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)

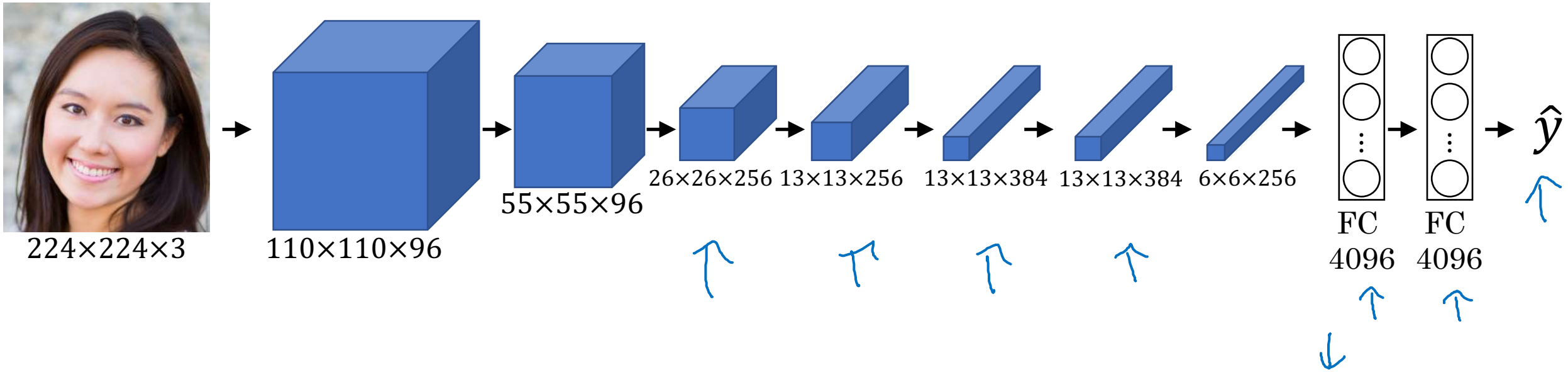
[Images generated by Justin Johnson]

deeplearning.ai

Neural Style Transfer

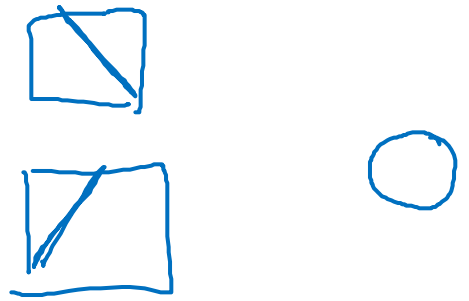
What are deep ConvNets learning?

Visualizing what a deep network is learning



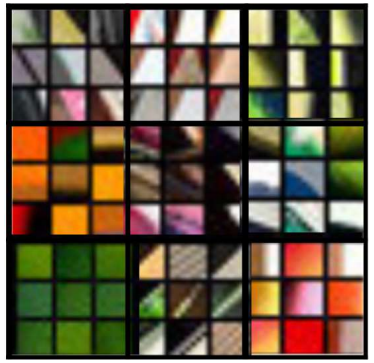
Pick a unit in layer 1. Find the nine image patches that maximize the unit's activation.

Repeat for other units.

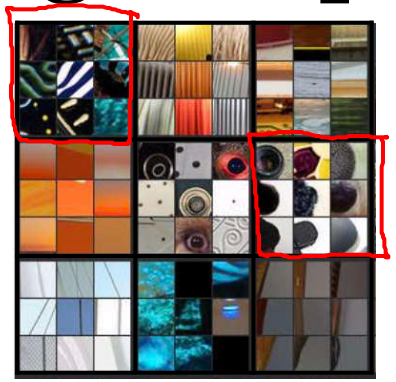


[Zeiler and Fergus., 2013, Visualizing and understanding convolutional networks]

Visualizing deep layers

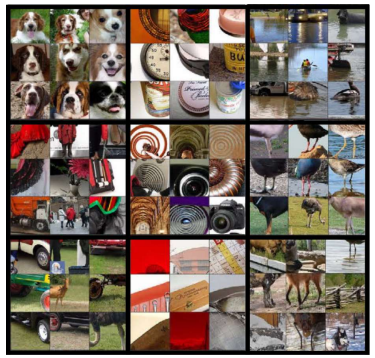


Layer 1

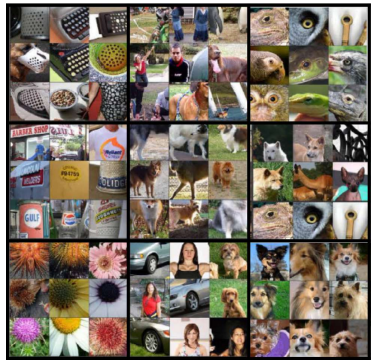


Layer 2

Layer 3

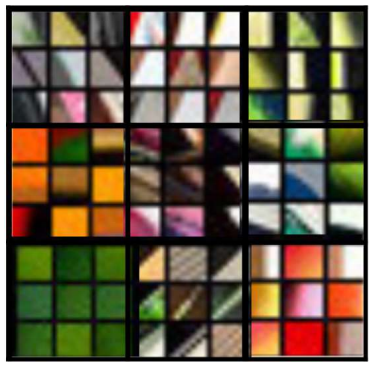


Layer 4



Layer 5

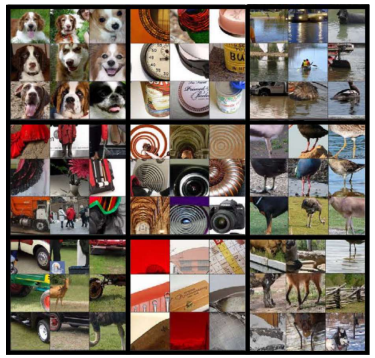
Visualizing deep layers: Layer 1



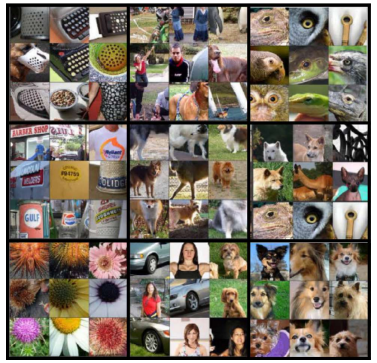
Layer 1

Layer 2

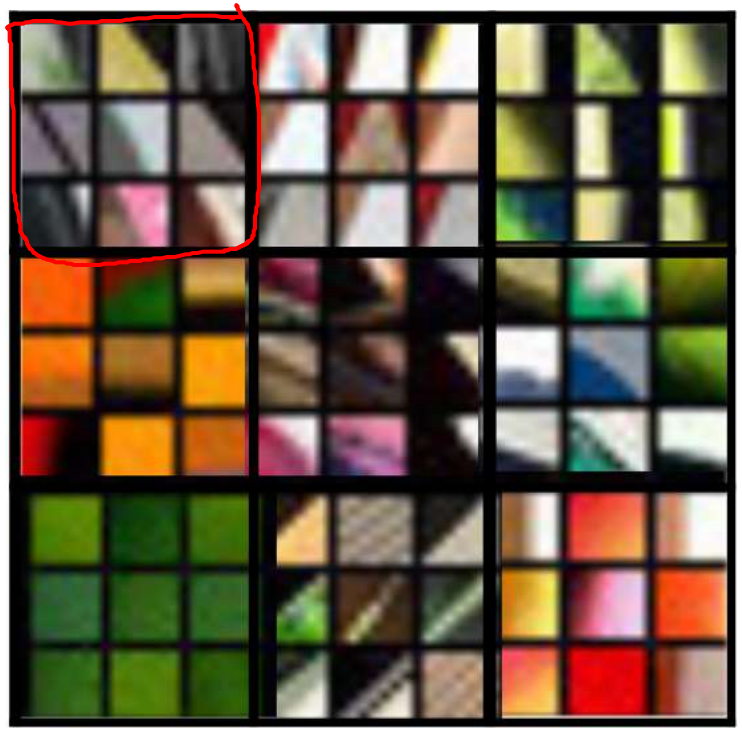
Layer 3



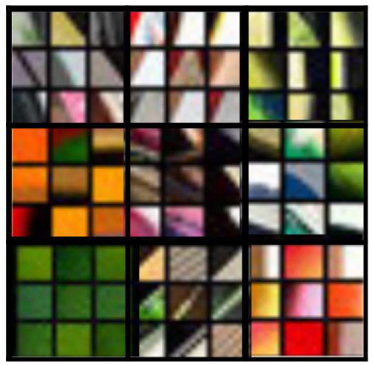
Layer 4



Layer 5



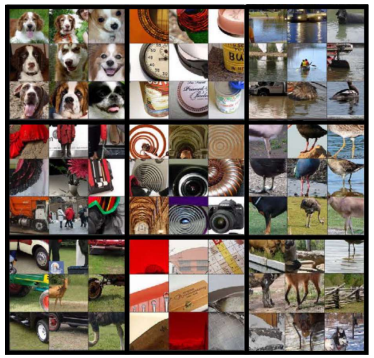
Visualizing deep layers: Layer 2



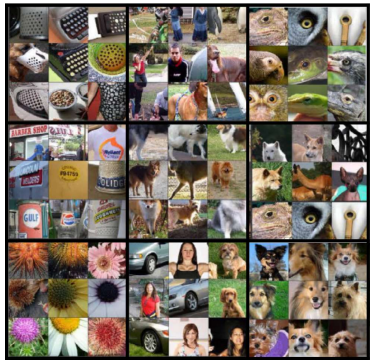
Layer 1

Layer 2

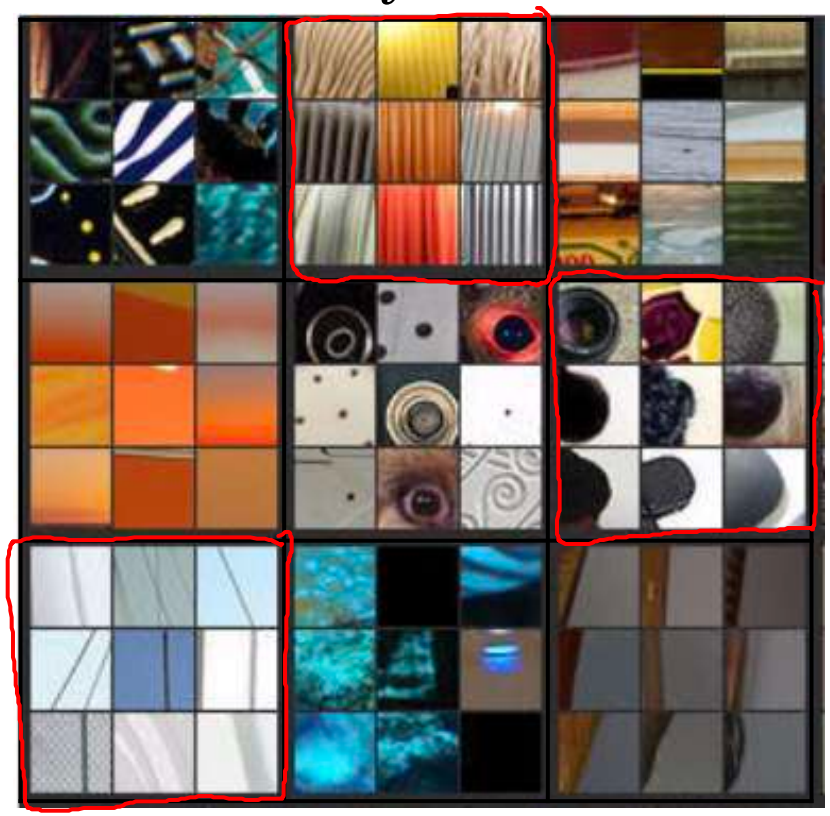
Layer 3



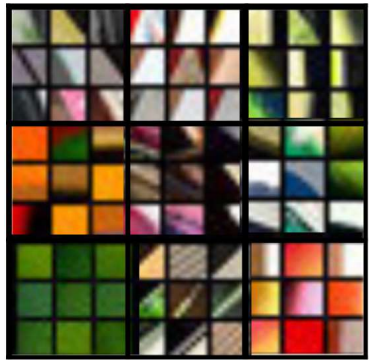
Layer 4



Layer 5



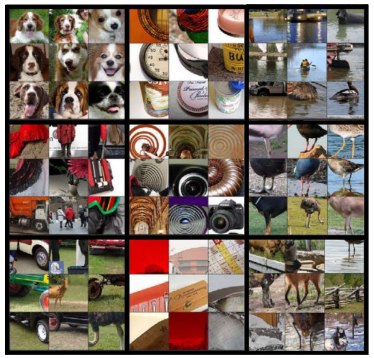
Visualizing deep layers: Layer 3



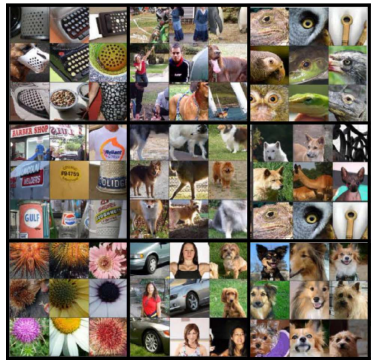
Layer 1

Layer 2

Layer 3

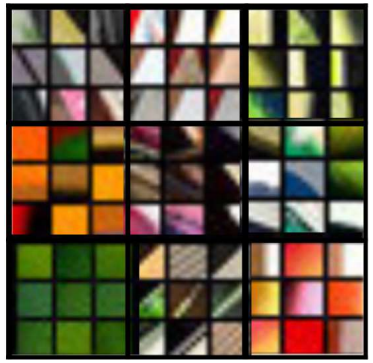


Layer 4

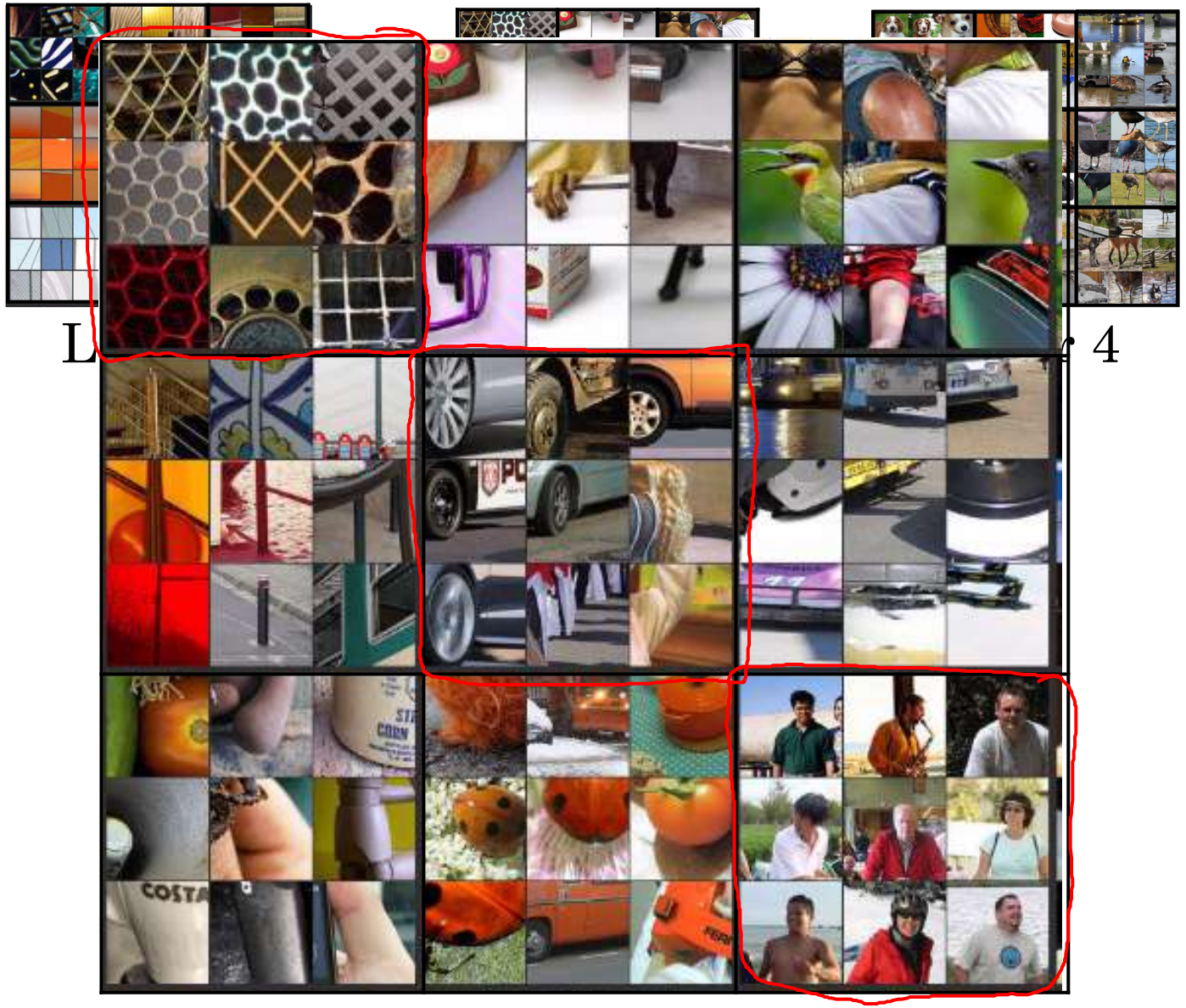


Layer 5

Visualizing deep layers: Layer 3

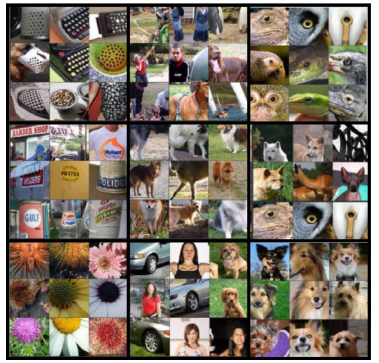


Layer 1



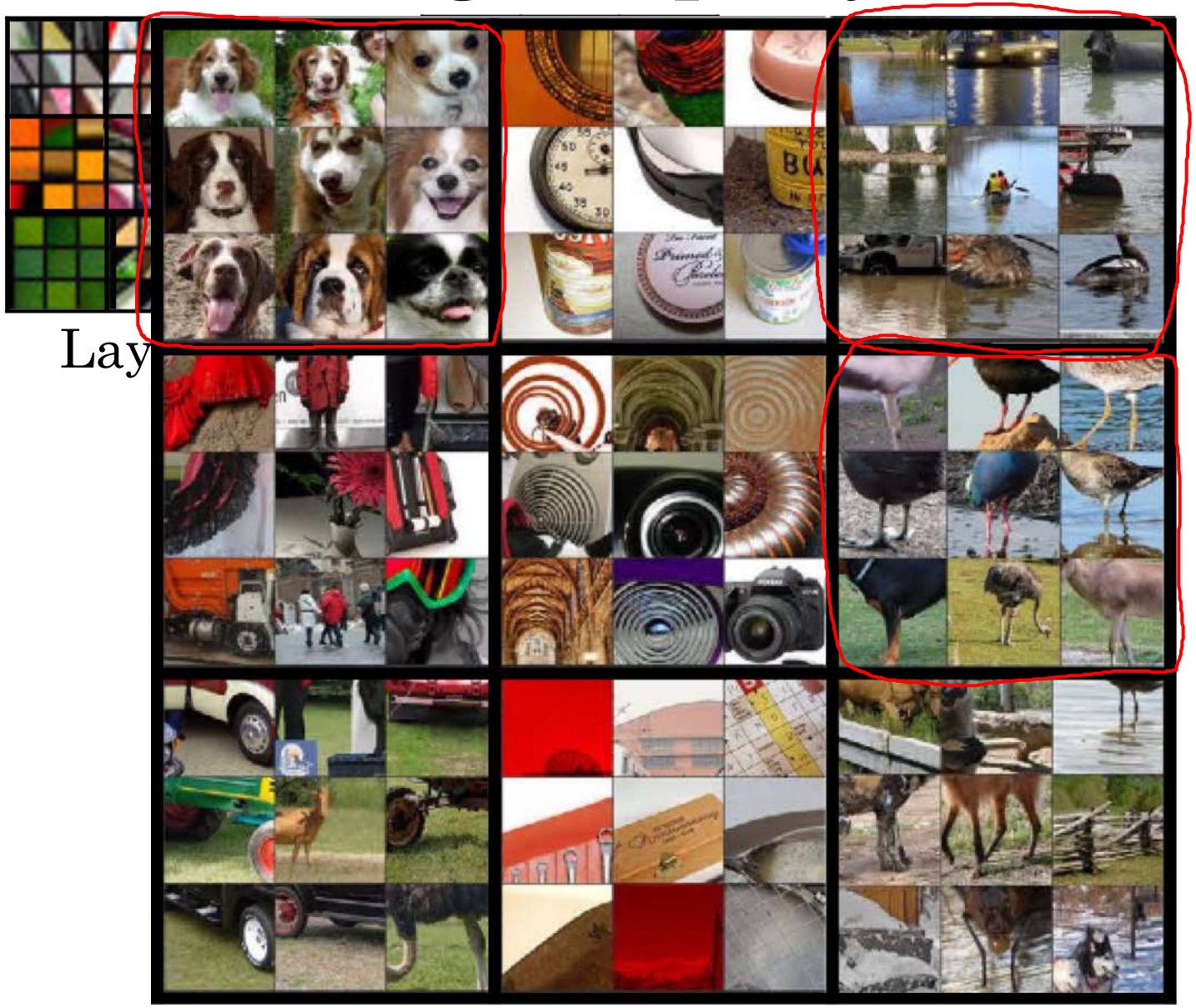
L

4

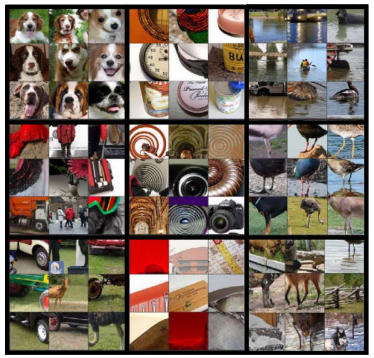


Layer 5

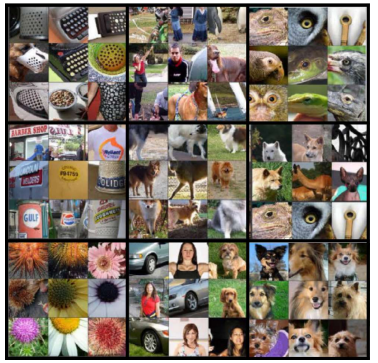
Visualizing deep layers: Layer 4



Lay

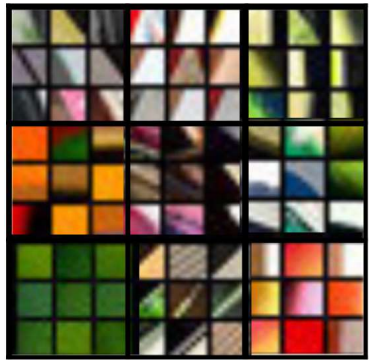


Layer 4

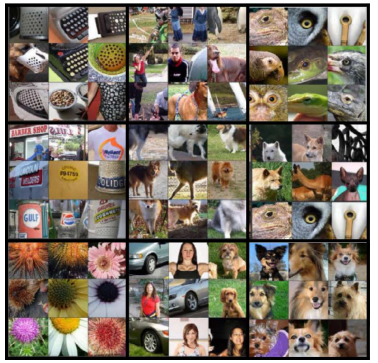


Layer 5

Visualizing deep layers: Layer 5



Layer 1



Layer 5

Andrew Ng

deeplearning.ai

Neural Style Transfer

Cost function

Neural style transfer cost function

Content C

Style S

Generated image G

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

[Gatys et al., 2015. A neural algorithm of artistic style. Images on slide generated by Justin Johnson]

Find the generated image G

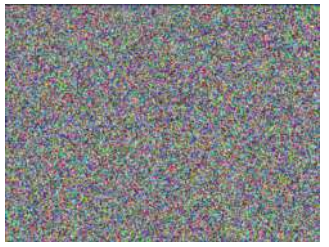
1. Initiate G randomly

   G: 100×100×3

   ↑ RGB

2. Use gradient descent to minimize $J(G)$

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



[Gatys et al., 2015. A neural algorithm of artistic style]

deeplearning.ai

Neural Style Transfer

Content cost function

Content cost function

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

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

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

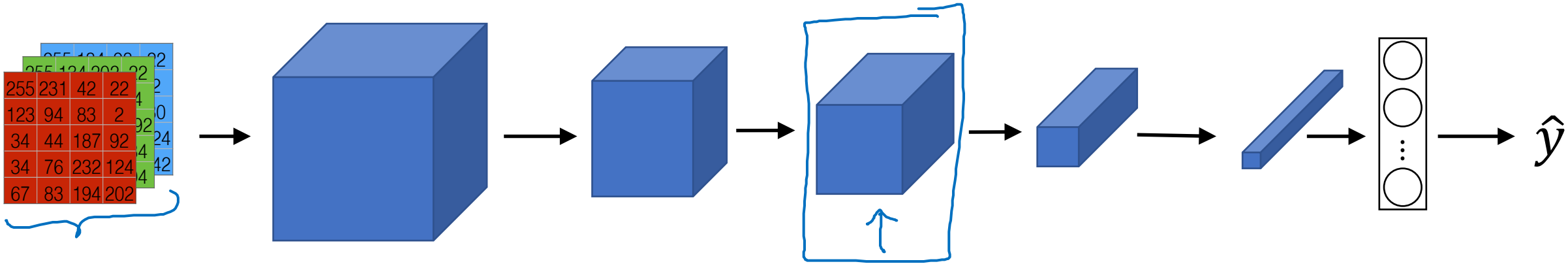
[Gatys et al., 2015. A neural algorithm of artistic style]

deeplearning.ai

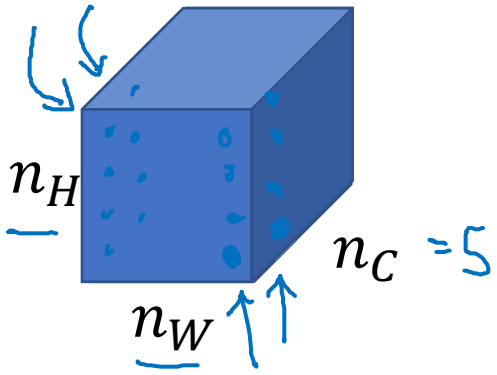
Neural Style Transfer

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.

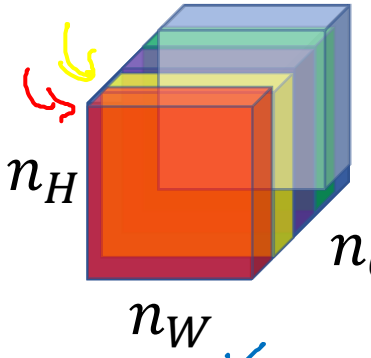


How correlated are the activations across different channels?

[Gatys et al., 2015. A neural algorithm of artistic style]

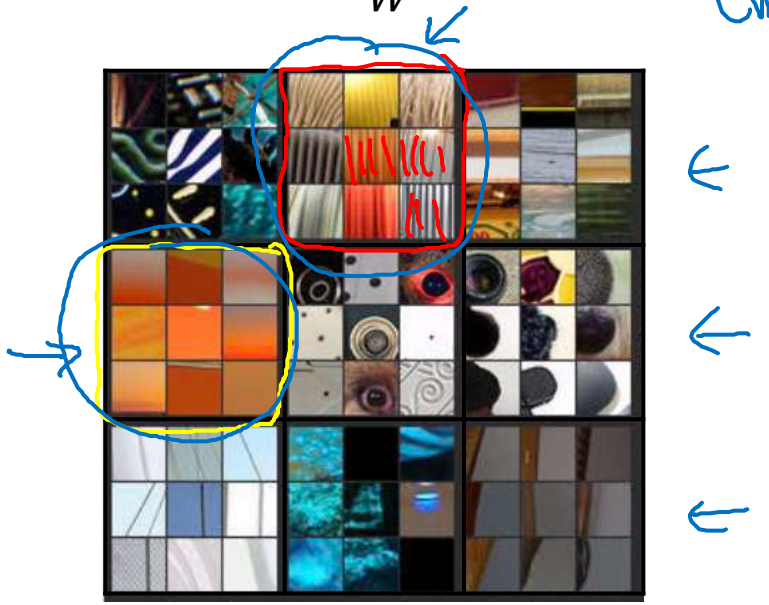
Intuition about style of an image

Style image

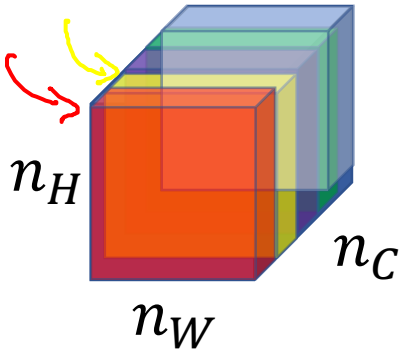


Correlated?

Uncorrelated



Generated Image



[Gatys et al., 2015. A neural algorithm of artistic style]

Style matrix

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

H W C

n_c

$G^{[l]}_{kk'}$

$k = 1, \ldots, n_c^{[l]}$

$$G^{[l](S)}_{kk'} = \sum_{i=1}^{n_H^{[l]}} \sum_{j=1}^{n_W^{[l]}} a^{[l](S)}_{ijk} \, a^{[l](S)}_{ijk'}$$

$$G^{[l](G)}_{kk'} = \sum_{i=1}^{n_H^{[l]}} \sum_{j=1}^{n_W^{[l]}} a^{[l](G)}_{ijk} \, a^{[l](G)}_{ijk}$$

"Gram matrix"

$$J^{[l]}_{style}(S, G) = \frac{1}{(\ldots)} \left\| G^{[l](S)} - G^{[l](G)} \right\|^2_F$$

$$= \frac{1}{(2 n_H^{[l]} n_W^{[l]} n_c^{[l]})^2} \sum_k \sum_{k'} \left(G^{[l](S)}_{kk'} - G^{[l](G)}_{kk'} \right)^2$$

β

[Gatys et al., 2015. A neural algorithm of artistic style]

Style cost function

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

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

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

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

G

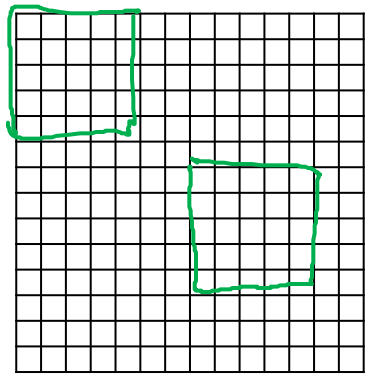
[Gatys et al., 2015. A neural algorithm of artistic style]

deeplearning.ai

Convolutional Networks in 1D or 3D

1D and 3D generalizations of models

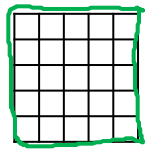
Convolutions in 2D and 1D



2D input image

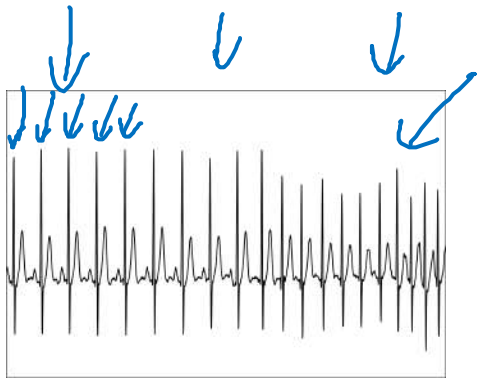
14×14

*

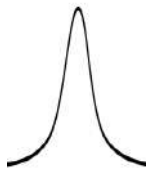


2D filter

5×5



*

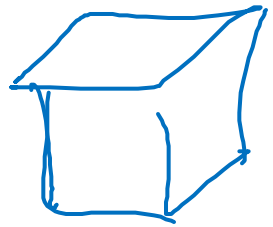


1	20	15	3	18	12	4	17

14

1	3	10	3	1

5



$14 \times 14 \times 3 \quad * \quad 5 \times 5 \times 3$

$\rightarrow 10 \times 10 \times 16$

$10 \times 10 \times 16 \quad * \quad 5 \times 5 \times 16$

$\rightarrow 6 \times 6 \times 32$

$14 \times 1 \quad * \quad 5 \times 1$

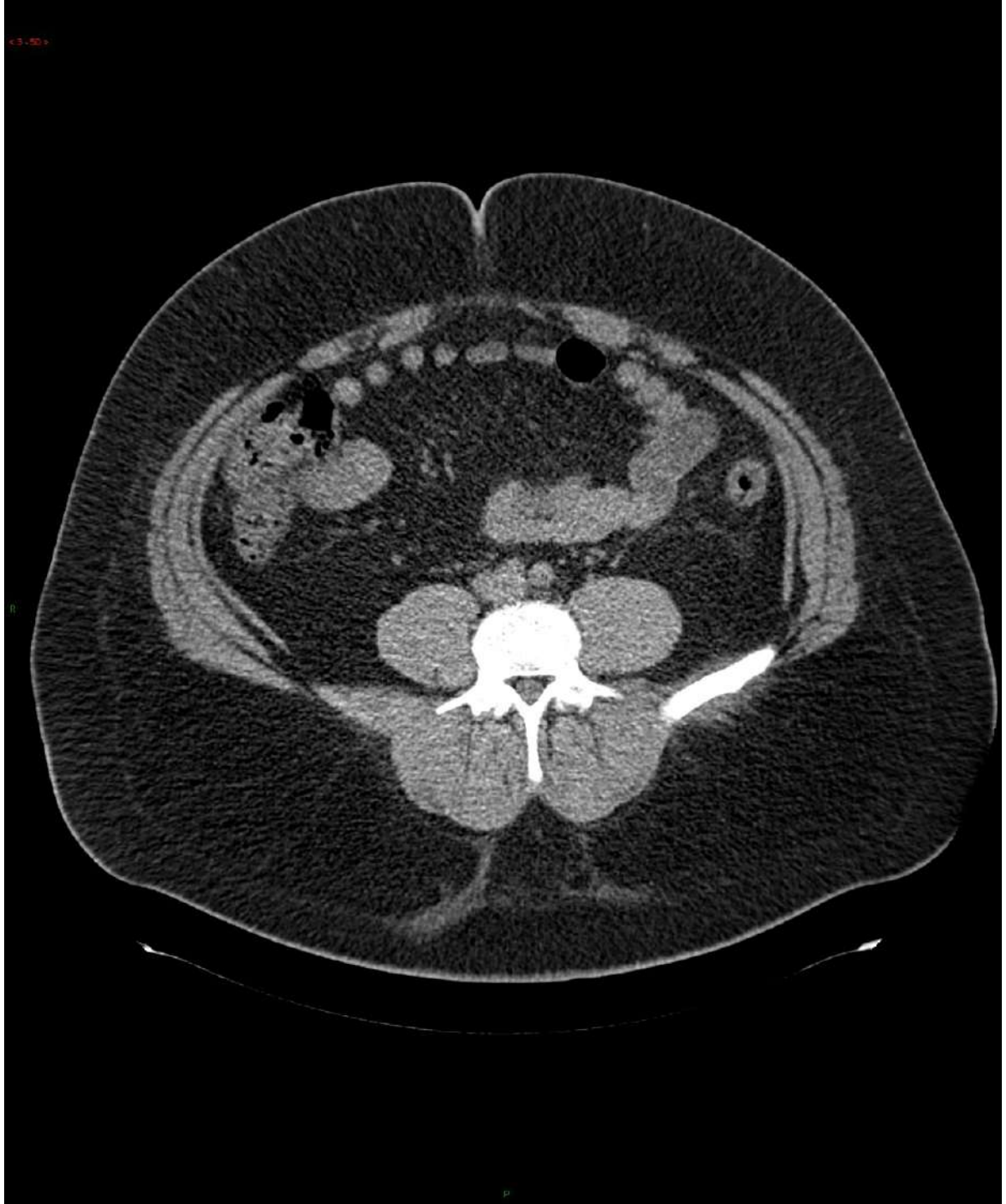
$\rightarrow 10 \times 16$

$10 \times 16 \quad * \quad 5 \times 16$

$\rightarrow 6 \times 32$

Andrew Ng

3D data

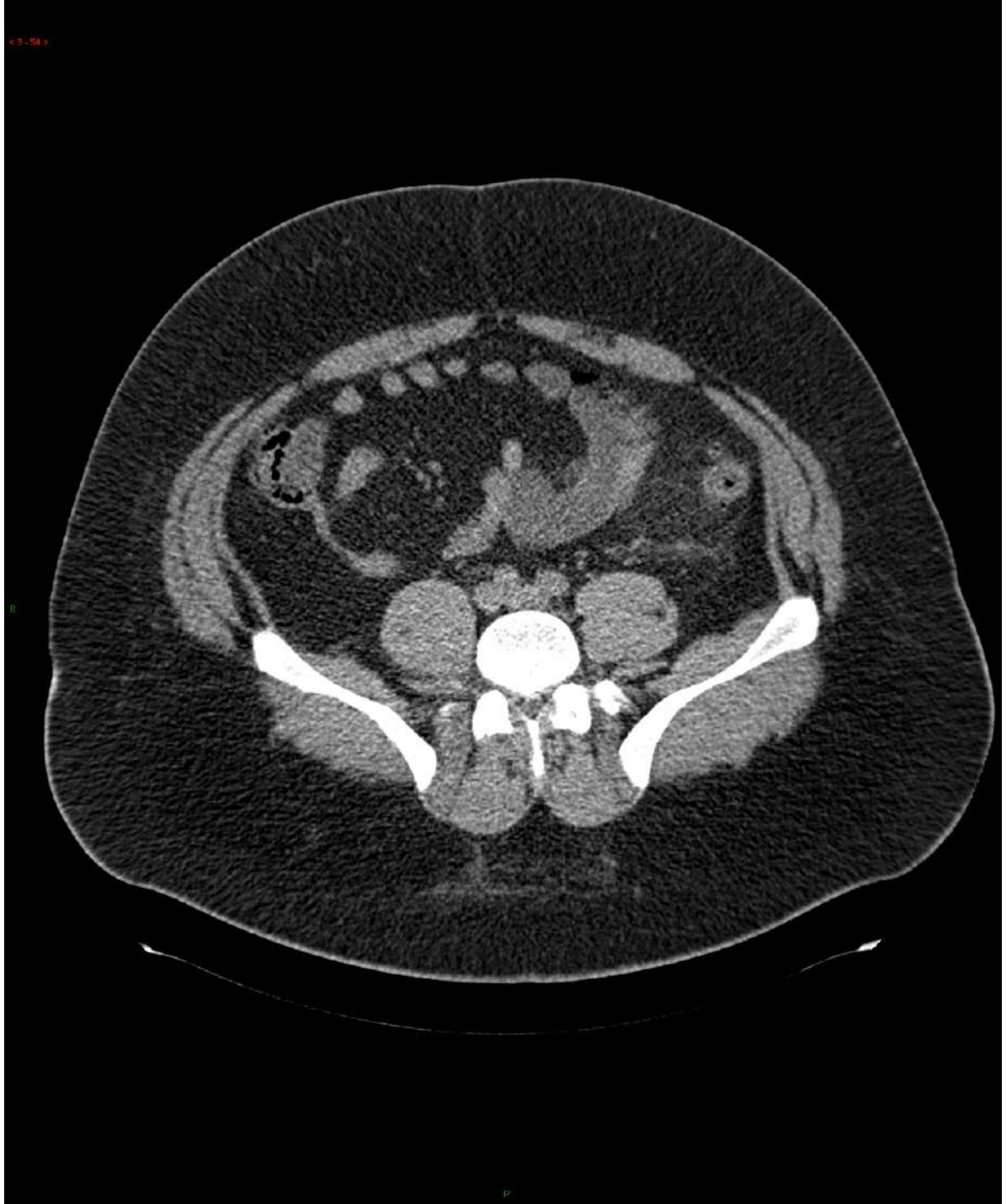


Andrew Ng

3D data

Andrew Ng

3D data

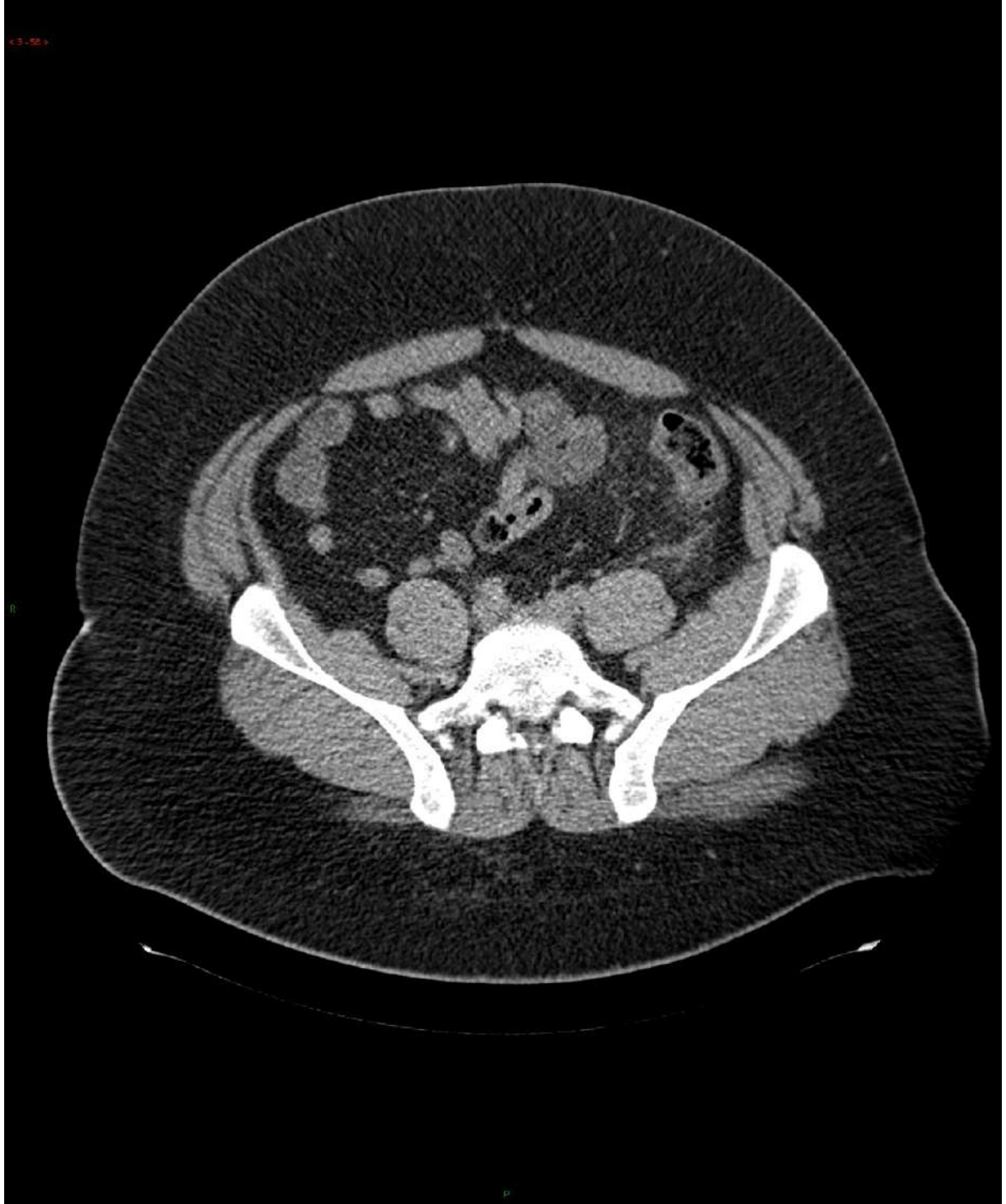


Andrew Ng

3D data

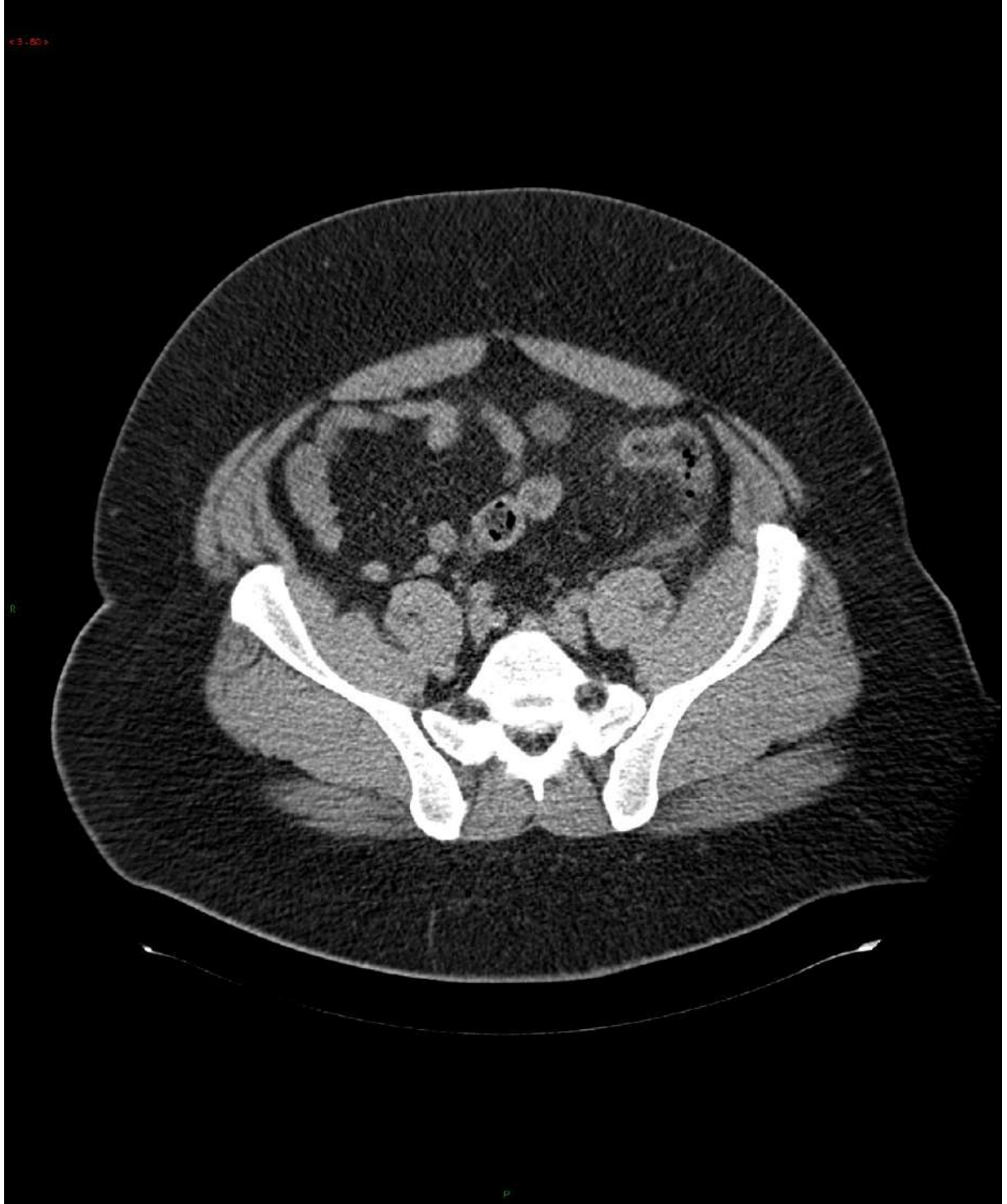
Andrew Ng

3D data



Andrew Ng

3D data



Andrew Ng

3D data

Andrew Ng

3D data

Andrew Ng

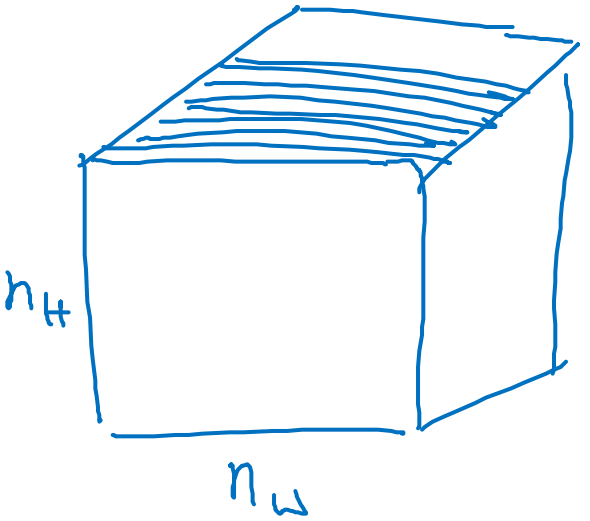
3D data

Andrew Ng

3D data

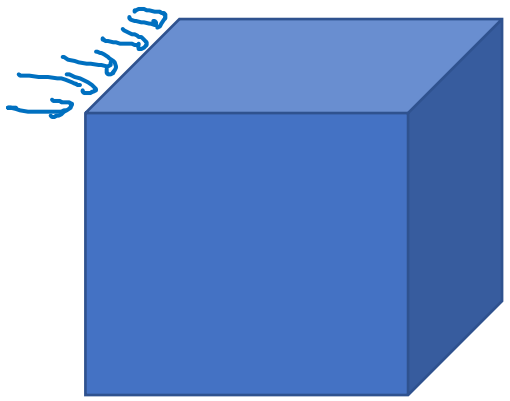
Andrew Ng

3D data



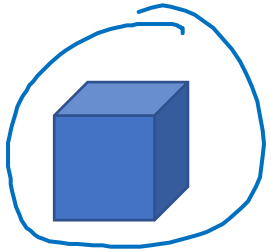
Andrew Ng

3D convolution



3D volume

*



3D filter

$14 \times 14 \times 14 \times 1$ (n_c)

$* \; 5 \times 5 \times 5 \times 1$

16 filters.

$\rightarrow 10 \times 10 \times 10 \times 16$

$* \; 5 \times 5 \times 5 \times 16$

32 filters

$\rightarrow 6 \times 6 \times 6 \times 32$