

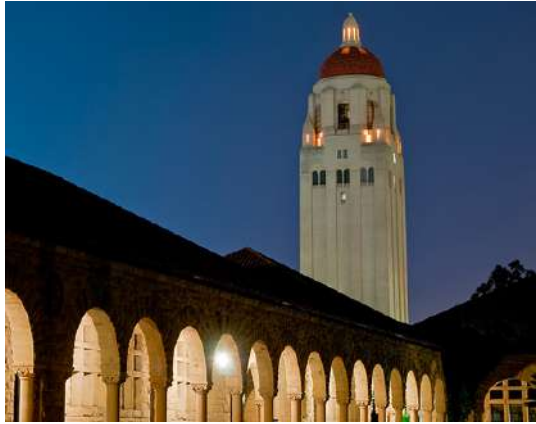


deeplearning.ai

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)

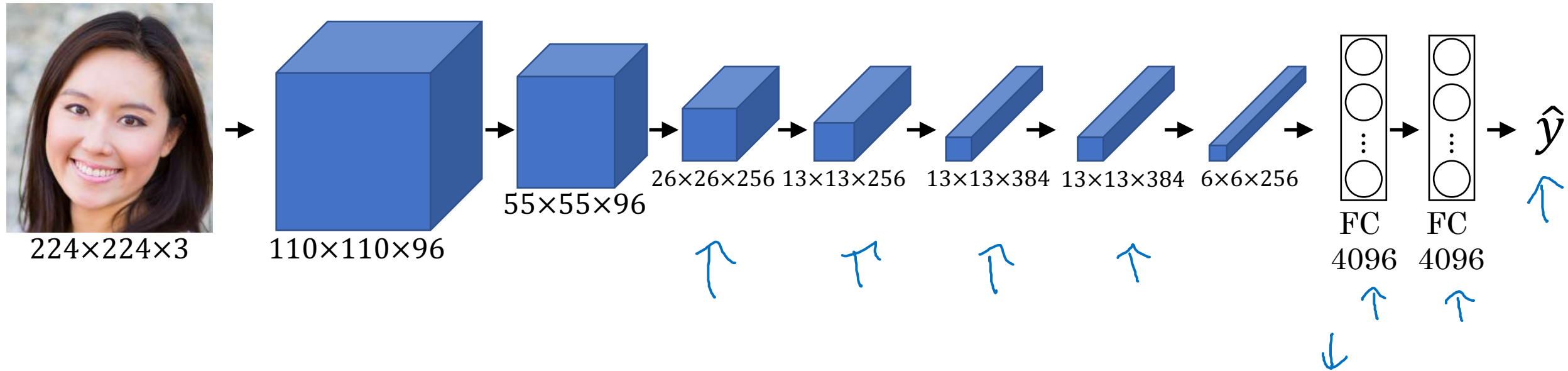


deeplearning.ai

Neural Style Transfer

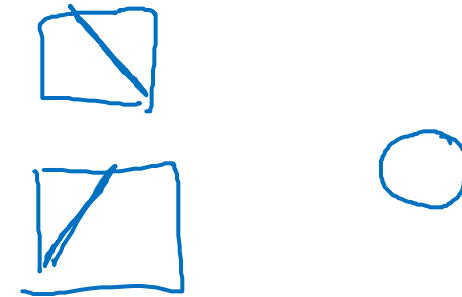
What are deep
ConvNets learning?

Visualizing what a deep network is learning

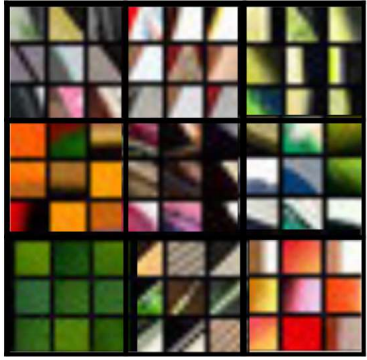


Pick a unit in layer 1. Find the nine image patches that maximize the unit's activation.

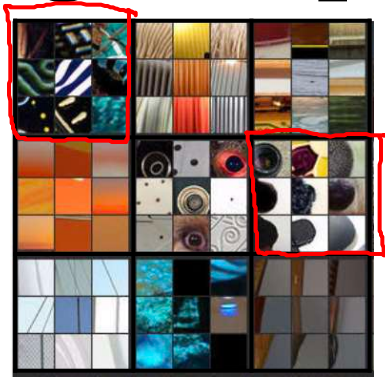
Repeat for other units.



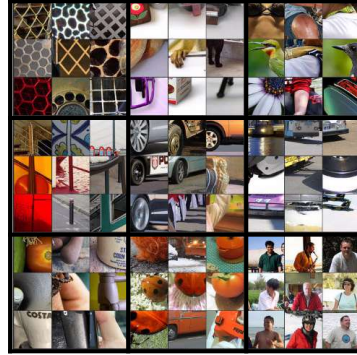
Visualizing deep layers



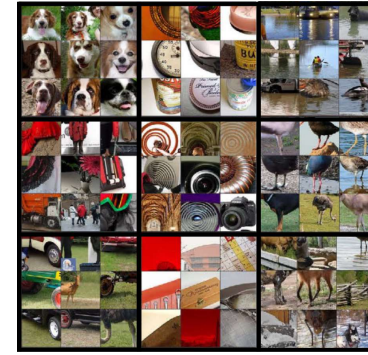
Layer 1



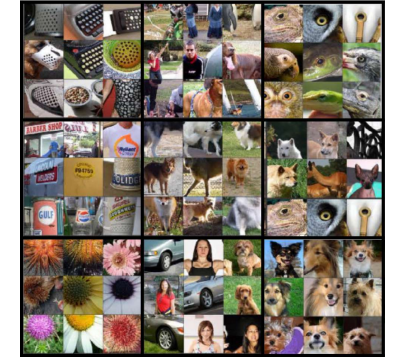
Layer 2



Layer 3



Layer 4



Layer 5

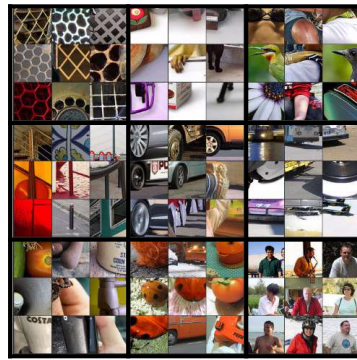
Visualizing deep layers: Layer 1



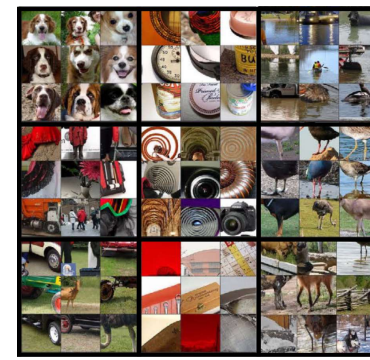
Layer 1



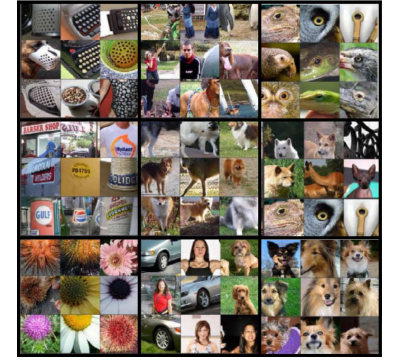
Layer 2



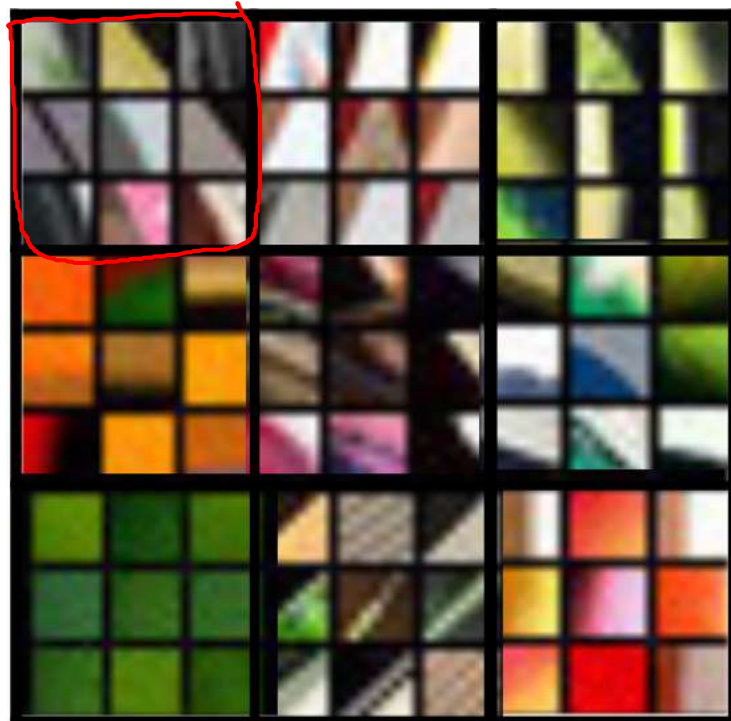
Layer 3



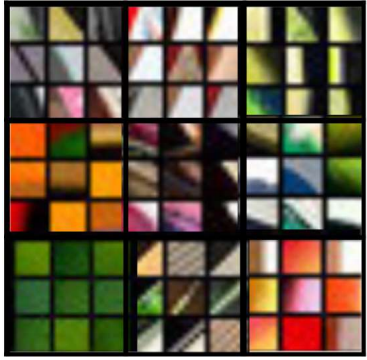
Layer 4



Layer 5



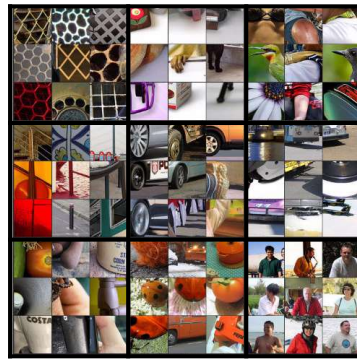
Visualizing deep layers: Layer 2



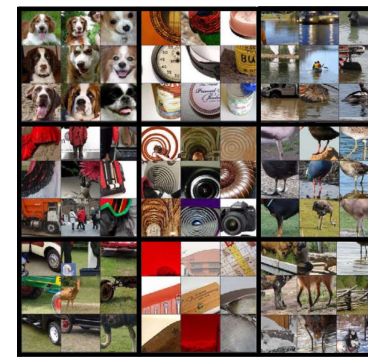
Layer 1



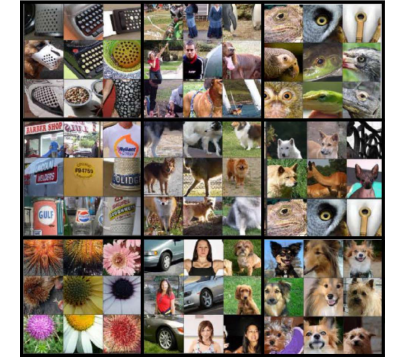
Layer 2



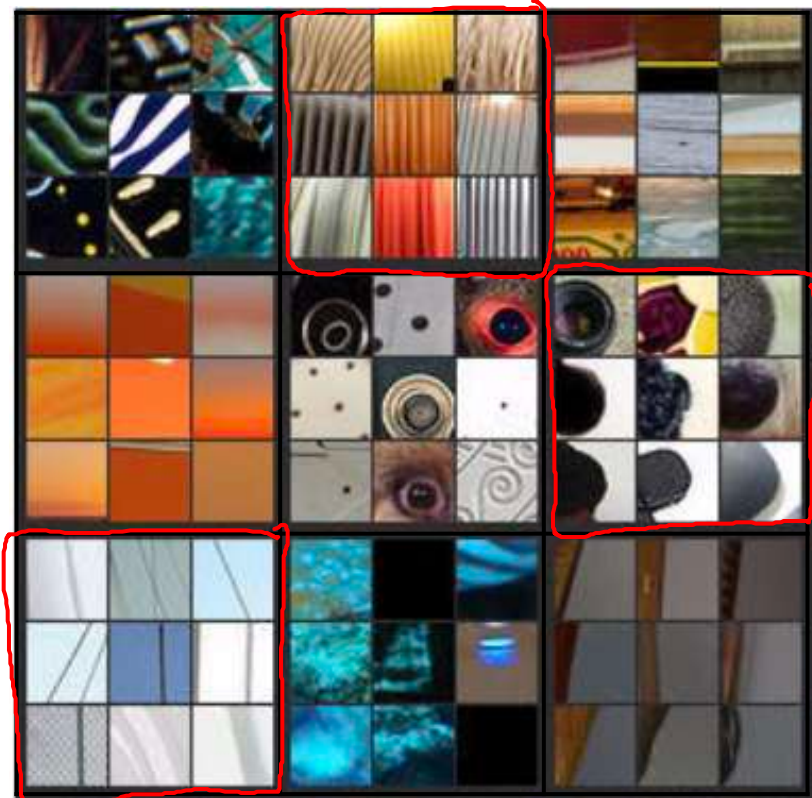
Layer 3



Layer 4



Layer 5



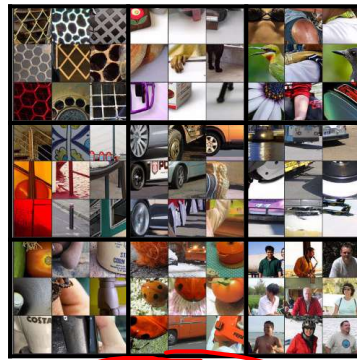
Visualizing deep layers: Layer 3



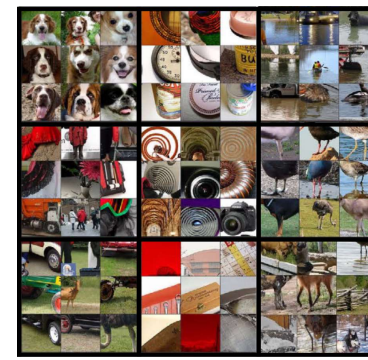
Layer 1



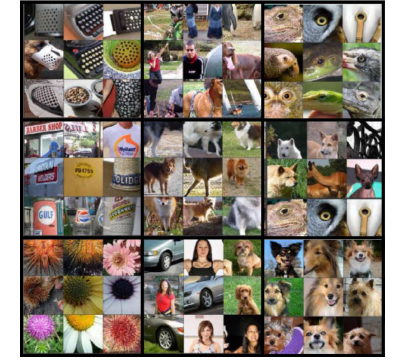
Layer 2



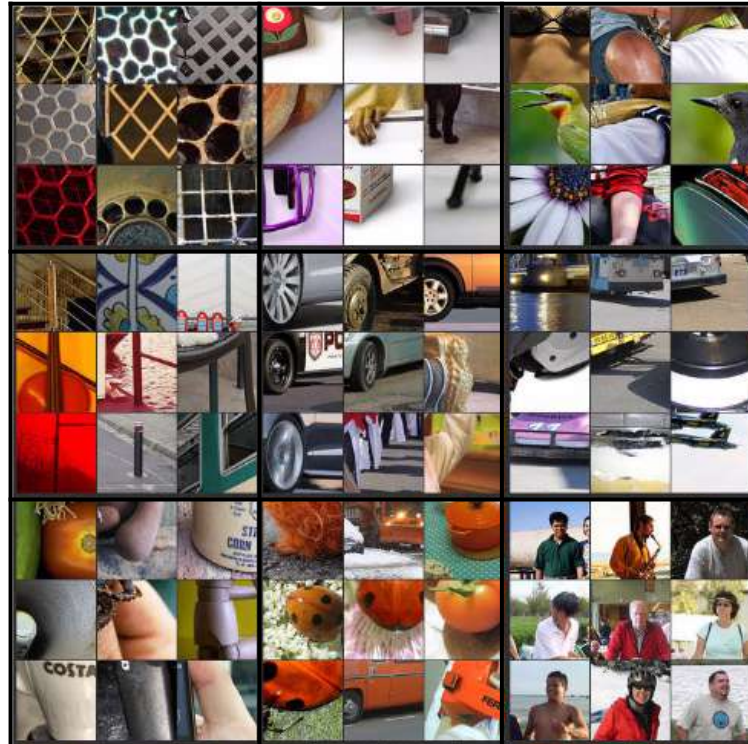
Layer 3



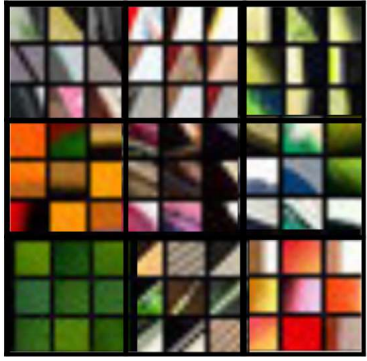
Layer 4



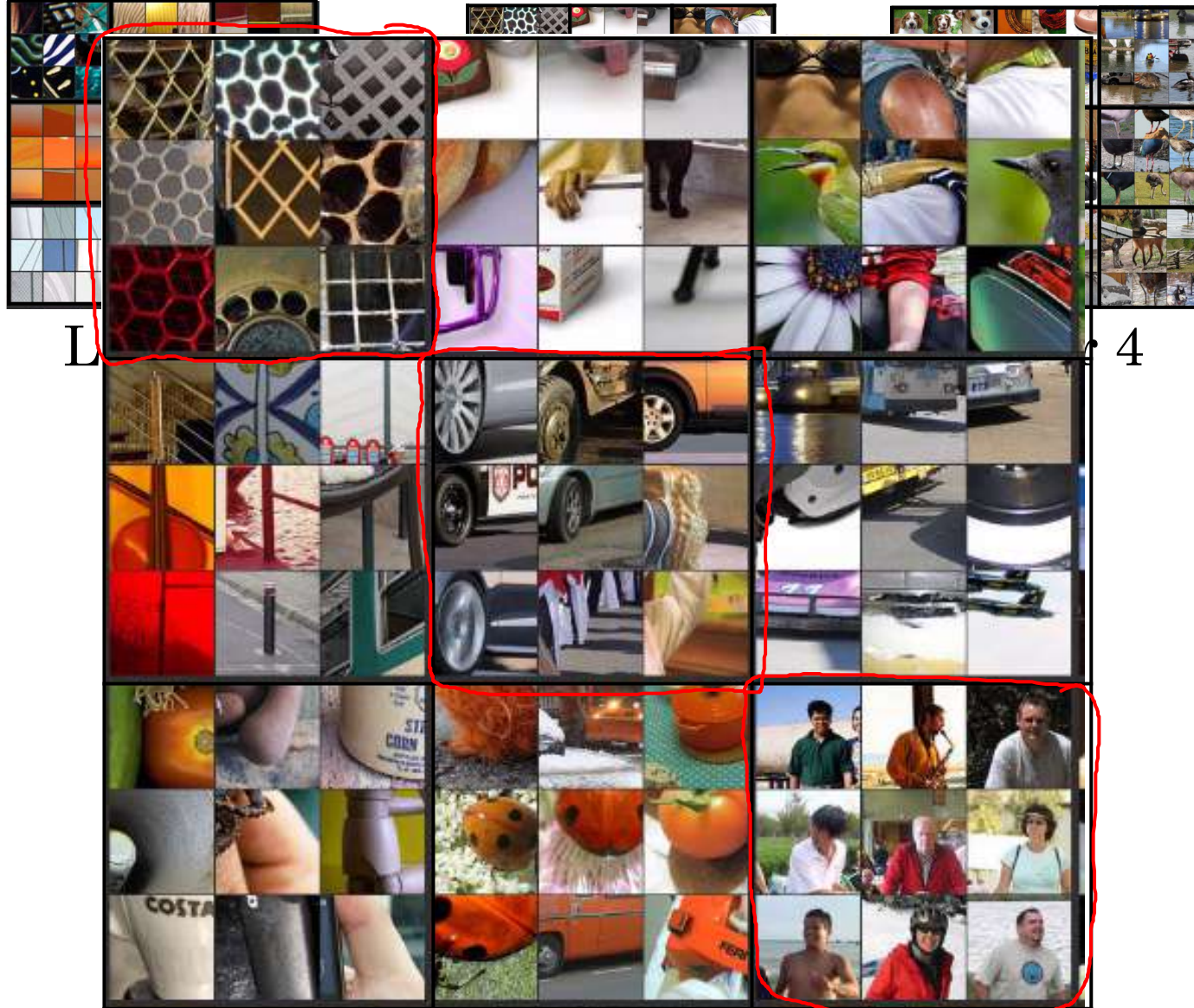
Layer 5



Visualizing deep layers: Layer 3

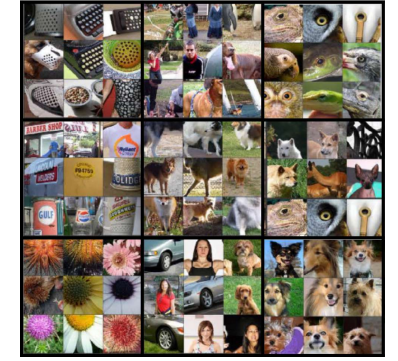


Layer 1



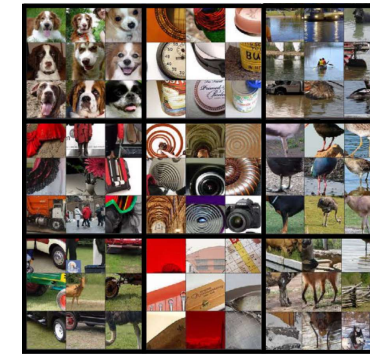
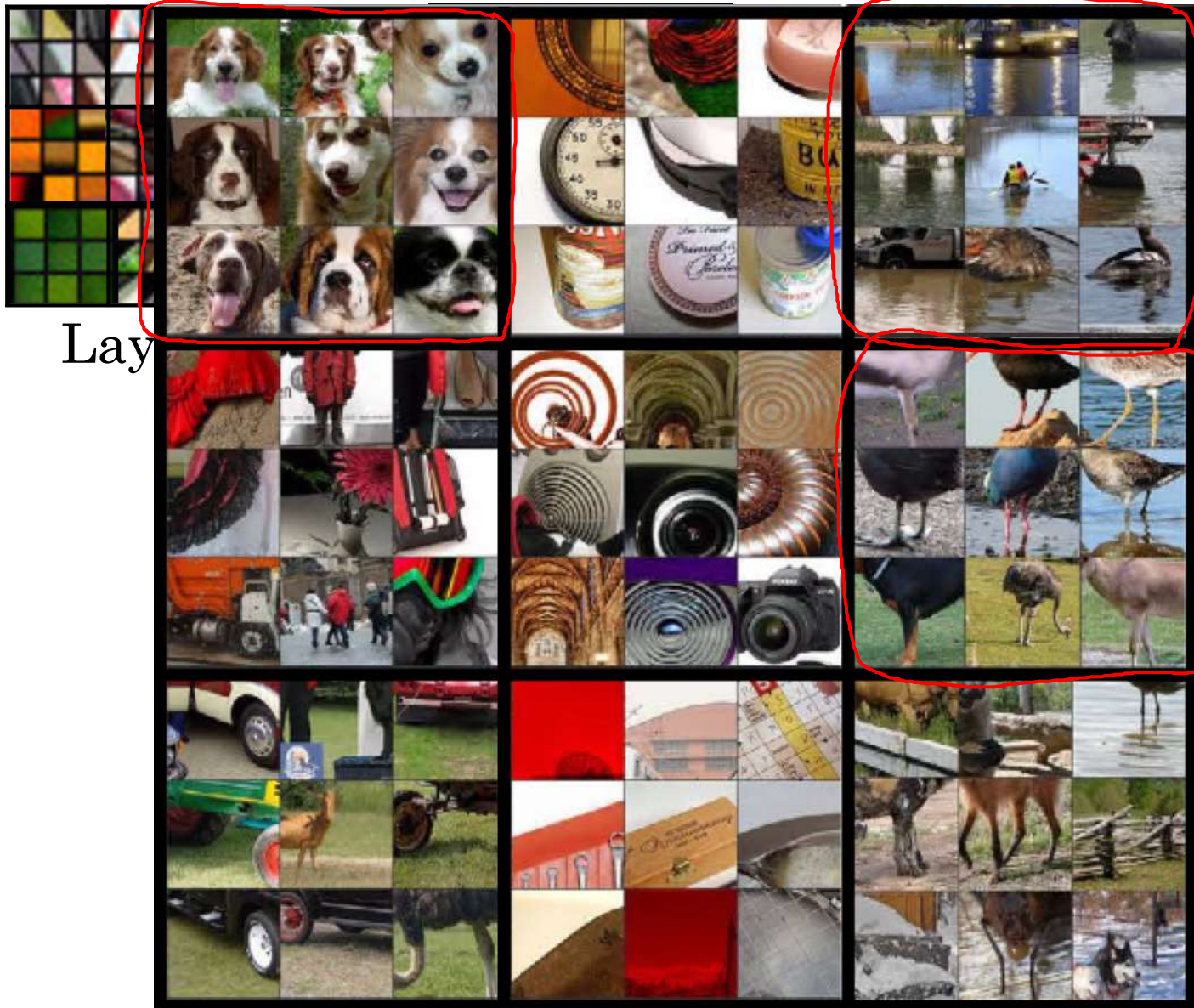
Layer 3

Layer 4

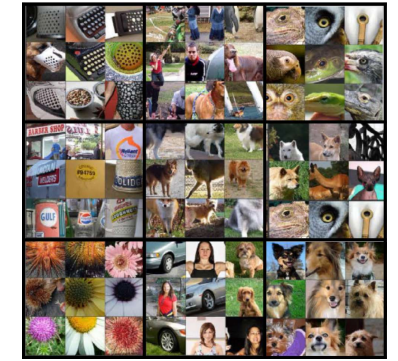


Layer 5

Visualizing deep layers: Layer 4

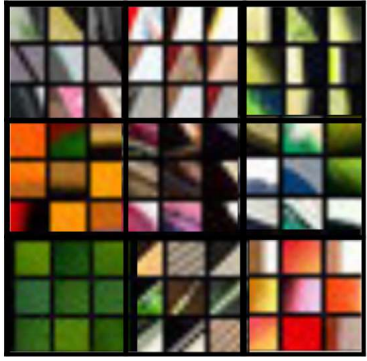


Layer 4

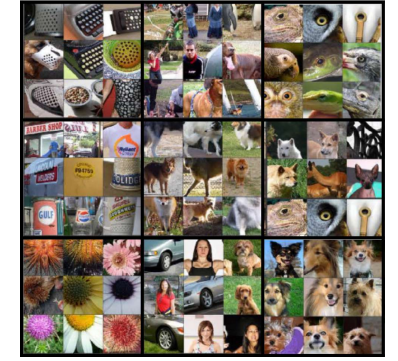


Layer 5

Visualizing deep layers: Layer 5



Layer 1



Layer 5



deeplearning.ai

Neural Style Transfer

Cost function

Neural style transfer cost function



Content C

Style S



Generated image G

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

Find the generated image G

1. Initiate G randomly

G : $100 \times 100 \times 3$

↑
RGB

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

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





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 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

$$J_{content}(C, G) = \frac{1}{2} \left\| \underbrace{a^{[l](C)}}_{\text{activation of layer } l \text{ on } C} - \underbrace{a^{[l](G)}}_{\text{activation of layer } l \text{ on } G} \right\|^2$$

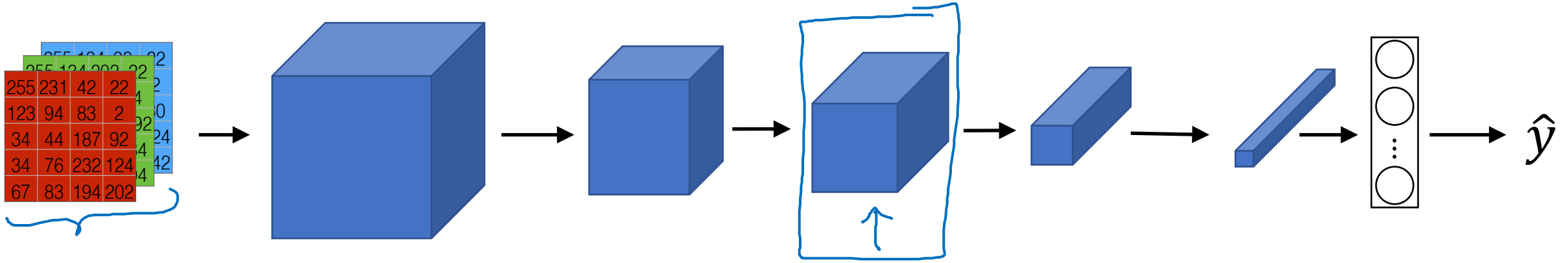


deeplearning.ai

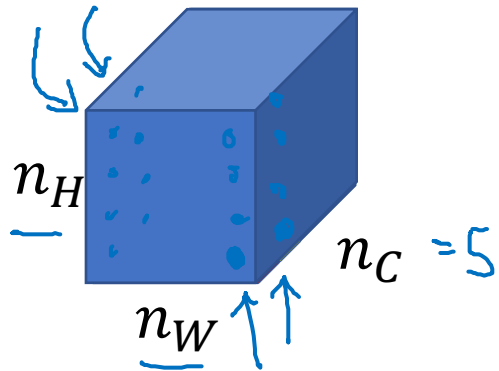
Neural Style Transfer

Style cost function

Meaning of the “style” of an image



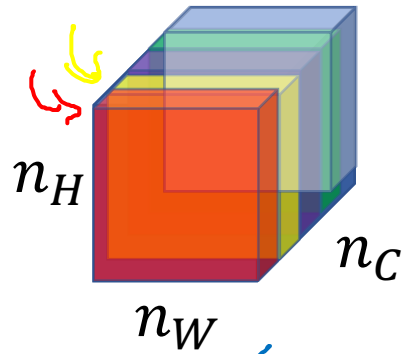
Say you are using layer l 's activation to measure “style.”
Define style as correlation between activations across channels.



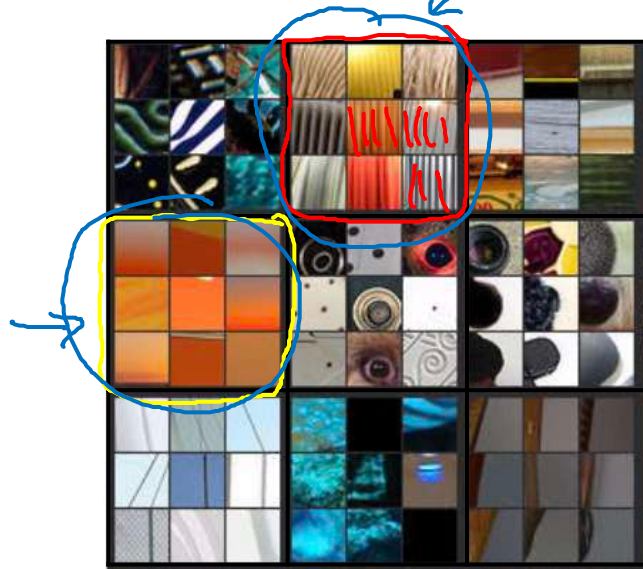
How correlated are the activations
across different channels?

Intuition about style of an image

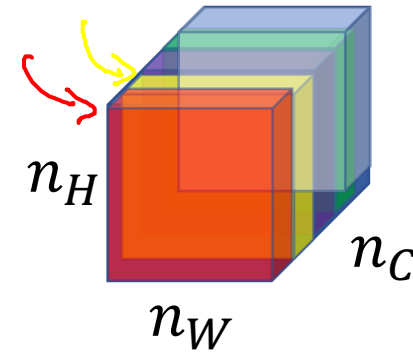
Style image



Correlated?
Uncorrelated



Generated Image



Style matrix

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

$$\begin{aligned} \rightarrow \underline{G_{kk'}^{[l](S)}} &= \sum_{i=1}^{n_H^{[l]}} \sum_{j=1}^{n_W^{[l]}} a_{ijk}^{[l](S)} a_{ijk'}^{[l](S)} \\ \rightarrow \underline{G_{kk'}^{[l](G)}} &= \sum_{i=1}^{n_H^{[l]}} \sum_{j=1}^{n_W^{[l]}} a_{ijk}^{[l](G)} a_{ijk}^{[l](G)} \end{aligned}$$

$$\begin{aligned} & n_c \\ & G_{kk'}^{[l]} \\ & \uparrow \uparrow \\ & k = 1, \dots, n_c \end{aligned}$$

"Gram matrix"

$$\begin{aligned} \beta \uparrow J_{\text{style}}^{[l]}(S, G) &= \frac{1}{(\dots)} \left\| G^{[l](S)} - G^{[l](G)} \right\|_F^2 \\ &= \frac{1}{(2 n_H^{[l]} n_W^{[l]} n_c^{[l]})^2} \sum_k \sum_{k'} \left(G_{kk'}^{[l](S)} - G_{kk'}^{[l](G)} \right)^2 \end{aligned}$$

Style cost function

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

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

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

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

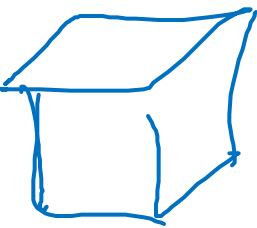


deeplearning.ai

Convolutional Networks in 1D or 3D

1D and 3D
generalizations of
models

Convolutions in 2D and 1D



$$14 \times 14 \times \underline{3} * 5 \times 5 \times \underline{3}$$

$$\rightarrow \underline{10 \times 10 \times 16}$$

$$\underline{10 \times 10 \times 16} * \underline{5 \times 5 \times 16}$$

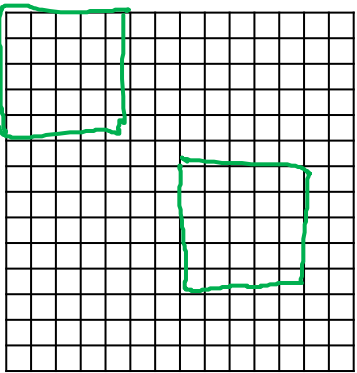
$$\rightarrow \underline{6 \times 6 \times 32}$$

$$14 \times \underline{1} * 5 \times \underline{1}$$

$$\rightarrow \underline{10 \times 16}$$

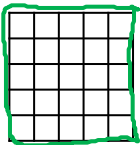
$$\underline{10 \times 16} * \underline{5 \times 16}$$

$$\rightarrow \underline{6 \times 32}$$

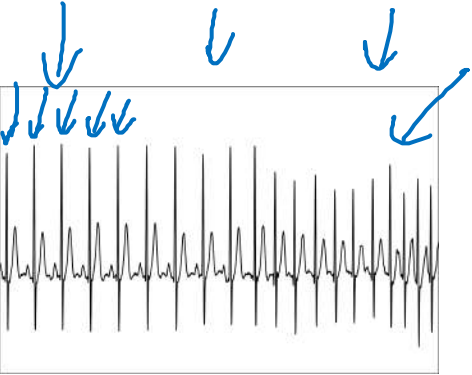


2D input image
14x14

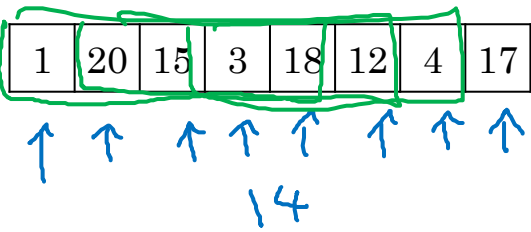
*



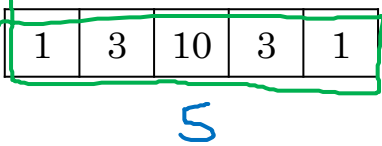
2D filter
5x5



*

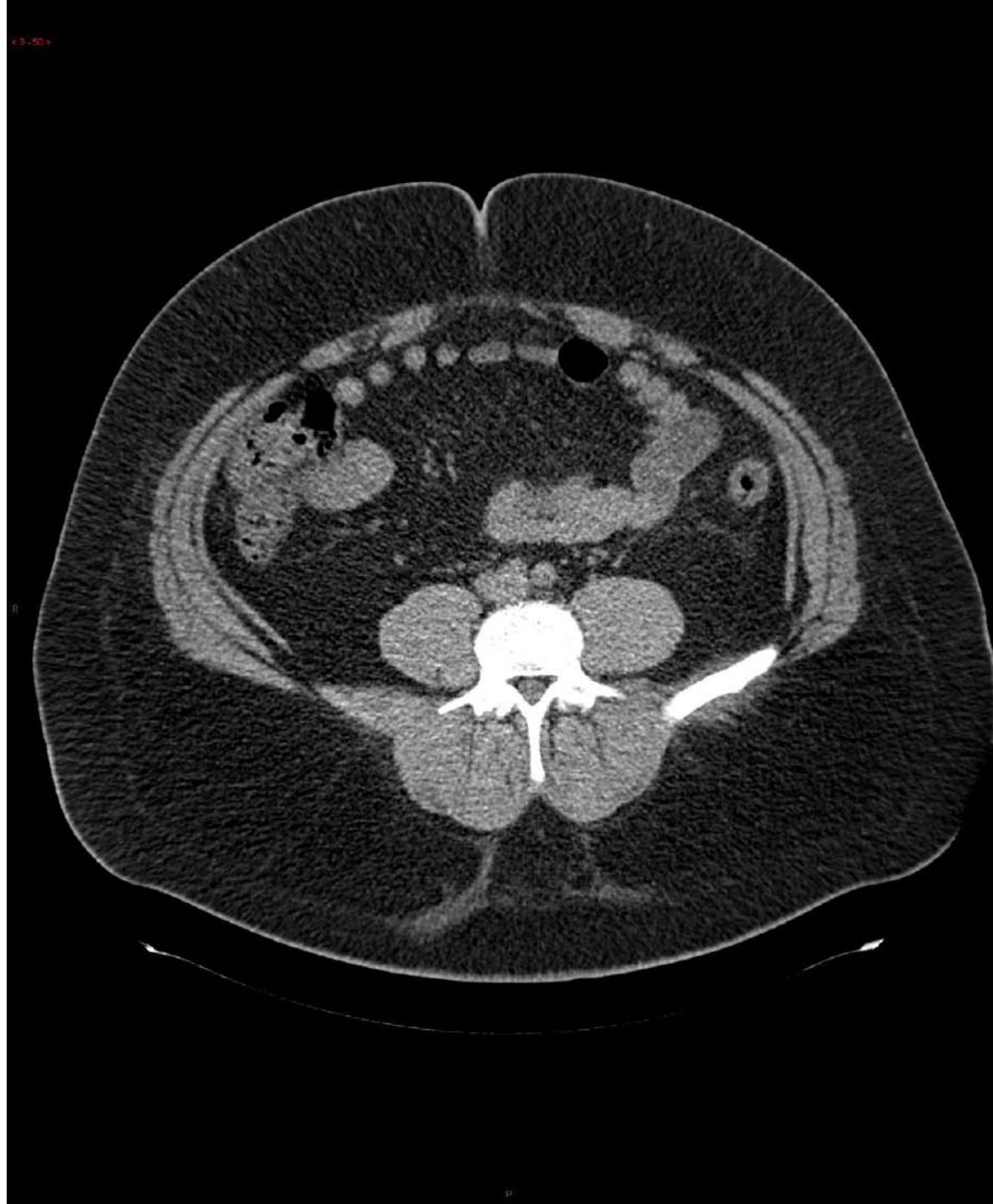


←



←

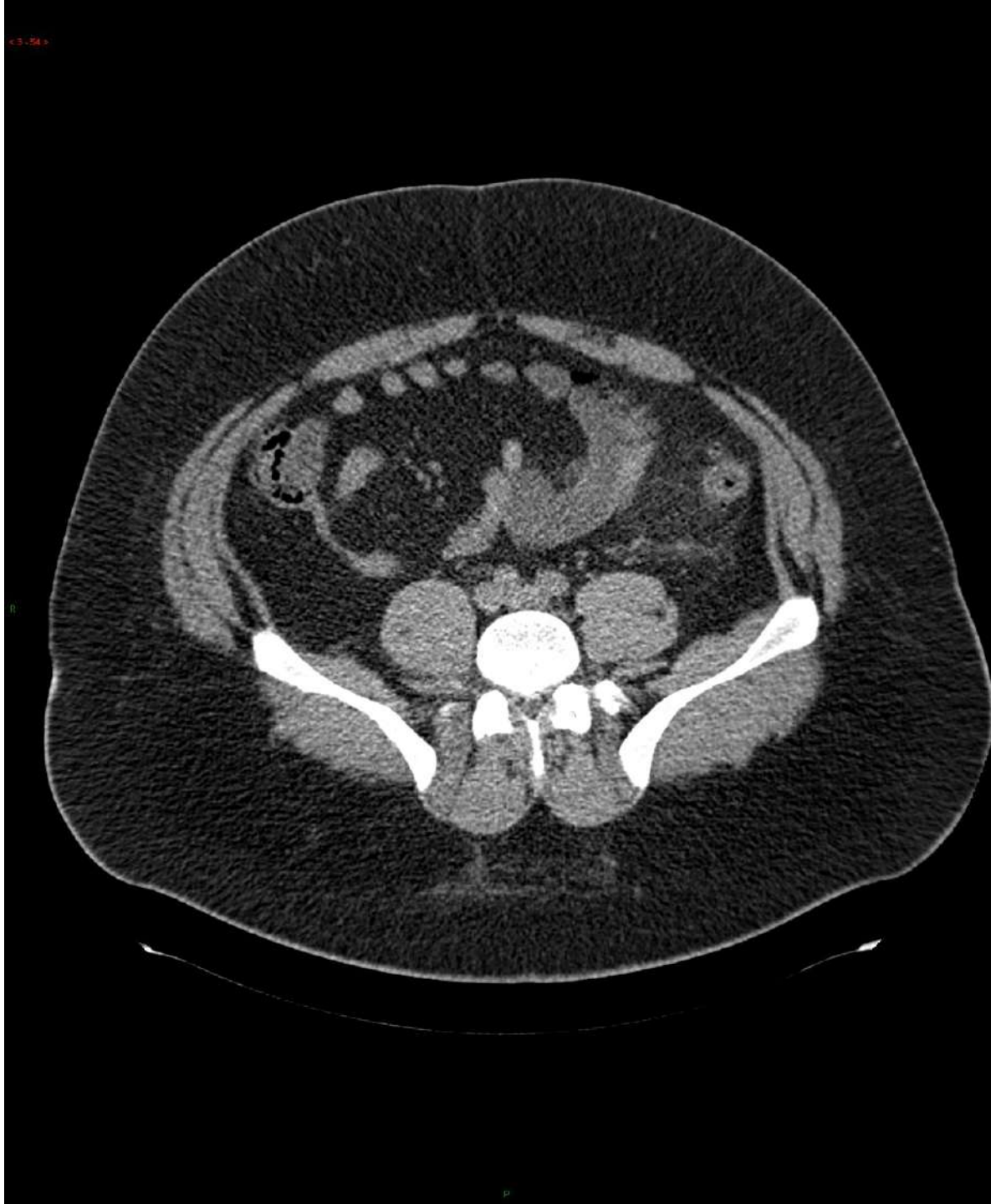
3D data



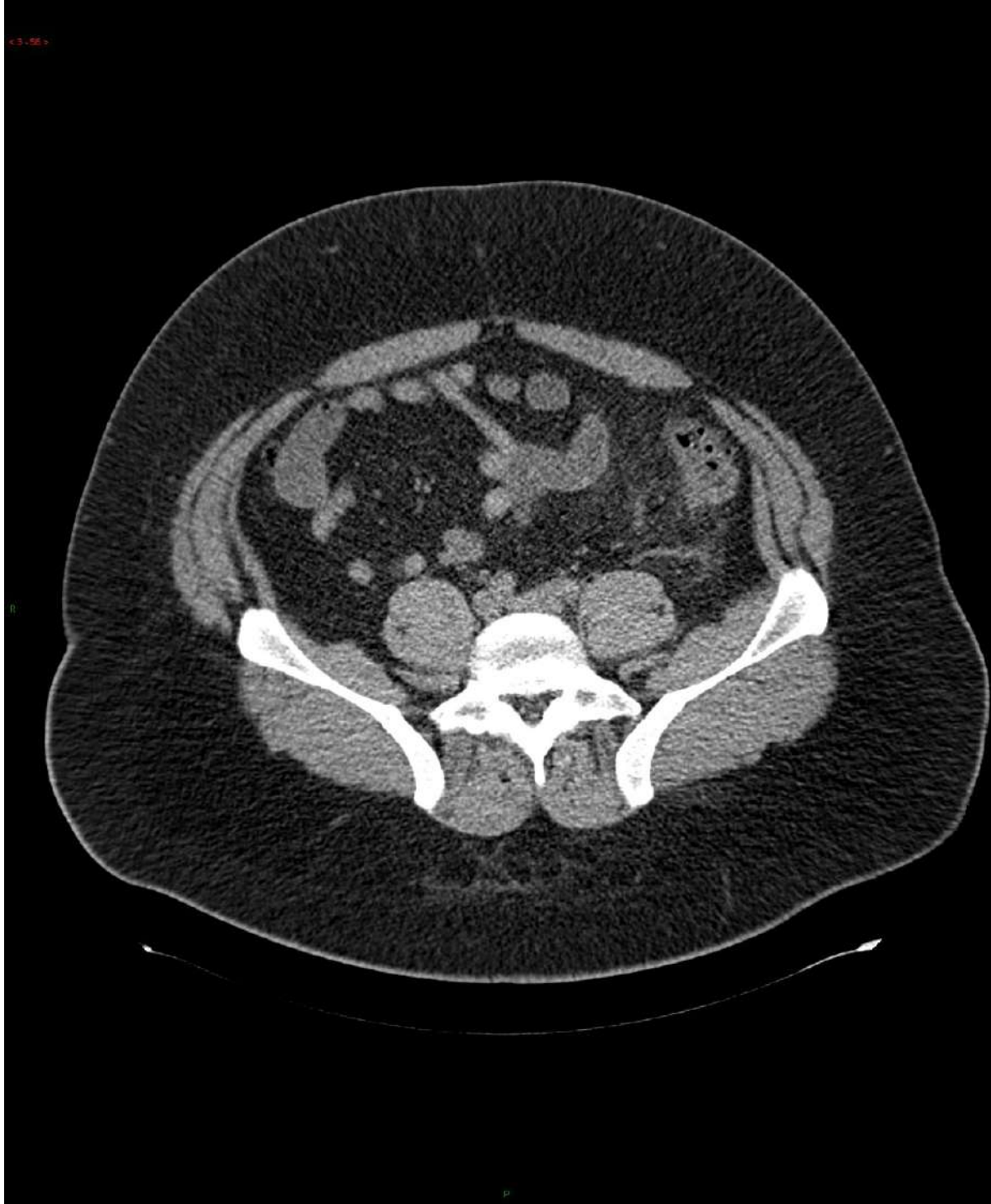
3D data



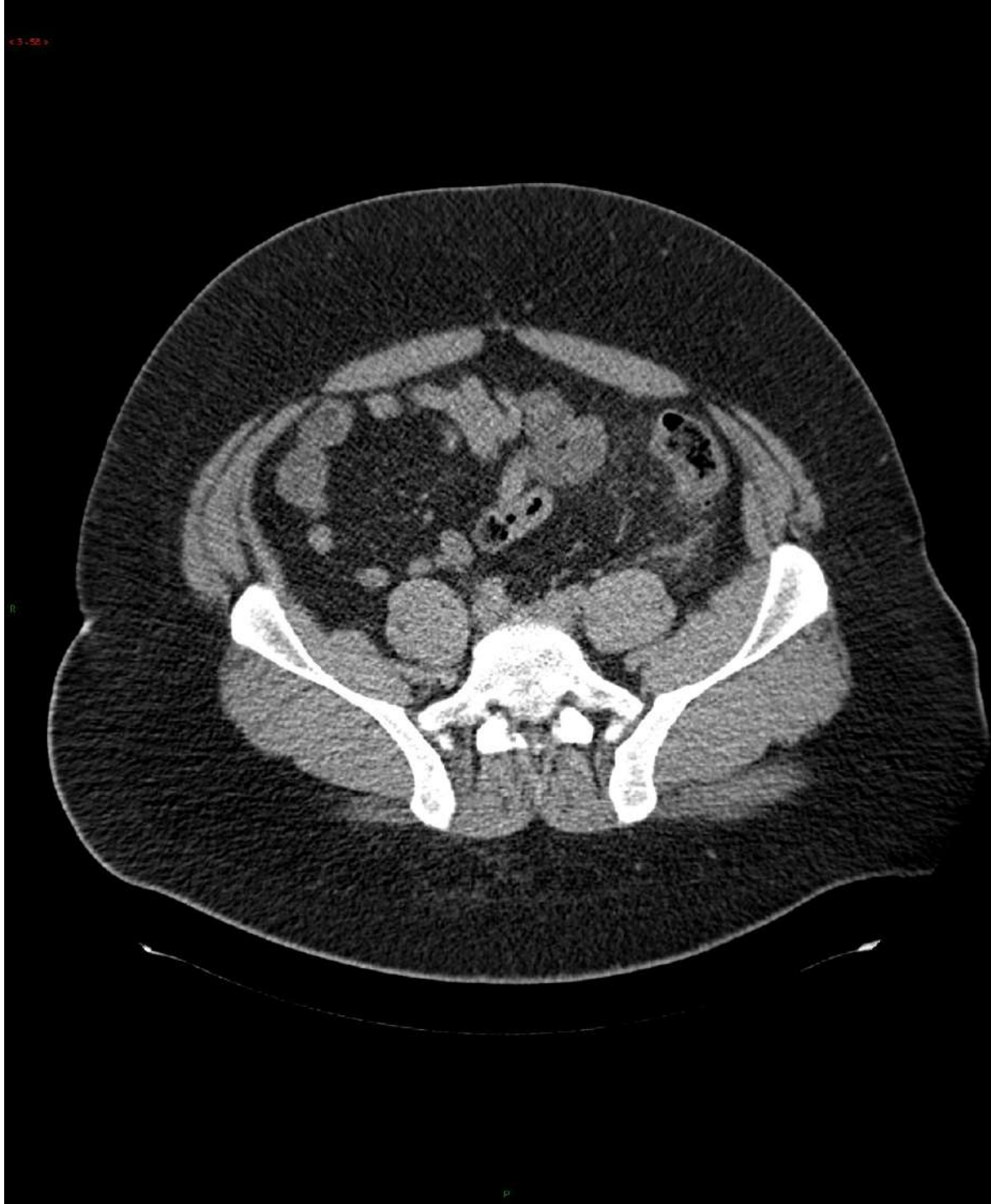
3D data



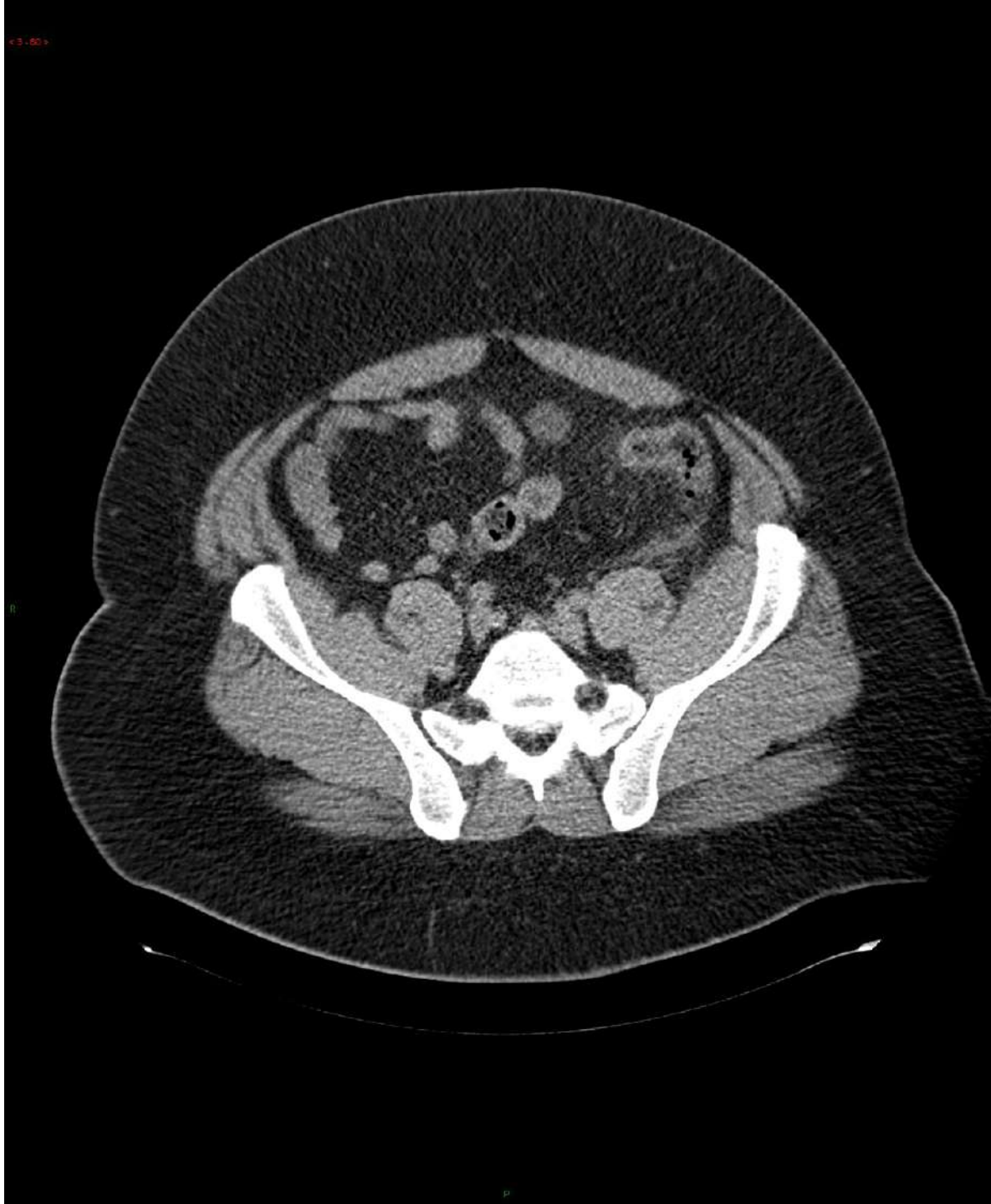
3D data



3D data



3D data



3D data



3D data



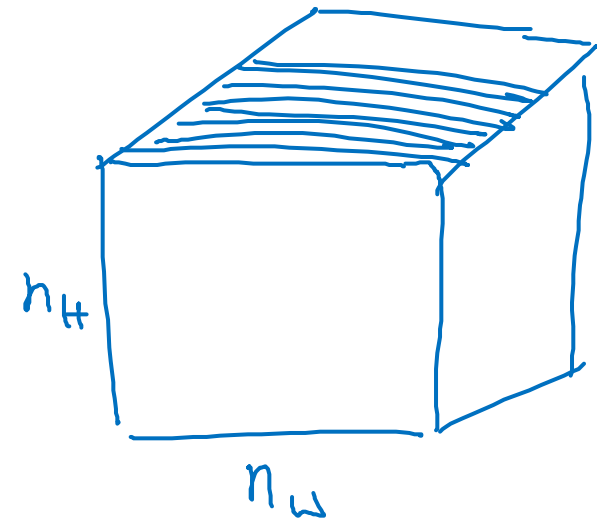
3D data



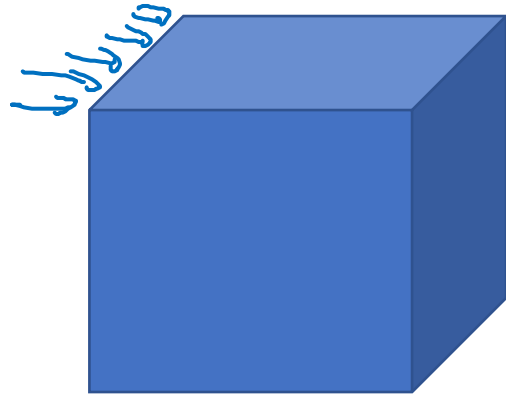
3D data



3D data



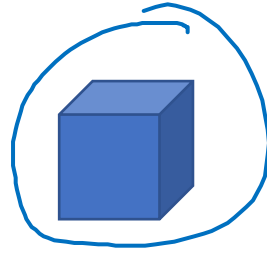
3D convolution



3D volume



*



3D filter

$$\begin{aligned} & \begin{array}{cccc} \downarrow & \downarrow & \downarrow & \downarrow^{n_c} \\ \underline{14 \times 14 \times 14} & \times & \underline{1} & \end{array} \\ & \quad \times \quad \underline{5 \times 5 \times 5 \times 1} \quad 16 \text{ filters} \\ & \rightarrow 10 \times 10 \times 10 \times \underline{16} \\ & \quad \times \quad \underline{5 \times 5 \times 5 \times 16} \\ & \rightarrow 6 \times 6 \times 6 \times 32 \quad 32 \text{ filters} \end{aligned}$$