



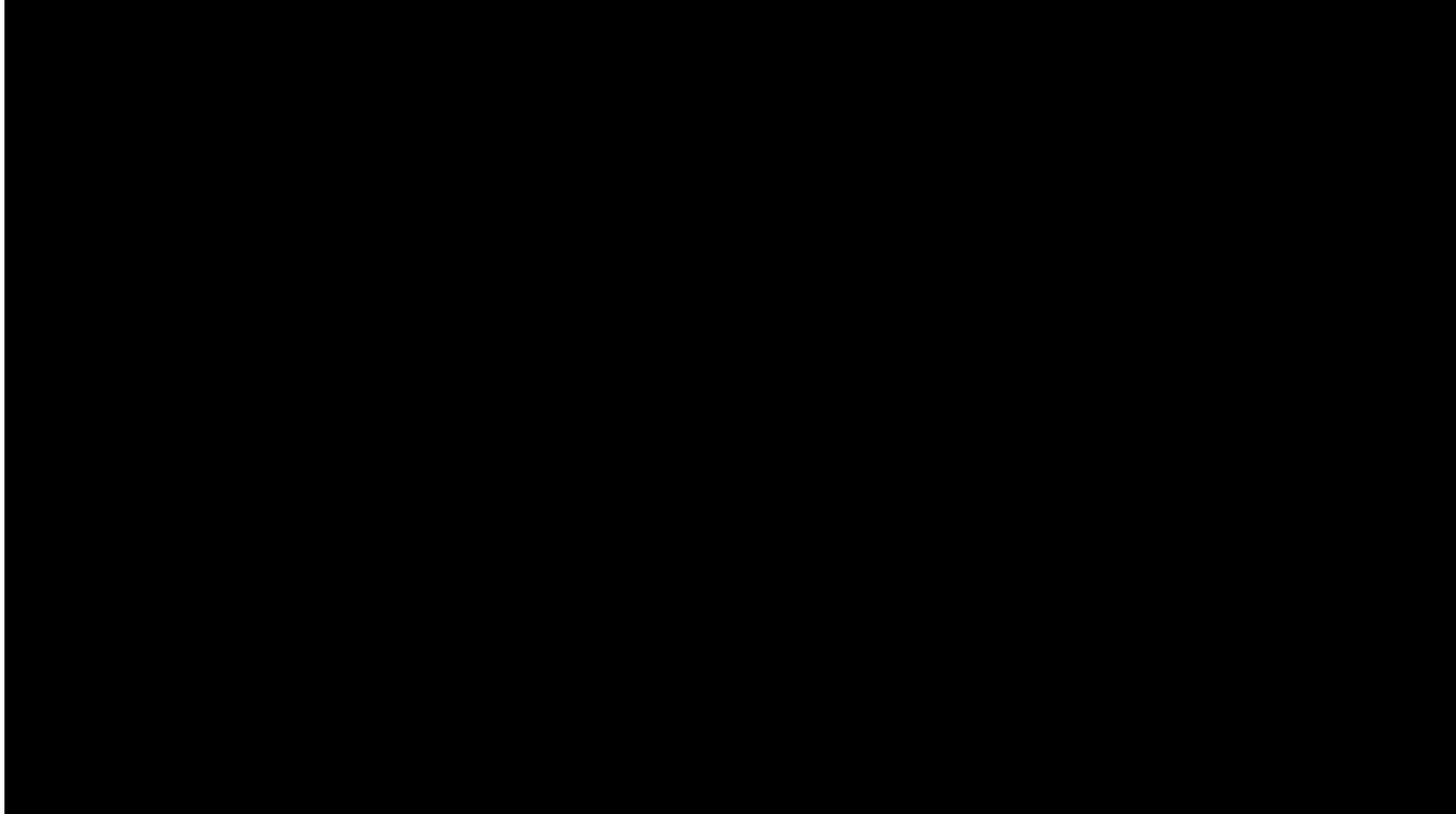
deeplearning.ai

# Face recognition

---

What is face  
recognition?

# Face recognition



# Face verification vs. face recognition

## → Verification

- Input image, name/ID
- Output whether the input image is that of the claimed person

1:1

99%

99.9  
~~~

## → Recognition

- Has a database of K persons
- Get an input image
- Output ID if the image is any of the K persons (or “not recognized”)

1:K

K=100 ←



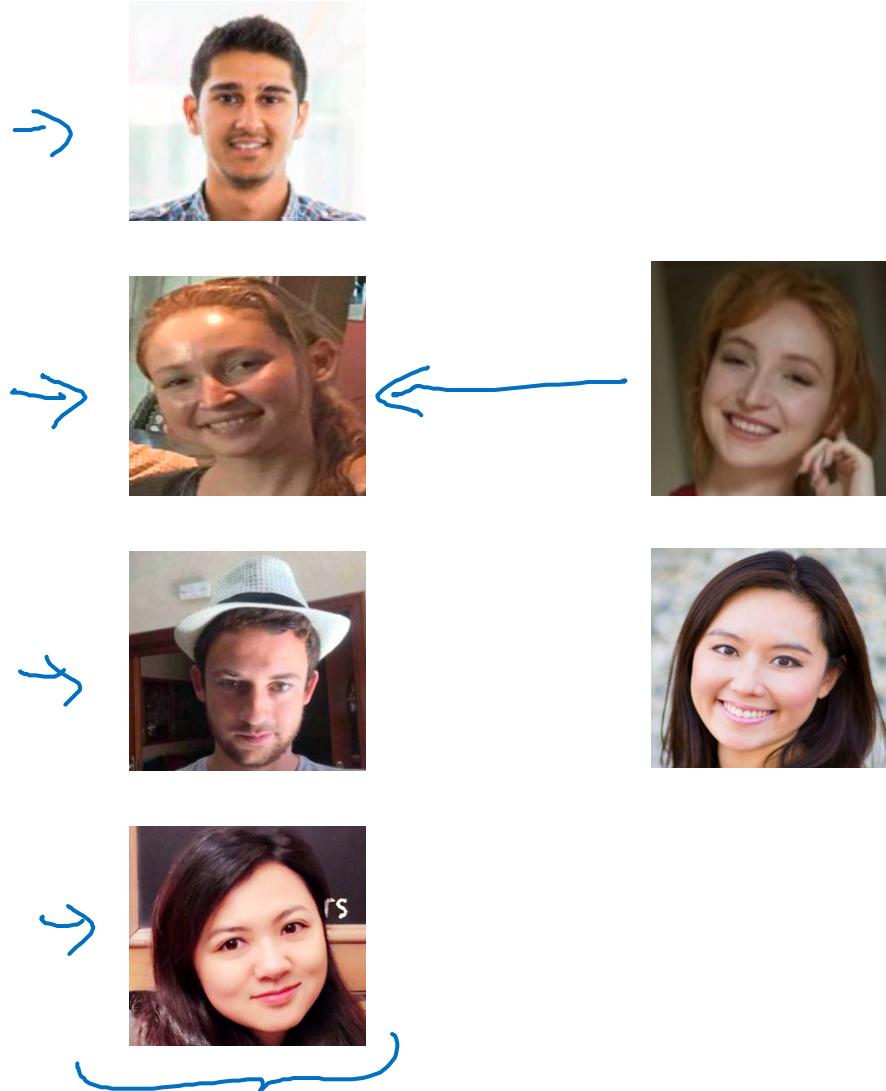
deeplearning.ai

# Face recognition

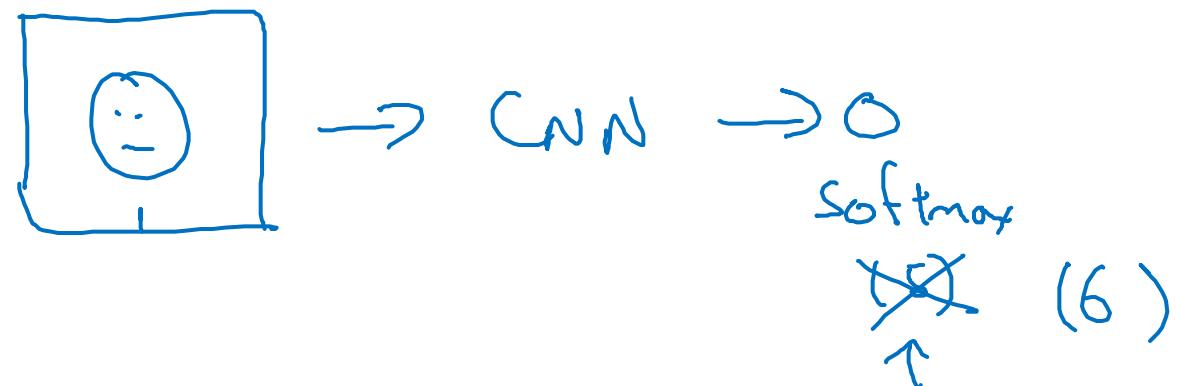
---

# One-shot learning

# One-shot learning



Learning from one example to recognize the person again



# Learning a “similarity” function

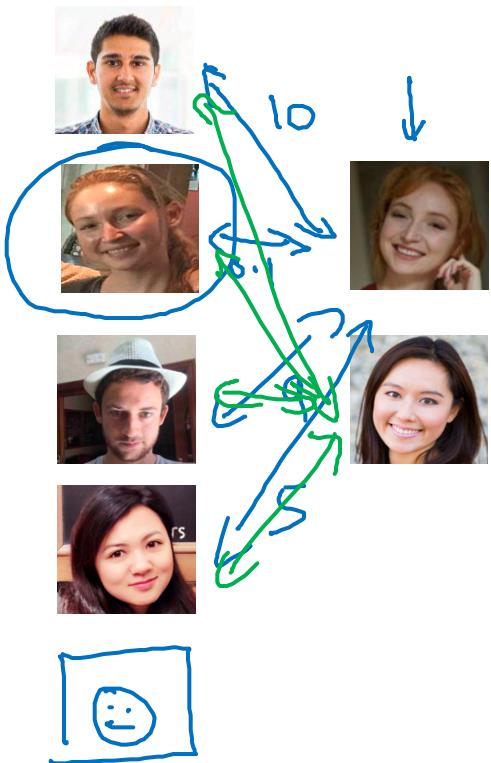
→  $d(\underline{\text{img1}}, \underline{\text{img2}})$  = degree of difference between images

If  $d(\text{img1}, \text{img2}) \leq \tau$

$$> \tau$$

"Some"  
"Different"

## Verification.



$d(\text{ing}1, \text{ing}2)$



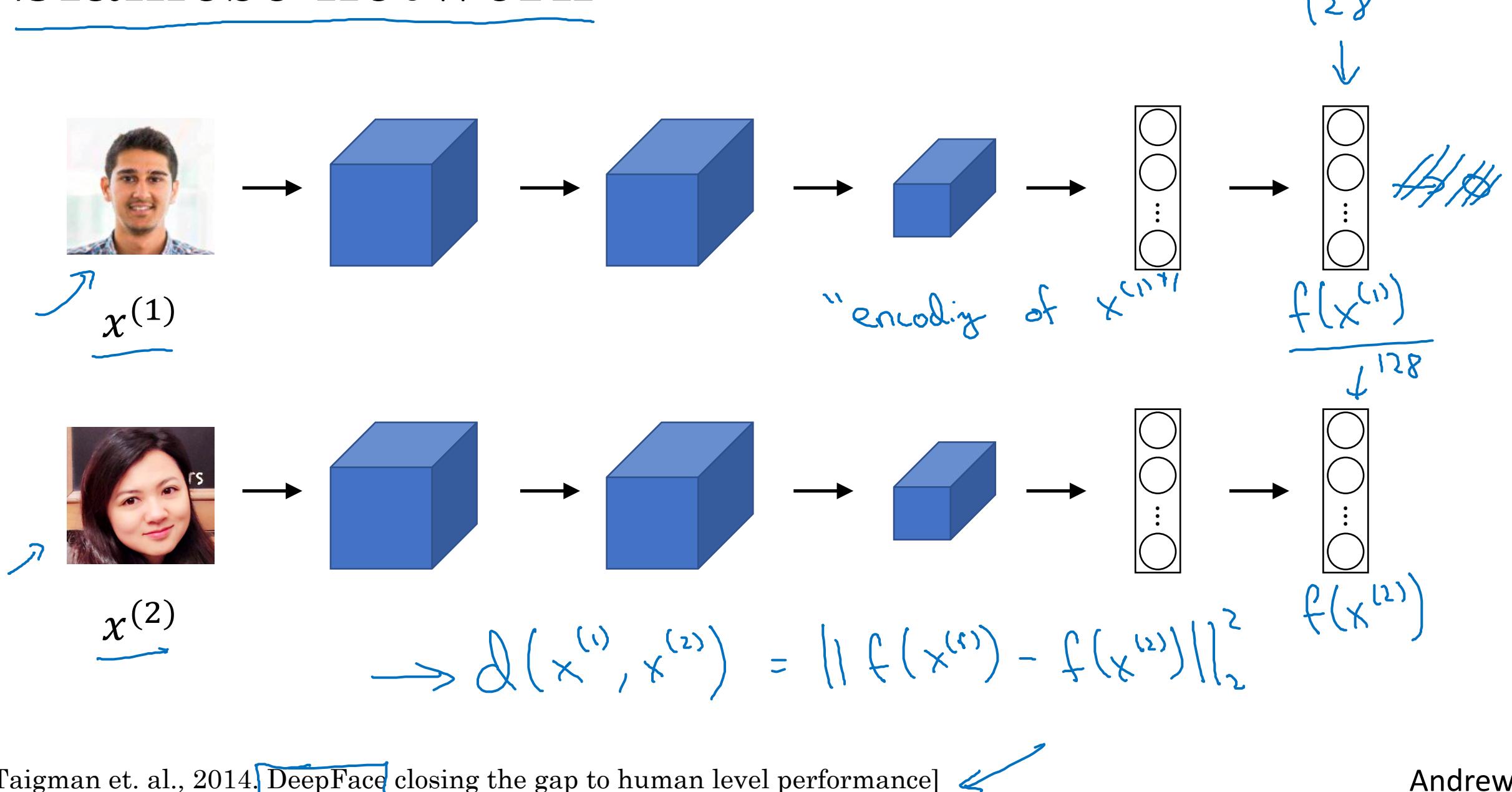
deeplearning.ai

# Face recognition

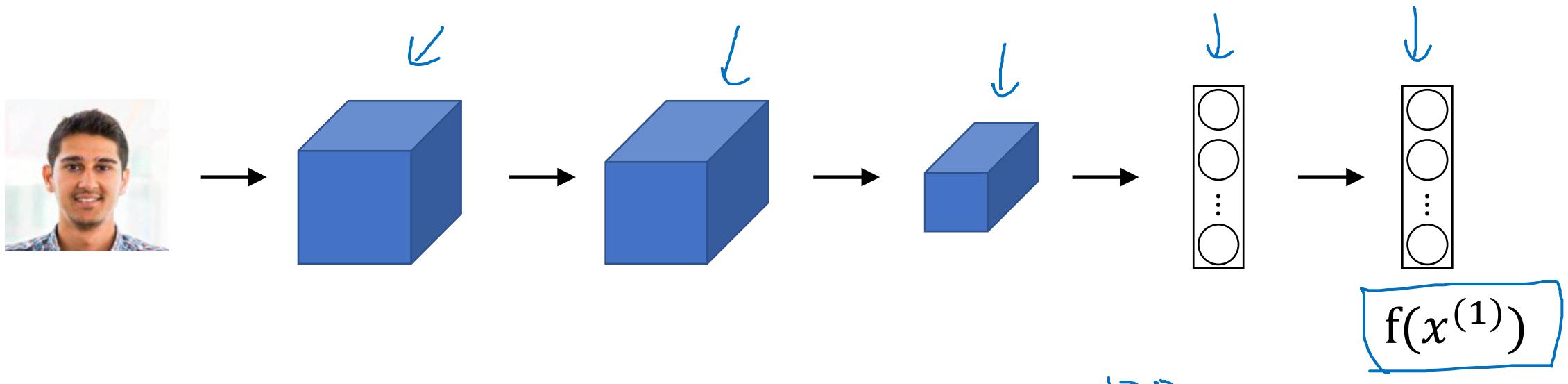
---

## Siamese network

# Siamese network



# Goal of learning



Parameters of NN define an encoding  $f(x^{(i)})$  128

Learn parameters so that:

If  $x^{(i)}, x^{(j)}$  are the same person,  $\|f(x^{(i)}) - f(x^{(j)})\|^2$  is small.

If  $x^{(i)}, x^{(j)}$  are different persons,  $\|f(x^{(i)}) - f(x^{(j)})\|^2$  is large.



deeplearning.ai

# Face recognition

---

## Triplet loss

# Learning Objective



Anchor Positive  
A  $d(A, P) = 0.5$

Want:  $\frac{\|f(A) - f(P)\|^2}{d(A, P)} + \lambda \leq \frac{\|f(A) - f(N)\|^2}{d(A, N)}$   $\rightarrow 0.2$

Anchor Negative  
A  $d(A, N) = \frac{N}{0.5} = 0.7$

$$\frac{\|f(A) - f(P)\|^2}{\circ} - \frac{\|f(A) - f(N)\|^2}{\circ} + \lambda \leq 0 \quad 4/4 \quad f(\text{img}) = \vec{0}$$

Margin

# Loss function

Given 3 images

$A, P, N$ :

$$\underline{L(A, P, N)} = \max \left( \left[ \|f(A) - f(P)\|^2 - \|f(A) - f(N)\|^2 + \lambda \right], 0 \right)$$

$$J = \sum_{i=1}^m L(A^{(i)}, P^{(i)}, N^{(i)})$$

$A, P$   
 $T$

Training set:  $\underbrace{10k}_{\infty}$  pictures of  $\frac{1k}{\infty}$  persons

# Choosing the triplets A,P,N



During training, if A,P,N are chosen randomly,  
 $d(A, P) + \alpha \leq d(A, N)$  is easily satisfied.

$$\underbrace{\|f(A) - f(P)\|^2}_{\text{ }} + \alpha \leq \underbrace{\|f(A) - f(N)\|^2}_{\text{ }}$$

Choose triplets that're “hard” to train on.

$$\begin{aligned} \cancel{d(A, P)} + \alpha &\leq \cancel{d(A, N)} \\ \frac{d(A, P)}{\downarrow} &\approx \frac{d(A, N)}{\uparrow} \end{aligned}$$

Face Net  
Deep Face



# Training set using triplet loss

Anchor



Positive



Negative



:

:

:



J

$$d(x^{(i)}, x^{(j)})$$



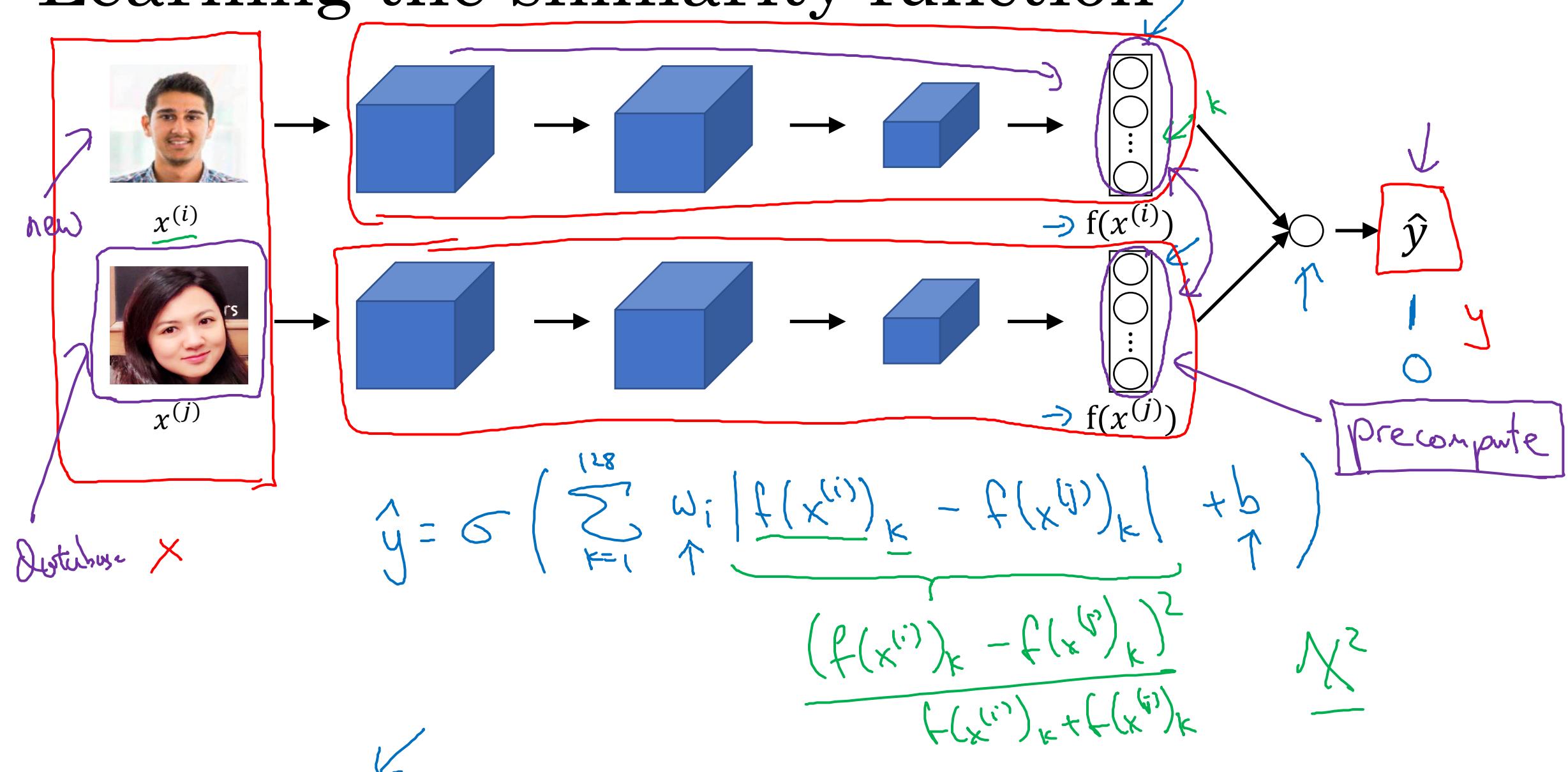
deeplearning.ai

## Face recognition

---

## Face verification and binary classification

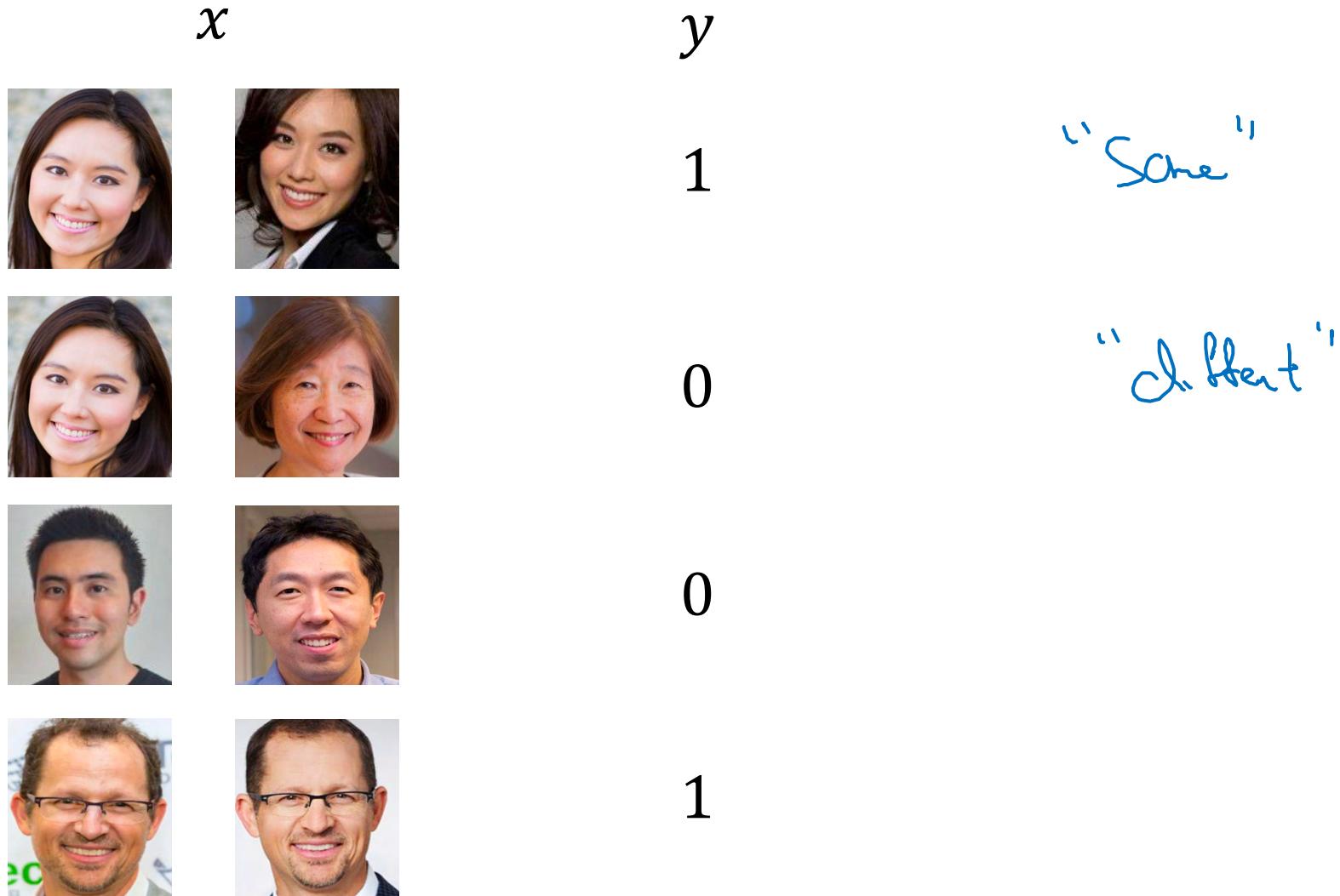
# Learning the similarity function



[Taigman et. al., 2014. DeepFace closing the gap to human level performance]

Andrew Ng

# Face verification supervised learning





deeplearning.ai

# Neural Style Transfer

---

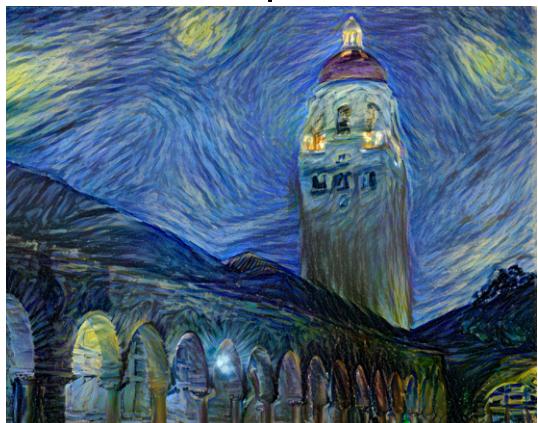
What is neural style  
transfer?

# Neural style transfer



Content ( $c$ )

Style ( $s$ )



Generated image ( $g$ )



Content ( $c$ )

Style ( $s$ )



Generated image ( $g$ )



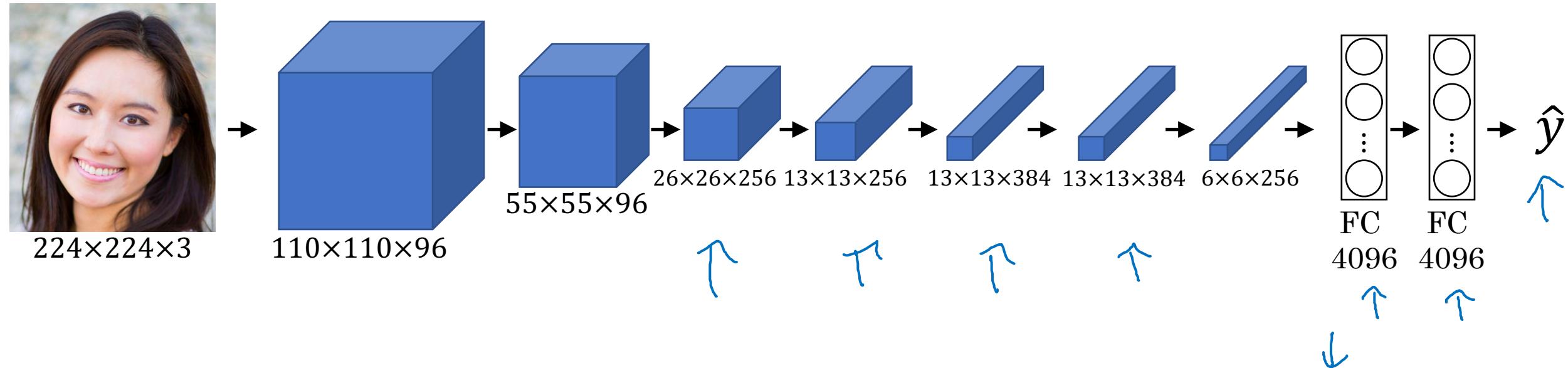
deeplearning.ai

# Neural Style Transfer

---

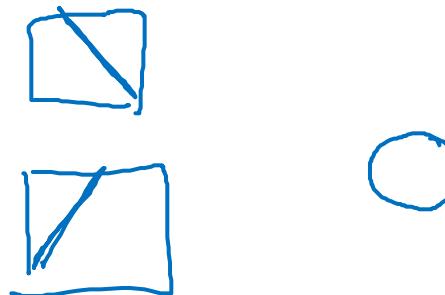
What are deep  
ConvNets learning?

# Visualizing what a deep network is learning

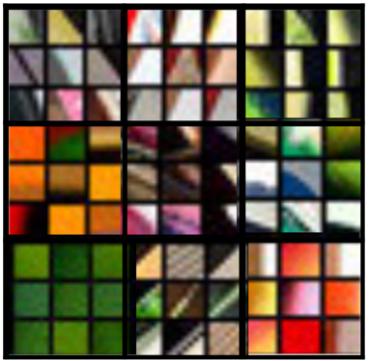


Pick a unit in layer 1. Find the nine image patches that maximize the unit's activation.

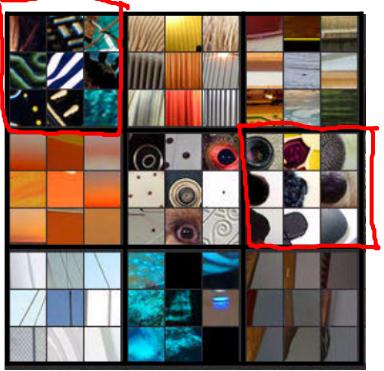
Repeat for other units.



# Visualizing deep layers



Layer 1



Layer 2



Layer 3

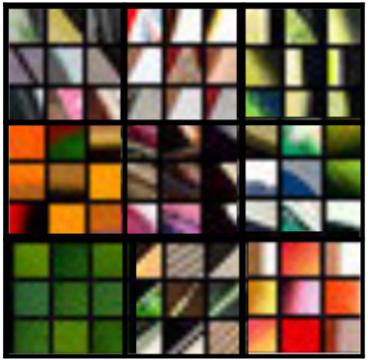


Layer 4

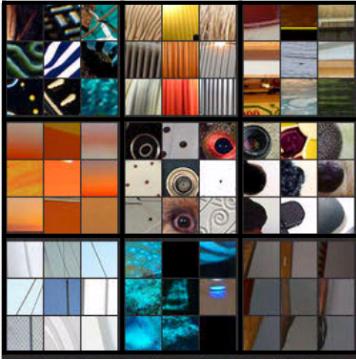


Layer 5

# Visualizing deep layers: Layer 1



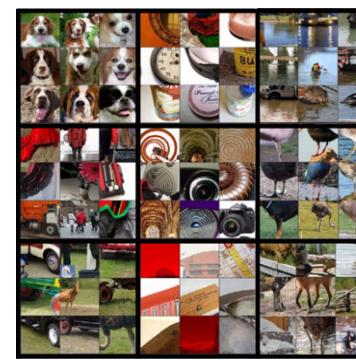
Layer 1



Layer 2



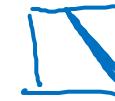
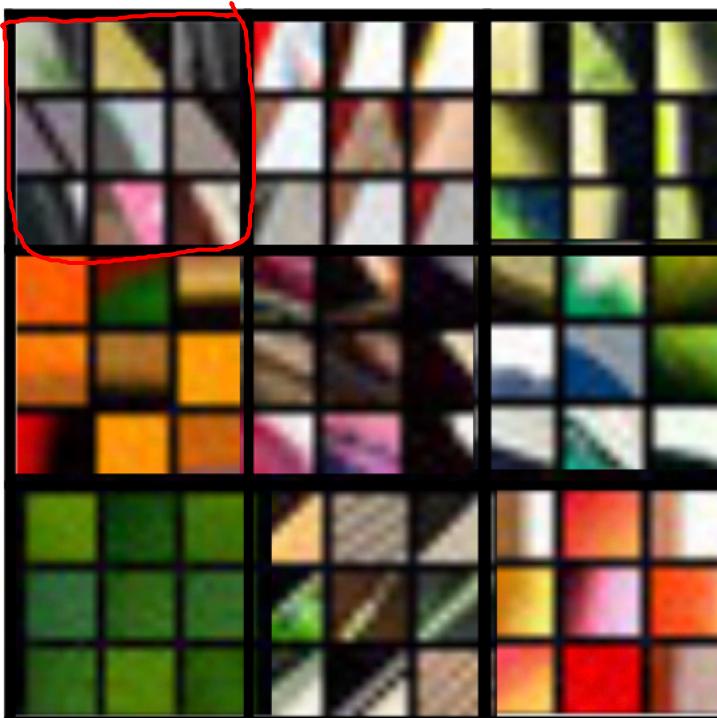
Layer 3



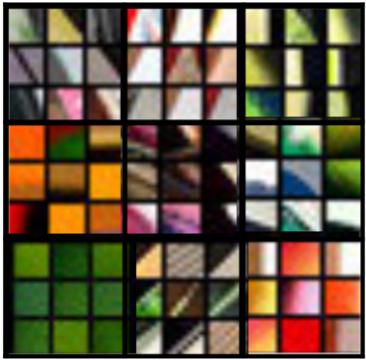
Layer 4



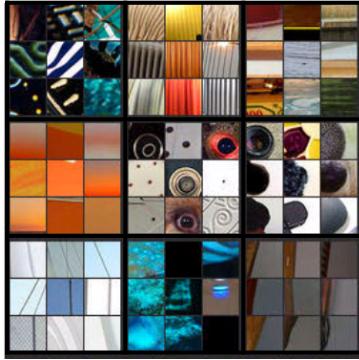
Layer 5



# Visualizing deep layers: Layer 2



Layer 1



Layer 2



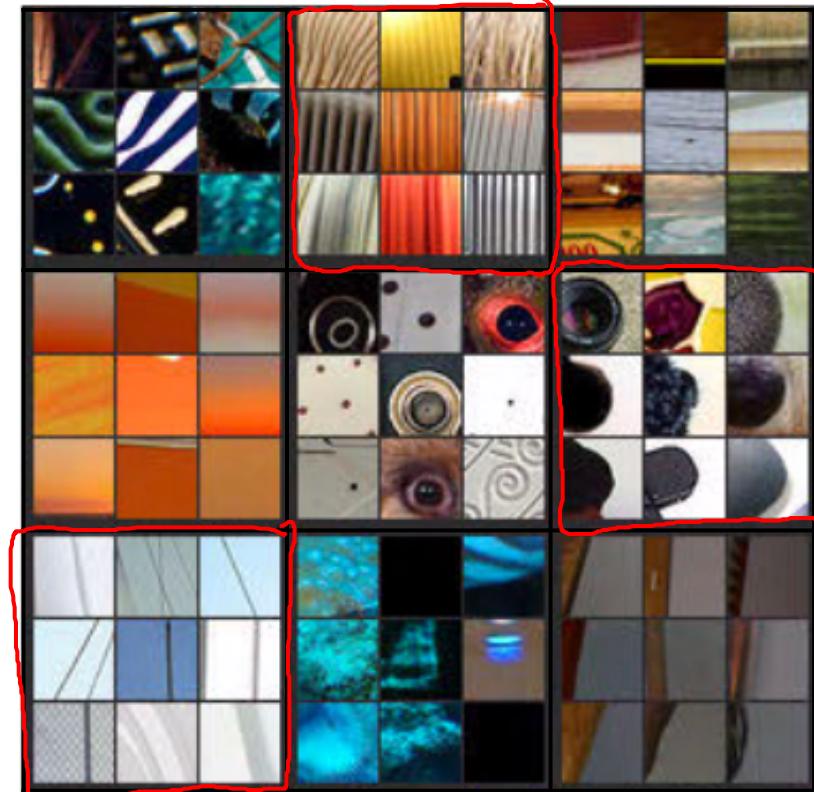
Layer 3



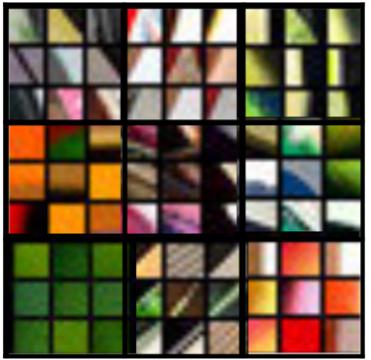
Layer 4



Layer 5



# Visualizing deep layers: Layer 3



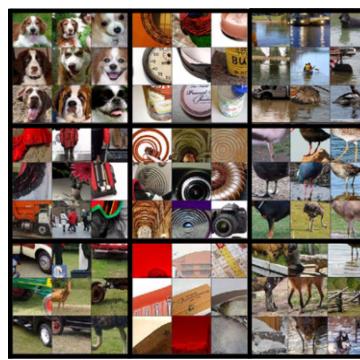
Layer 1



Layer 2



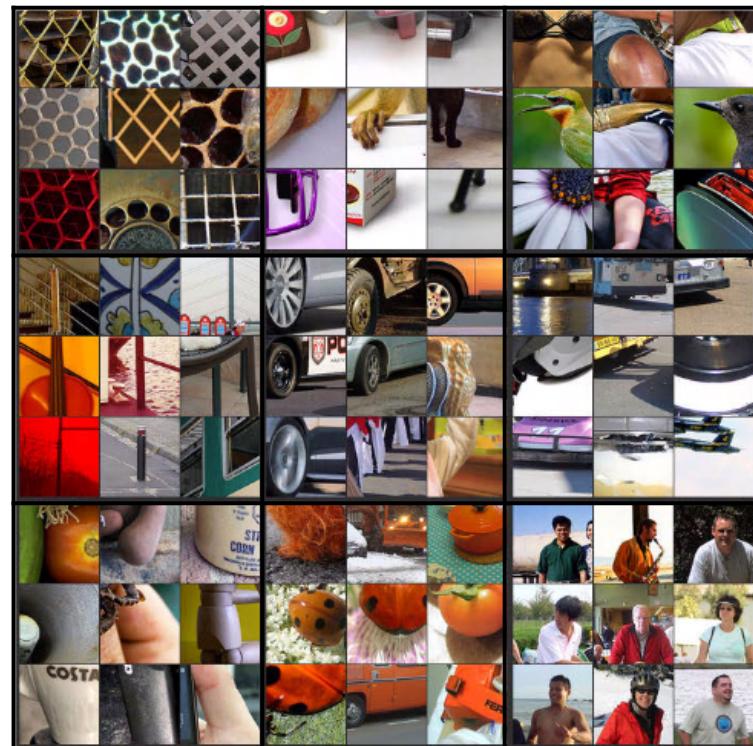
Layer 3



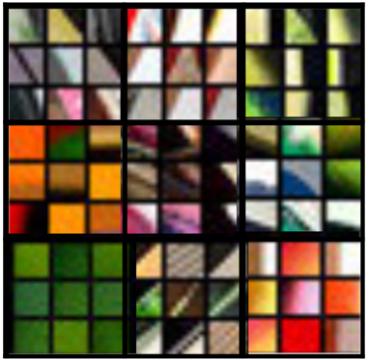
Layer 4



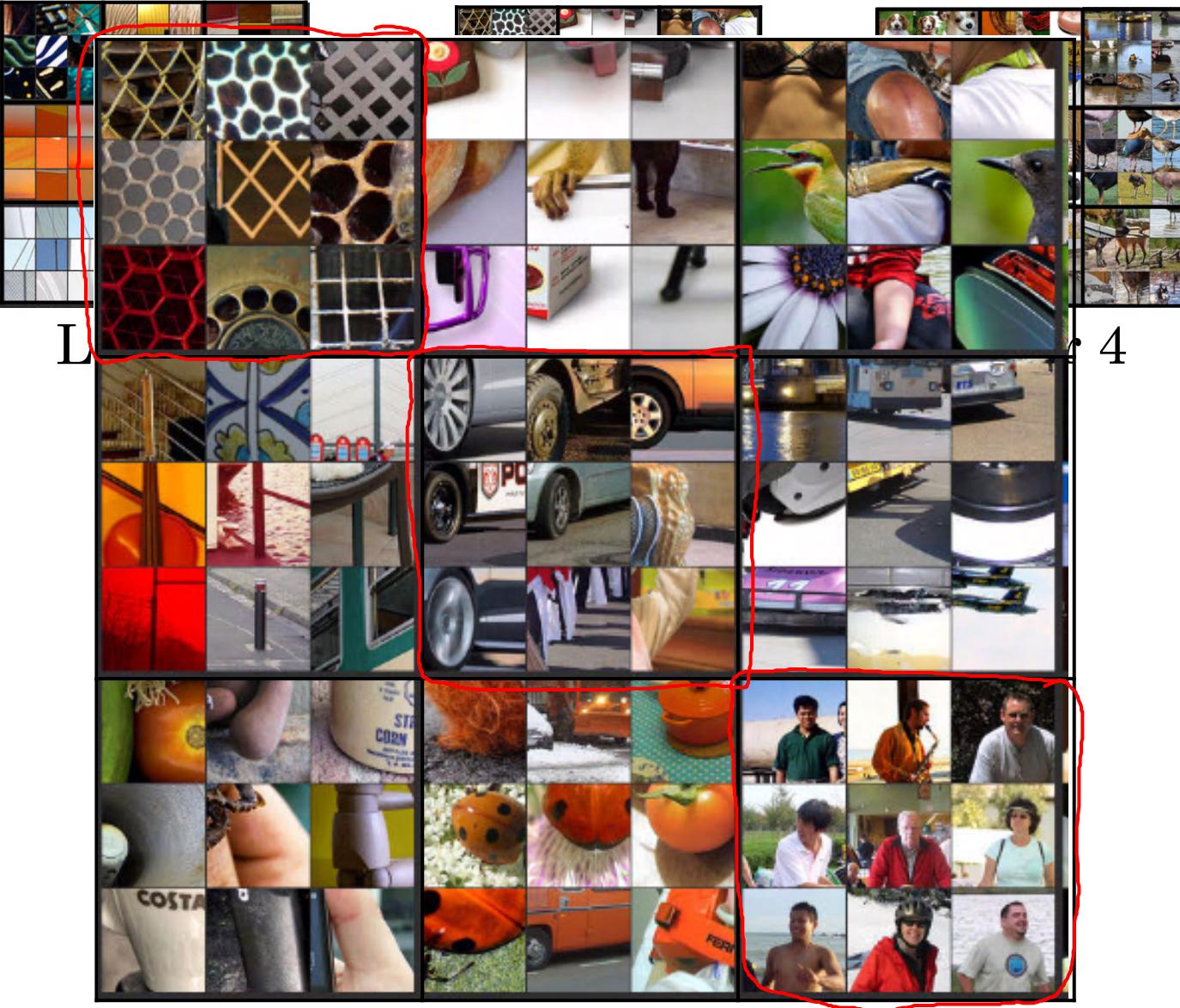
Layer 5



# Visualizing deep layers: Layer 3



Layer 1



Layer 5

# Visualizing deep layers: Layer 4

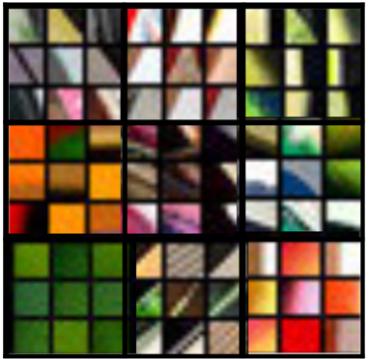


Layer 4



Layer 5

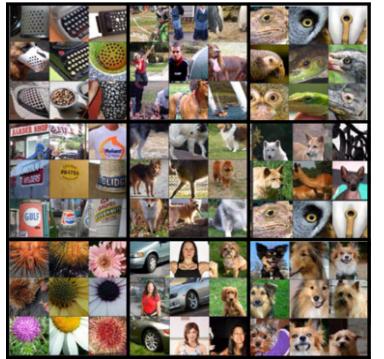
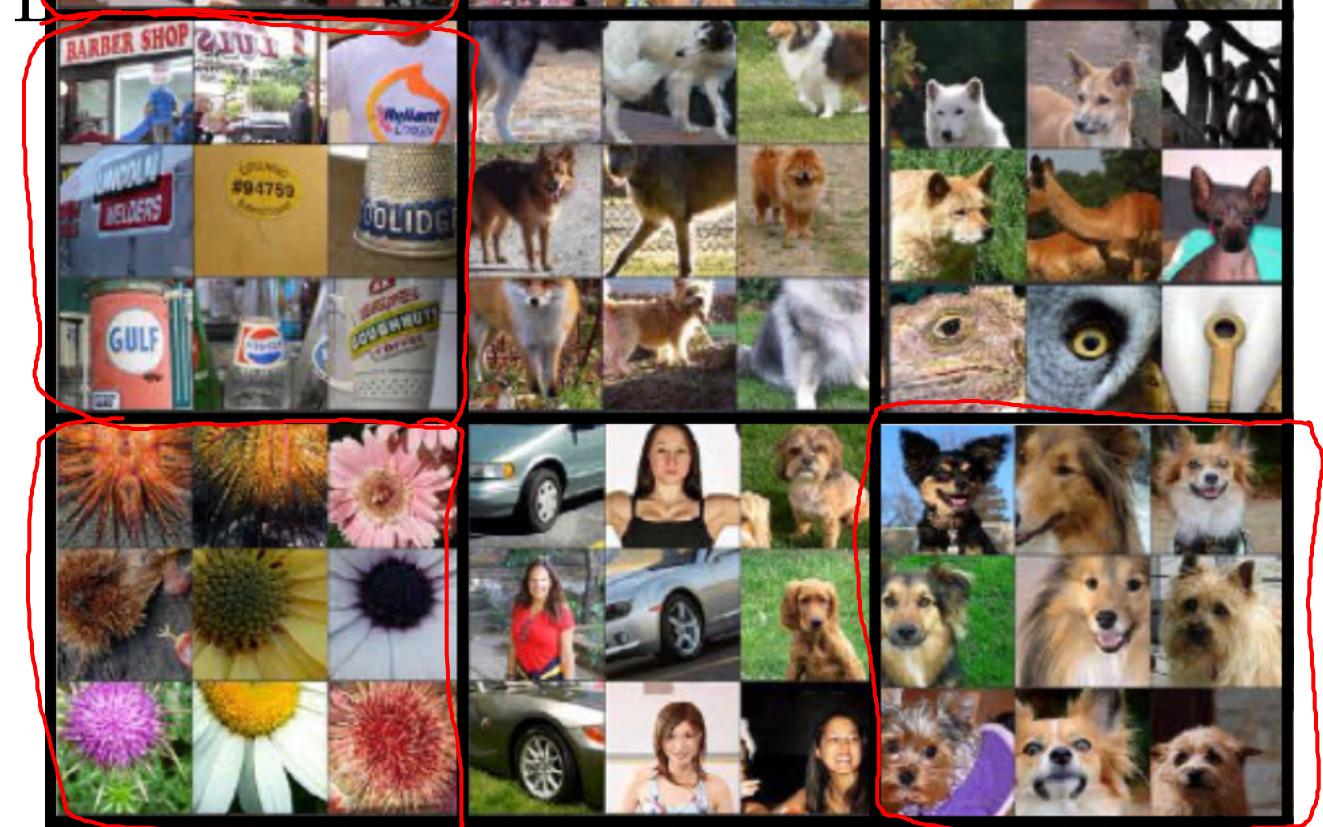
# Visualizing deep layers: Layer 5



Layer 1



L



Layer 5



deeplearning.ai

# Neural Style Transfer

---

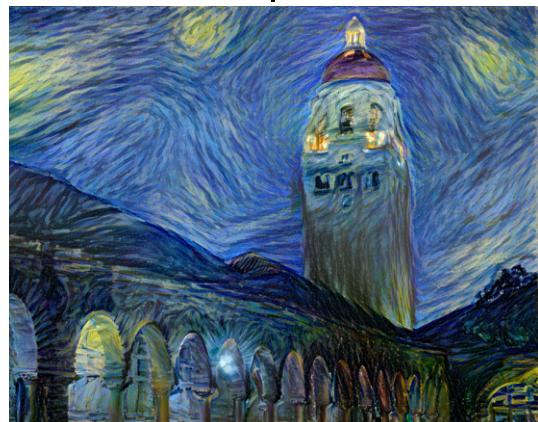
## Cost function

# Neural style transfer cost function



Content C

Style S



Generated image G

$$J(G) = \alpha J_{\text{Content}}(C, G) + \beta J_{\text{Style}}(S, G)$$

# Find the generated image G

1. Initiate G randomly

$G: \underbrace{100 \times 100}_{\text{---}} \times \underbrace{3}_{\text{---}}$

$\uparrow$   
RGB



2. Use gradient descent to minimize  $\underline{J(G)}$

$$G_t := G - \frac{\partial}{\partial G} J(G)$$





deeplearning.ai

# Neural Style Transfer

---

## Content cost function

# Content cost function

$$\underline{J}(G) = \alpha \underline{J}_{content}(C, G) + \beta J_{style}(S, G)$$

- Say you use hidden layer  $\underline{l}$  to compute content cost.
- Use pre-trained ConvNet. (E.g., VGG network)
- Let  $\underline{a}^{[l](C)}$  and  $\underline{a}^{[l](G)}$  be the activation of layer  $\underline{l}$  on the images
- If  $\underline{a}^{[l](C)}$  and  $\underline{a}^{[l](G)}$  are similar, both images have similar content

$$J_{content}(C, G) = \frac{1}{2} \left\| \underline{a}^{[l](C)} - \underline{a}^{[l](G)} \right\|^2$$



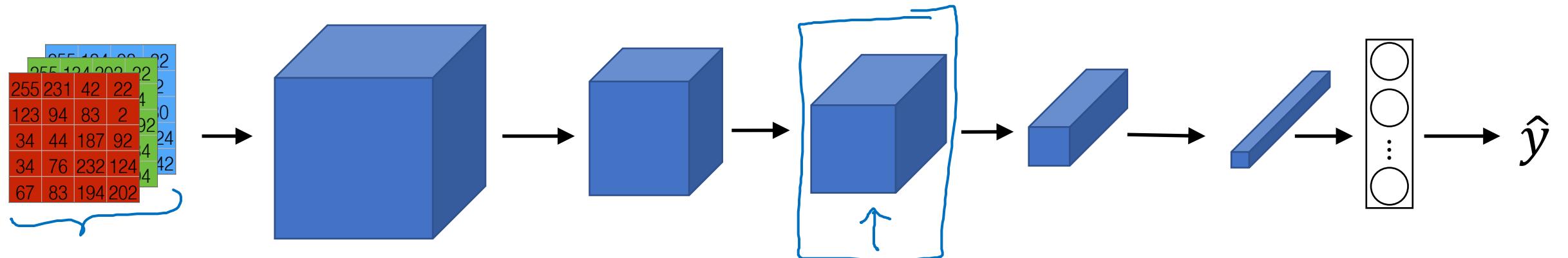
deeplearning.ai

# Neural Style Transfer

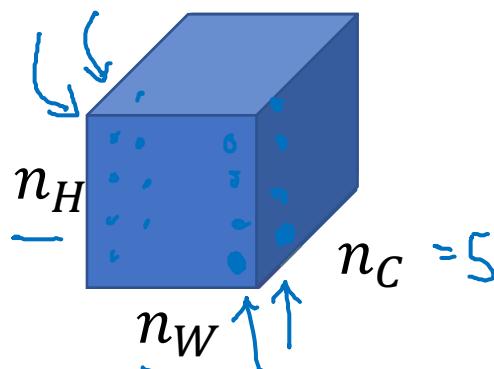
---

## Style cost function

# Meaning of the “style” of an image

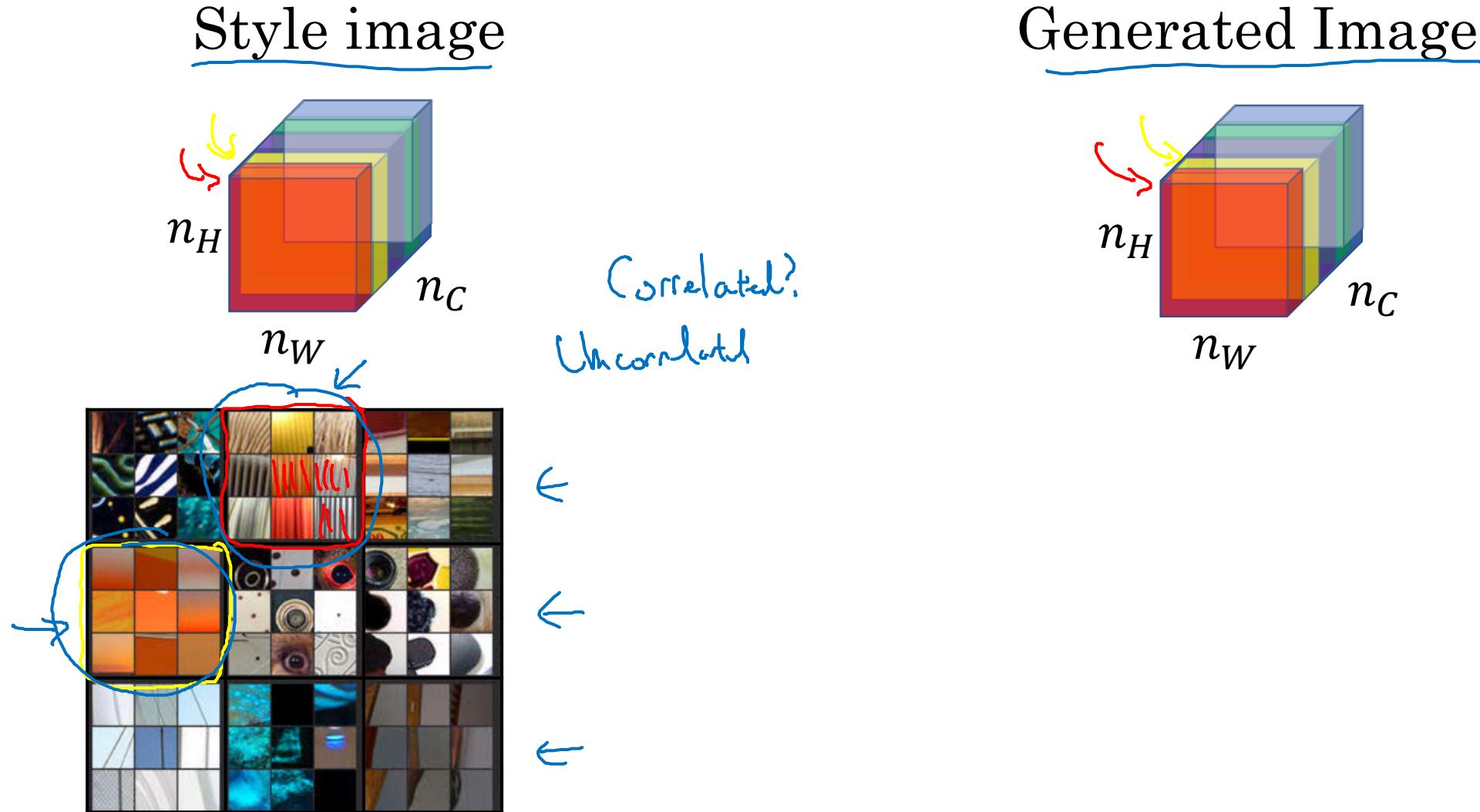


Say you are using layer  $l$ 's activation to measure “style.”  
Define style as correlation between activations across channels.



How correlated are the activations  
across different channels?

# Intuition about style of an image



# Style matrix

Let  $a_{i,j,k}^{[l]}$  = activation at  $(i, j, k)$ .  $G^{[l]}$  is  $n_c^{[l]} \times n_c^{[l]}$

$$\rightarrow G_{kk'}^{[l](s)} = \sum_{i=1}^{n_h^{[l]}} \sum_{j=1}^{n_w^{[l]}} a_{ijk}^{[l](s)} a_{ijk'}^{[l](s)}$$

$$\rightarrow G_{kk'}^{[l](G)} = \sum_{i=1}^{n_h^{[l]}} \sum_{j=1}^{n_w^{[l]}} a_{ijk}^{[l](G)} a_{ijk'}^{[l](G)}$$

H W C  
↓ ↓ ↗

$n_c$

$$G_{kk'}^{[l]} \quad \text{for } k, k' = 1, \dots, n_c^{[l]}$$

"Gram matrix"

$$\begin{aligned} J_{\text{style}}^{[l]}(S, G) &= \frac{1}{(\dots)} \| G_{kk'}^{[l](s)} - G_{kk'}^{[l](G)} \|_F^2 \\ &= \frac{1}{(2n_h^{[l]} n_w^{[l]} n_c^{[l]})^2} \sum_k \sum_{k'} (G_{kk'}^{[l](s)} - G_{kk'}^{[l](G)})^2 \end{aligned}$$

# Style cost function

$$\left\| G^{[l](s)} - G^{[l](G)} \right\|_F^2$$

$$J_{style}^{[l]}(S, G) = \frac{1}{\left(2n_H^{[l]} n_W^{[l]} n_C^{[l]}\right)^2} \sum_k \sum_{k'} (G_{kk'}^{[l](s)} - G_{kk'}^{[l](G)})$$

$$J_{style}(S, G) = \sum_l \lambda^{[l]} J_{style}^{[l]}(S, G)$$

$$\underbrace{J(G)}_G = \alpha J_{content}(S, G) + \beta J_{style}(S, G)$$



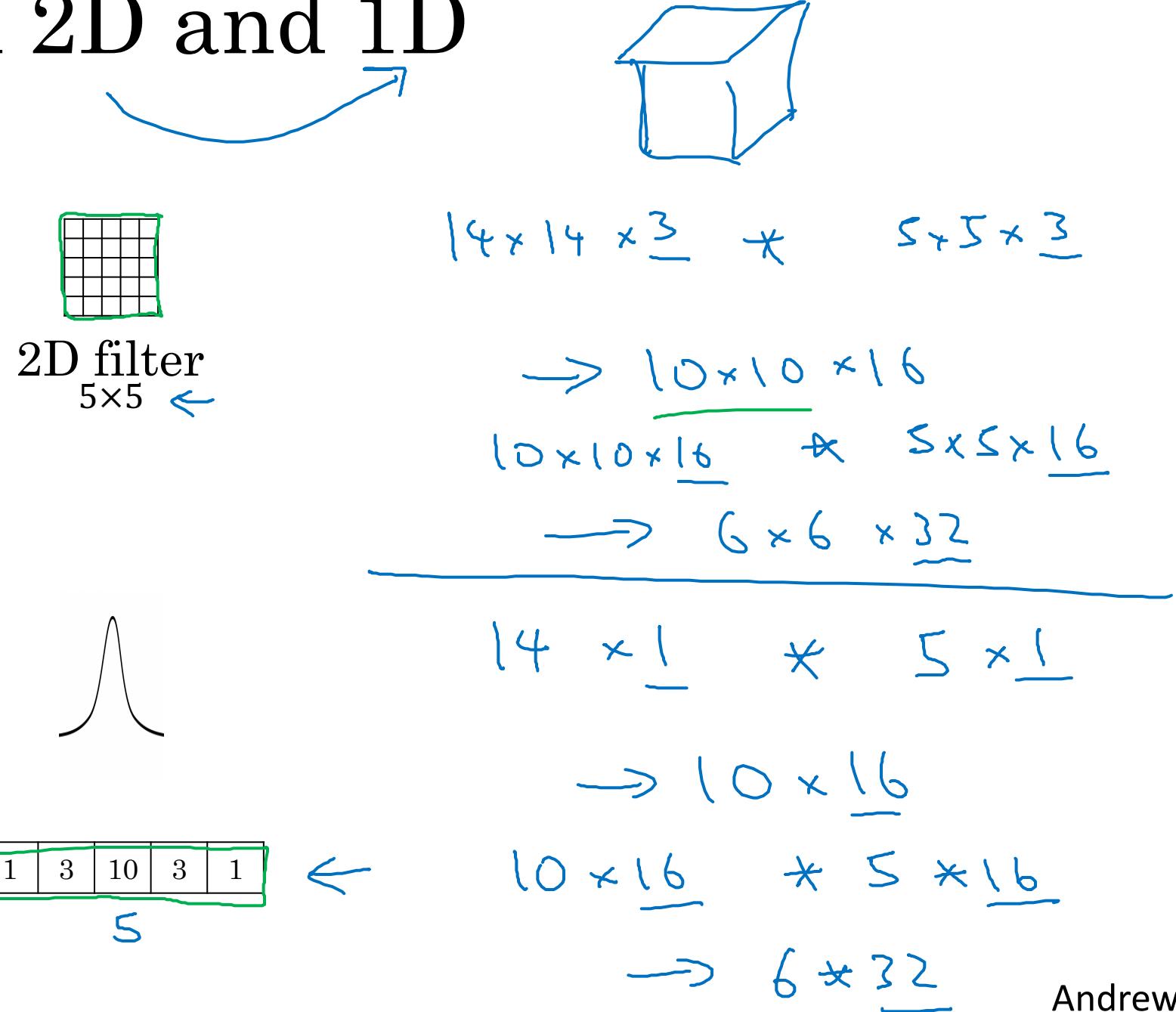
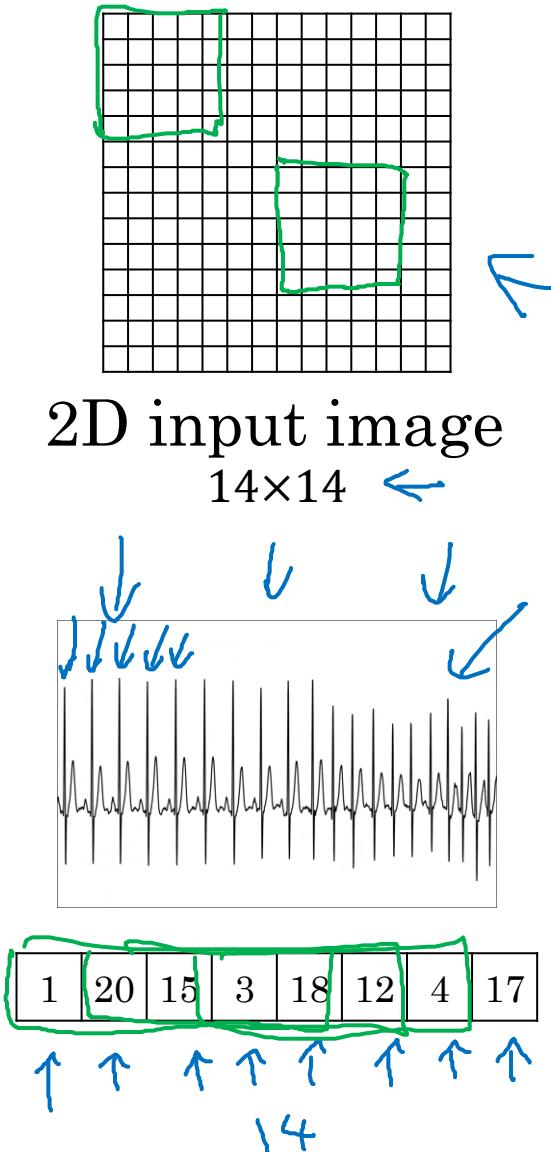
deeplearning.ai

# Convolutional Networks in 1D or 3D

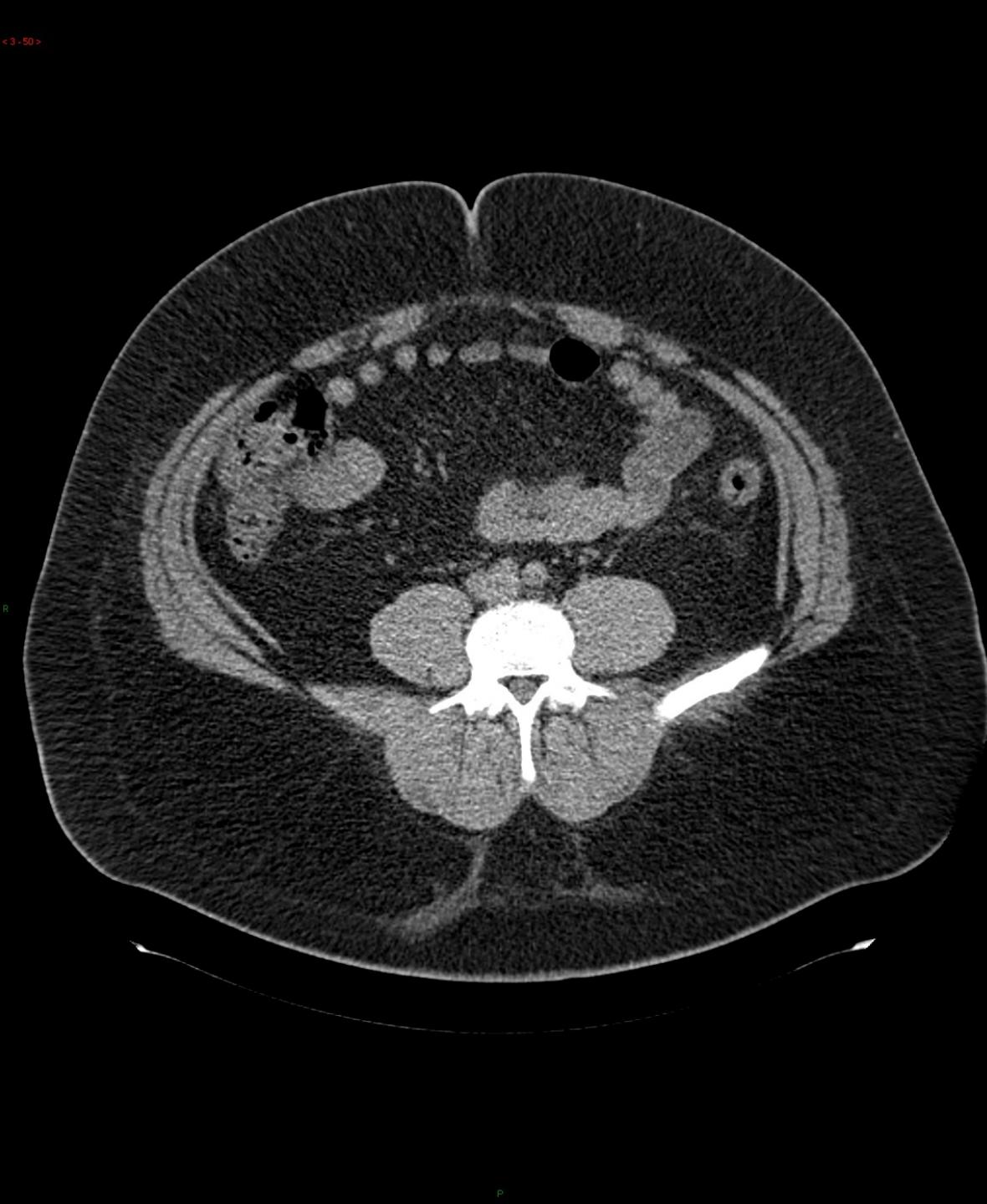
---

1D and 3D  
generalizations of  
models

# Convolutions in 2D and 1D



# 3D data

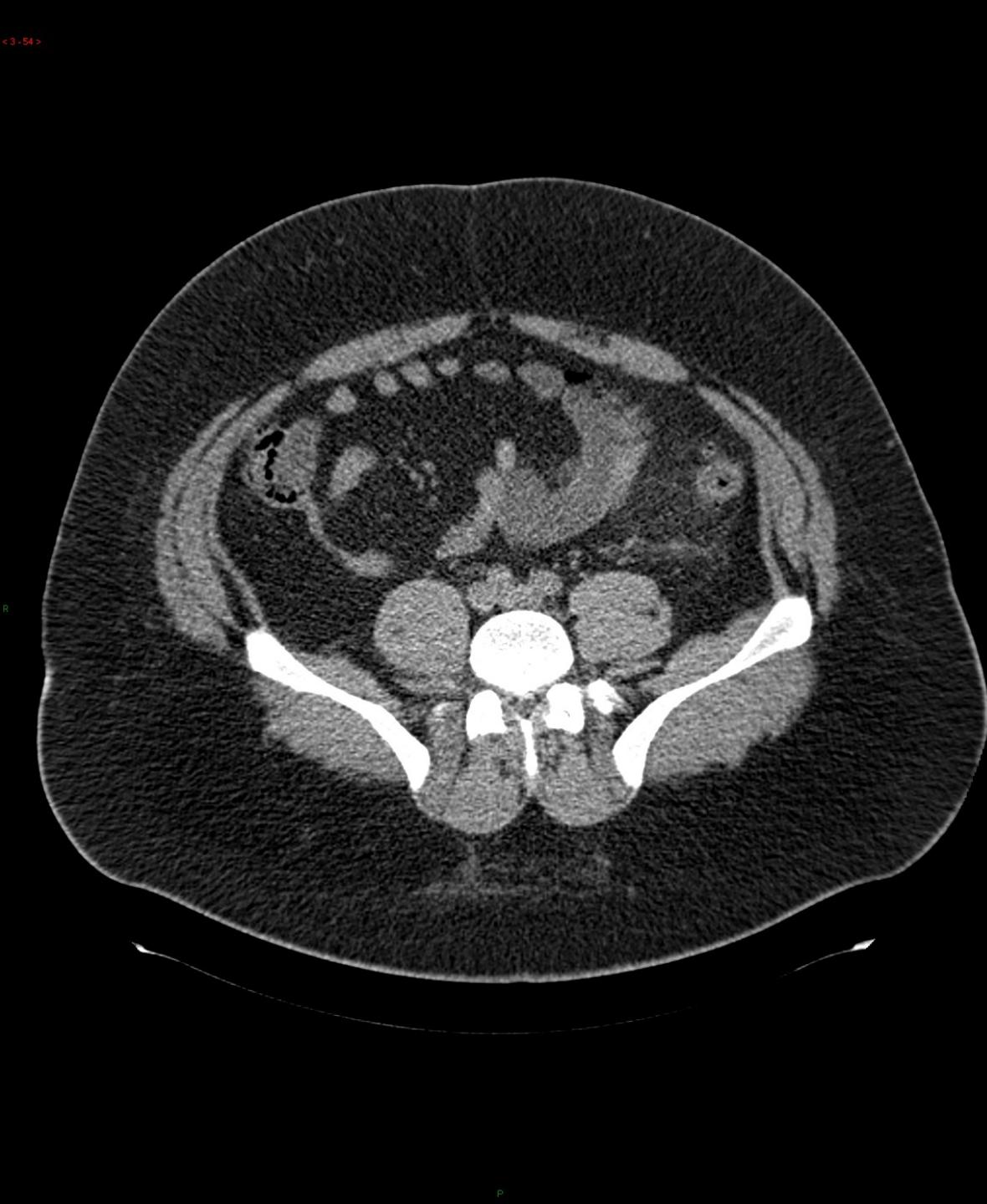


# 3D data



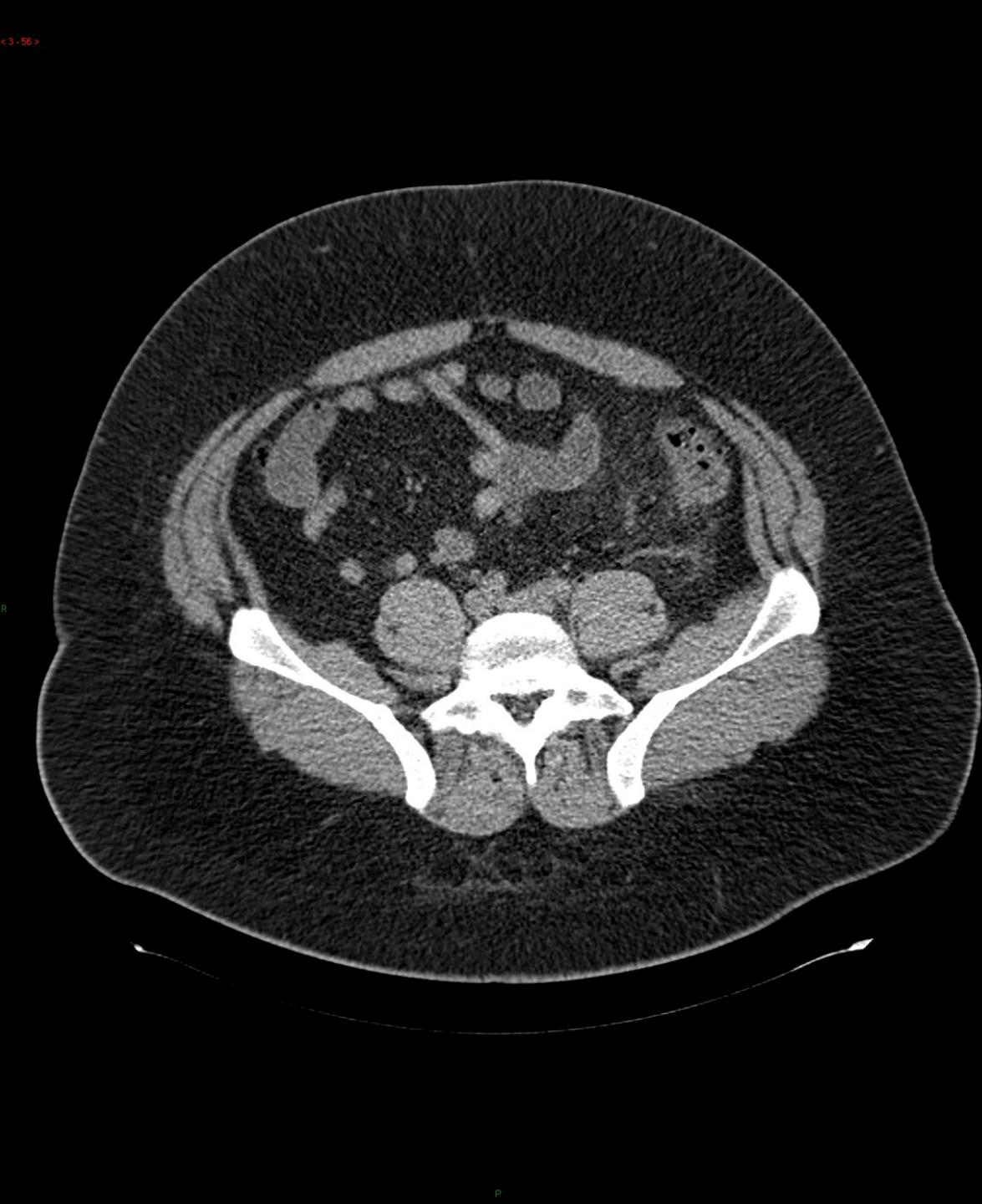
Andrew Ng

# 3D data

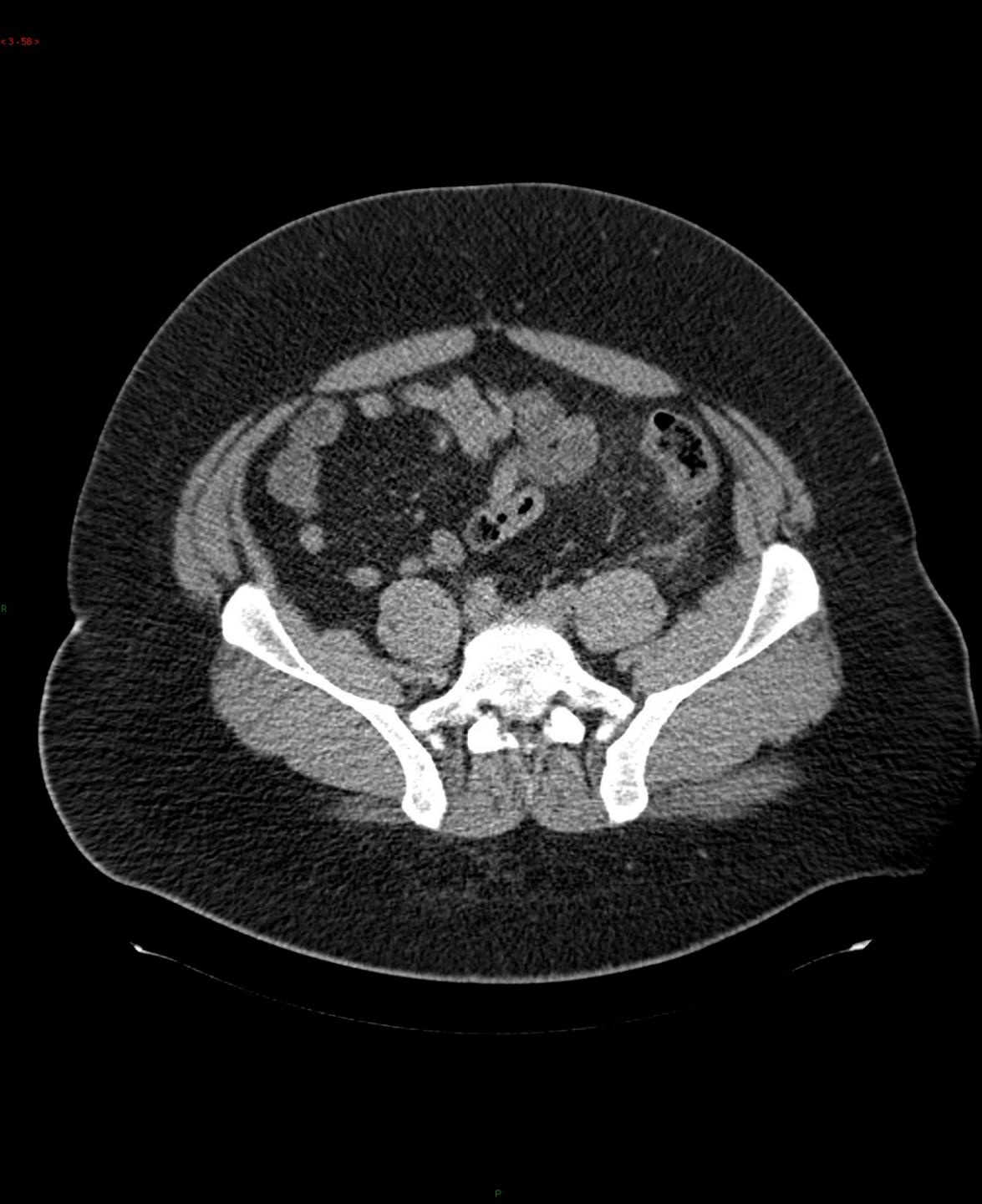


Andrew Ng

# 3D data

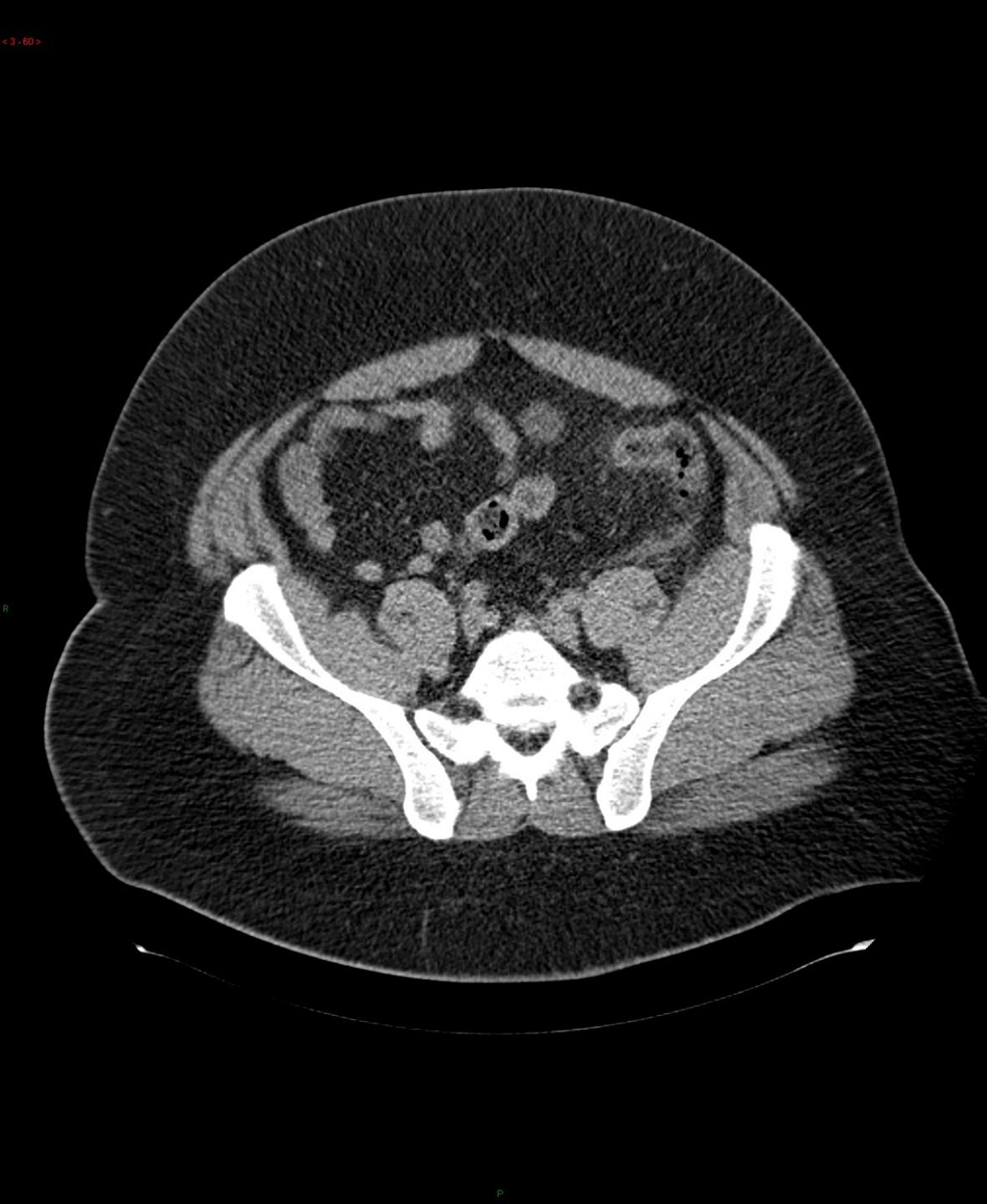


# 3D data



Andrew Ng

# 3D data

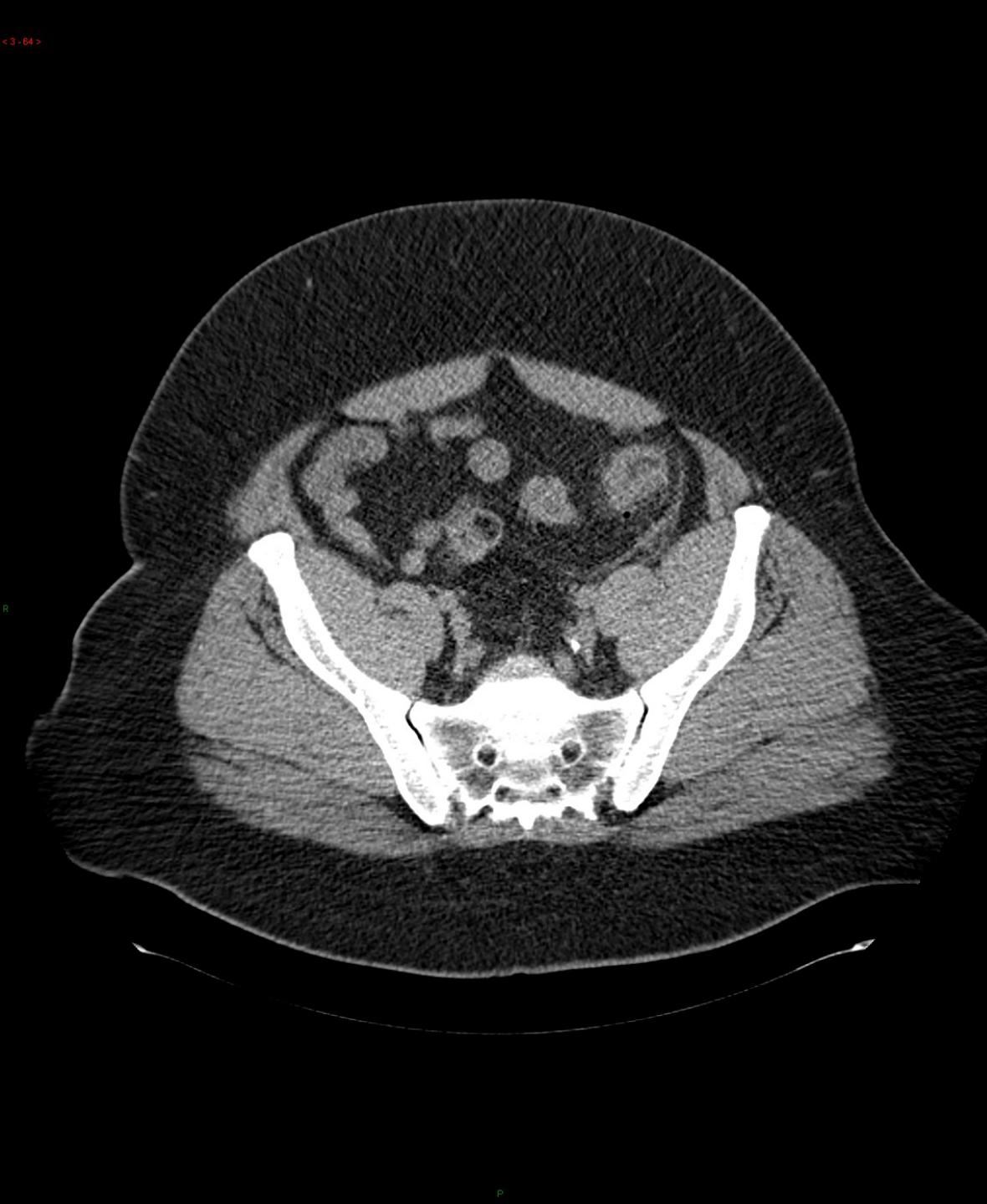


# 3D data



Andrew Ng

# 3D data



Andrew Ng

# 3D data



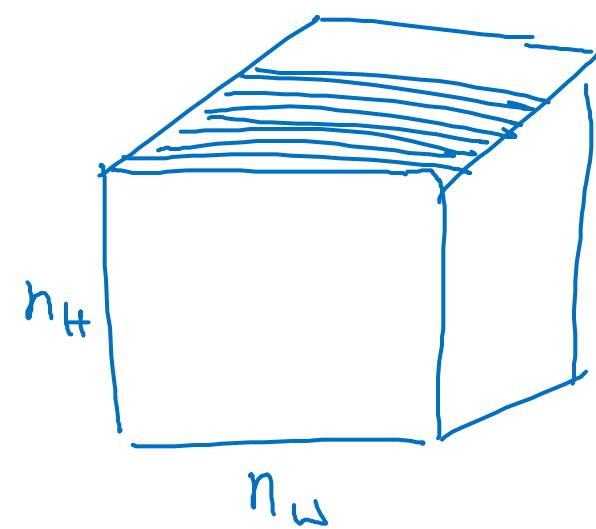
Andrew Ng

# 3D data

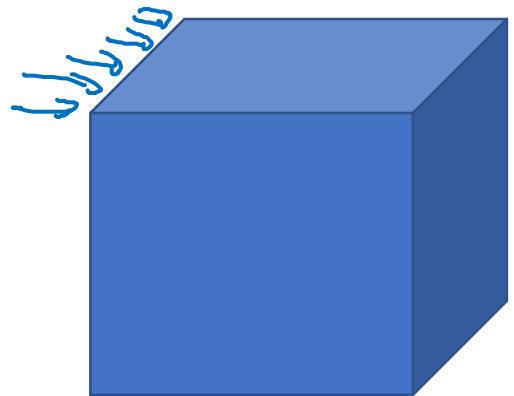


Andrew Ng

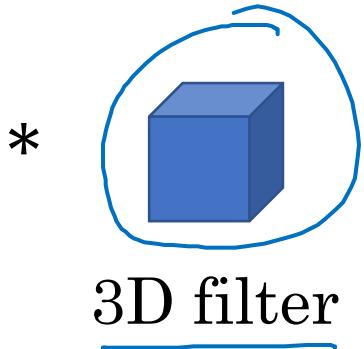
# 3D data



# 3D convolution



3D volume



$$\begin{array}{c} \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \\ \underbrace{4 \times 4 \times 4}_{\text{Input}} \times 1 \\ * \quad \underbrace{5 \times 5 \times 5}_{\text{Filter}} \times 1 \quad 16 \text{ filters.} \\ \rightarrow 10 \times 10 \times 10 \times 16 \\ * \quad \underbrace{5 \times 5 \times 5}_{\text{Stride}} \times 16 \\ \rightarrow 6 \times 6 \times 6 \times 32 \end{array}$$

Diagram illustrating the 3D convolution process:

- The input is a 3D volume of size  $4 \times 4 \times 4$ .
- The input is processed by a 3D filter of size  $5 \times 5 \times 5$ , resulting in 16 filters of size  $10 \times 10 \times 10$ .
- The output is then processed by another 3D filter of size  $5 \times 5 \times 5$  with stride 2, resulting in 32 filters of size  $6 \times 6 \times 6$ .