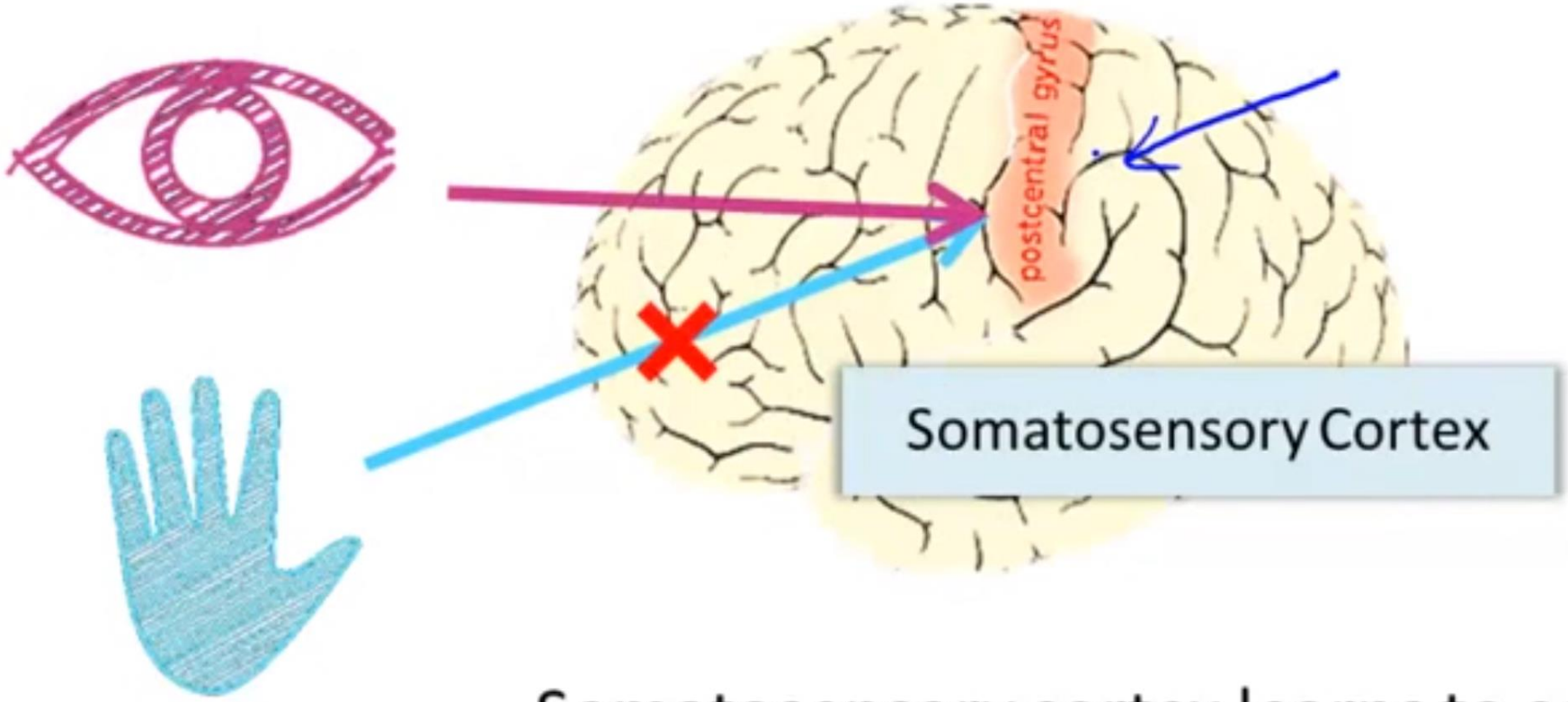
The “one learning algorithm” hypothesis



[Roe et al. 1992]

Slide credit: Andrew Ng

The “one learning algorithm” hypothesis



Somatosensory cortex learns to see

[Metin and Frost 1989]

Slide credit: Andrew Ng

Sensor representations in the brain

Seeing with your tongue

Human echolocation (sonar)

Haptic belt: Direction sense

Implanting a 3rd eye

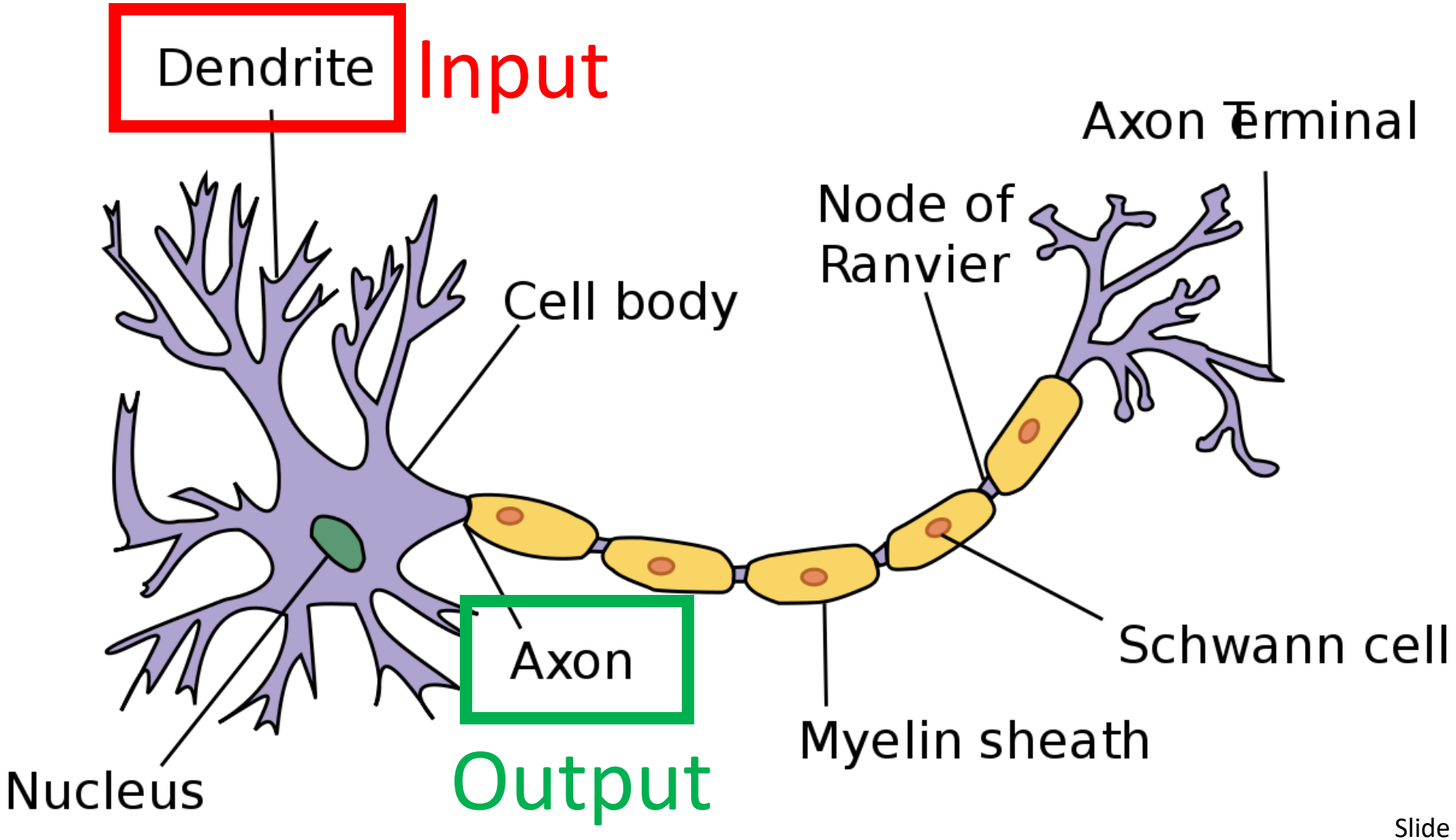
[BrainPort; Welsh & Blasch, 1997; Nagel et al., 2005; Constantine-Paton & Law, 2009]

Slide credit: Andrew Ng

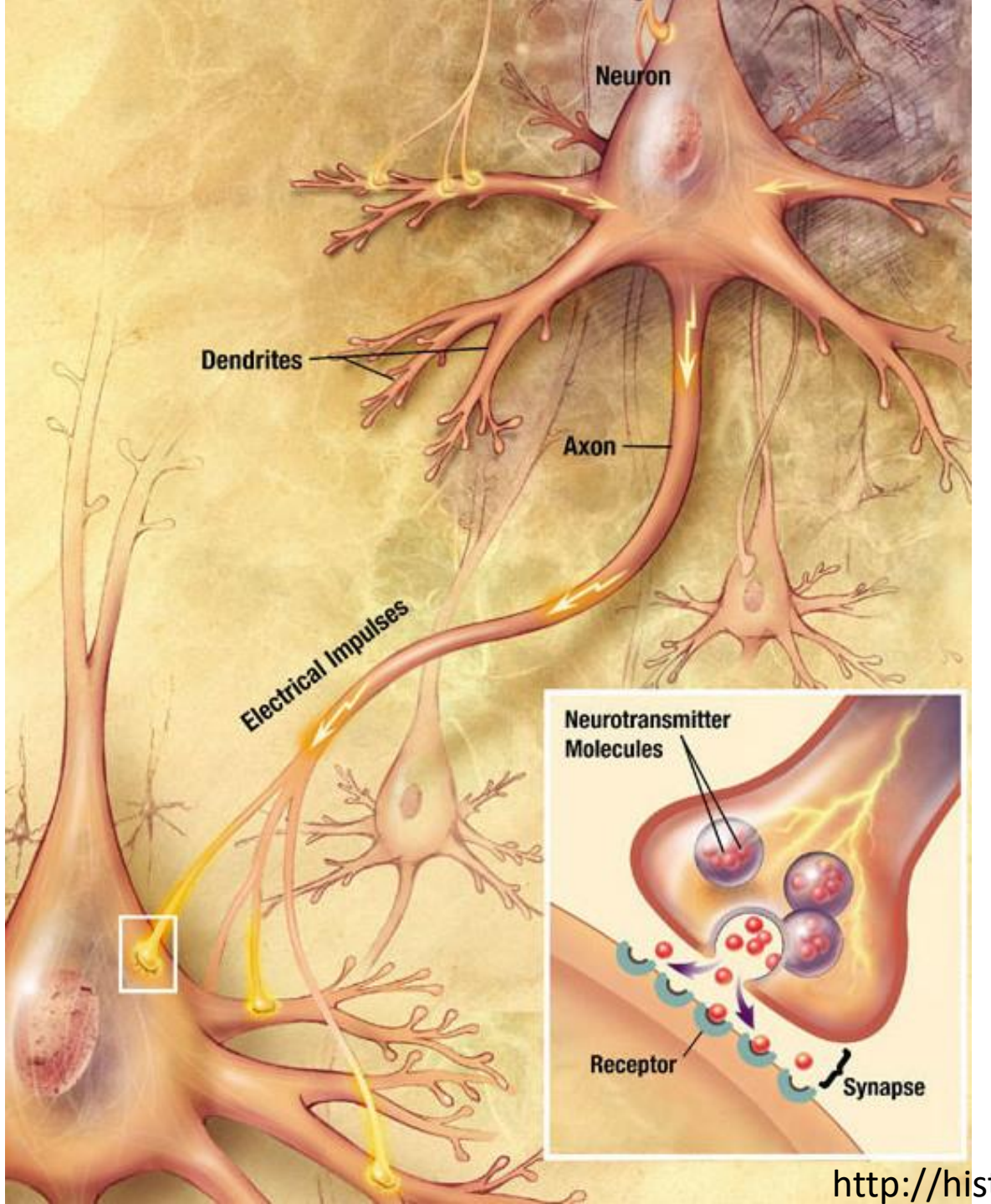
Neural Networks

- Why neural networks?
- **Model representation**
- Examples and intuitions
- Multi-class classification

A single neuron in the brain

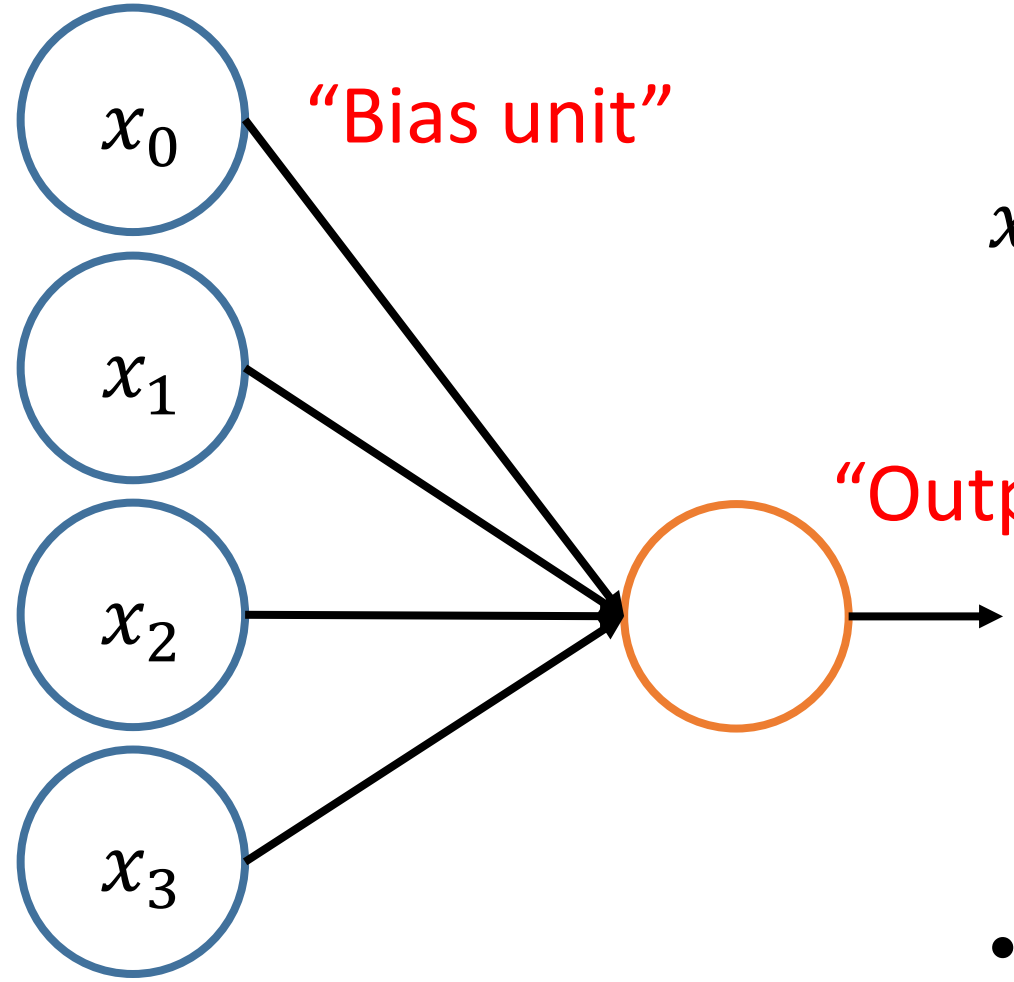


Slide credit: Andrew Ng



http://histoweb.co.za/082/082img002.html

An artificial neuron: Logistic unit



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

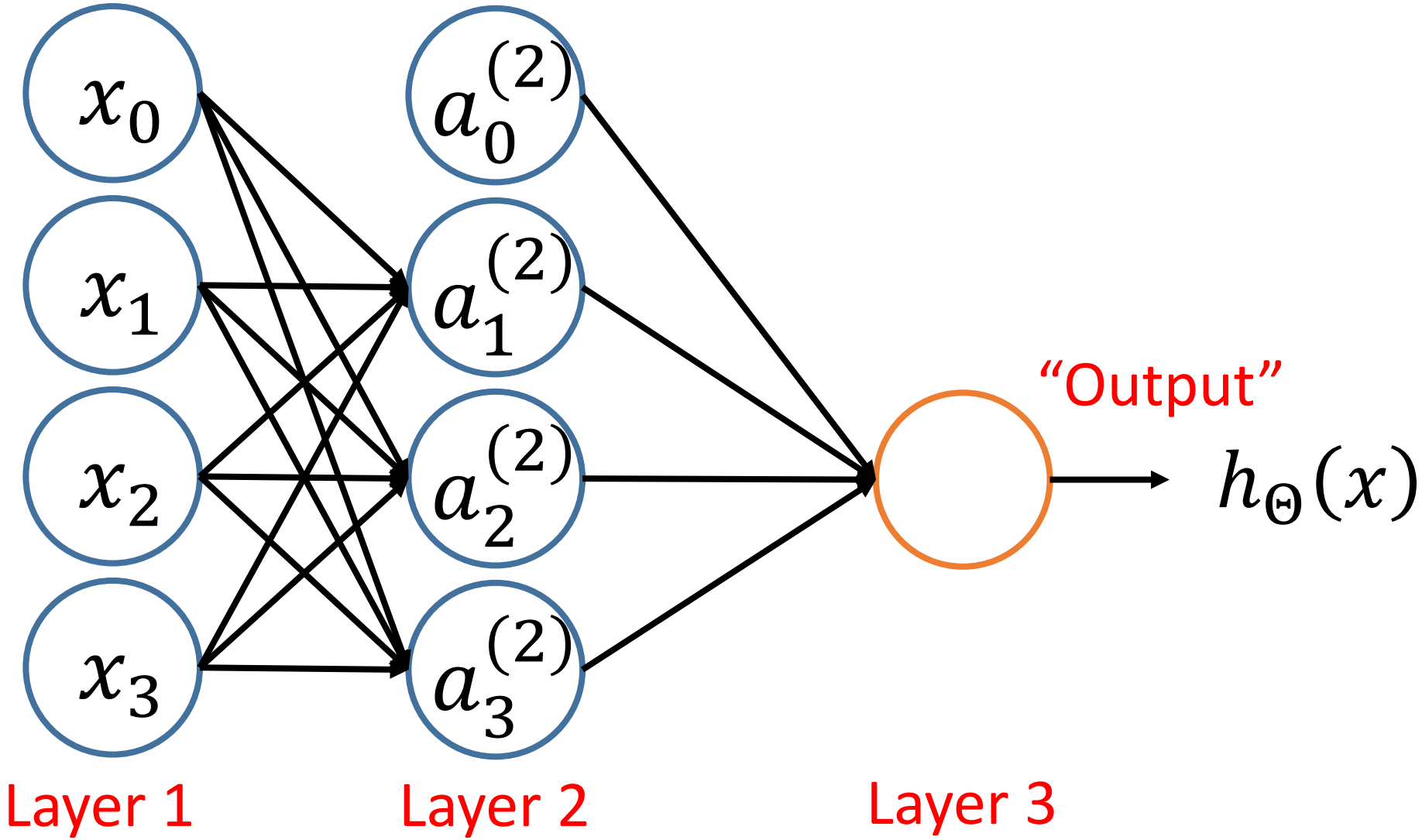
"Weights"
"Parameters"

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

- Sigmoid (logistic) activation function

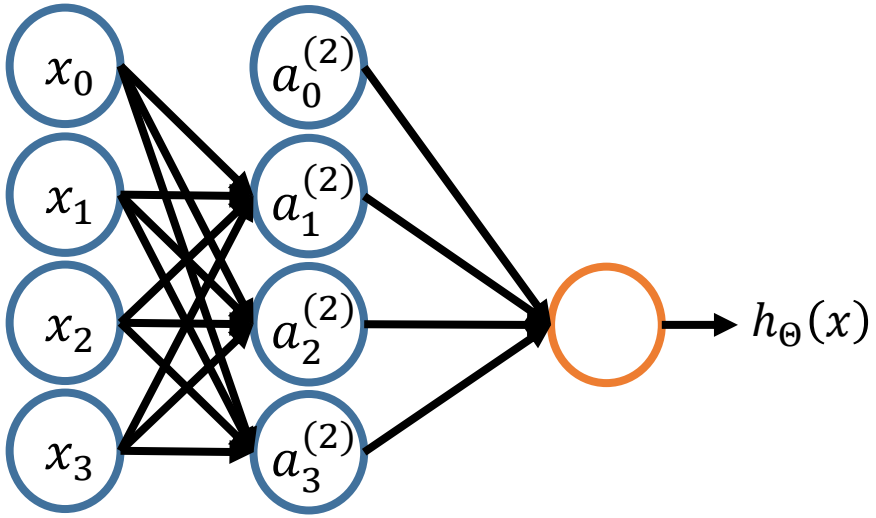
Slide credit: Andrew Ng

Neural network



Slide credit: Andrew Ng

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 $j+1$

s_j unit in layer j

s_{j+1} units in layer $j+1$

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

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

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

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

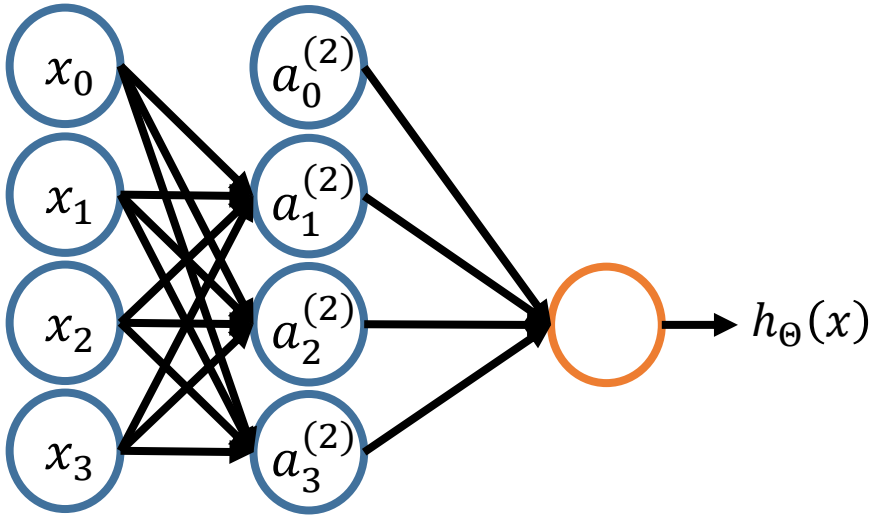
Size of $\Theta^{(j)}$?

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

Slide credit: Andrew Ng

Neural network

"Pre-activation"



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

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

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

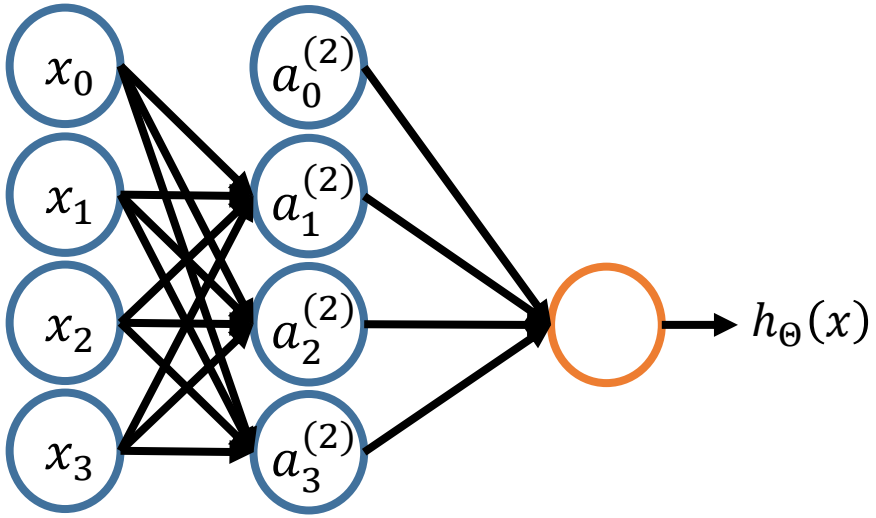
$$a_3^{(2)} = g\left(\Theta_{30}^{(1)} x_0 + \Theta_{31}^{(1)} x_1 + \Theta_{32}^{(1)} x_2 + \Theta_{33}^{(1)} x_3\right) = g(z_3^{(2)})$$

$$h_\Theta(x) = g\left(\Theta_{10}^{(2)} a_0^{(2)} + \Theta_{11}^{(1)} a_1^{(2)} + \Theta_{12}^{(1)} a_2^{(2)} + \Theta_{13}^{(1)} a_3^{(2)}\right) = g(z^{(3)})$$

Slide credit: Andrew Ng

Neural network

"Pre-activation"



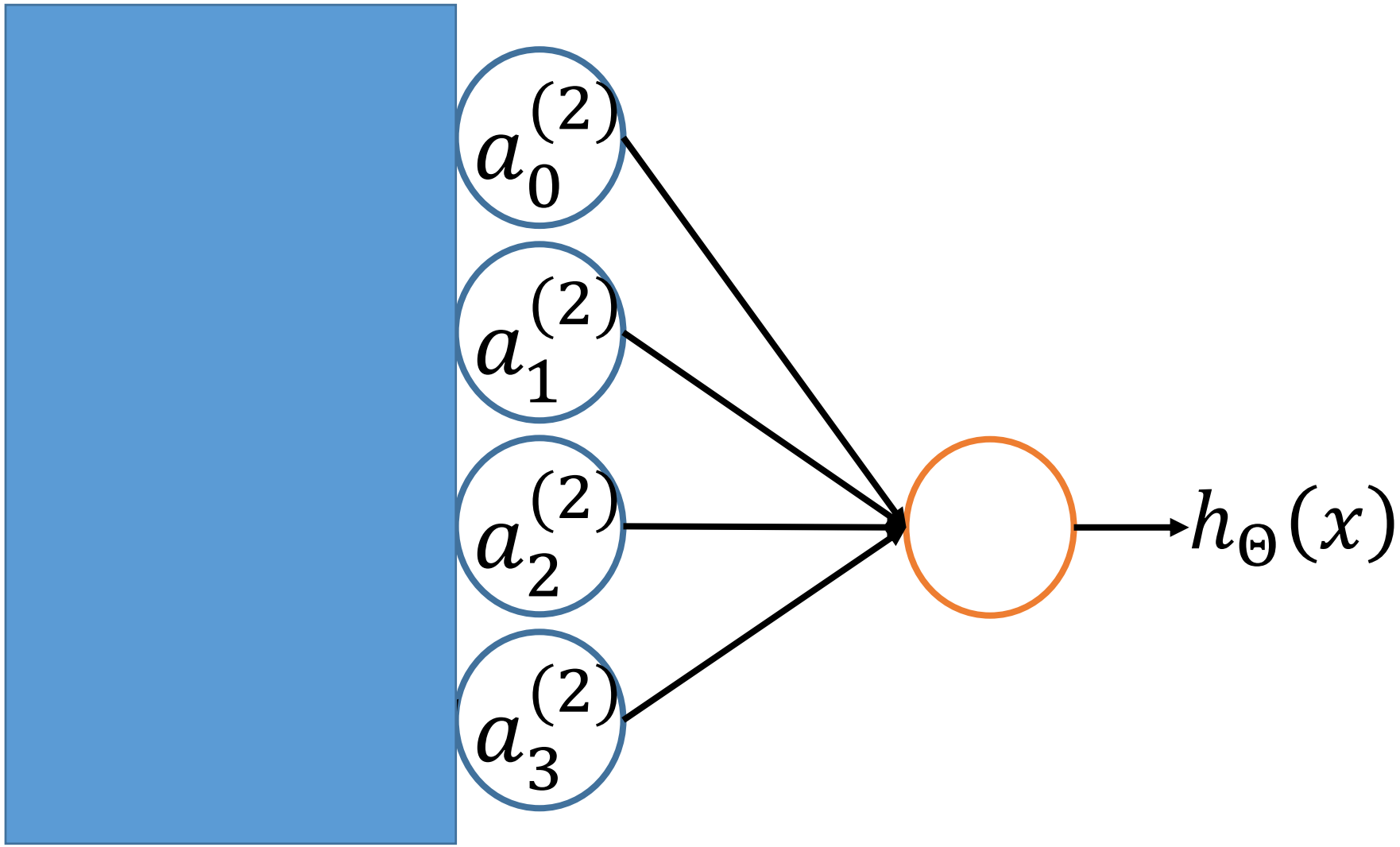
$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad z^{(2)} = \begin{bmatrix} z_1^{(2)} \\ z_2^{(2)} \\ z_3^{(2)} \end{bmatrix}$$

$$a_1^{(2)} = g(z_1^{(2)})$$
$$a_2^{(2)} = g(z_2^{(2)})$$
$$a_3^{(2)} = g(z_3^{(2)})$$
$$h_\Theta(x) = g(z^{(3)})$$

$$z^{(2)} = \Theta^{(1)} x = \Theta^{(1)} a^{(1)}$$
$$a^{(2)} = g(z^{(2)})$$
$$\text{Add } a_0^{(2)} = 1$$
$$z^{(3)} = \Theta^{(2)} a^{(2)}$$
$$h_\Theta(x) = a^{(3)} = g(z^{(3)})$$

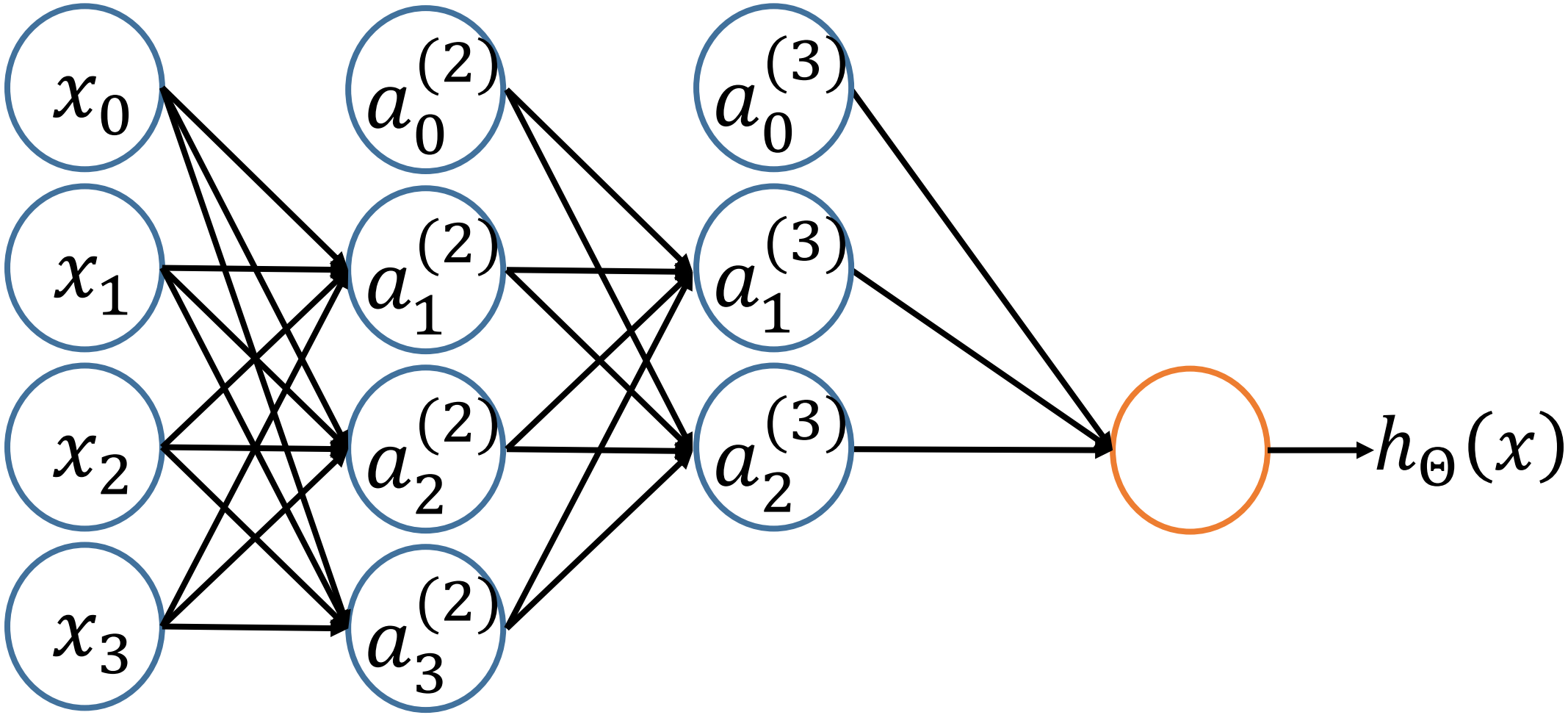
Slide credit: Andrew Ng

Neural network learning its own features



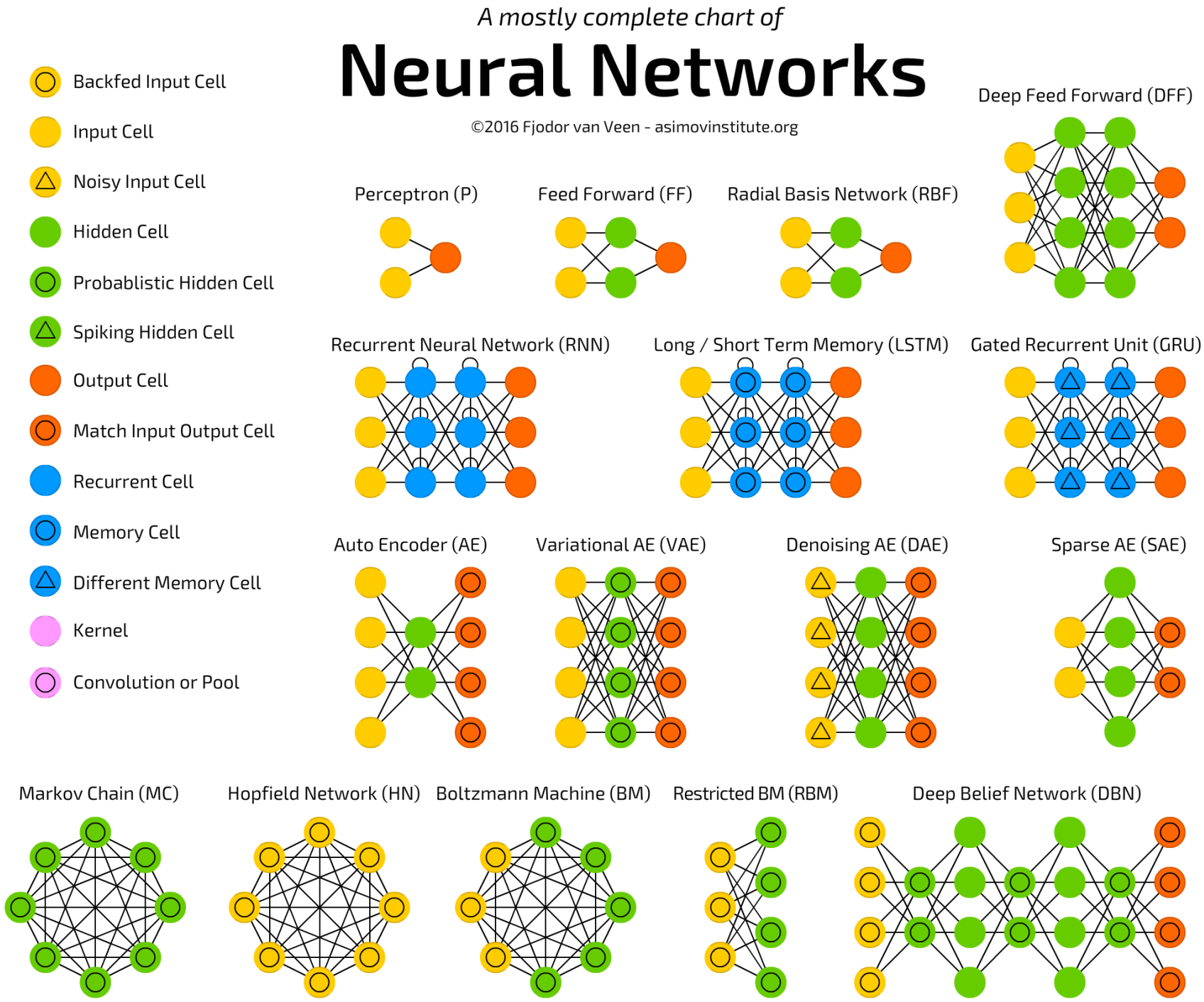
Slide credit: Andrew Ng

Other network architectures

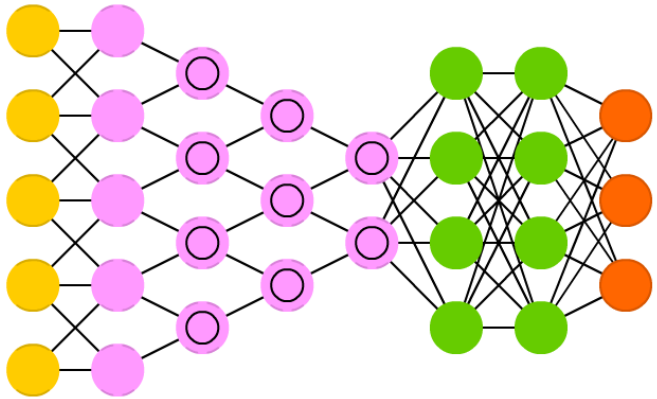


Slide credit: Andrew Ng

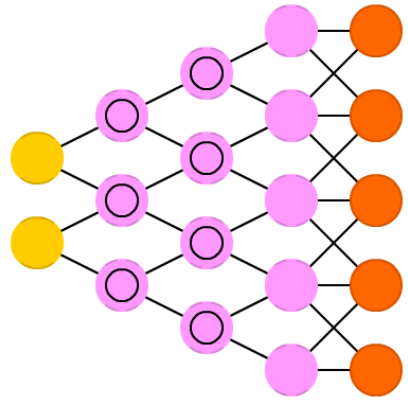
- Archi


A mostly complete chart of
Neural Networks
©2016 Fjodor van Veen - asimovinstitute.org
Backfed Input Cell
Input Cell
Noisy Input Cell
Hidden Cell
Probablistic Hidden Cell
Spiking Hidden Cell
Output Cell
Match Input Output Cell
Recurrent Cell
Memory Cell
Different Memory Cell
Kernel
Convolution or Pool
Perceptron (P)
Feed Forward (FF)
Radial Basis Network (RBF)
Deep Feed Forward (DFF)
Recurrent Neural Network (RNN)
Long / Short Term Memory (LSTM)
Gated Recurrent Unit (GRU)
Auto Encoder (AE)
Variational AE (VAE)
Denoising AE (DAE)
Sparse AE (SAE)
Markov Chain (MC)
Hopfield Network (HN)
Boltzmann Machine (BM)
Restricted BM (RBM)
Deep Belief Network (DBN)

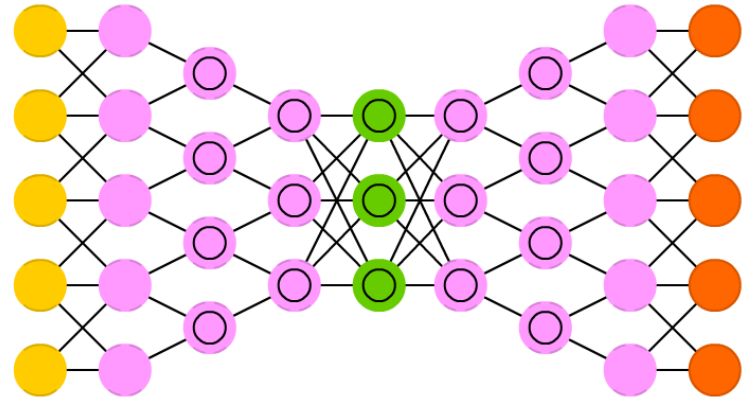
Deep Convolutional Network (DCN)



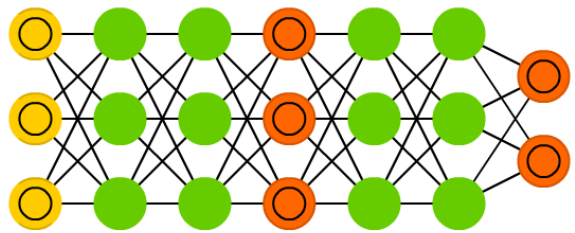
Deconvolutional Network (DN)



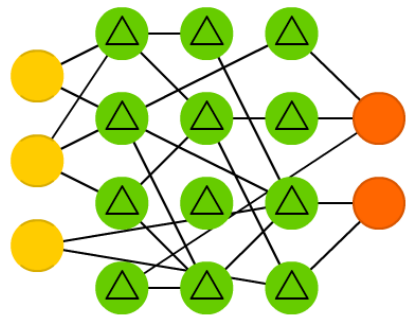
Deep Convolutional Inverse Graphics Network (DCIGN)



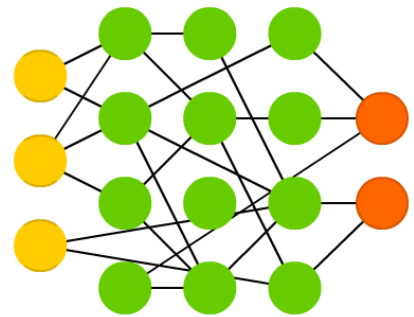
Generative Adversarial Network (GAN)



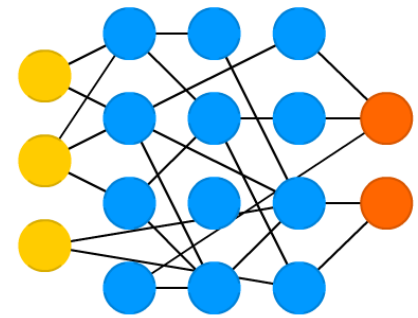
Liquid State Machine (LSM)



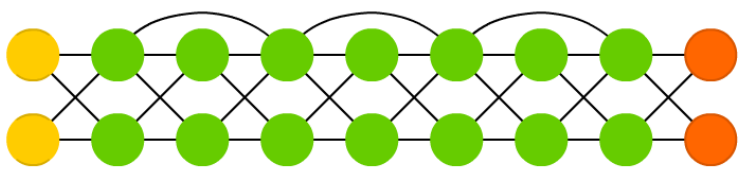
Extreme Learning Machine (ELM)



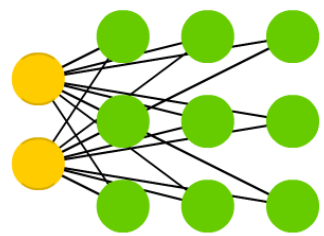
Echo State Network (ESN)



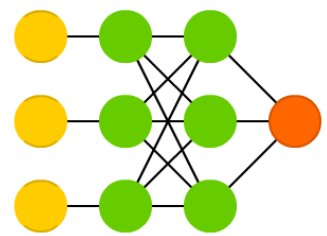
Deep Residual Network (DRN)



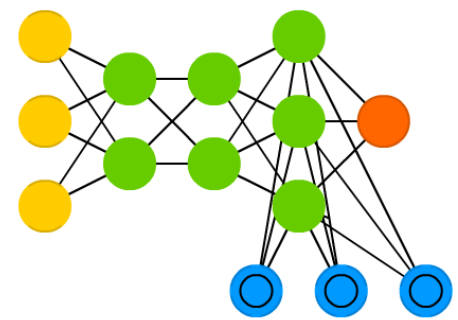
Kohonen Network (KN)



Support Vector Machine (SVM)



Neural Turing Machine (NTM)

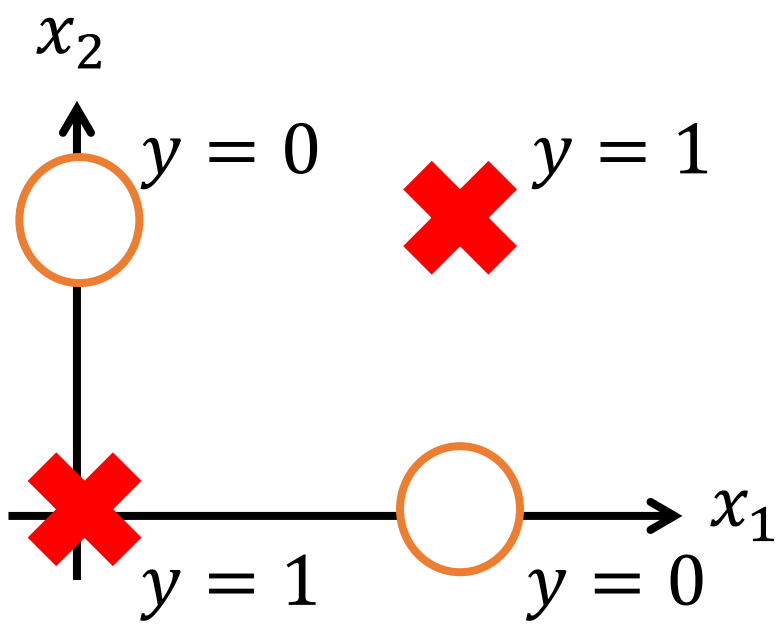


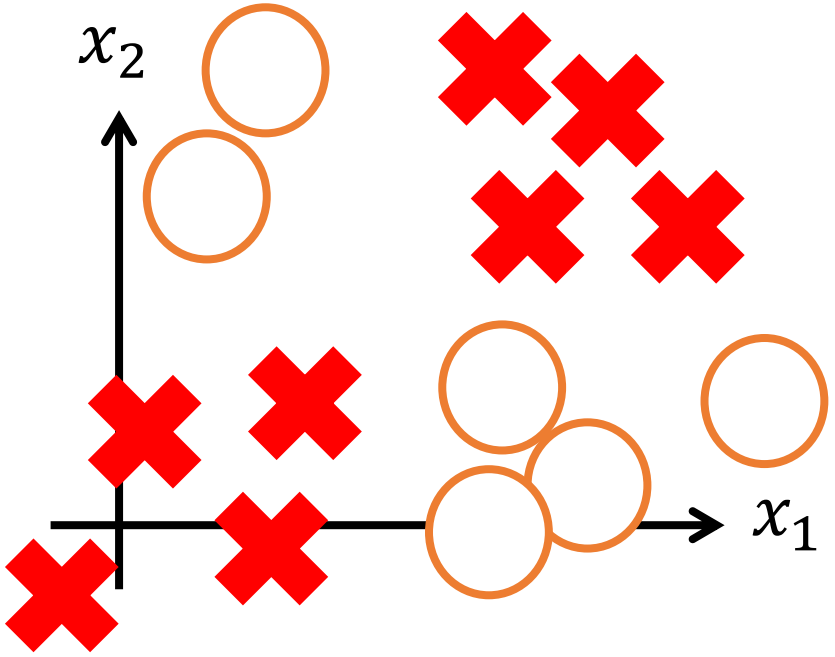
Neural Networks

- Why neural networks?
- Model representation
- **Examples and intuitions**
- Multi-class classification

Non-linear classification example: XOR/XNOR

- x_1, x_2 are binary (0 or 1)
- $y = XOR(x_1, x_2)$

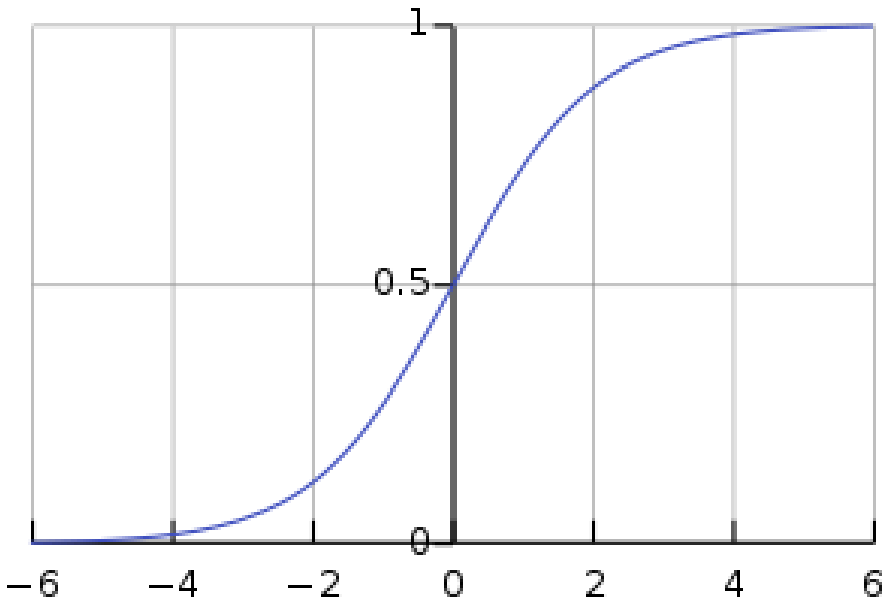


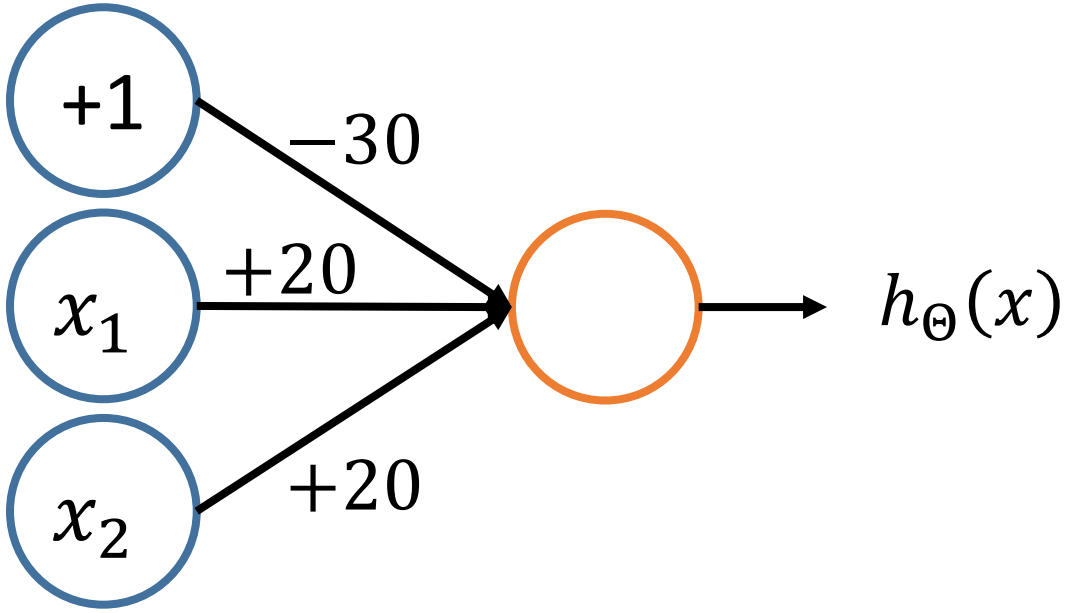


Slide credit: Andrew Ng

Simple example: AND

- $x_1, x_2 \in \{0, 1\}$
- $y = x_1 \text{ AND } x_2$





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	$g(10) \approx 1$

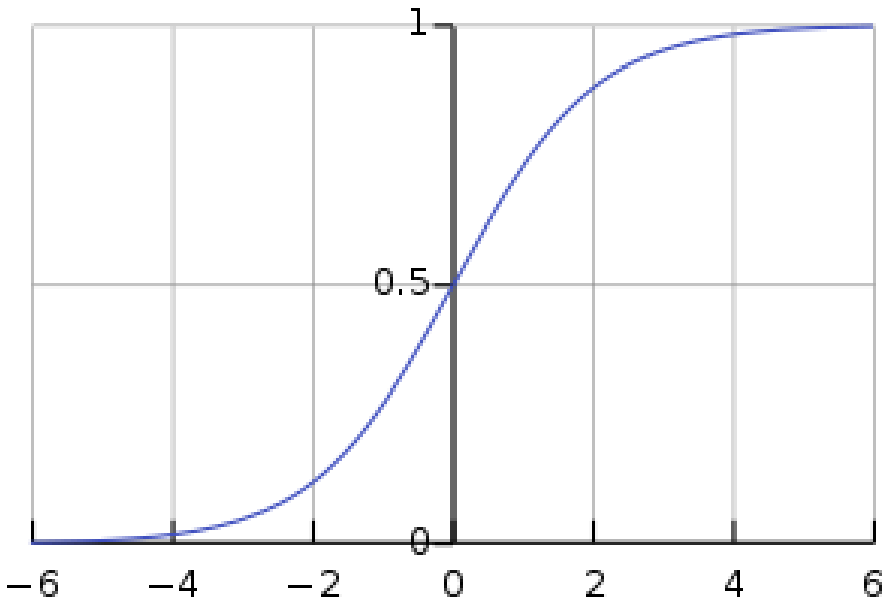
$$h_\Theta(x) = g(-30 + 20x_1 + 20x_2)$$

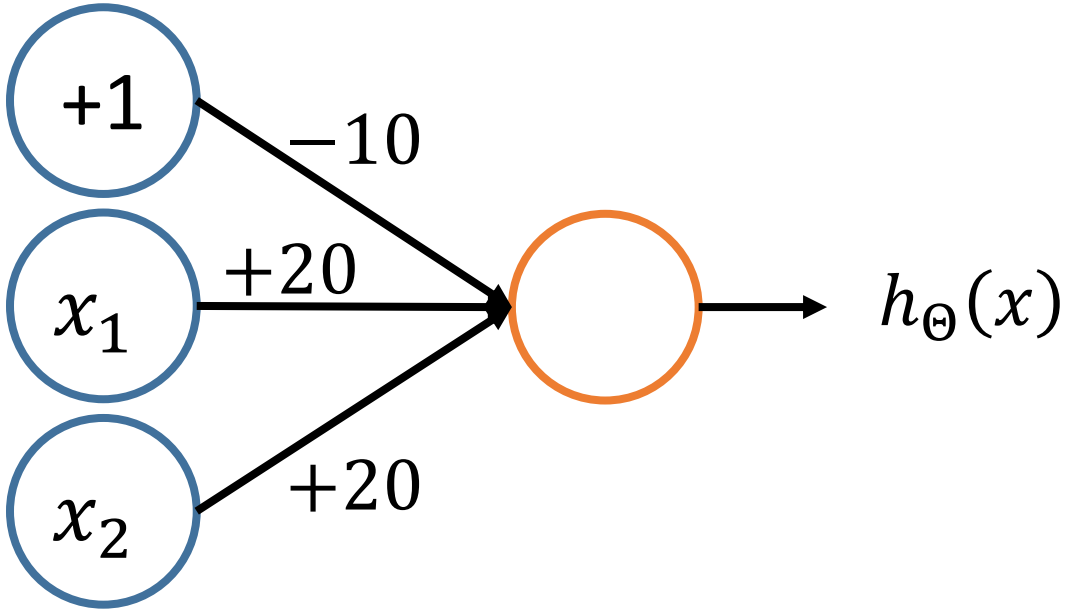
$$h_\Theta(x) \approx x_1 \text{ AND } \mathrm{x}_2$$

Slide credit: Andrew Ng

Simple example: OR

- $x_1, x_2 \in \{0, 1\}$
- $y = x_1 \text{ AND } x_2$





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

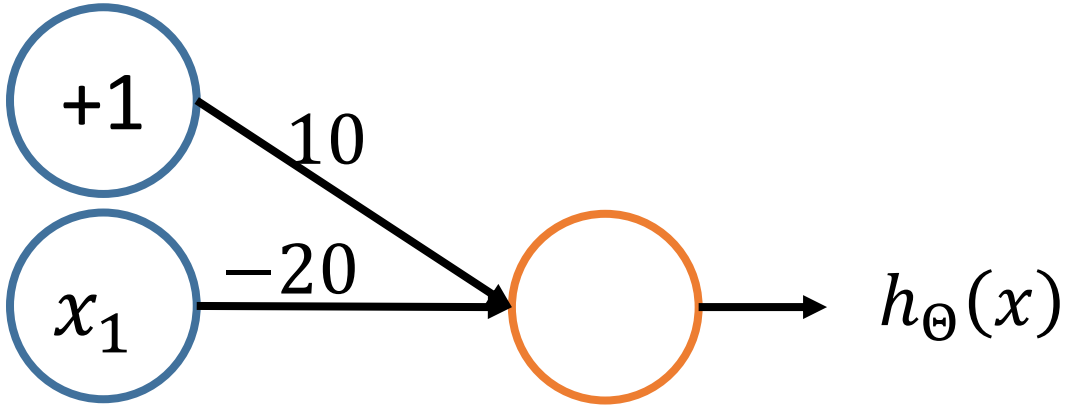
$$h_\Theta(x) = g(-10 + 20x_1 + 20x_2)$$

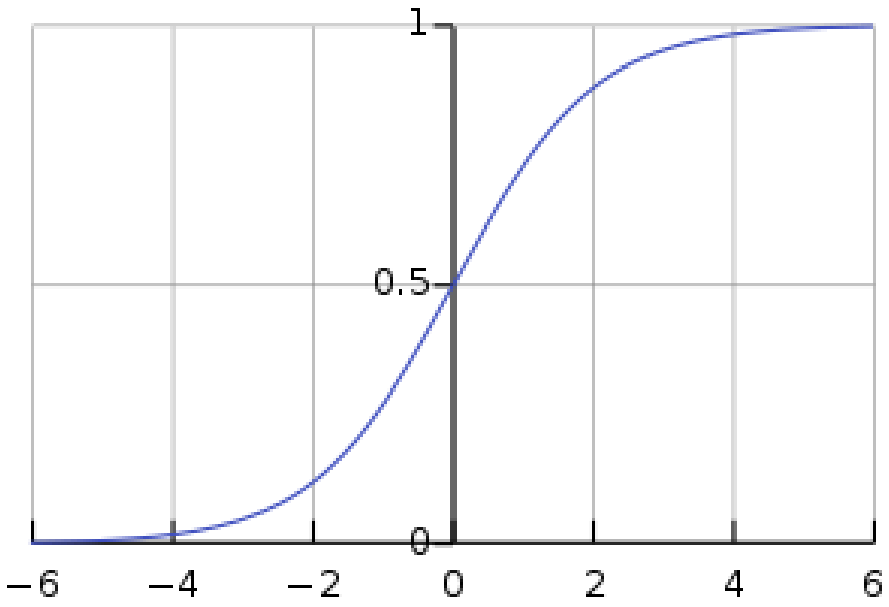
$$h_\Theta(x) \approx x_1 \text{ OR } x_2$$

Slide credit: Andrew Ng

Simple example: NOT

- $x_1, x_2 \in \{0, 1\}$
- $y = x_1 \text{ AND } x_2$





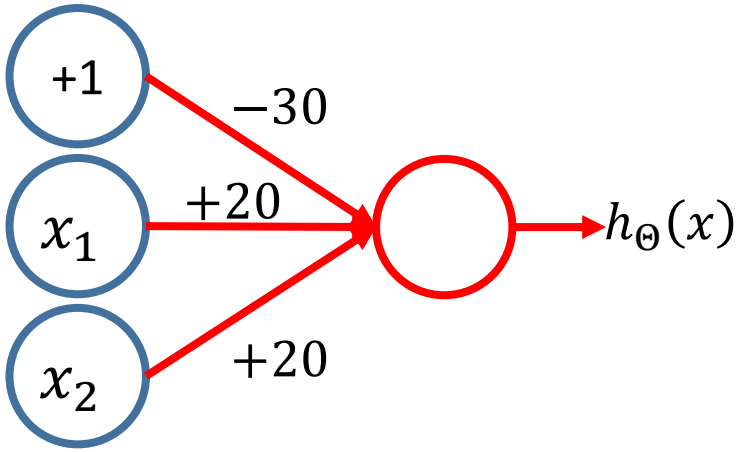
x_1	$h_\Theta(x)$
0	$g(10) \approx 1$
1	$g(-10) \approx 0$

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

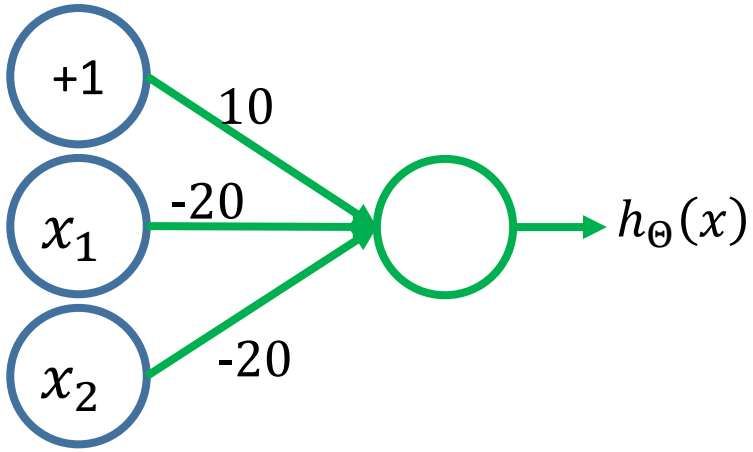
$$h_\Theta(x) \approx \text{NOT } x_1$$

Slide credit: Andrew Ng

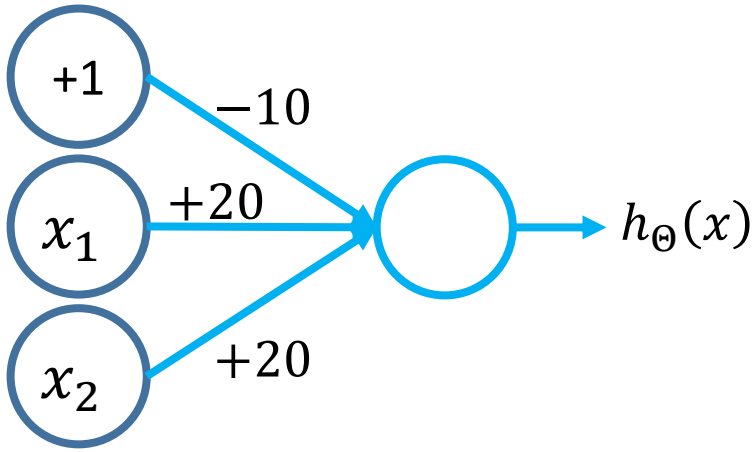
Putting it together: x_1 XNOR x_2



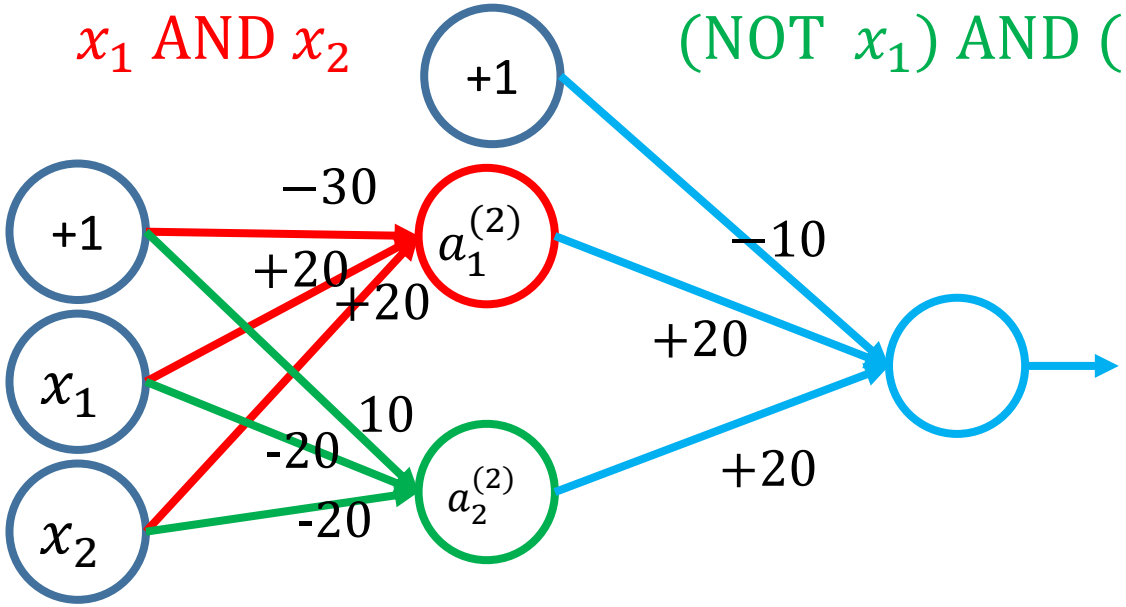
x_1 AND x_2



(NOT x_1) AND (NOT x_2)



x_1 OR 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

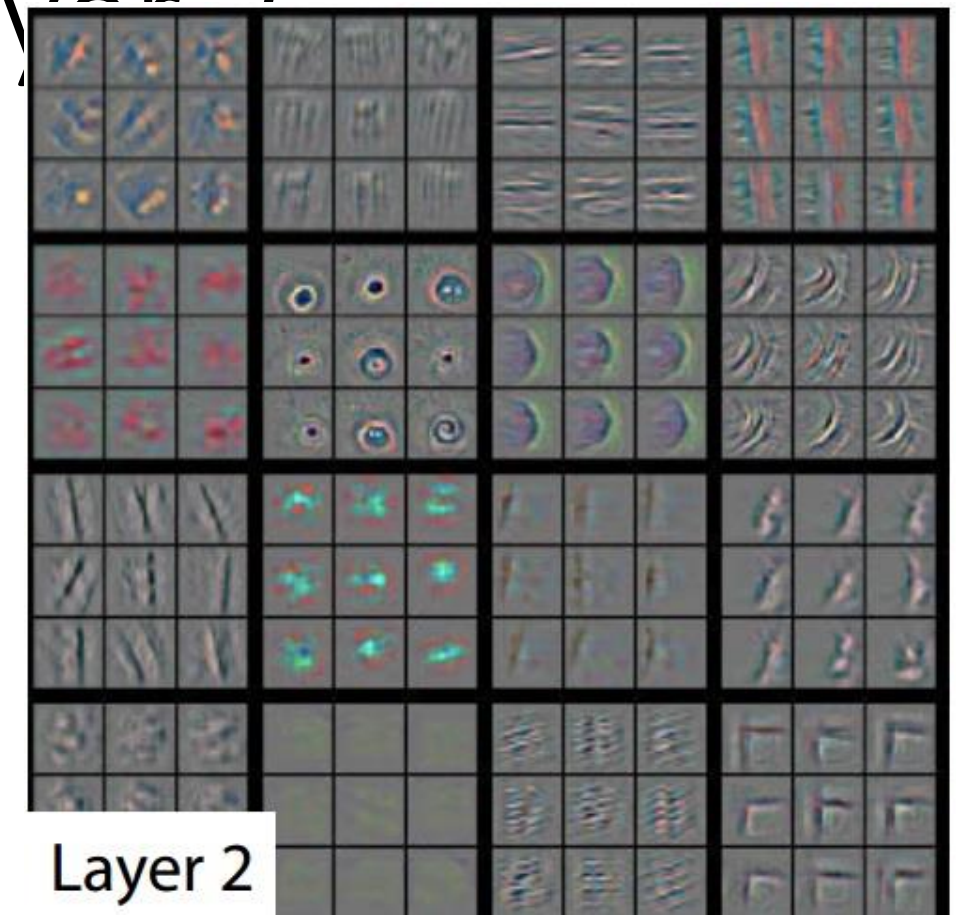
Slide credit: Andrew Ng

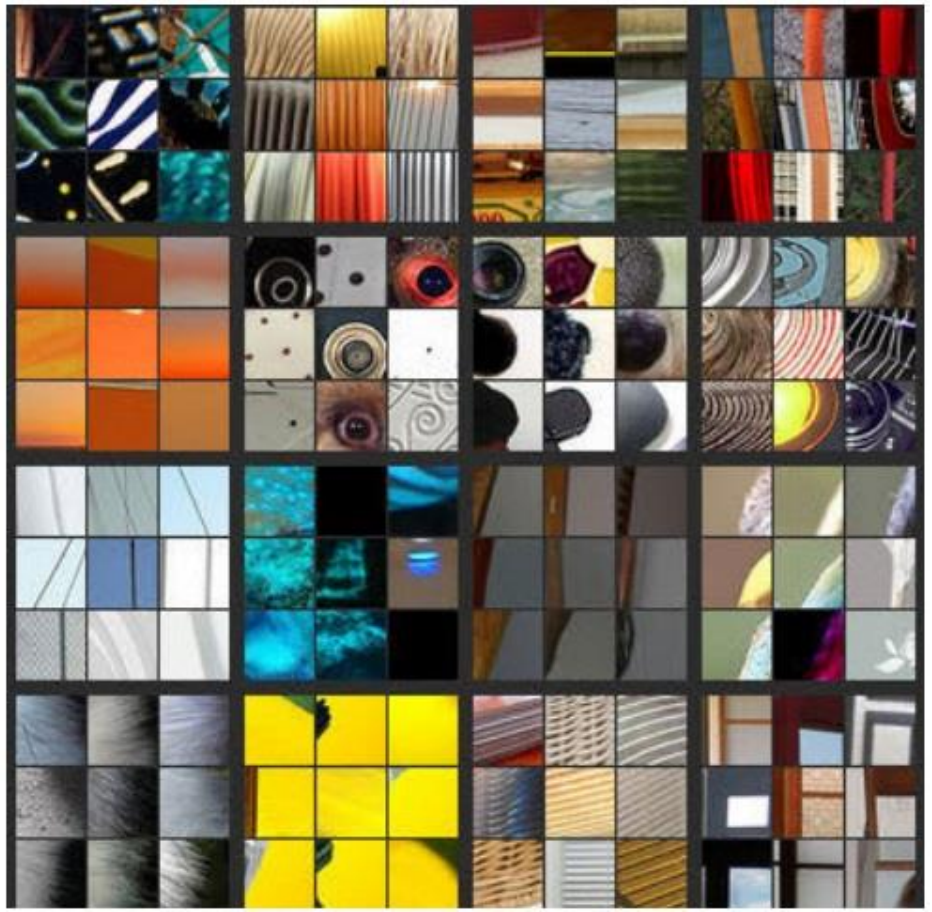
Layer 1

Layer 1

Visualizing and Understanding Convolutional Networks [Zeiler and Fergus, ECCV 2014]

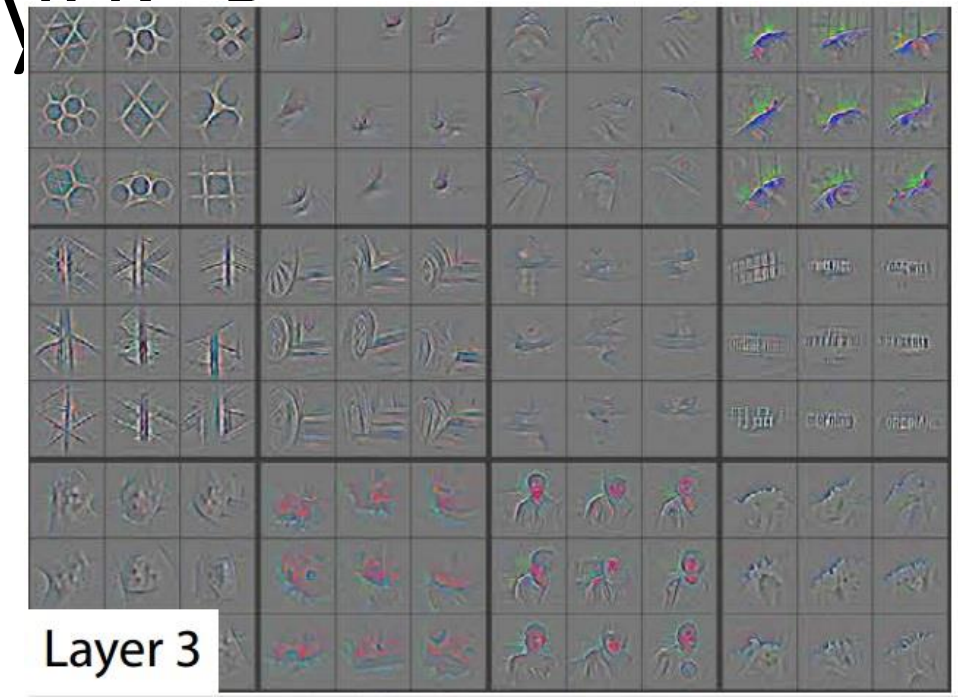
Layer 2





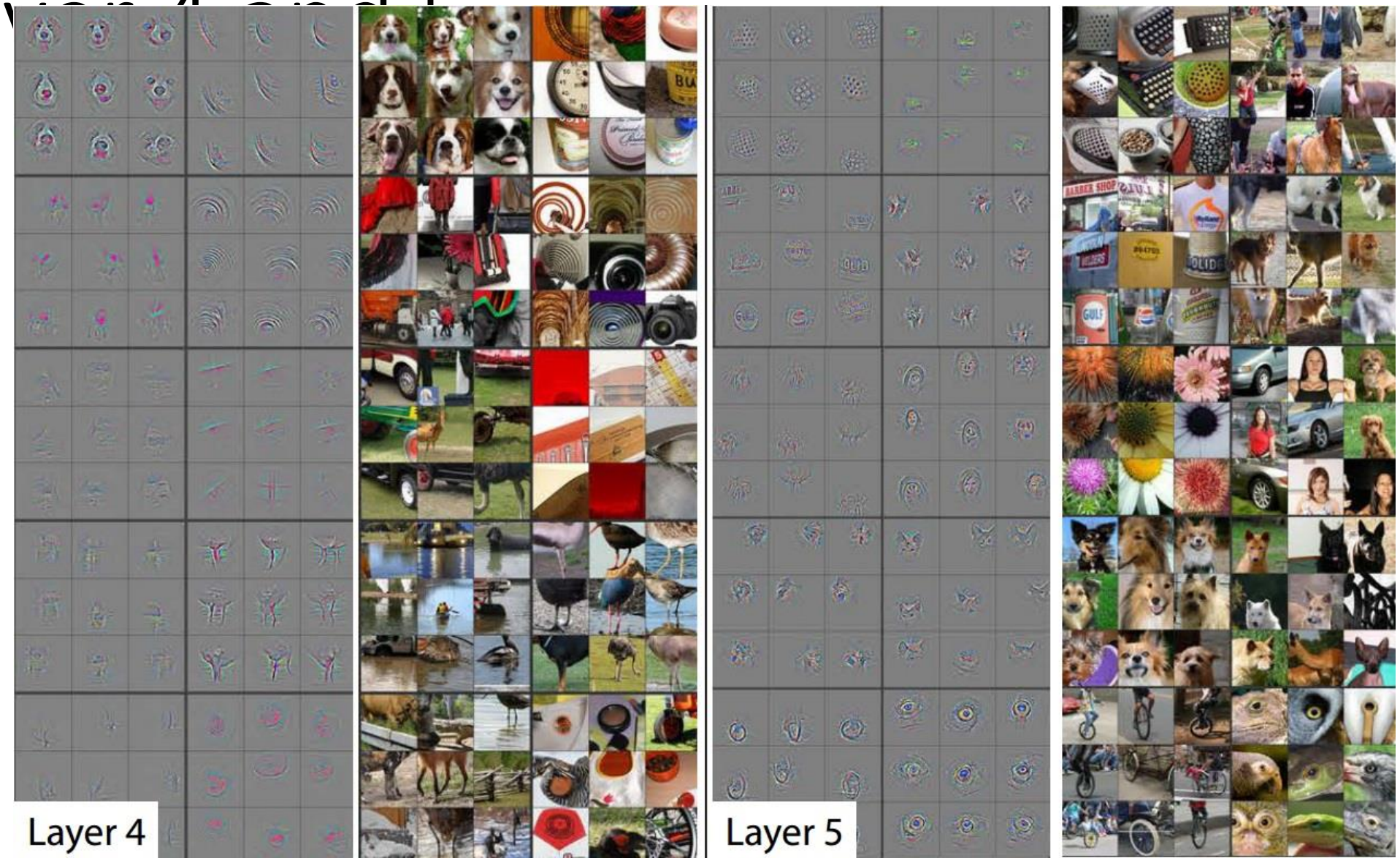
Visualizing and Understanding Convolutional Networks [Zeiler and Fergus, ECCV 2014]

Layer 3



Visualizing and Understanding Convolutional Networks [Zeiler and Fergus, ECCV 2014]

Layer 4 and 5



Visualizing and Understanding Convolutional Networks [Zeiler and Fergus, ECCV 2014]

Neural Networks

- Why neural networks?
- Model representation
- Examples and intuitions
- **Multi-class classification**

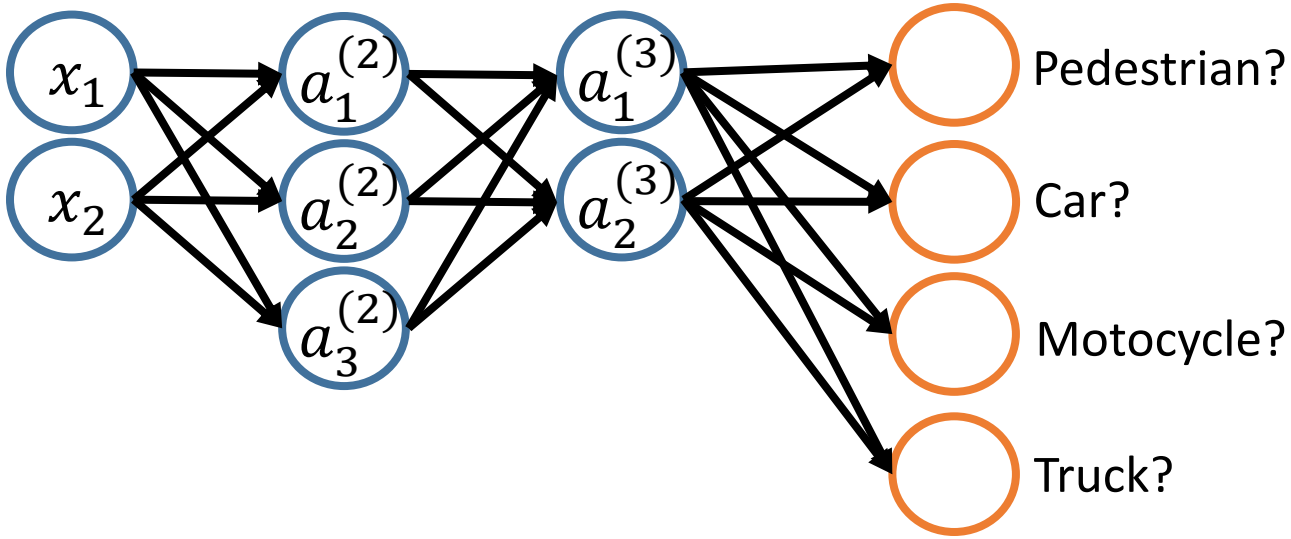
Multiple output units: One-vs-all

Pedestrian

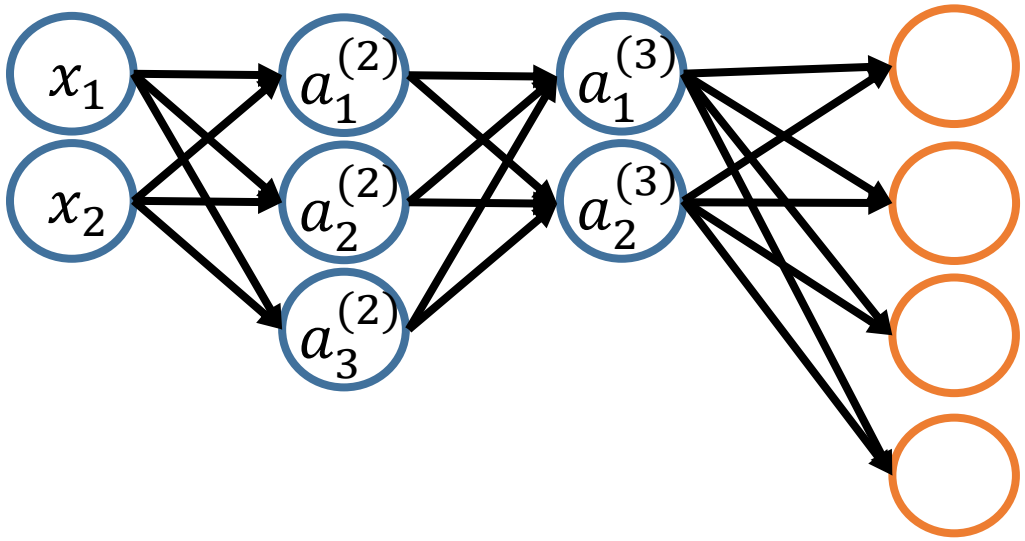
Car

Motorcycle

Truck



Multiple output units: One-vs-all



$$h_\Theta(x) \in R^4$$

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

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

Things to remember

- Why neural networks?
- Model representation
- Examples and intuitions
- Multi-class classification