

DS 6050 Deep Learning

3. Convolutional Neural Networks

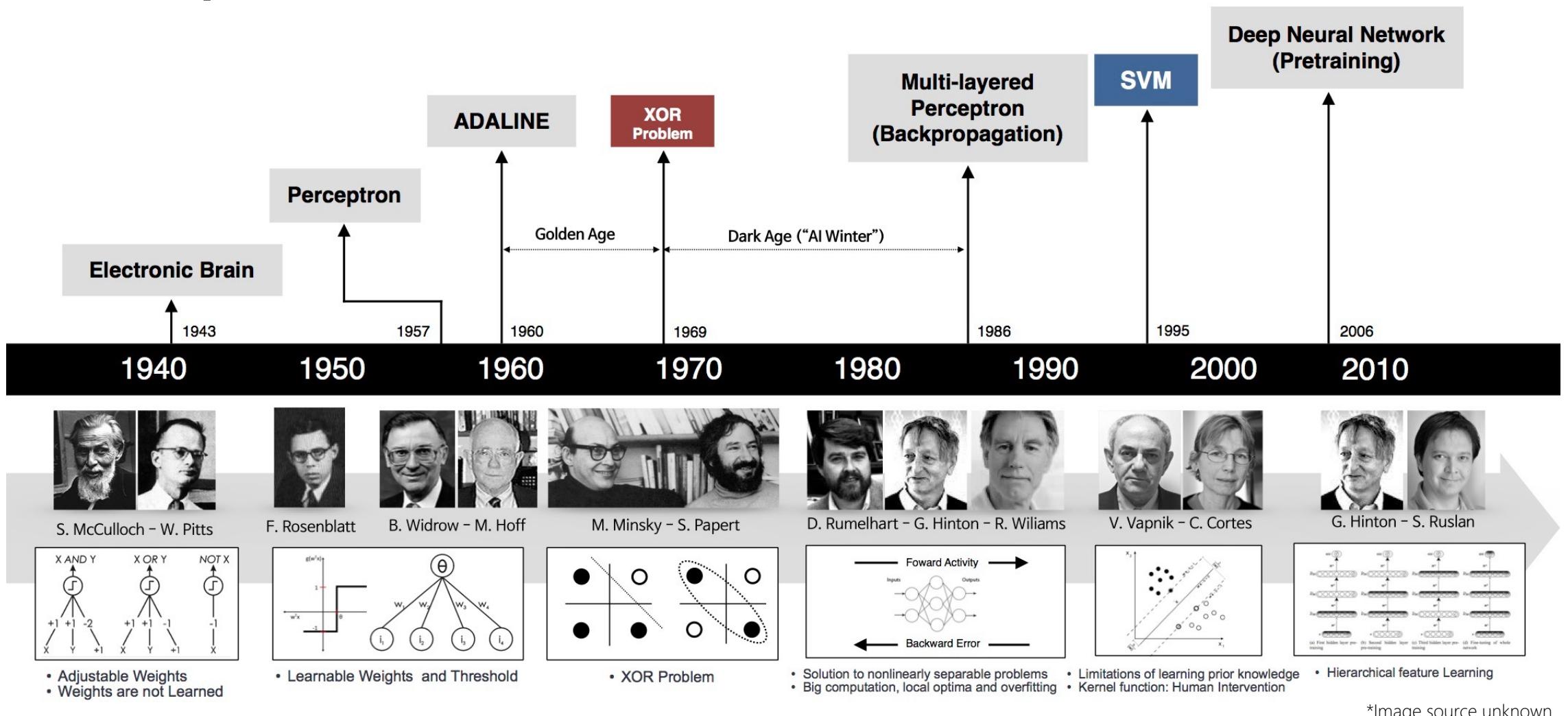
Sheng Li

Associate Professor

School of Data Science

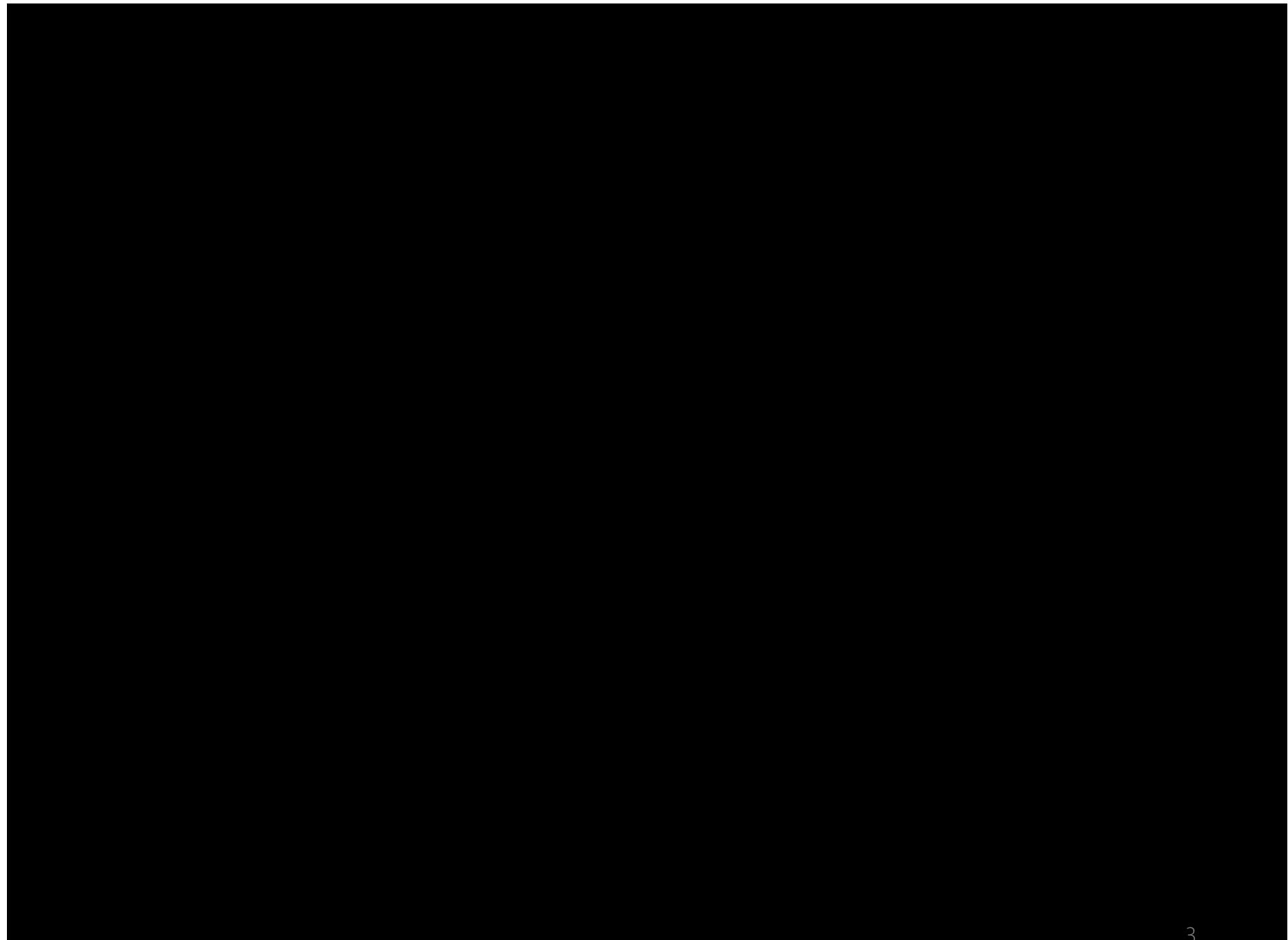
University of Virginia

Recap

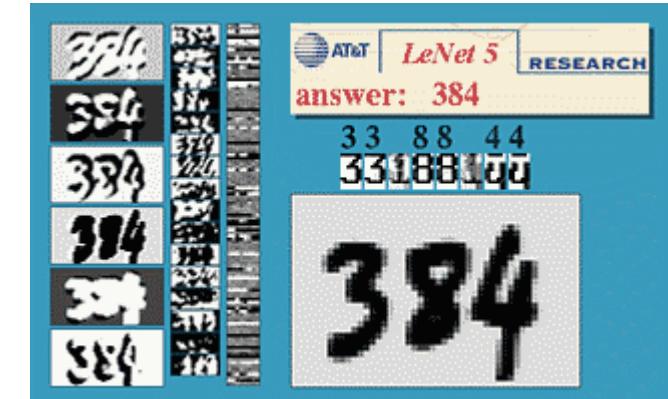


LeNet

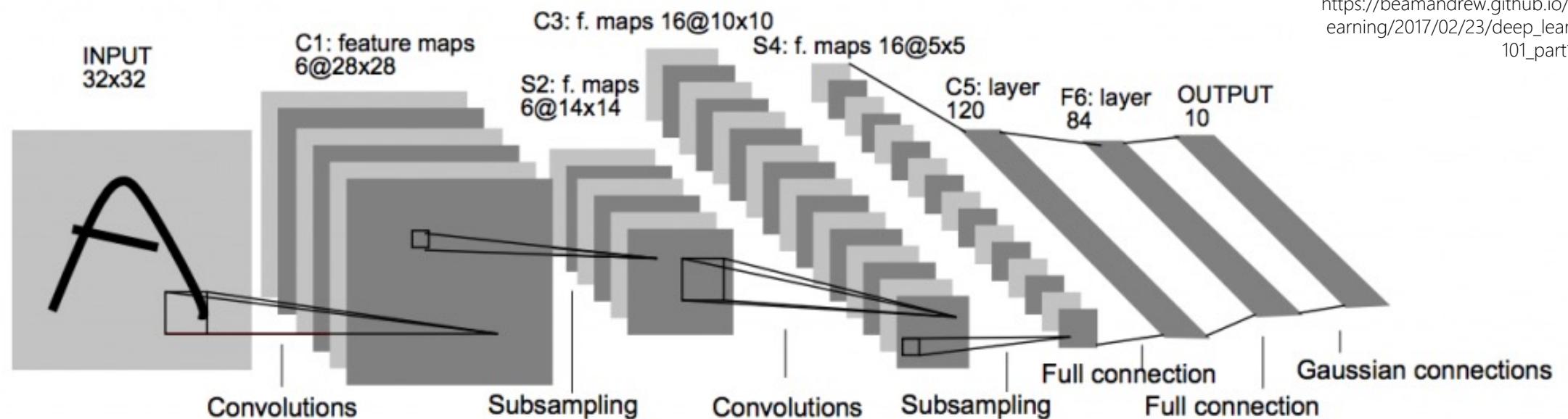
LeCun et al. (1989).
Handwritten digit
recognition with a
back-propagation
network. NuerIPS.



Convolutional Neural Networks



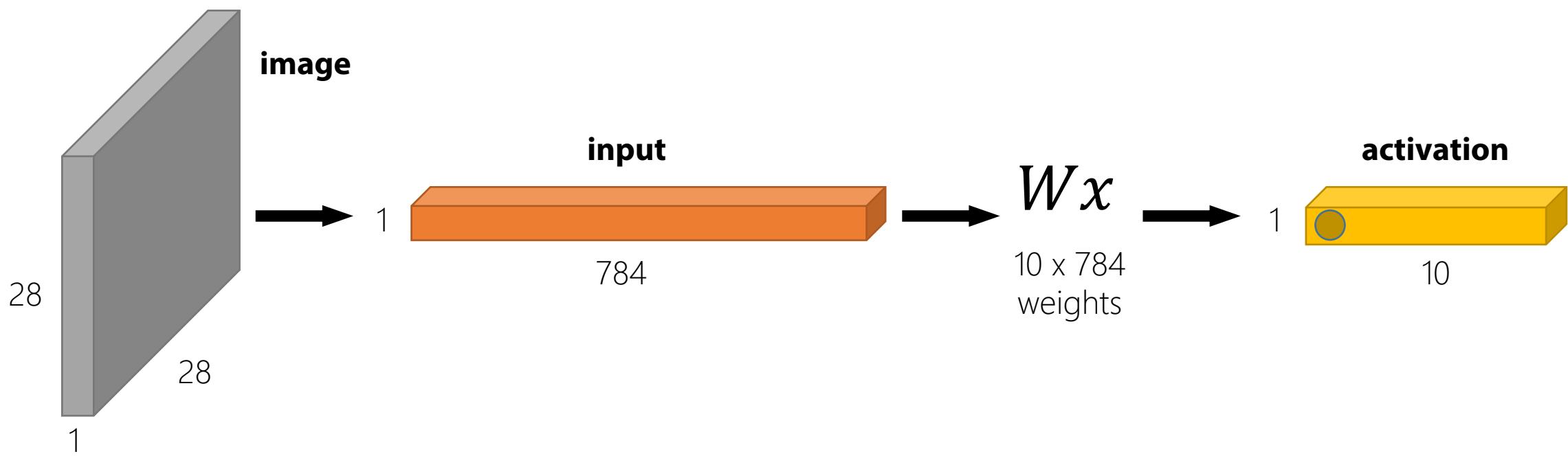
https://beamandrew.github.io/deep-learning/2017/02/23/deep_learning_101_part1.html



LeCun, Bottou, Bengio, & Haffner. (1998). Gradient-based learning applied to document recognition. Proc. IEEE.

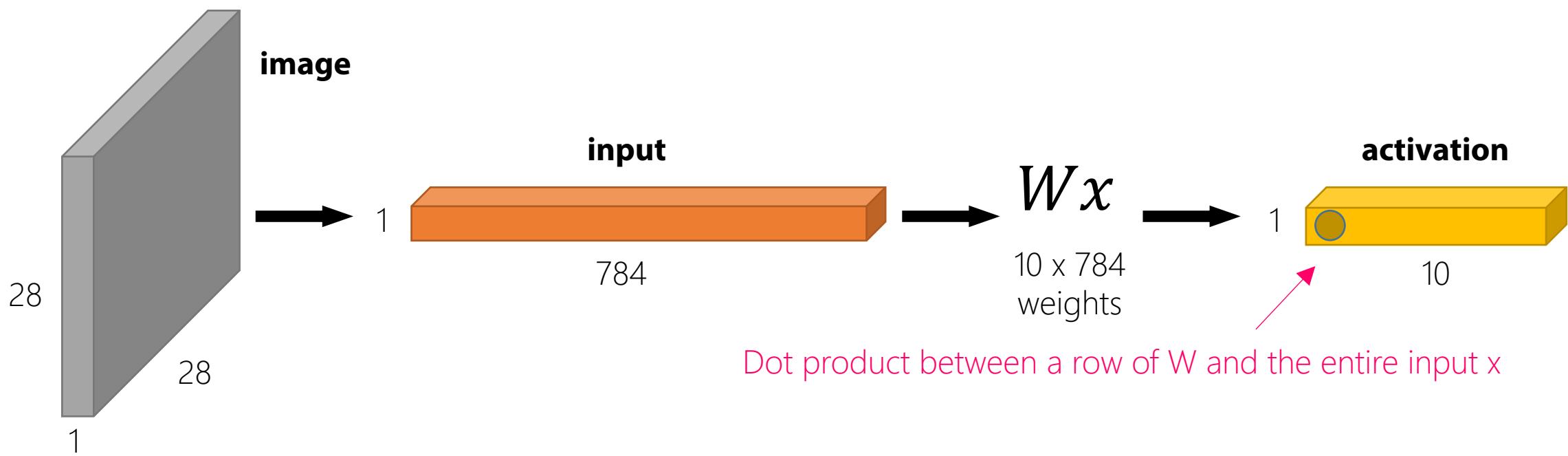
Fully Connected Layer

- 28×28 image → stretch to 784×1
- $64 \times 64 \times 3$ image → stretch to 12288×1
- ...



Fully Connected Layer

- 28×28 image → stretch to 784×1
- $64 \times 64 \times 3$ image → stretch to 12288×1
- ...



Draw your number here



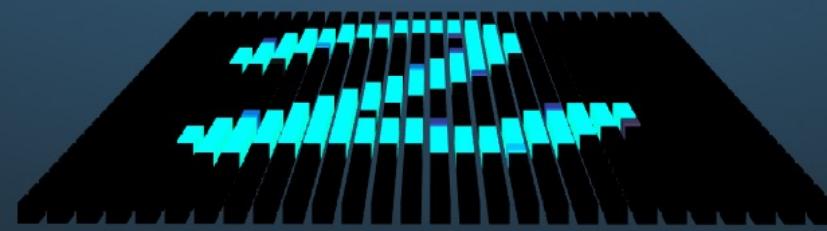
0 1 2 3 4 5 6 7 8 9
█ █ █ █ █ █ █ █ █ █



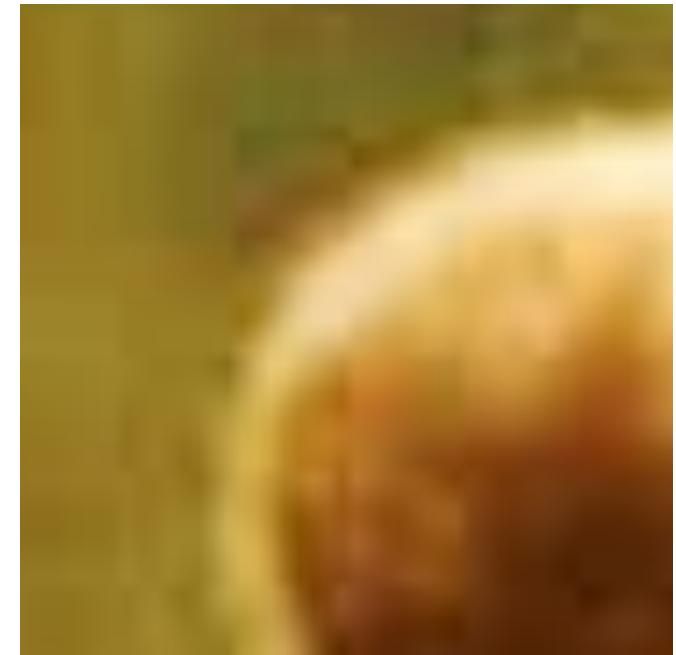
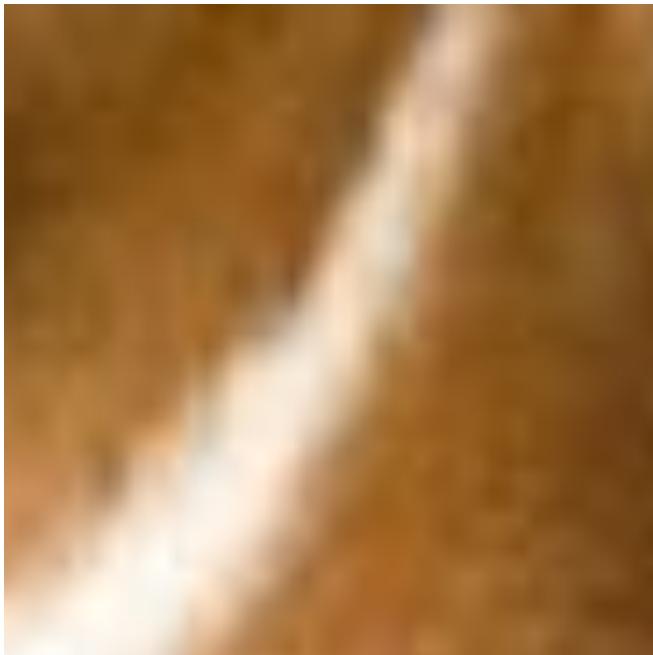
Downsampled drawing:

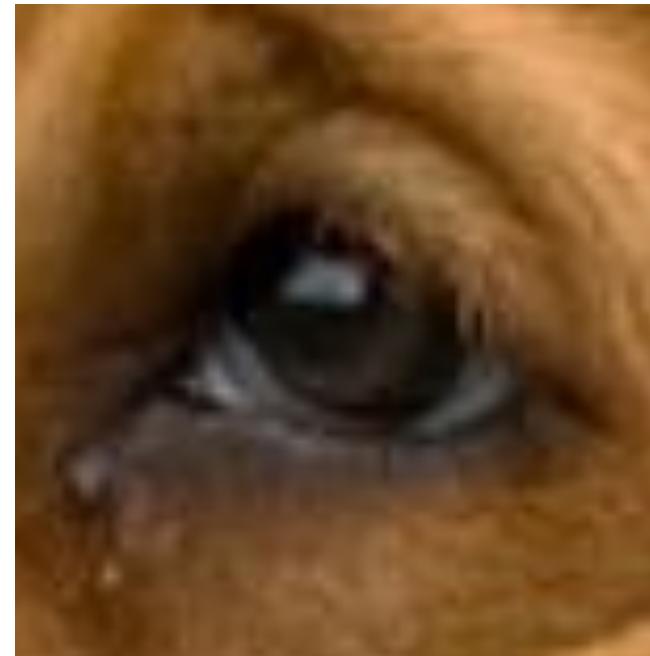
First guess:

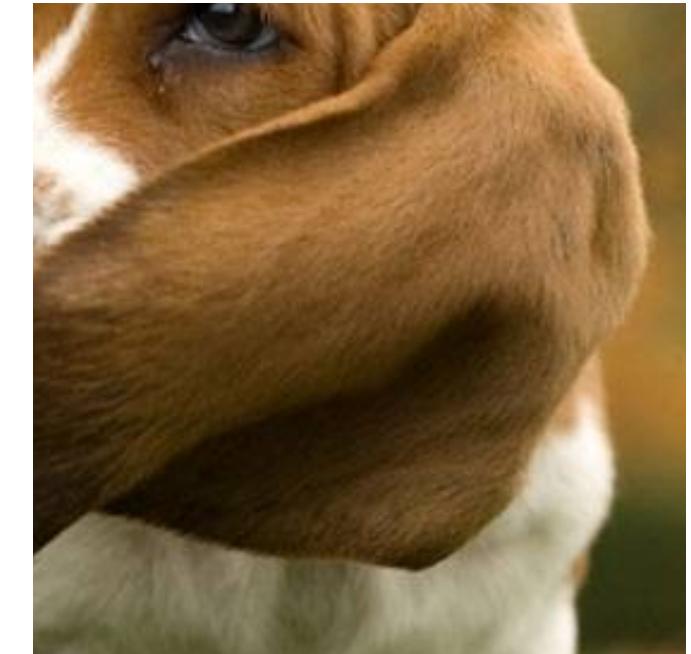
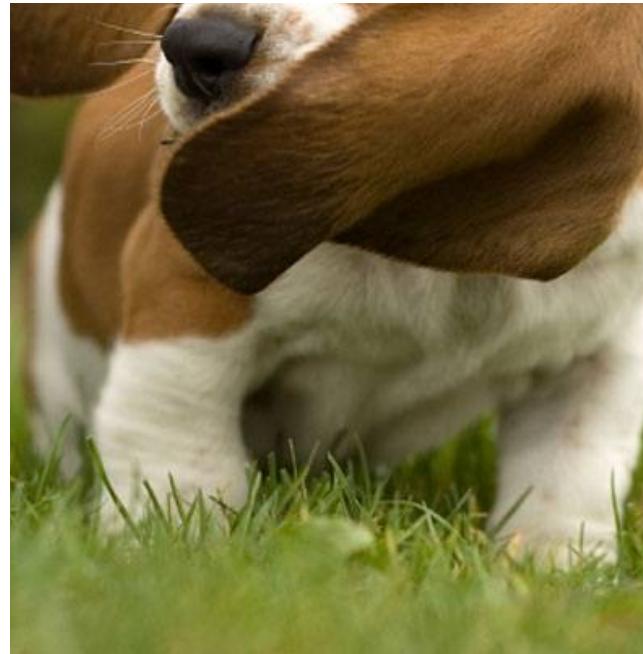
Second guess:





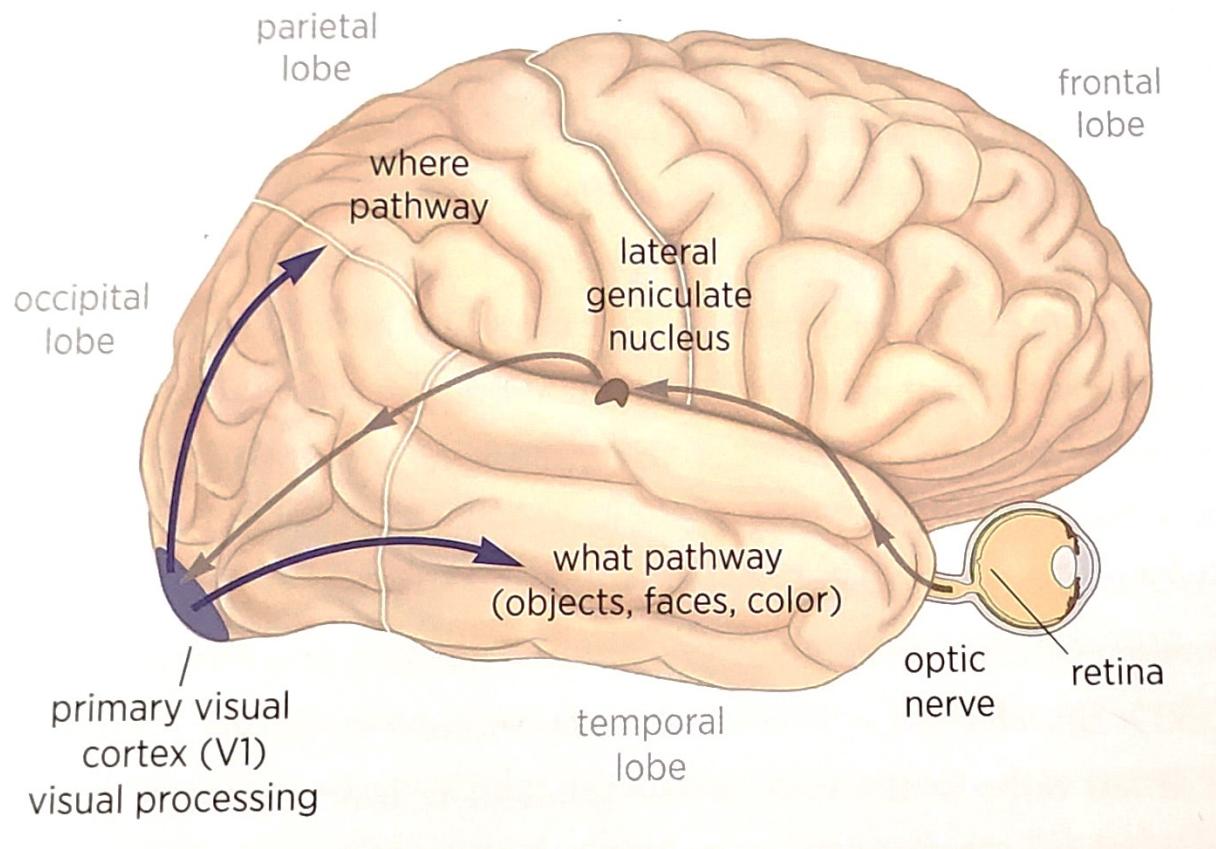








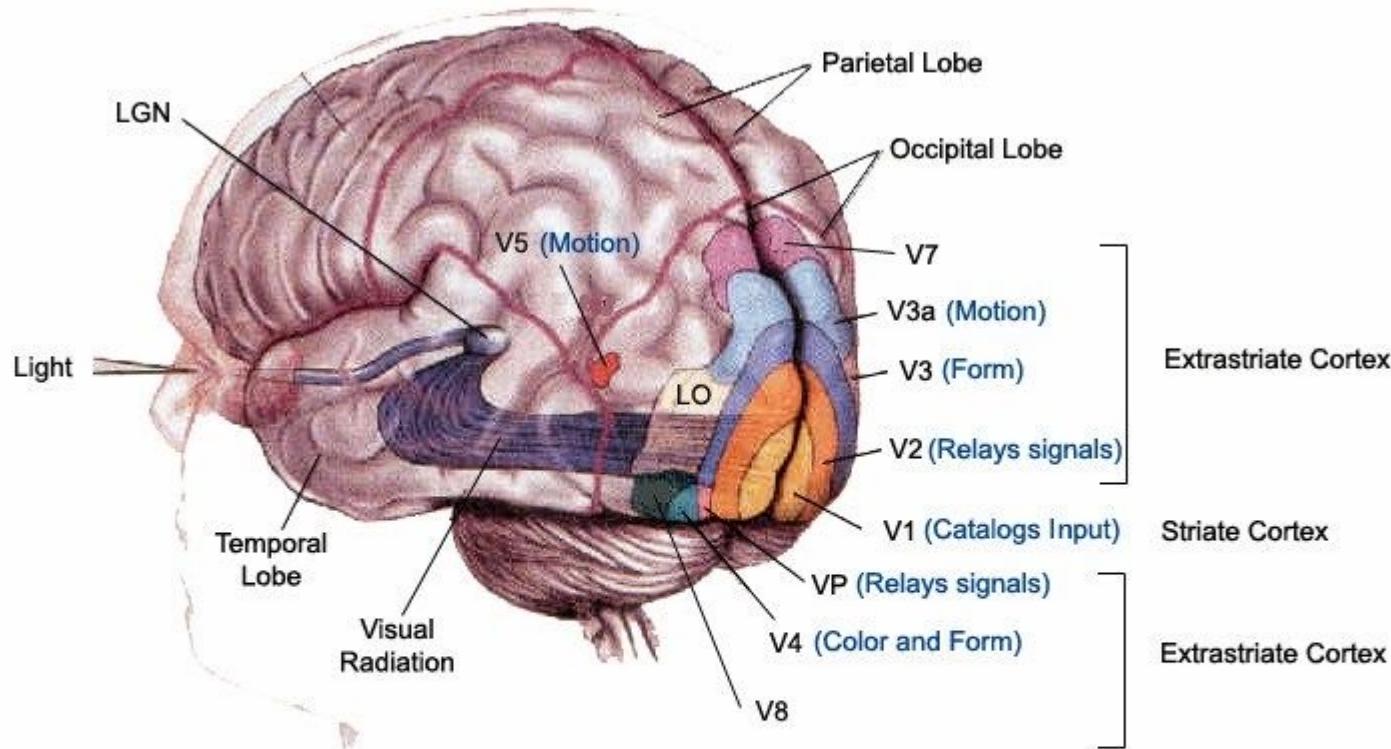
The human visual system



- Retina: visual input
- Retina → Lateral Geniculate Nucleus
 - Visual information flows through the optic nerve
- Lateral Geniculate Nucleus (LGN):
 - A small, ovoid object at the end of the optic tract
 - One on each side of the brain
 - In humans, each LGN has six layers of neurons
 - Sends information to the primary visual cortex (V1)

Image Courtesy: Kandel, "Reductionism in Art and Brain Science," 2016

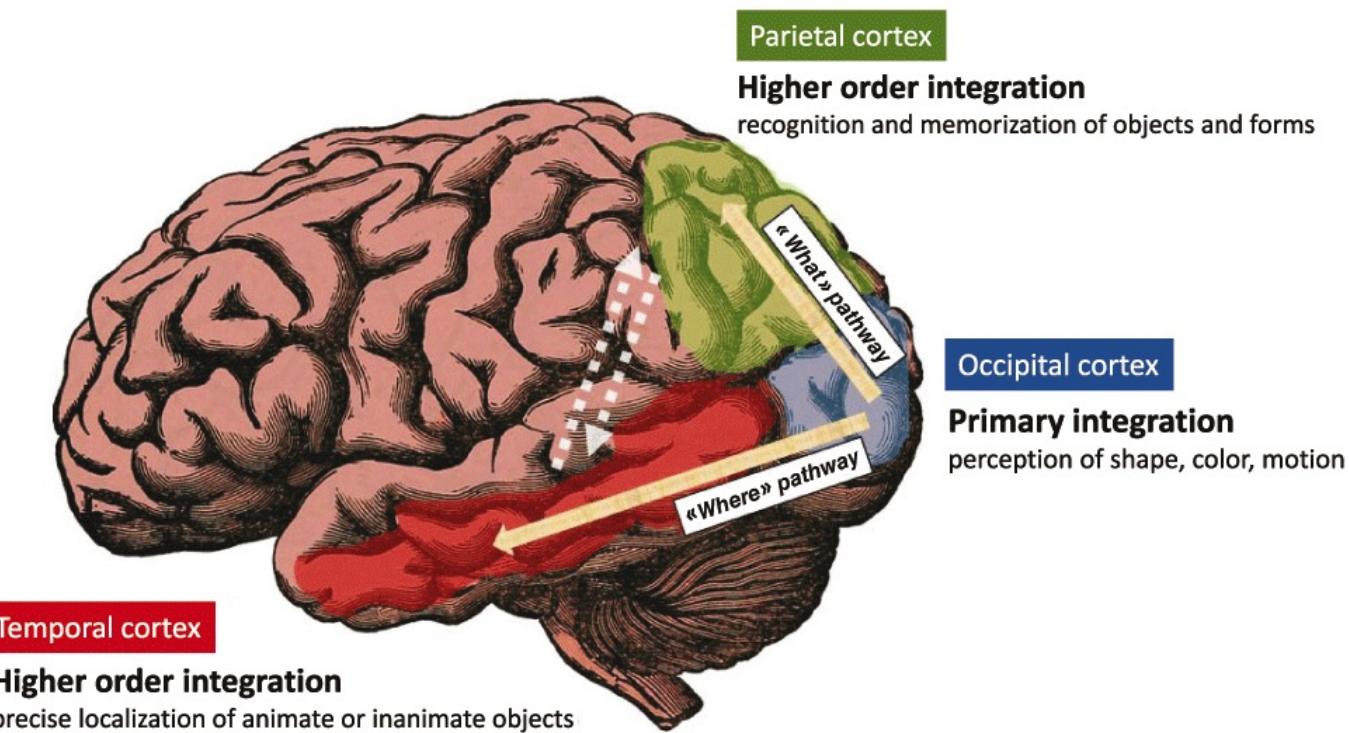
The human visual system



- Primary Visual Cortex (V1):
 - Highly specialized for processing visual information about static/moving objects
 - Pattern recognition happens
 - In average, the adult human V1 in each hemisphere has around 140 million neurons*
 - Transforms visual inputs to neural firing rates of those 140 million neurons → saliency determined by the visual location signaled by the highest firing neuron ("saliency map")

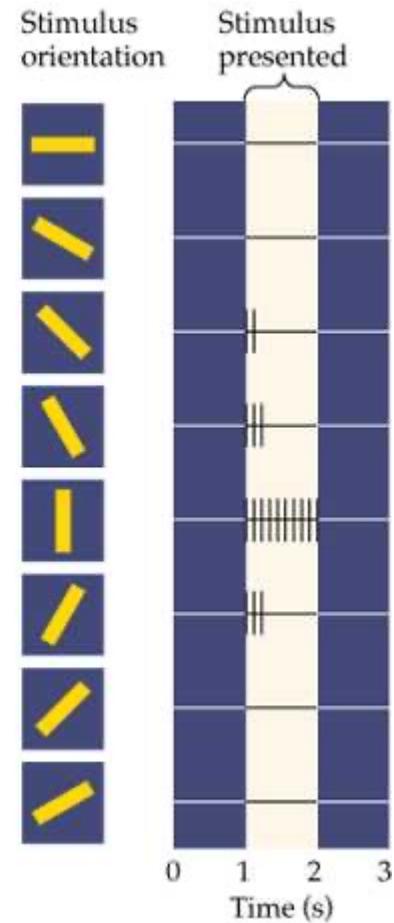
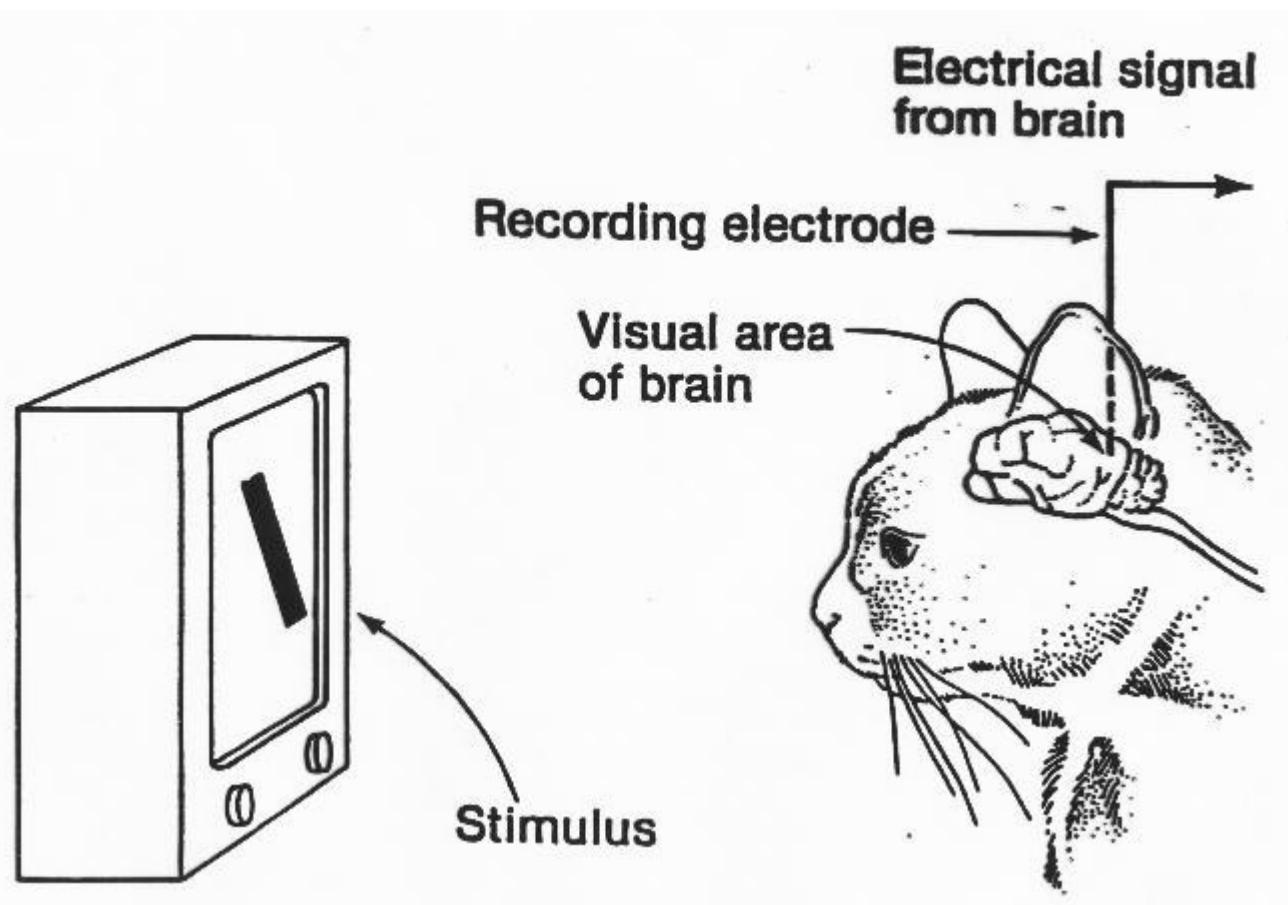
* Leuba G; Kraftsik R (1994). "Changes in volume, surface estimate, three-dimensional shape and total number of neurons of the human primary visual cortex from midgestation until old age". *Anatomy and Embryology*. 190 (4): 351–366.

The human visual system



- Visual pathways
 - V1 gives rise to two major pathways: the *where* and *what* pathways
- The “where” pathway:
 - Concerned with where an object or a person is located
- The “what” pathway:
 - Concerned with what an object is or who a person is.

Hubel & Wiesel (1959 ~)



Neurons in the visual cortex respond selectively to oriented edges. Neurons in visual cortex typically respond vigorously to a bar of light oriented at a particular angle and weakly or not at all to other orientations.

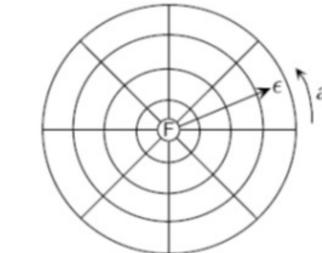
Hubel & Wiesel (1959 ~)

Warning: DISTURBING CONTENT

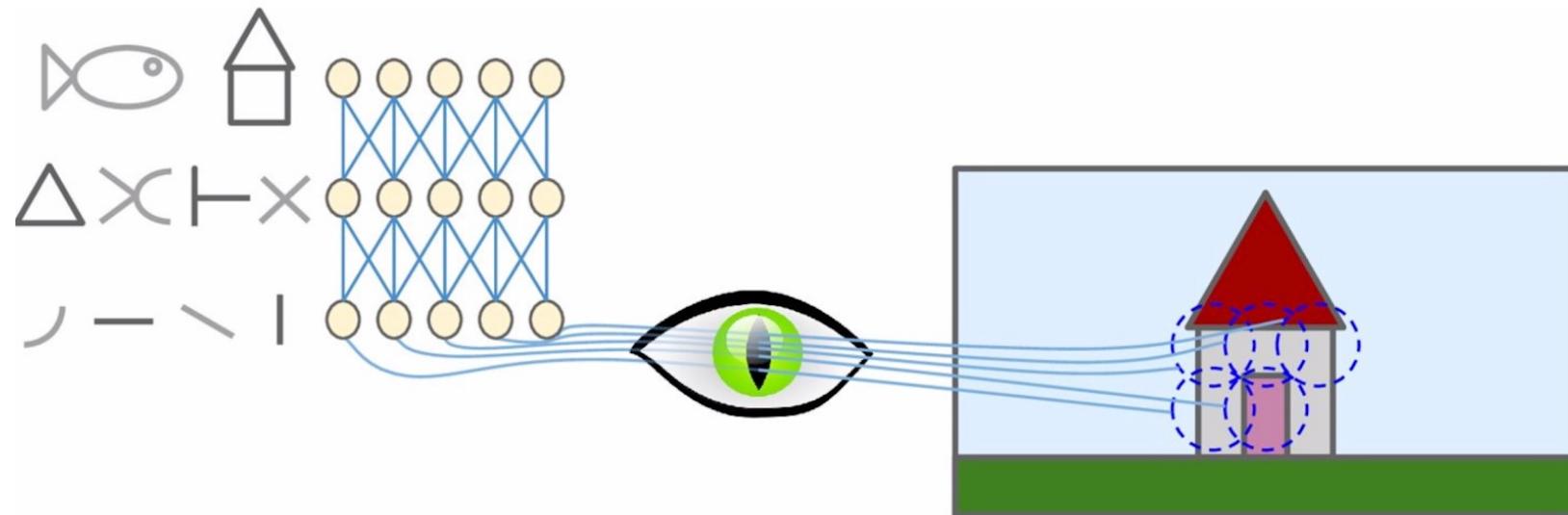
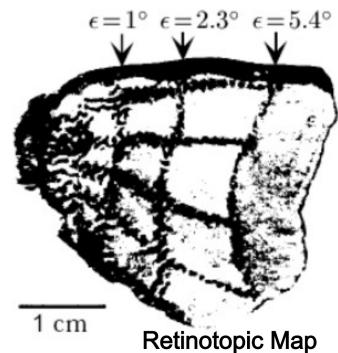


Topographic Maps in Cortex

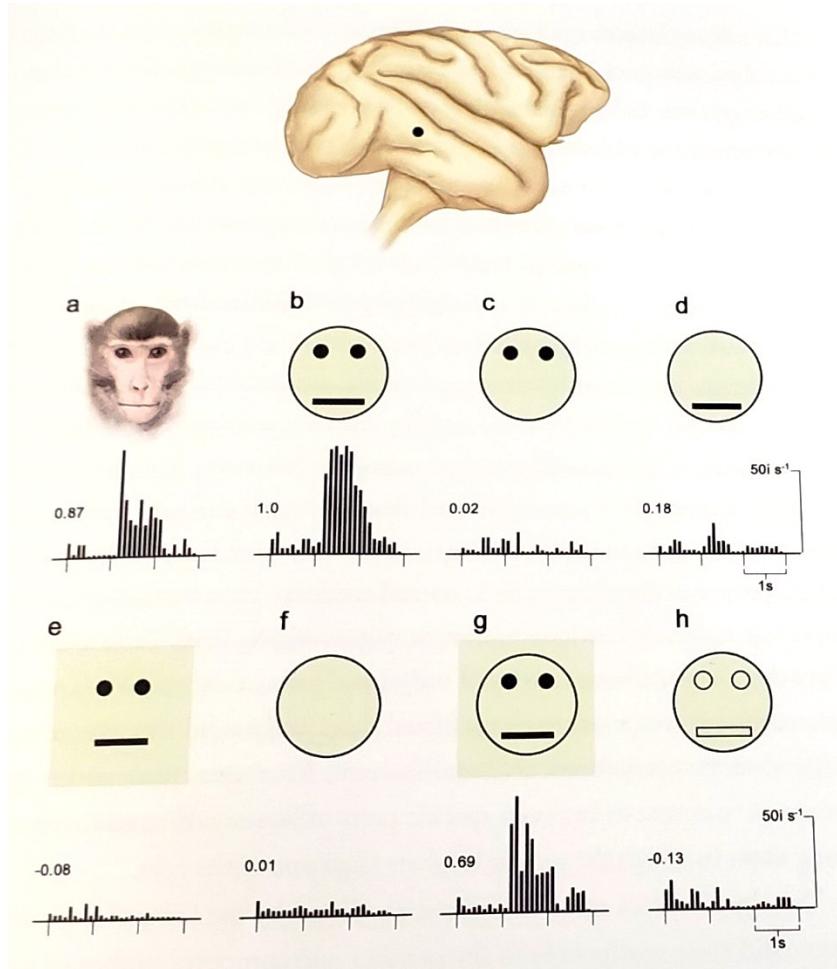
- Each visual sensitive cell only responses to stimuli of a limited region (receptive field)
- Neighboring cells have partially overlapping receptive fields
- Neighboring points in a visual image evoke activity in neighboring regions of visual cortex
- In this manner, the visual system easily maintain the information of the spatial location of stimulus



(Dayan and Abbott 2001)



e.g. Facial Recognition

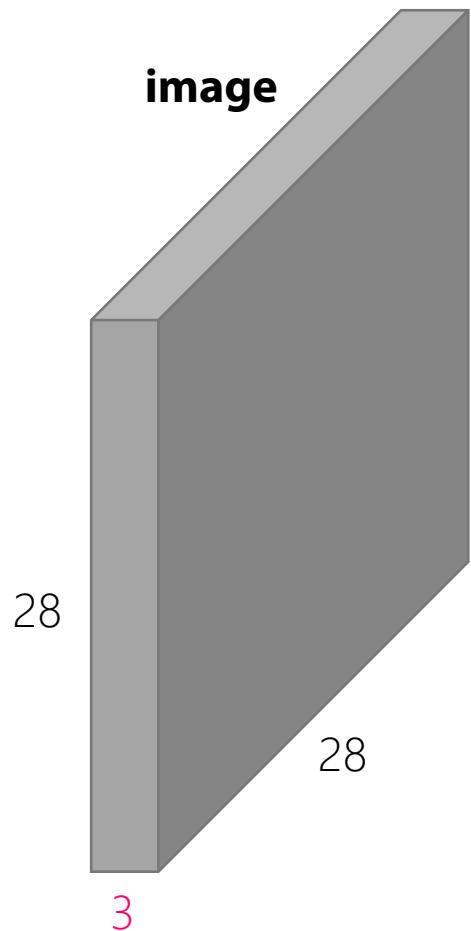


- Holistic face detection
 - "Face cell" in the inferior temporal cortex
 - Fires when there is a face-like object

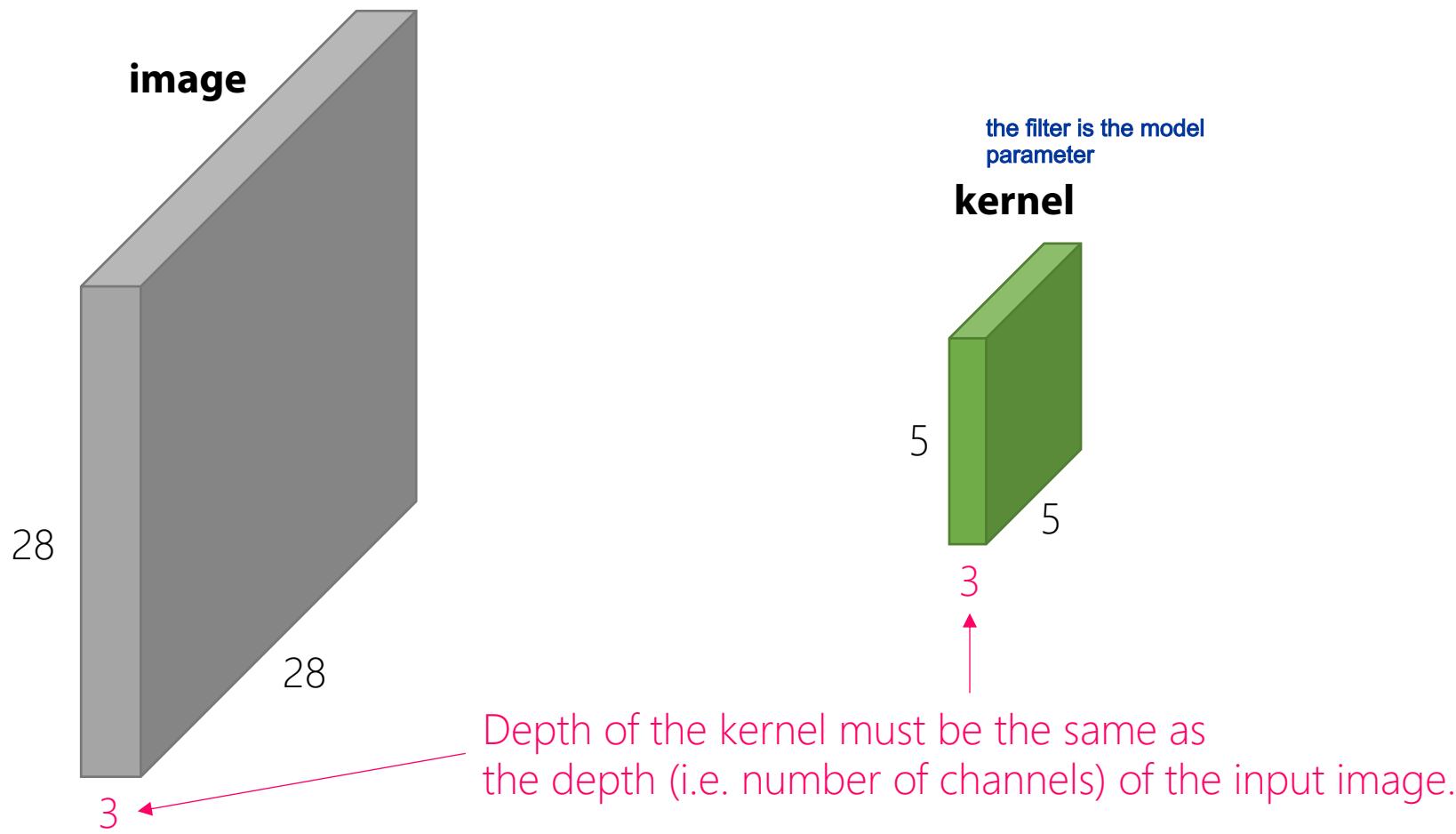
Convolutional Neural Networks

- Key idea:
 - As like how we (humans) understand a visual scene, if neural nets could see small pieces, understand patterns and textures, combine the pieces to see a bigger picture, computers should be able to recognize images.
- A bonus:
 - Typical neural networks are “fully connected”.
 - In an image domain, this means all the pixels are interconnected.
 - However, pixels far apart have no significant meaning...
 - By connecting only the nearing neighbors, computational load could be much lower.

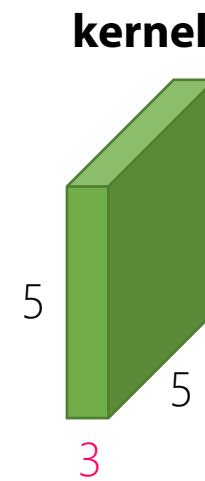
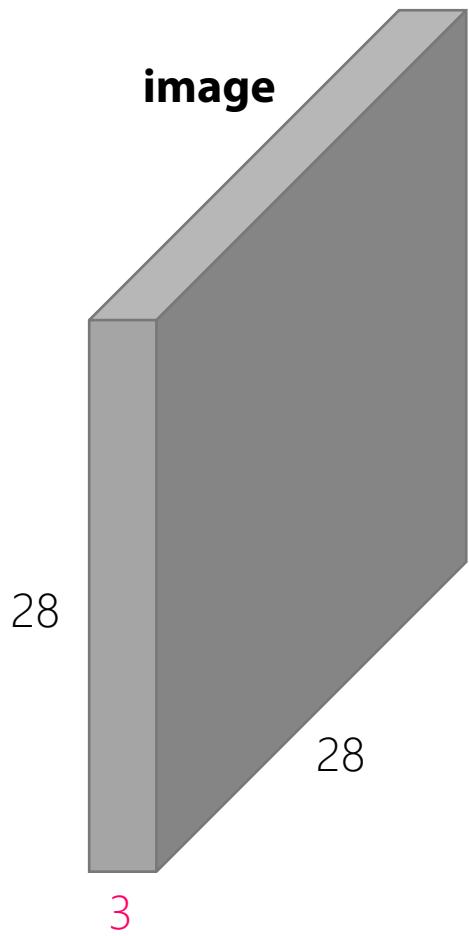
Convolution Layer



Convolution Layer

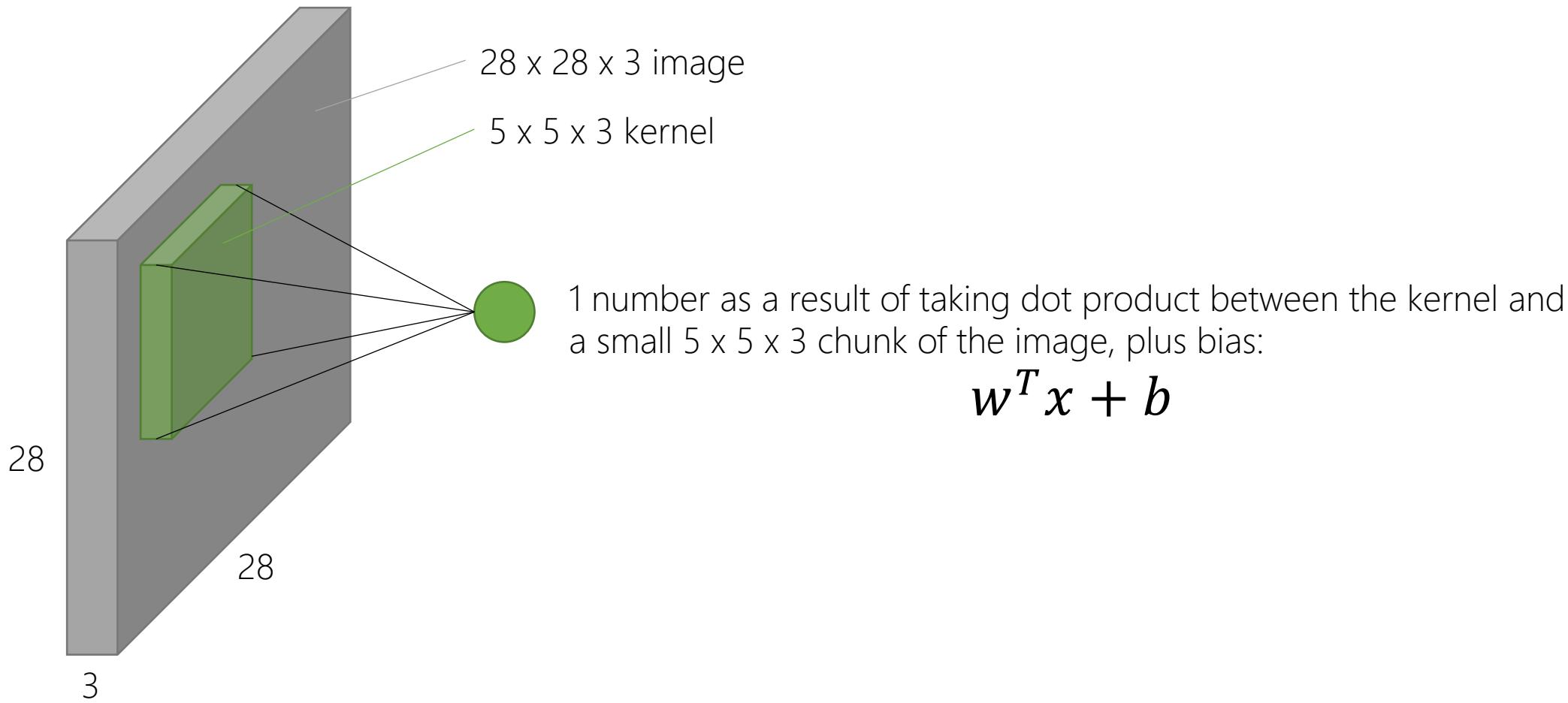


Convolution Layer

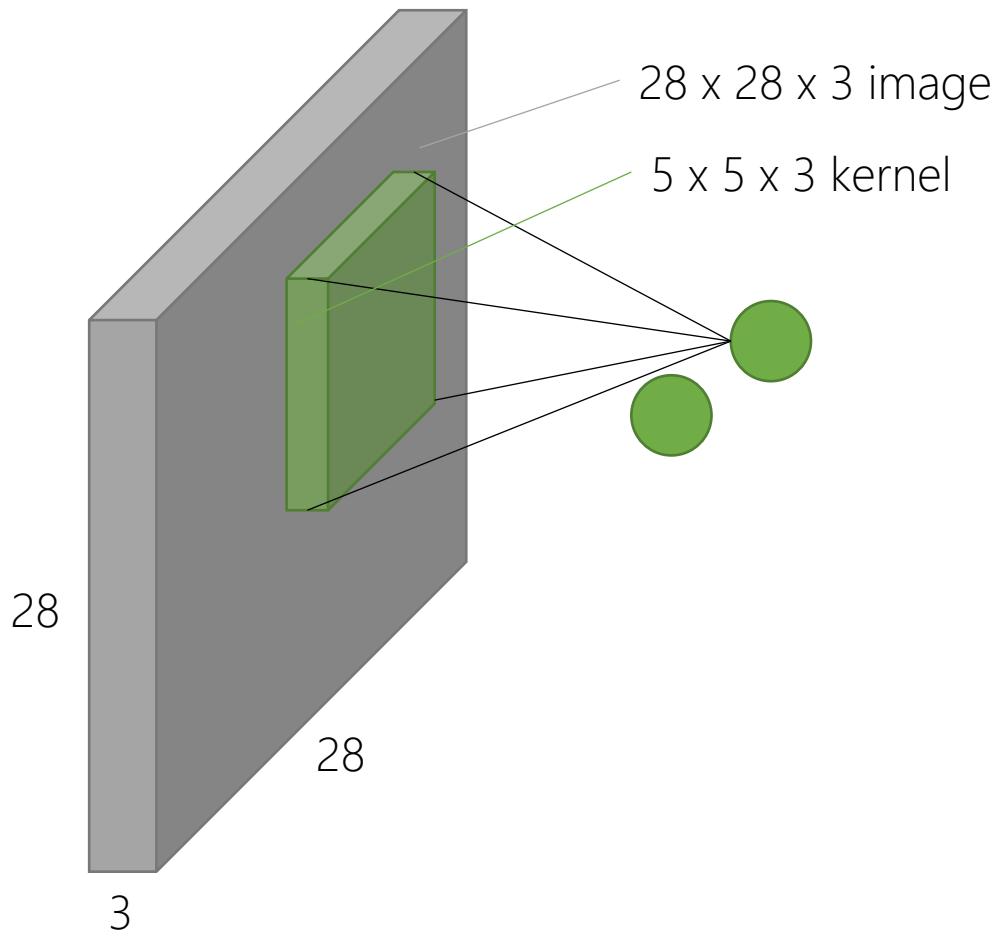


Convolve the kernel with the image!
In other words...
“slide the kernel over the image,
compute dot products each time.”

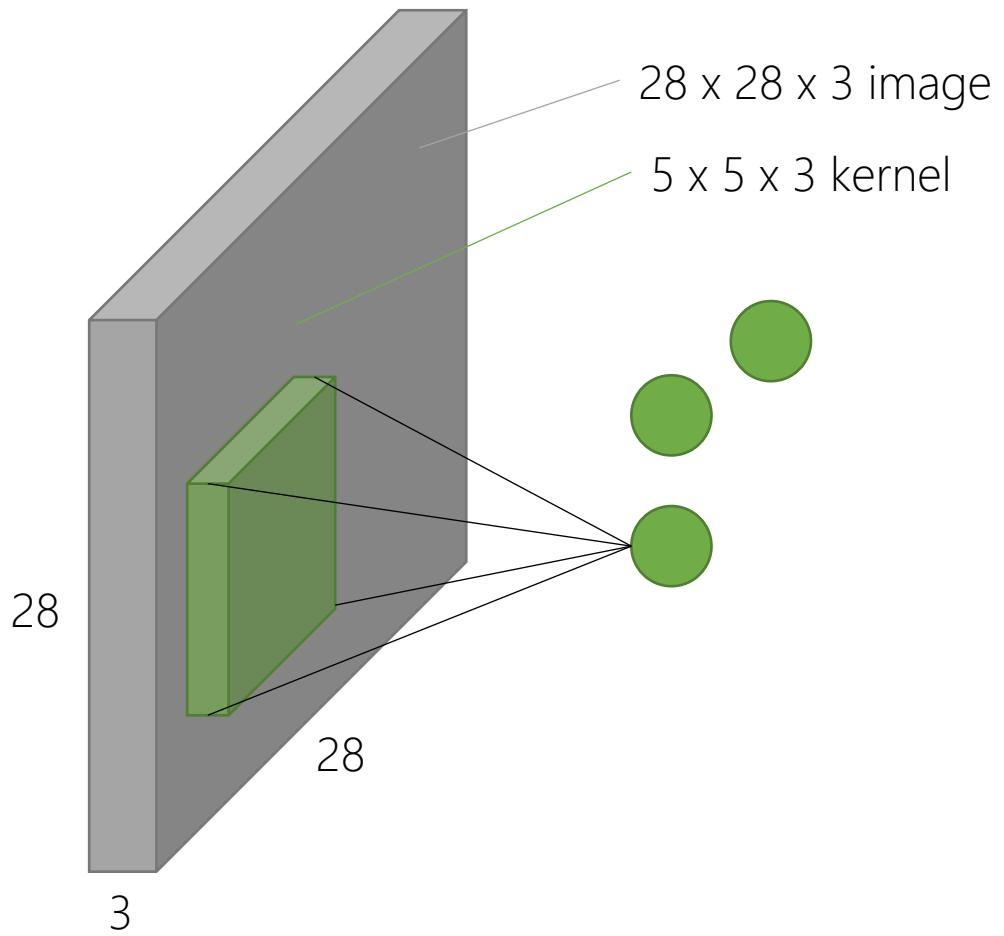
Convolution Layer



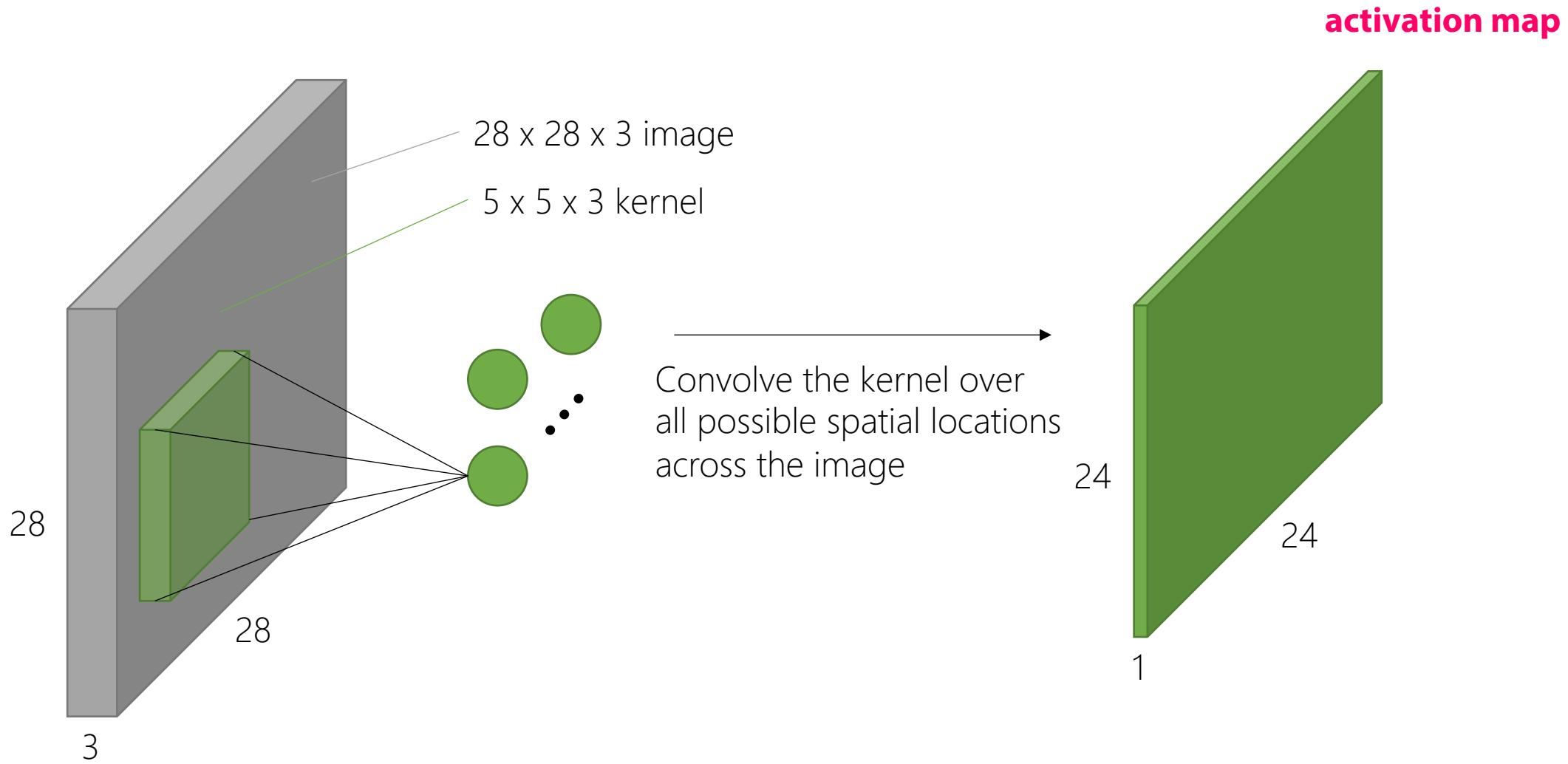
Convolution Layer



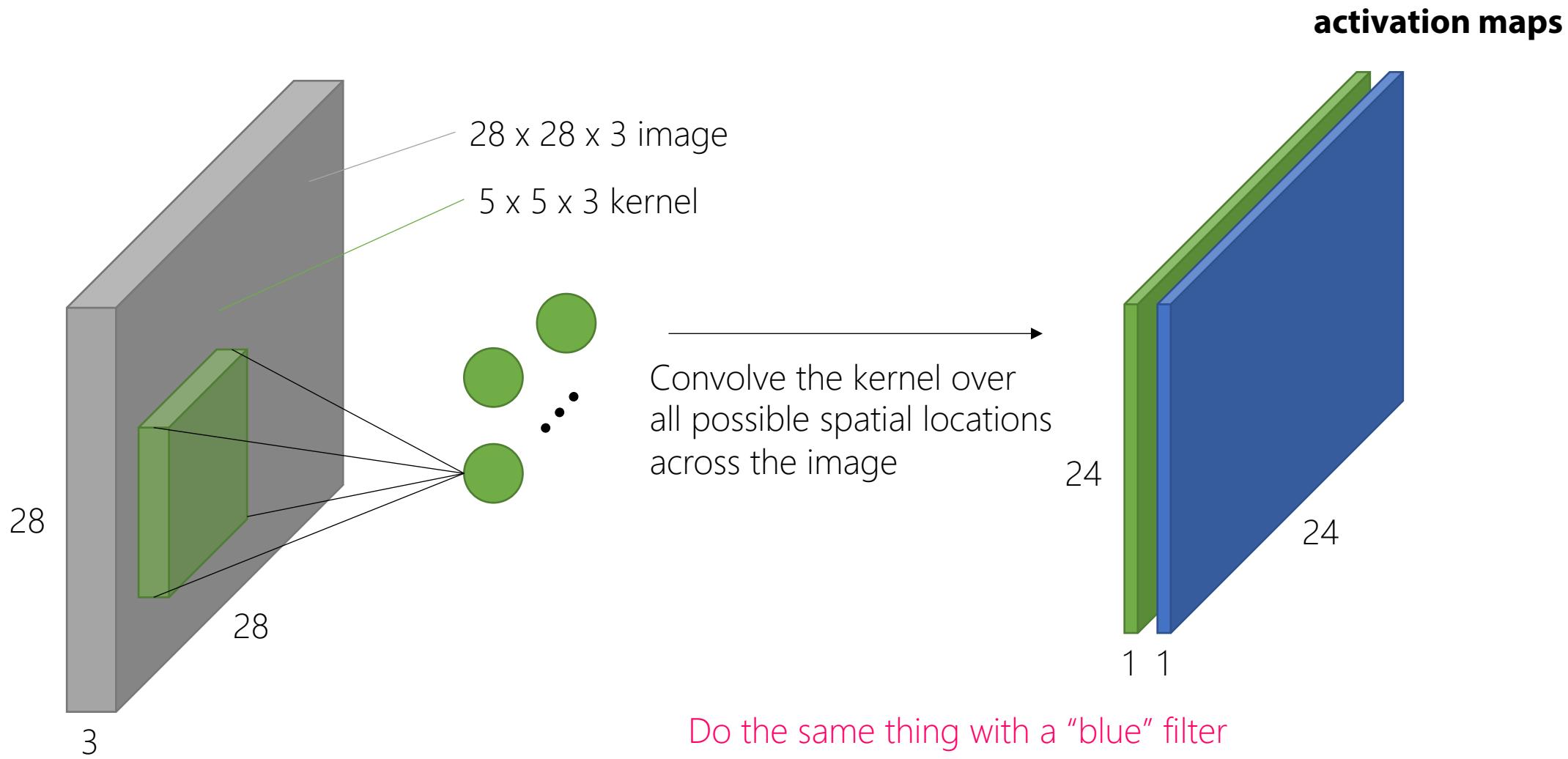
Convolution Layer



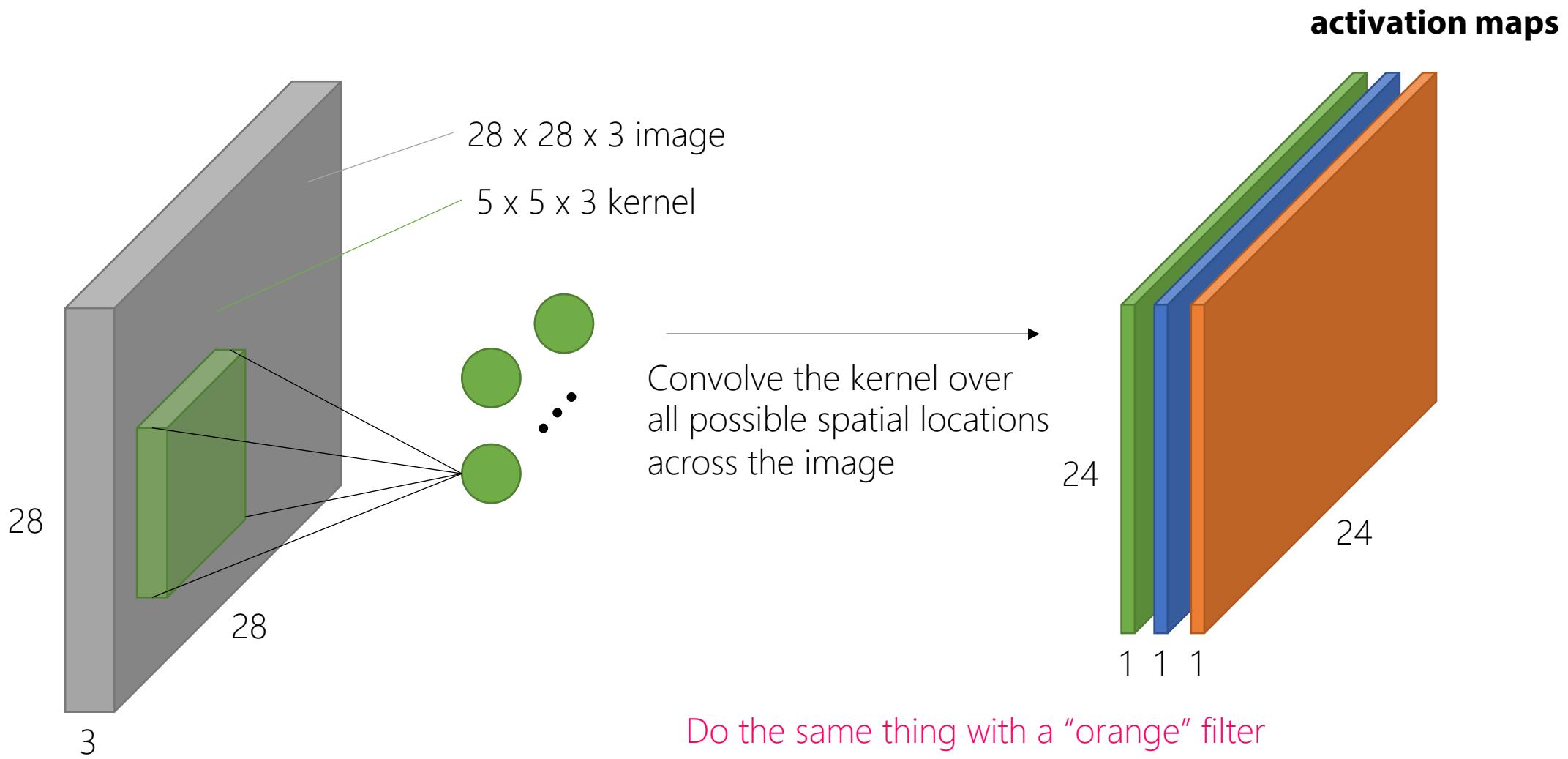
Convolution Layer



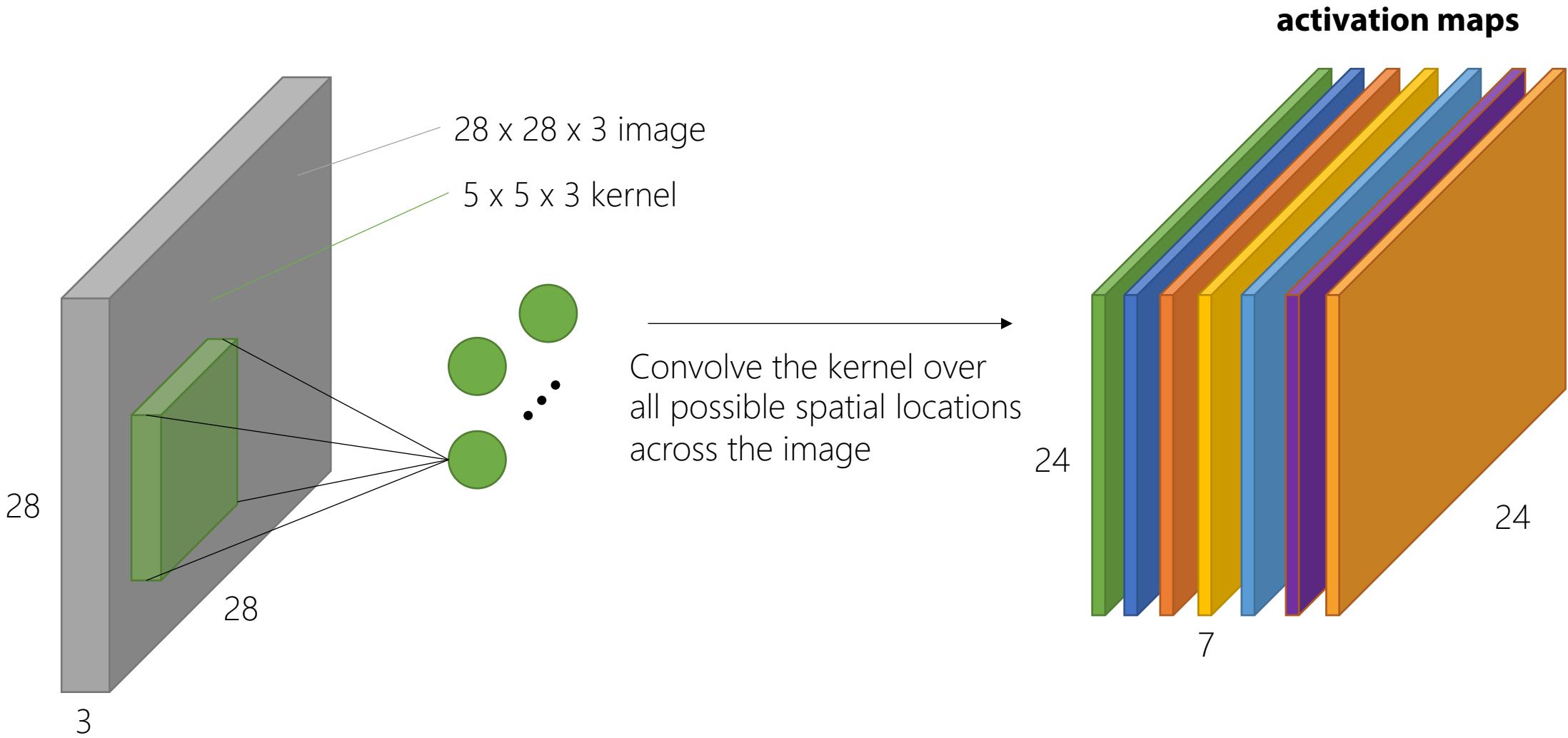
Convolution Layer



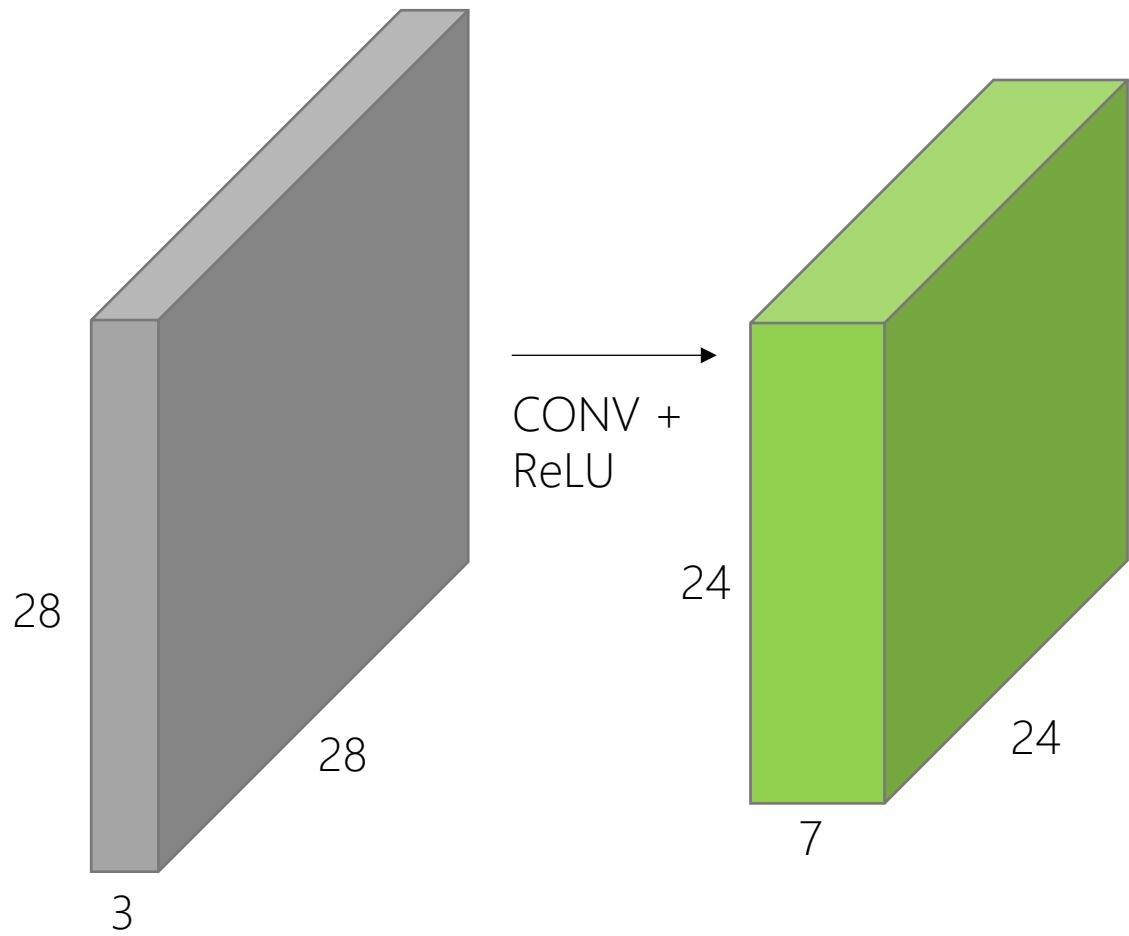
Convolution Layer



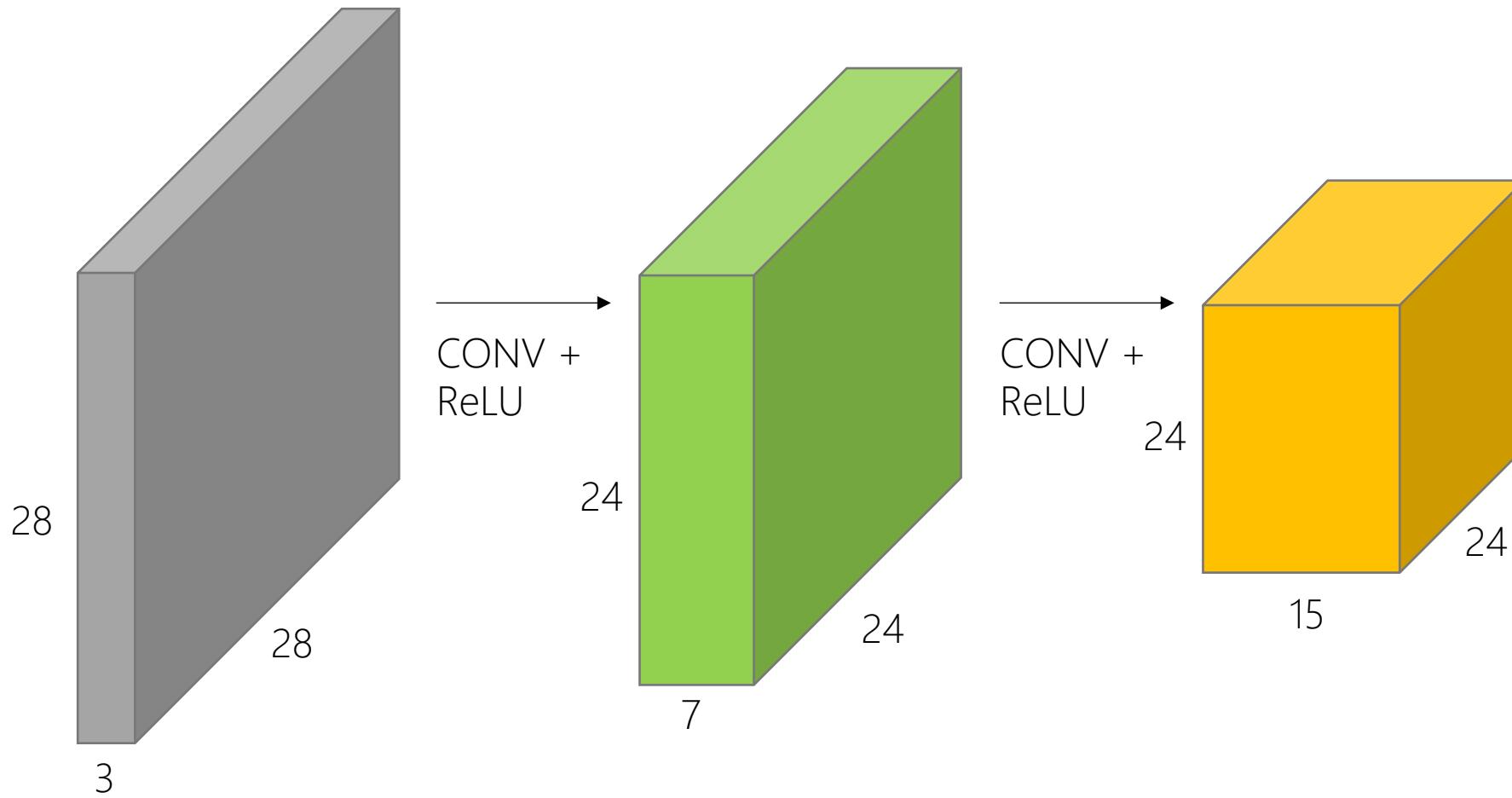
Convolution Layer



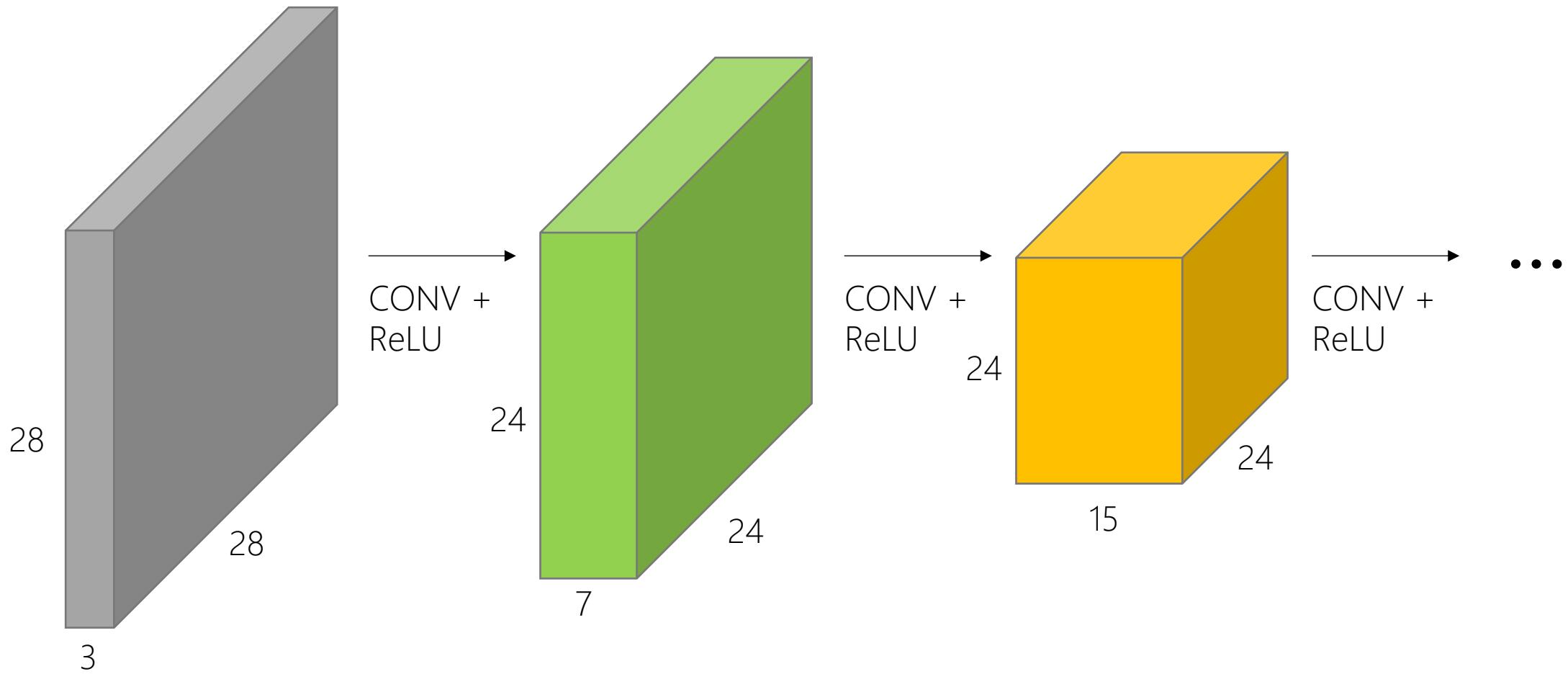
ConvNet is a sequence of convolution layers



ConvNet is a sequence of convolution layers



ConvNet is a sequence of convolution layers



Draw your number here



Downsampled drawing: 2

First guess: 2

Second guess: 1

Layer visibility

Input layer

Show

Convolution layer 1

Show

Downsampling layer 1

Show

Convolution layer 2

Show

Downsampling layer 2

Show

Fully-connected layer 1

Show

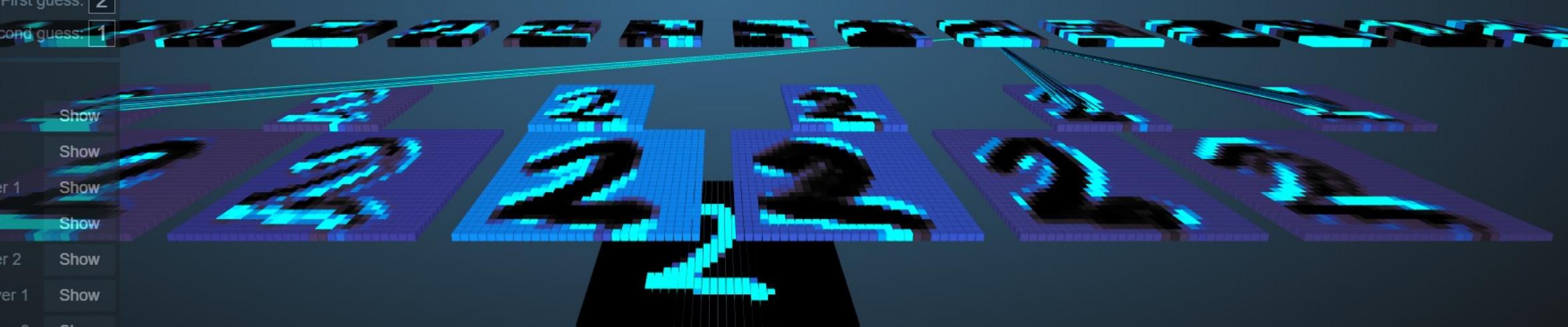
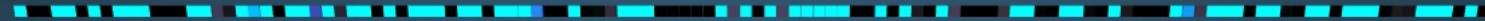
Fully-connected layer 2

Show

Output layer

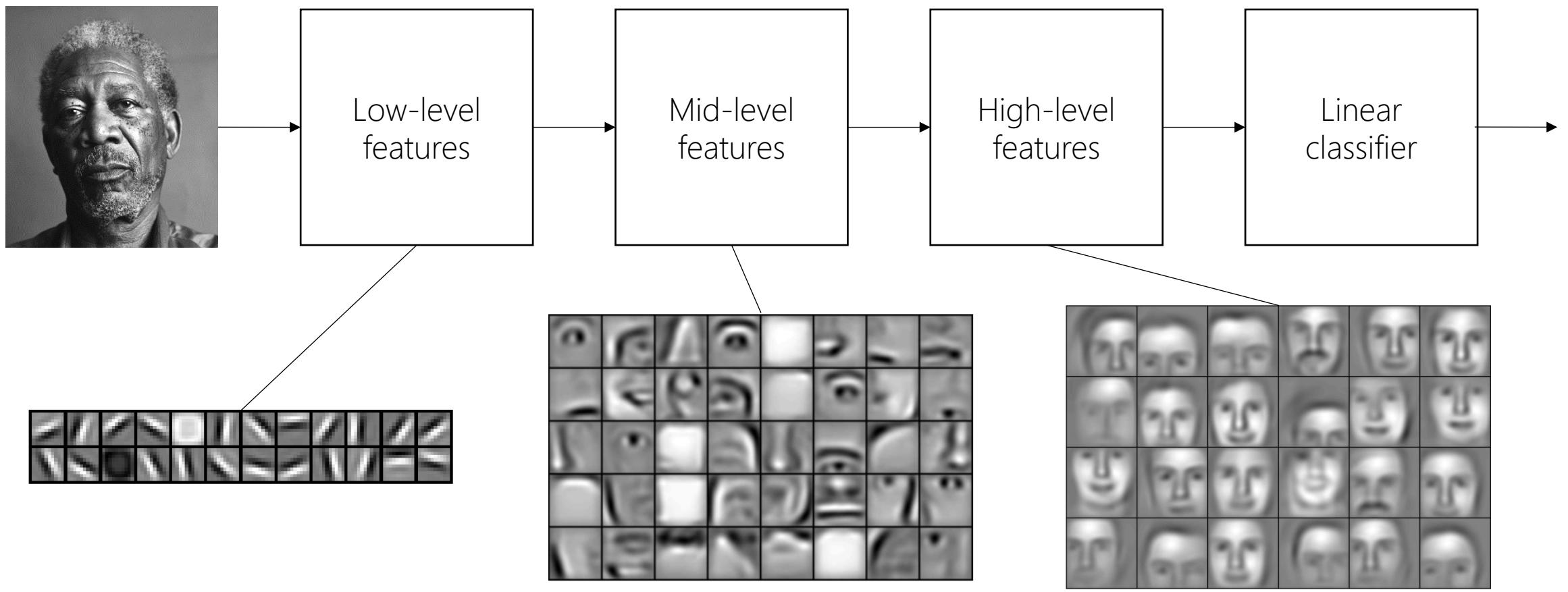
Show

0 1 2 3 4 5 6 7 8 9



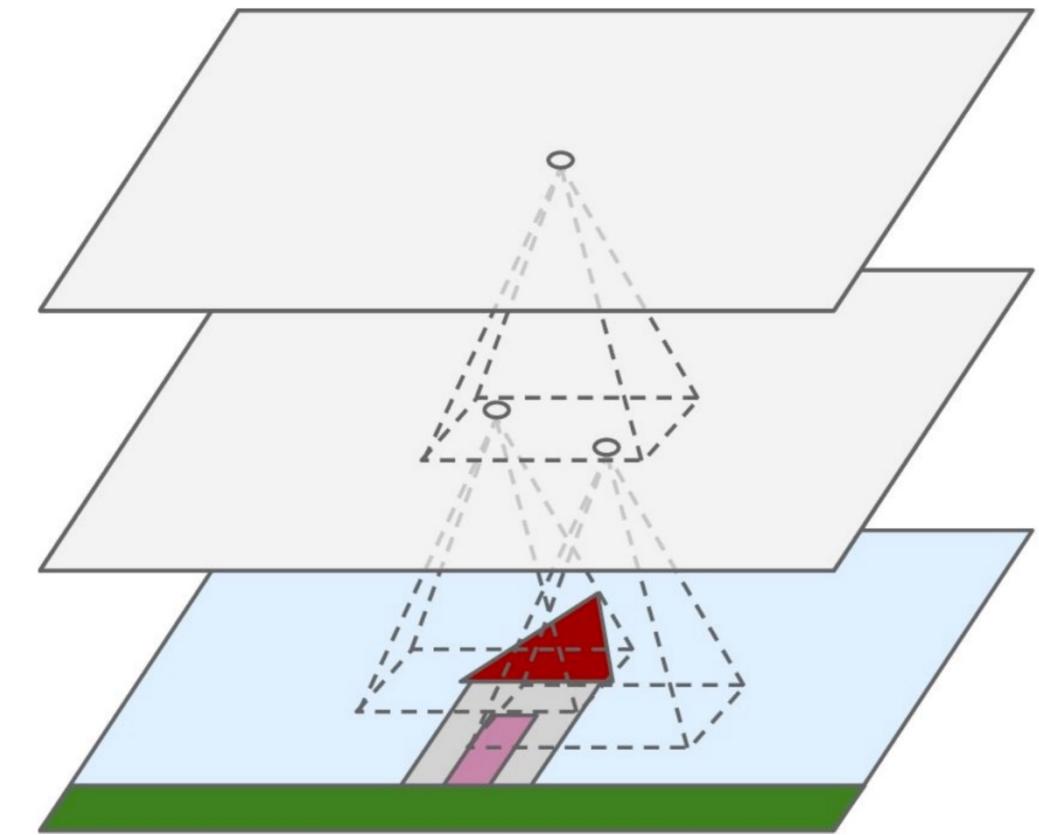
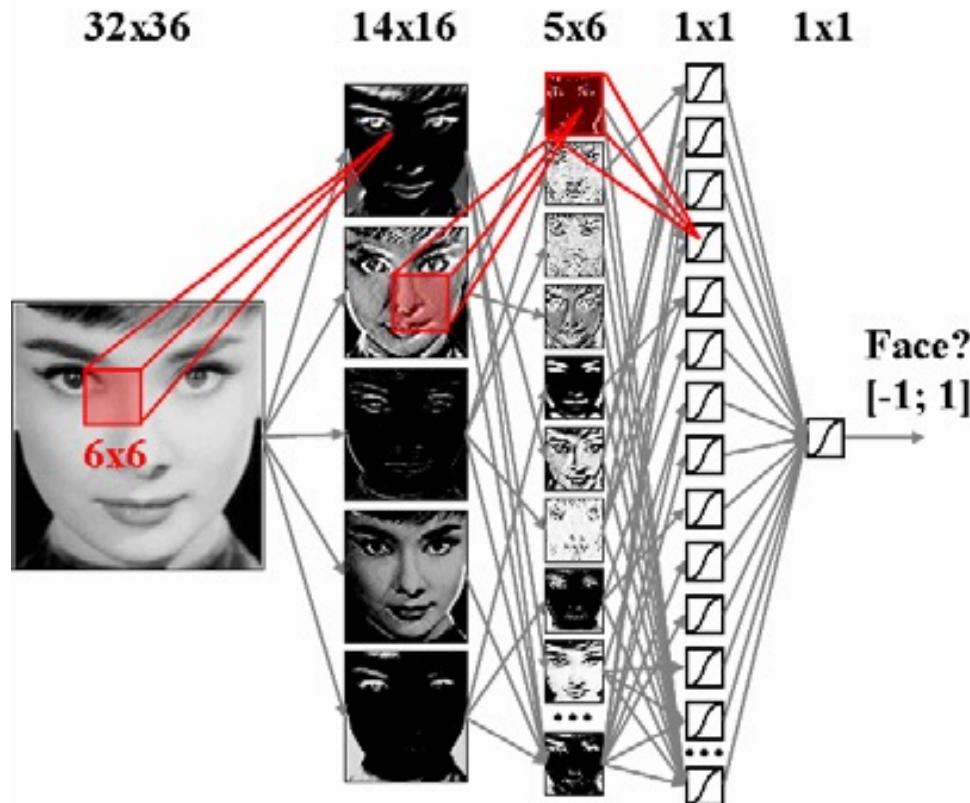
Made by Adam Harley. [Project details](#)

ConvNet is a sequence of convolution layers

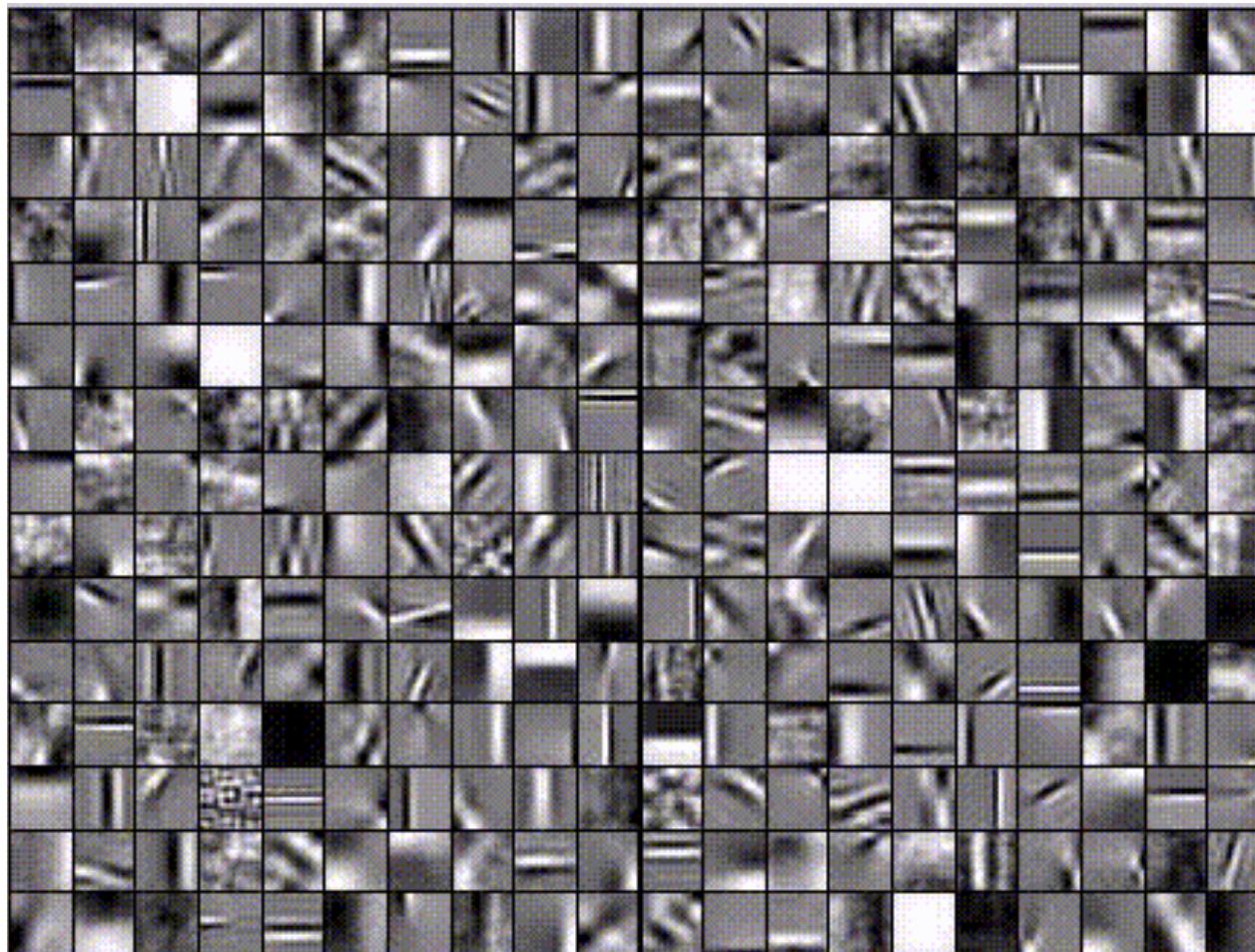


Lee et al. (2009)

ConvNet is a sequence of convolution layers



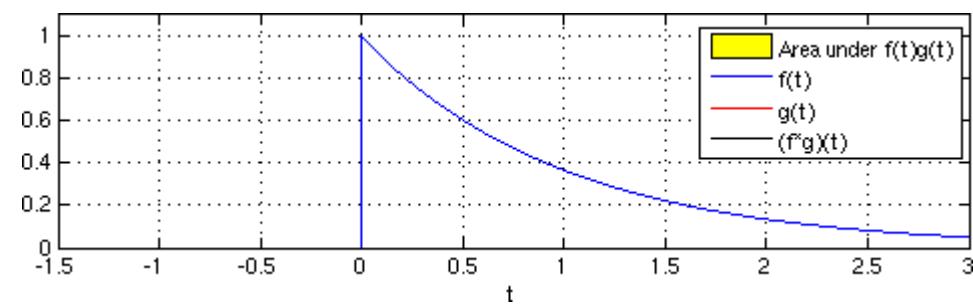
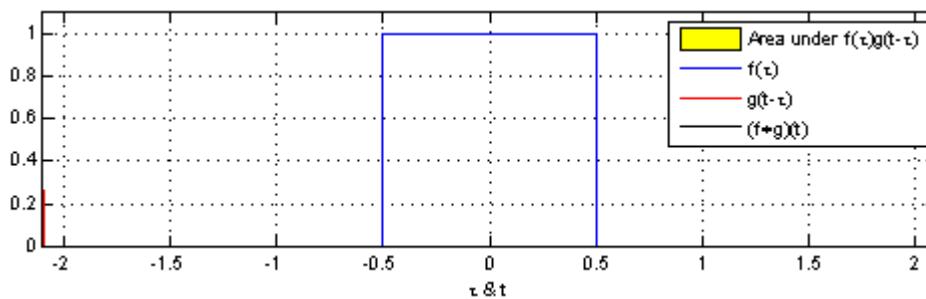
Conv kernels are trainable



A closer look...

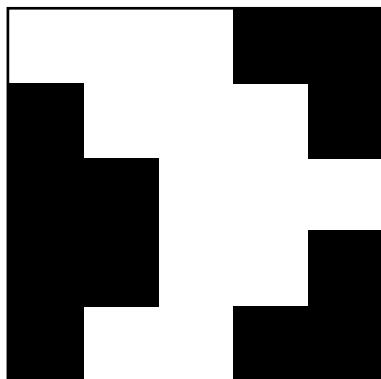
- Convolution

$$(f * g)(t) := \int_{-\infty}^{\infty} f(\tau)g(t - \tau) d\tau$$



A closer look...

- 2D Discrete Convolution



*

$$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$



1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

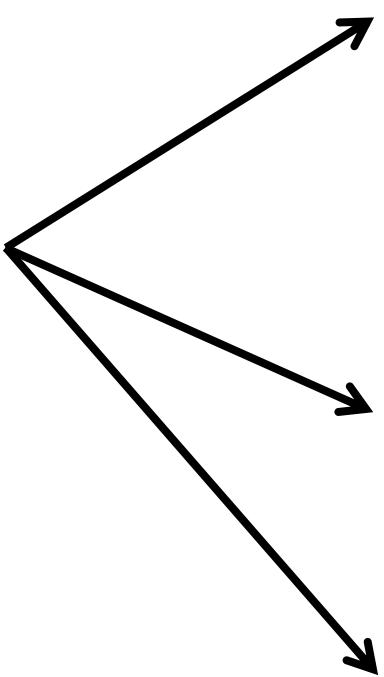
Image

4			

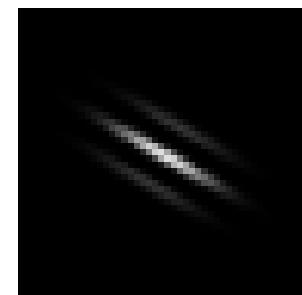
Convolved
Feature

A closer look...

- 2D Discrete Convolution



$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} =$$



*

=



⋮

Case Study of Convolution

X or X?

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1



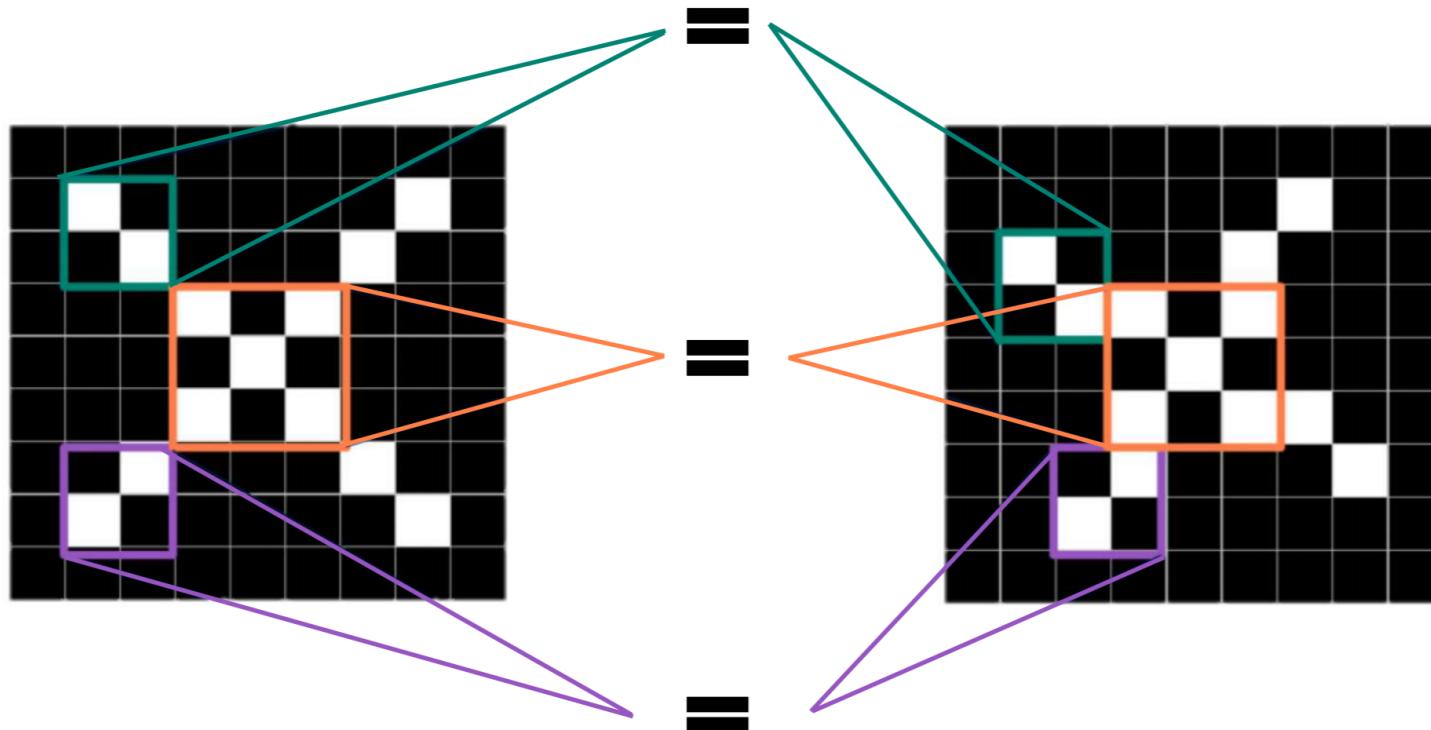
-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	1	-1
-1	1	-1	-1	-1	-1	1	-1	-1
-1	-1	1	1	-1	-1	1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	1	-1	-1
-1	-1	-1	1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

Image is represented as matrix of pixel values... and computers are literal!
We want to be able to classify an X as an X even if it's shifted, shrunk, rotated, deformed.

Case Study of Convolution

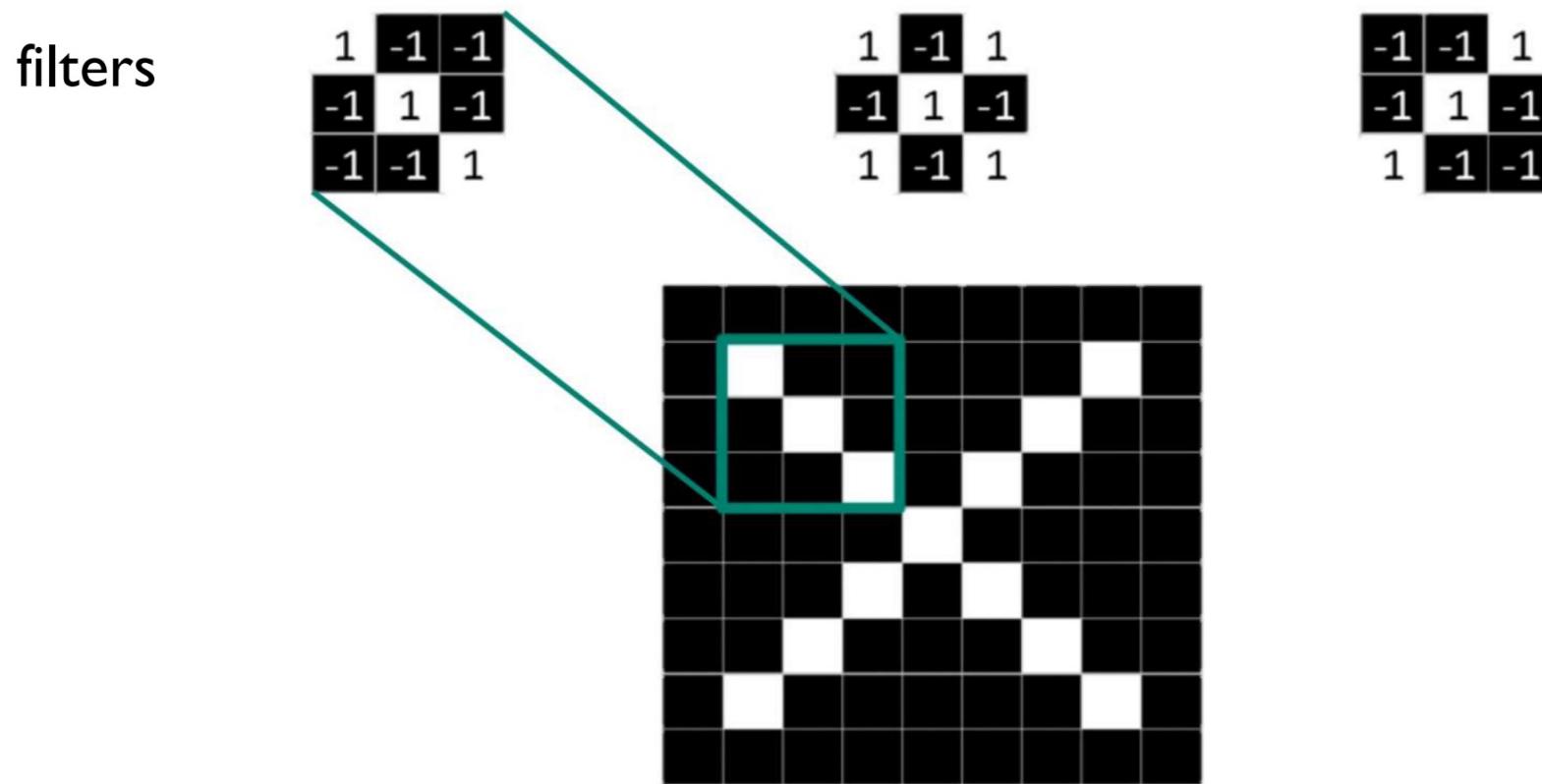
Features of X

small filters that can match patterns within the image (compared to matching the entire image) helps with reducing error in prediction

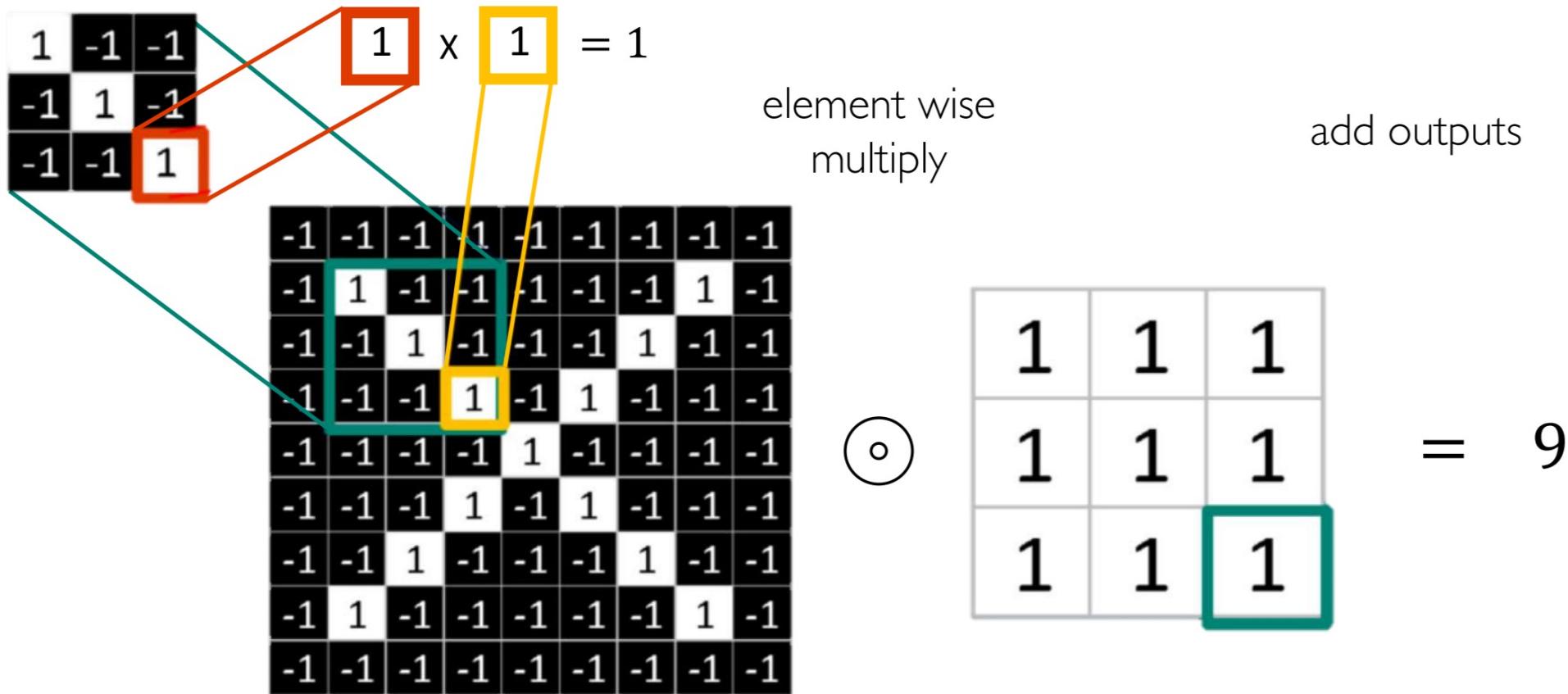


Case Study of Convolution

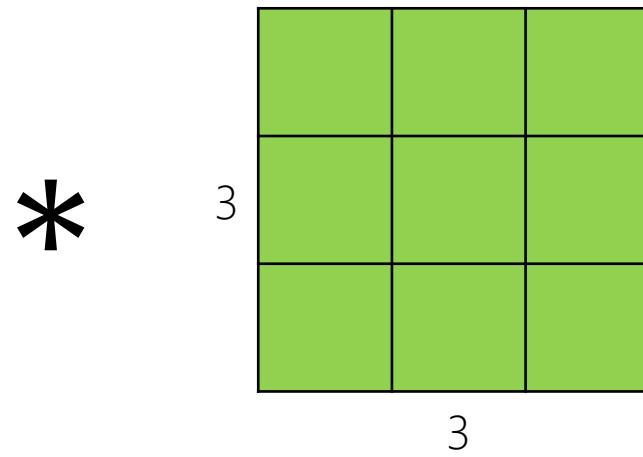
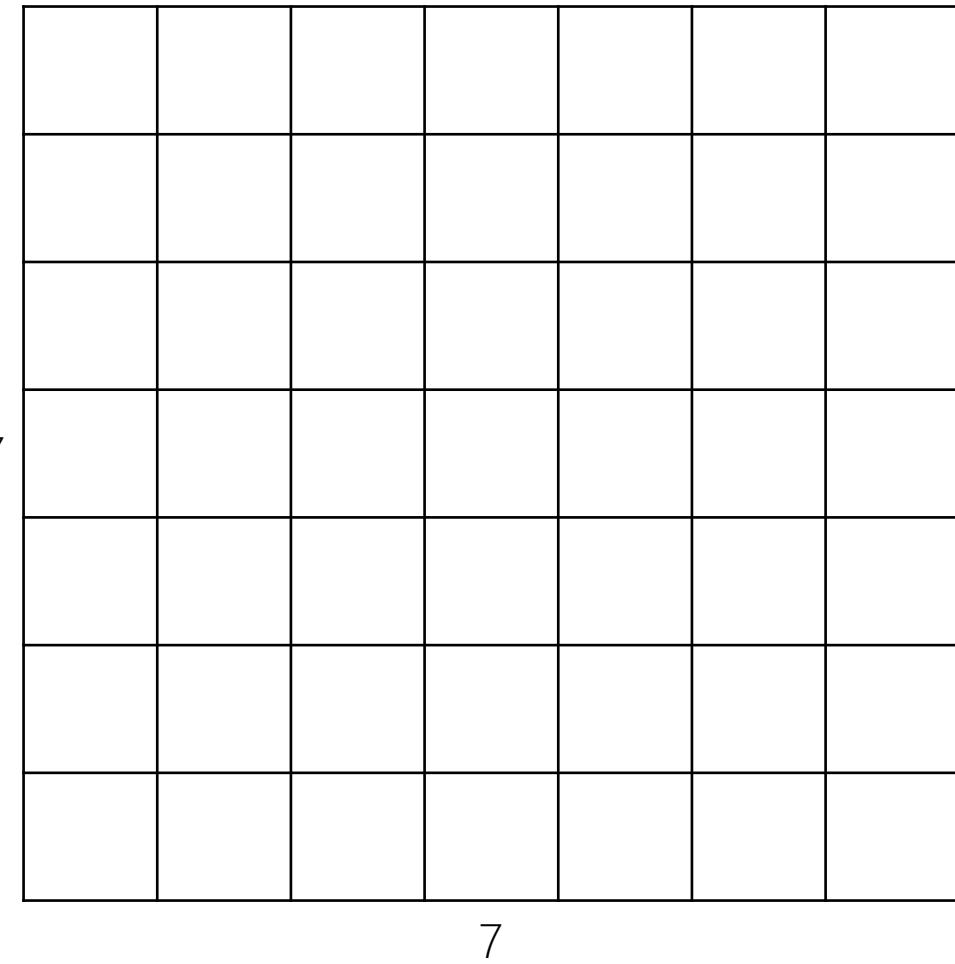
Filters to Detect X Features



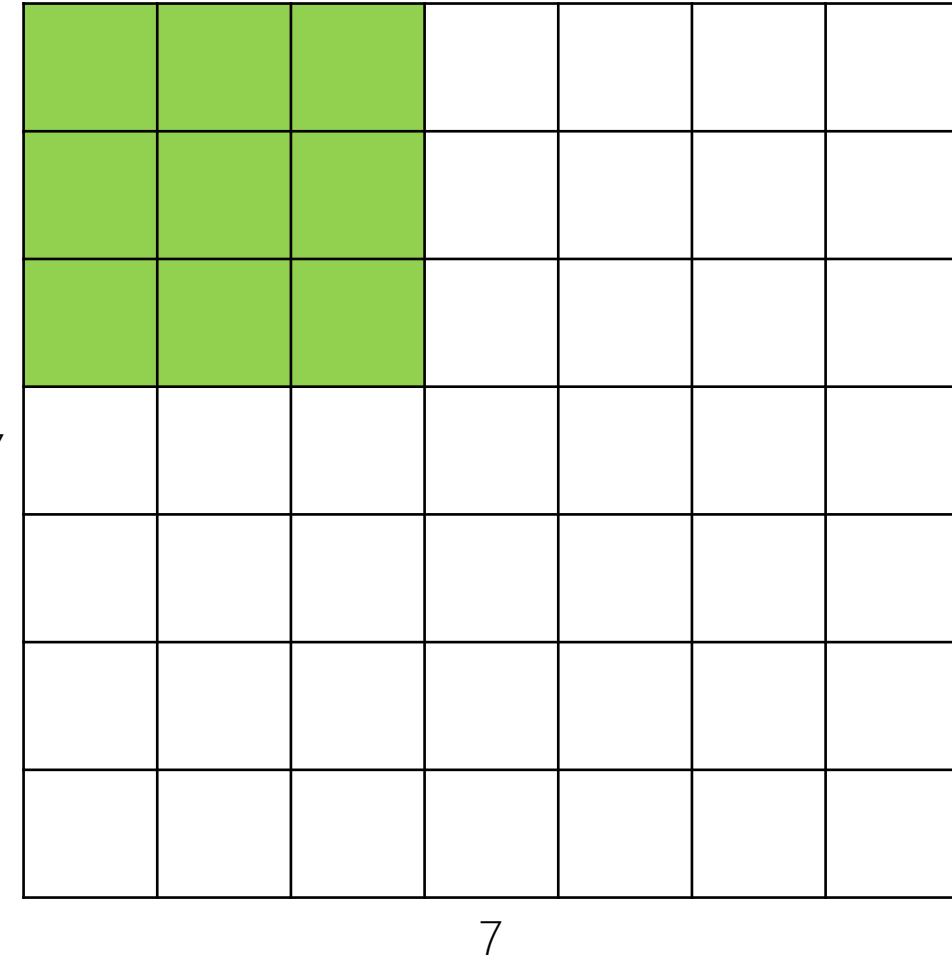
Case Study of Convolution



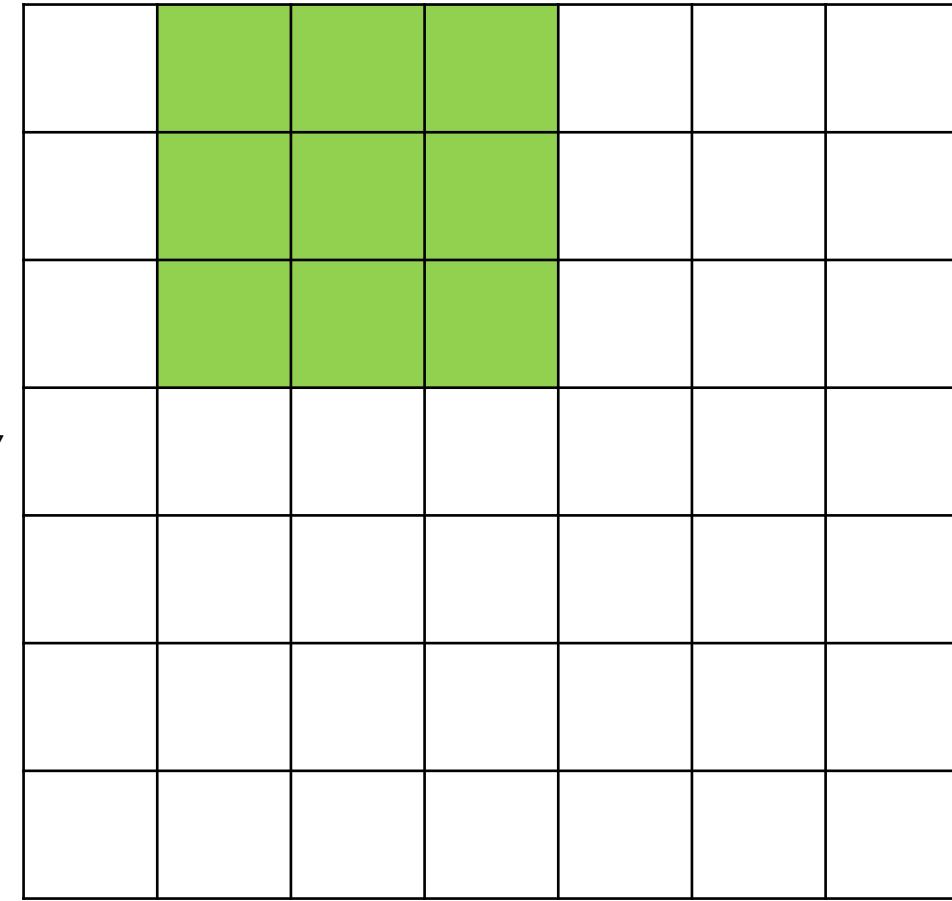
Stride & Padding



Stride & Padding

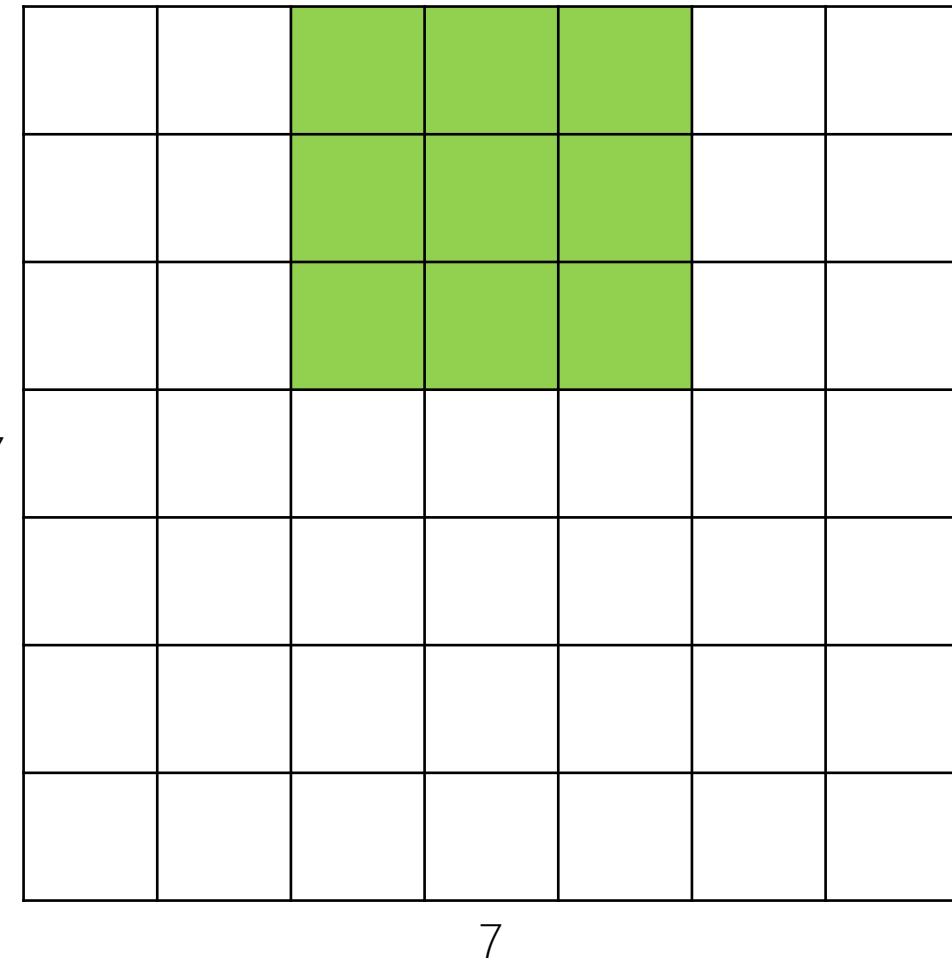


Stride & Padding



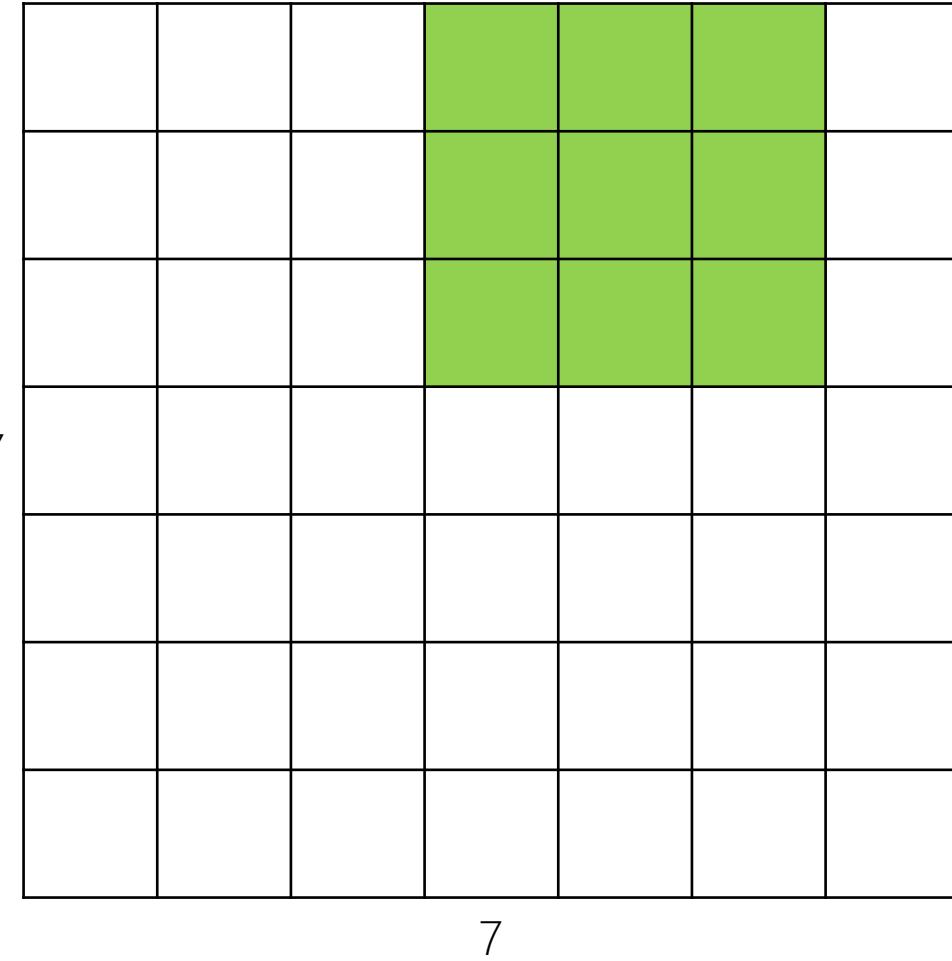
7

Stride & Padding

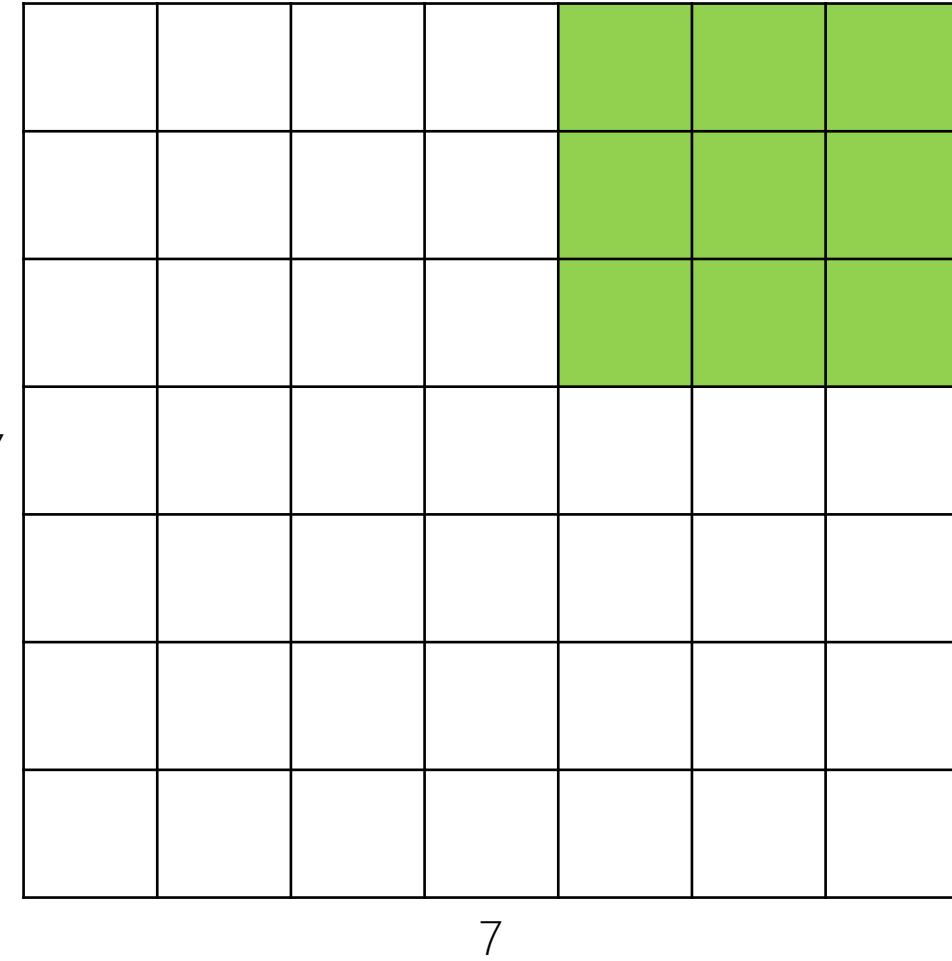


7

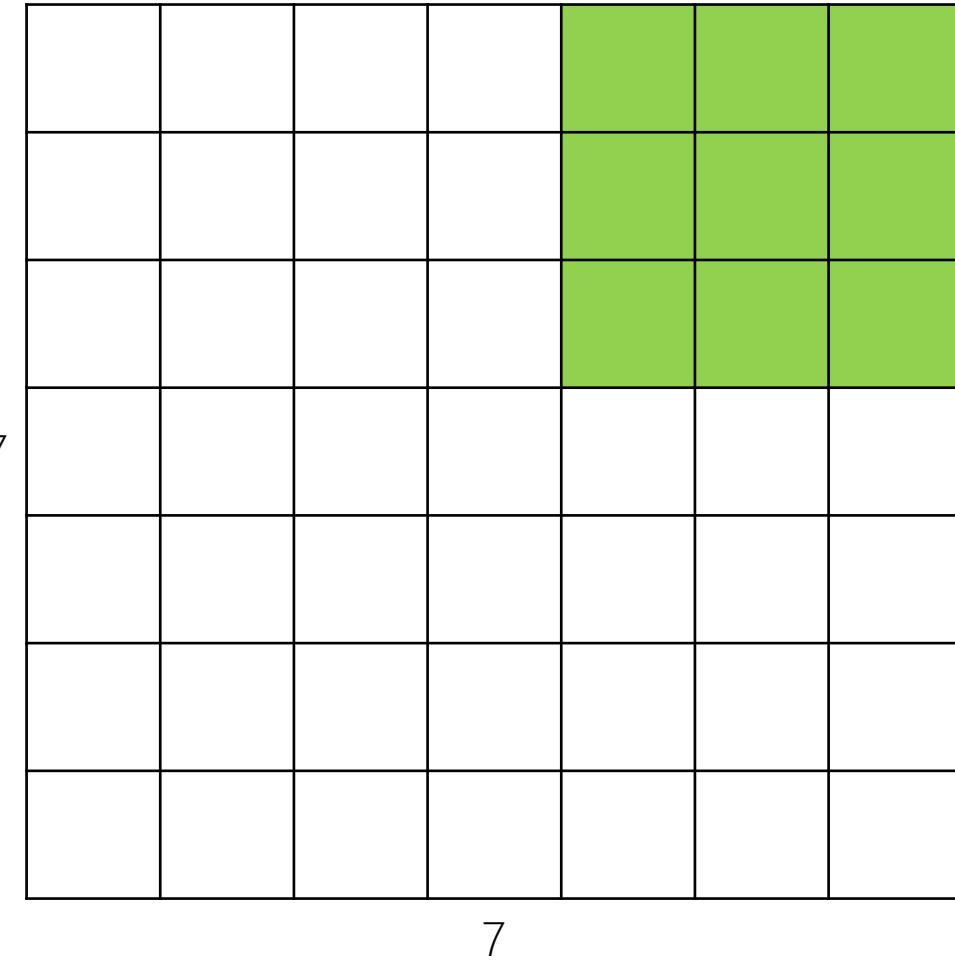
Stride & Padding



Stride & Padding

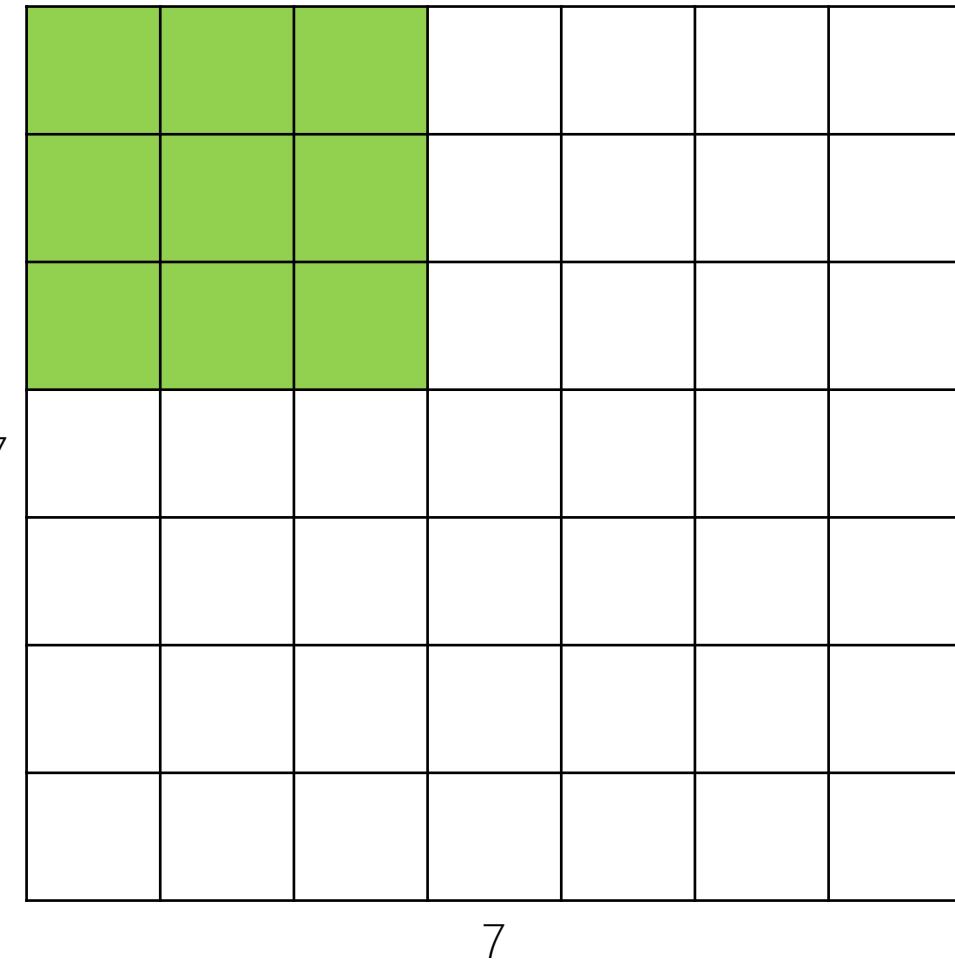


Stride & Padding



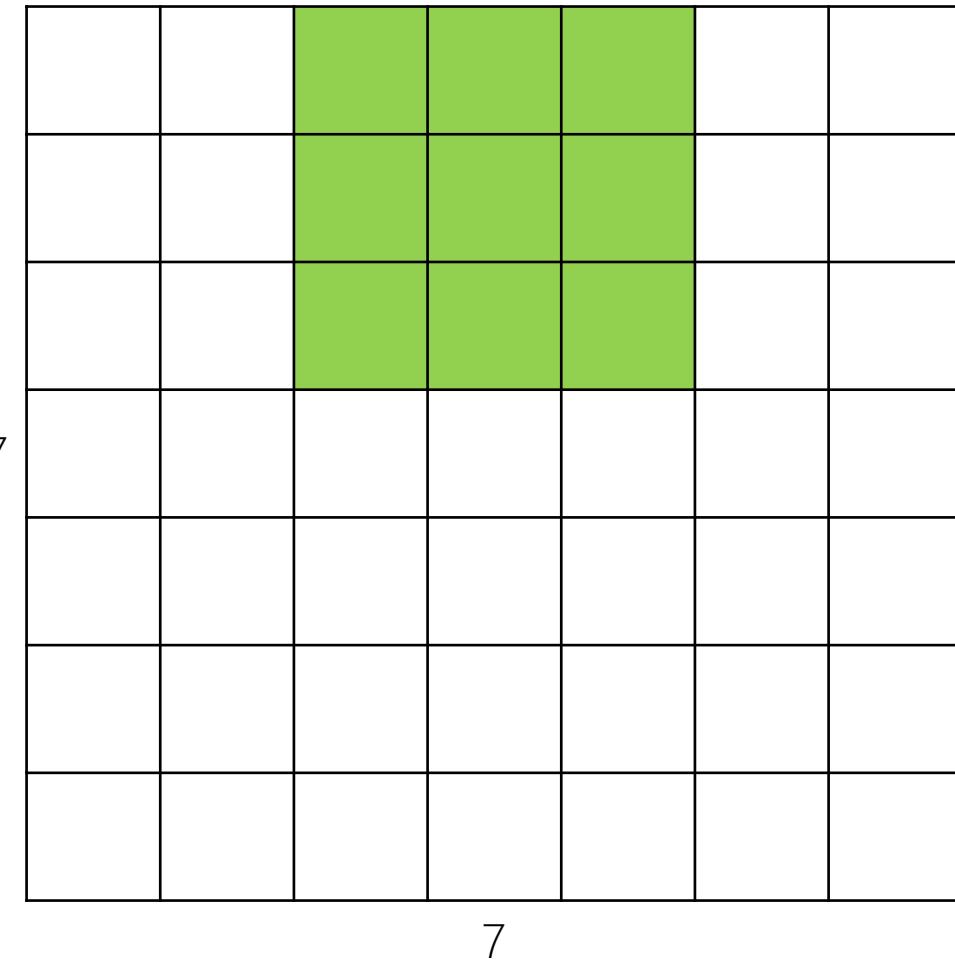
→ 5x5 output

Stride & Padding



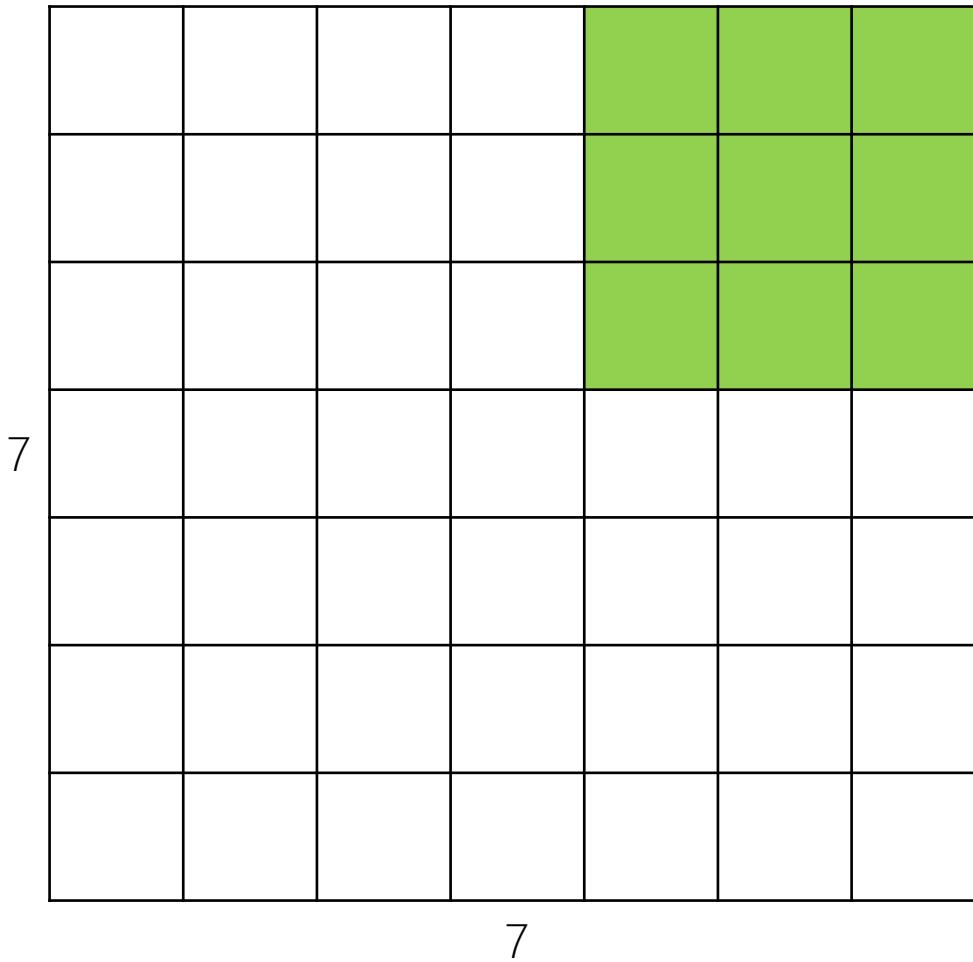
Now with **stride 2**

Stride & Padding



Now with **stride 2**

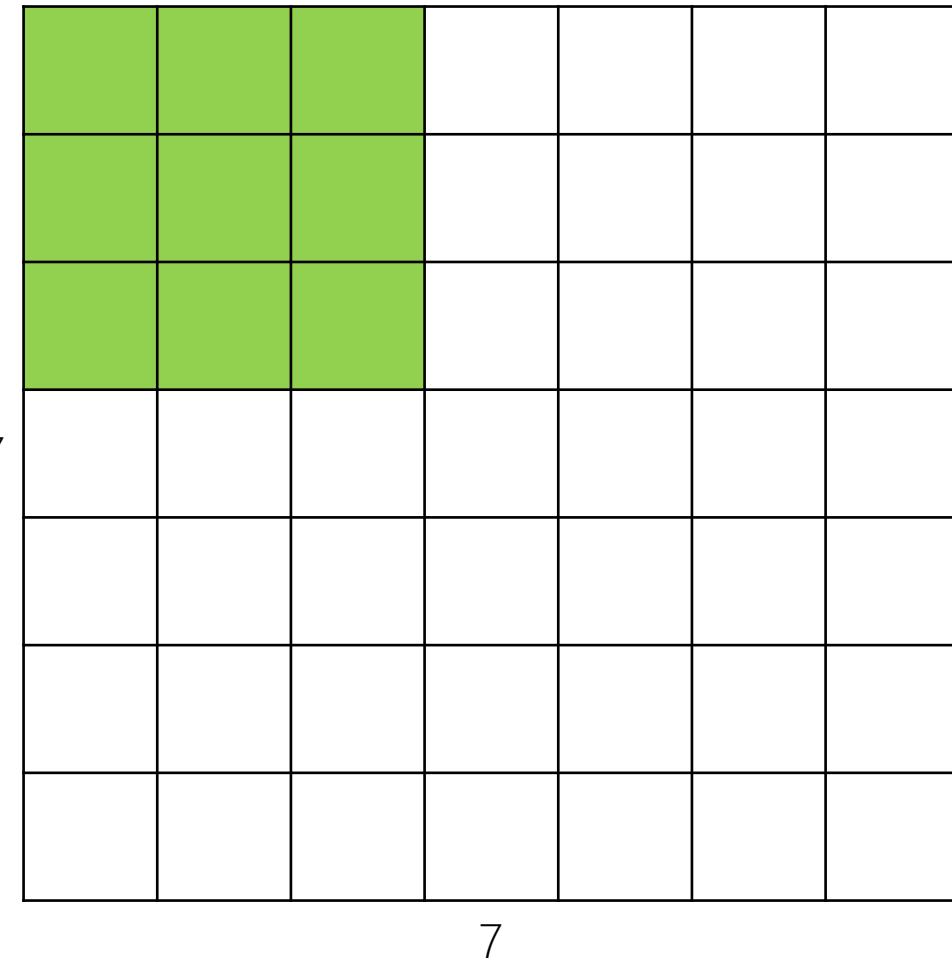
Stride & Padding



Now with **stride 2**

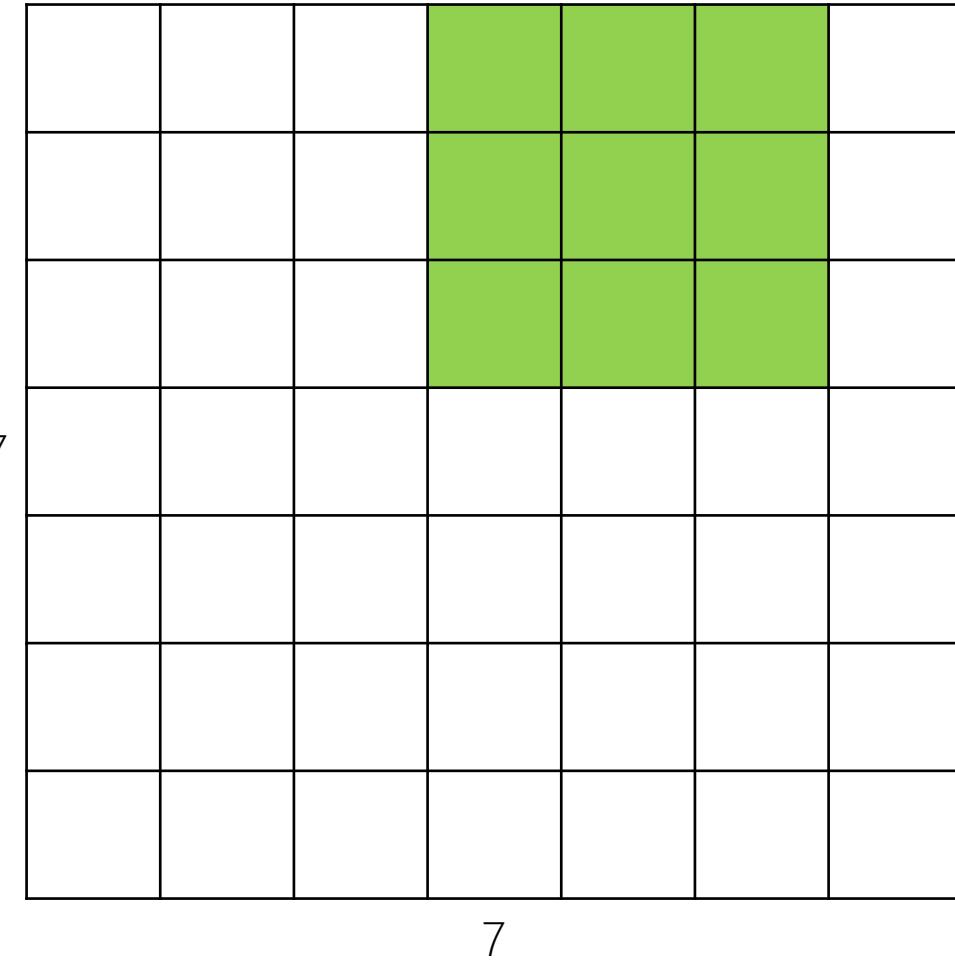
→ 3x3 output

Stride & Padding



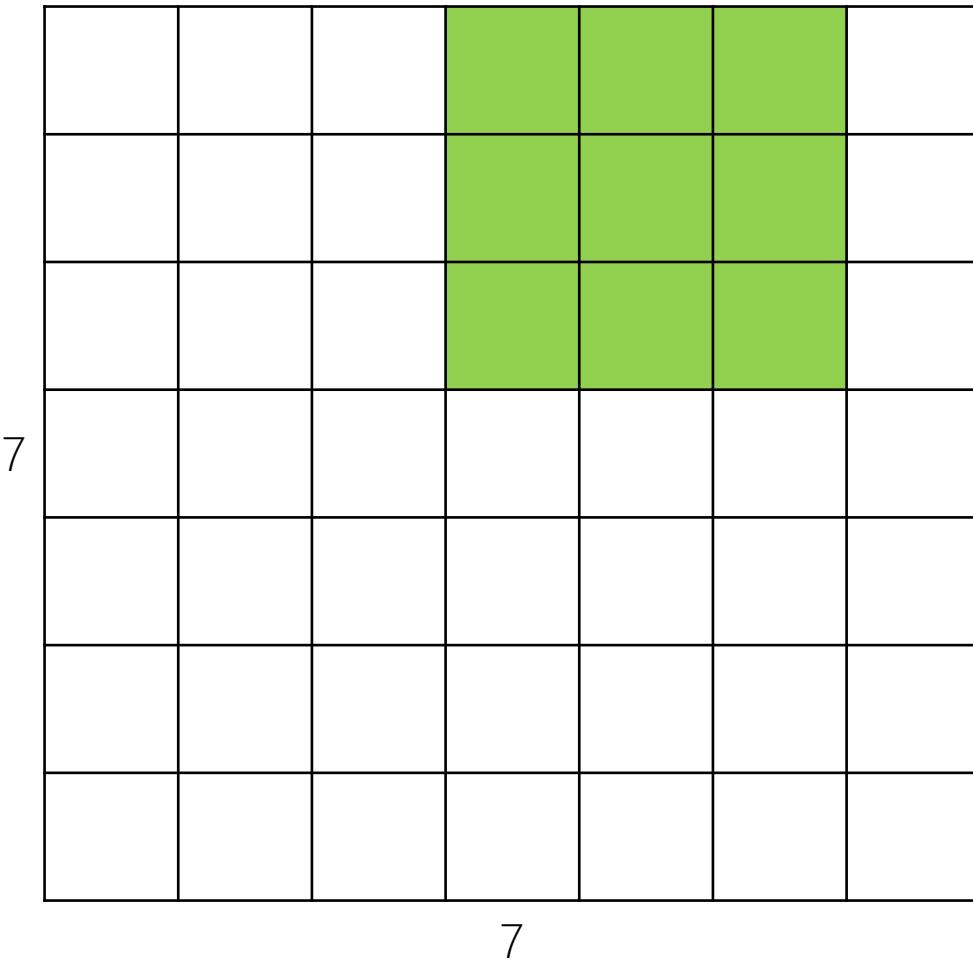
How about with **stride 3**?

Stride & Padding



How about with **stride 3**?

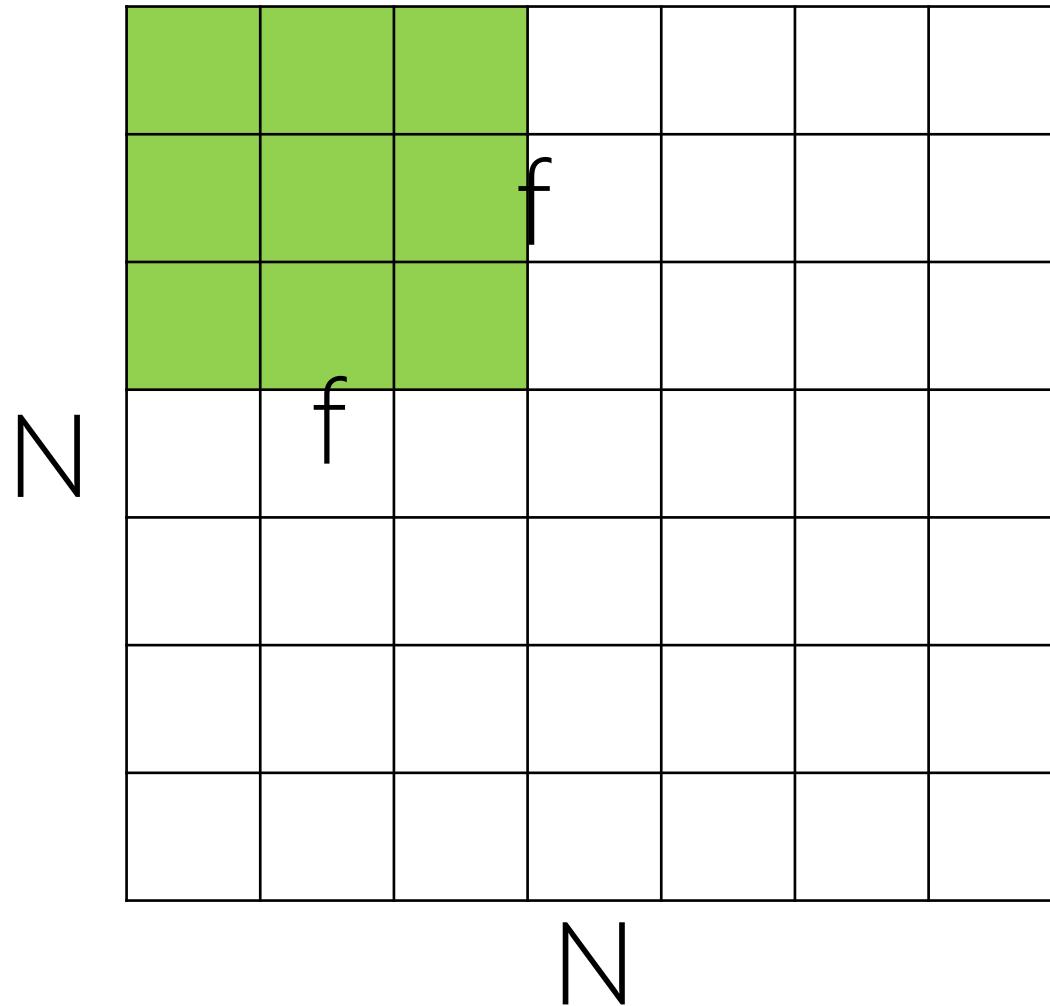
Stride & Padding



How about with **stride 3**?

→ ...?

Stride & Padding



Output size:
 $(N-f)/\text{stride} + 1$

Example, $N=7$, $f=3$:

$$\text{Stride 1: } (7-3)/1 + 1 = 5$$

$$\text{Stride 2: } (7-3)/2 + 1 = 3$$

$$\text{Stride 3: } (7-3)/3 + 1 = 2.3333$$

Stride & Padding

0	0	0	0	0	0	0	0	0
0								0
0								0
0								0
0								0
0								0
0								0
0								0
0	0	0	0	0	0	0	0	0

Output size:

$$(N-f)/\text{stride} + 1$$

Q. 7x7 input, 3x3 kernel, with stride 1,
padded with 1 pixel. Output size?

Stride & Padding

0	0	0	0	0	0	0	0	0
0								0
0								0
0								0
0								0
0								0
0								0
0								0
0	0	0	0	0	0	0	0	0

Output size:

$$(N-f)/\text{stride} + 1$$

Q. 7x7 input, 3x3 kernel, with stride 1,
padded with 1 pixel. Output size?

→ 7x7

Stride & Padding

0	0	0	0	0	0	0	0	0
0								0
0								0
0								0
0								0
0								0
0								0
0								0
0	0	0	0	0	0	0	0	0

Output size:
 $(N-f)/\text{stride} + 1$

Q. 7x7 input, 3x3 kernel, with stride 1,
padded with 1 pixel. Output size?

→ 7x7

Q. 7x7 input, 3x3 kernel, with stride 3.
You want to make 3x3 output, padding?

Stride & Padding

0	0	0	0	0	0	0	0	0
0								0
0								0
0								0
0								0
0								0
0								0
0								0
0	0	0	0	0	0	0	0	0

Output size:

$$(N-f)/\text{stride} + 1$$

Q. 7x7 input, 3x3 kernel, with stride 1,
padded with 1 pixel. Output size?

→ 7x7

Q. 7x7 input, 3x3 kernel, with stride 3.
You want to make 3x3 output, padding?

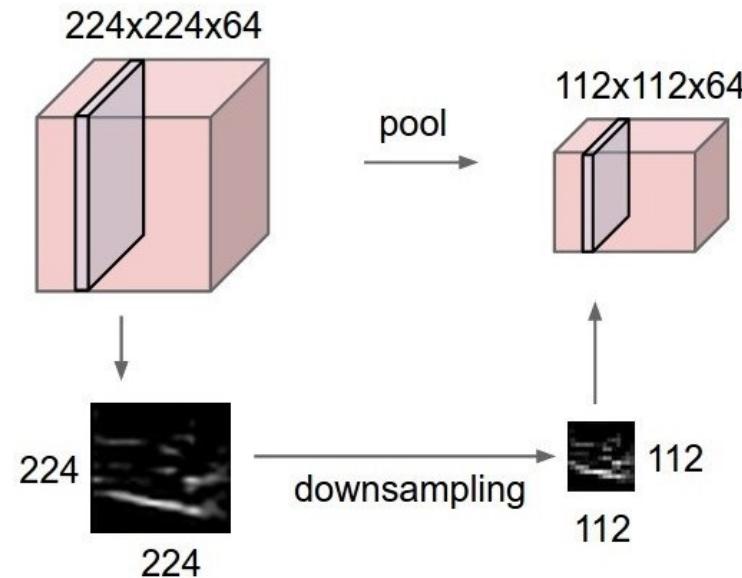
→ 1

More questions:

- Input volume: $50 \times 50 \times 3$
- $10 \times 5 \times 3$ filters with stride 1 , pad 2
- Output size?
 - $(50 + 2*2 - 5)/1 + 1 = 50$
 - Therefore, $50 \times 50 \times 10$
- Total number of parameters in this layer?
 - Each filter: $5 \times 5 \times 3 + 1 = 76$ parameters
 - Therefore, $76 * 10 = 760$

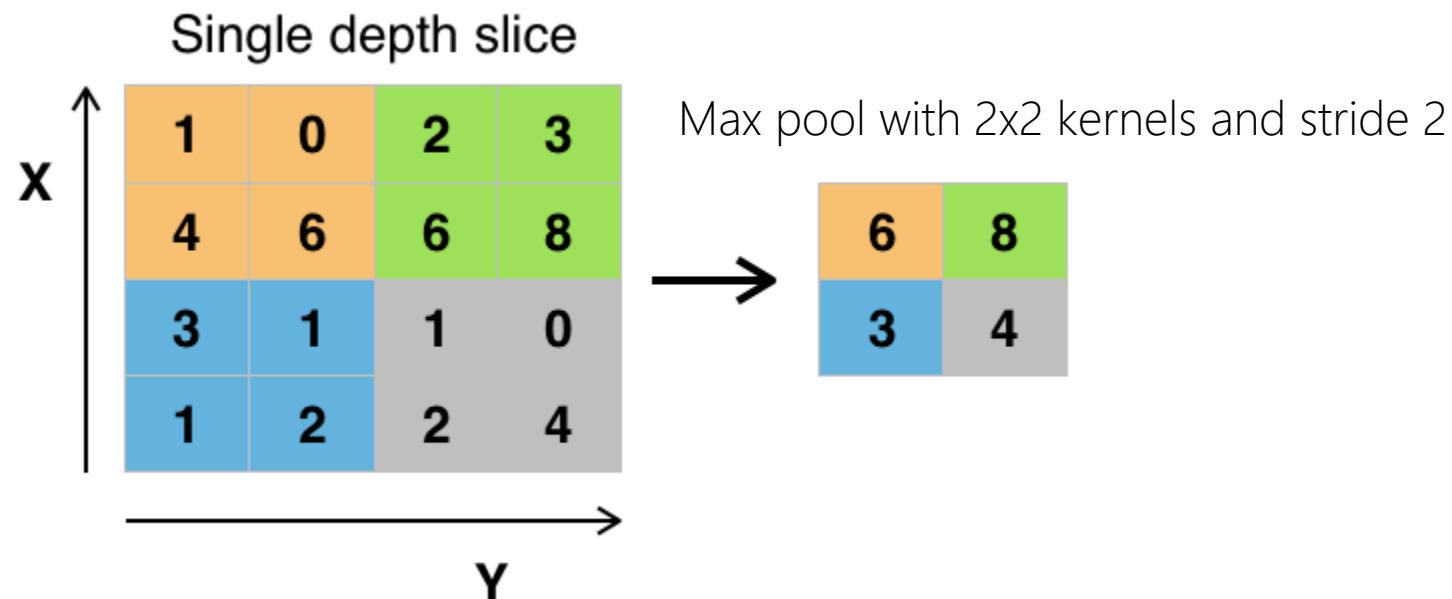
Pooling Layers

- Pooling == ConvNet way of downsampling
- “Reduce the image size” to obtain smaller but more manageable features
- Receptive fields becomes relatively larger as the image size gets smaller
- Operates over each activation map independently

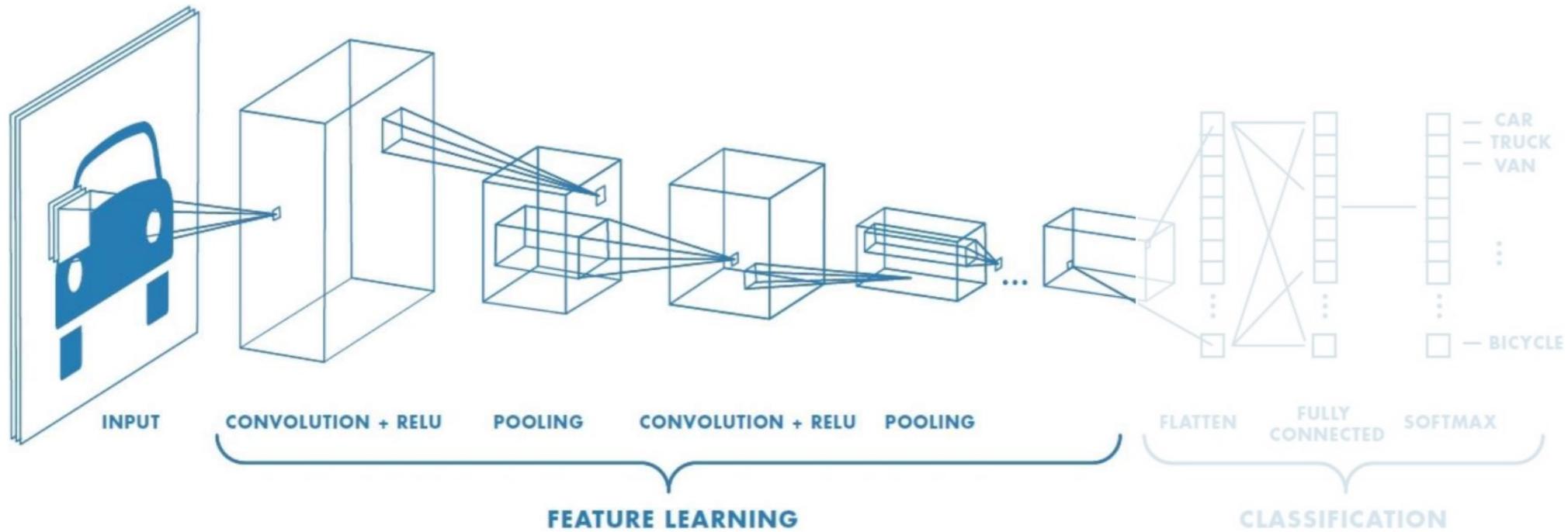


Max Pooling

- A non-linear down-sampling method
- An image is partitioned into a set of (non-overlapping) rectangles.
- The maximum of each such sub-region is sampled.

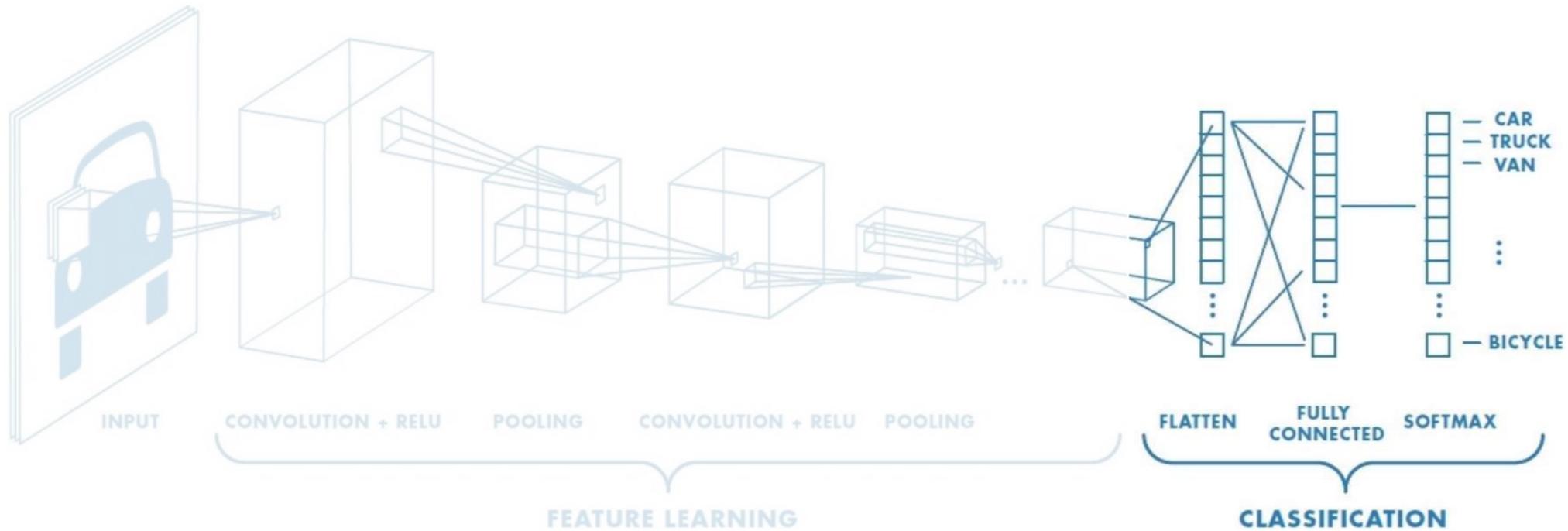


CNN for Classification: Feature Learning



1. Learn features in input image through **convolution**
2. Introduce **non-linearity** through activation function (real-world data is non-linear!)
3. Reduce dimensionality and preserve spatial invariance with **pooling**

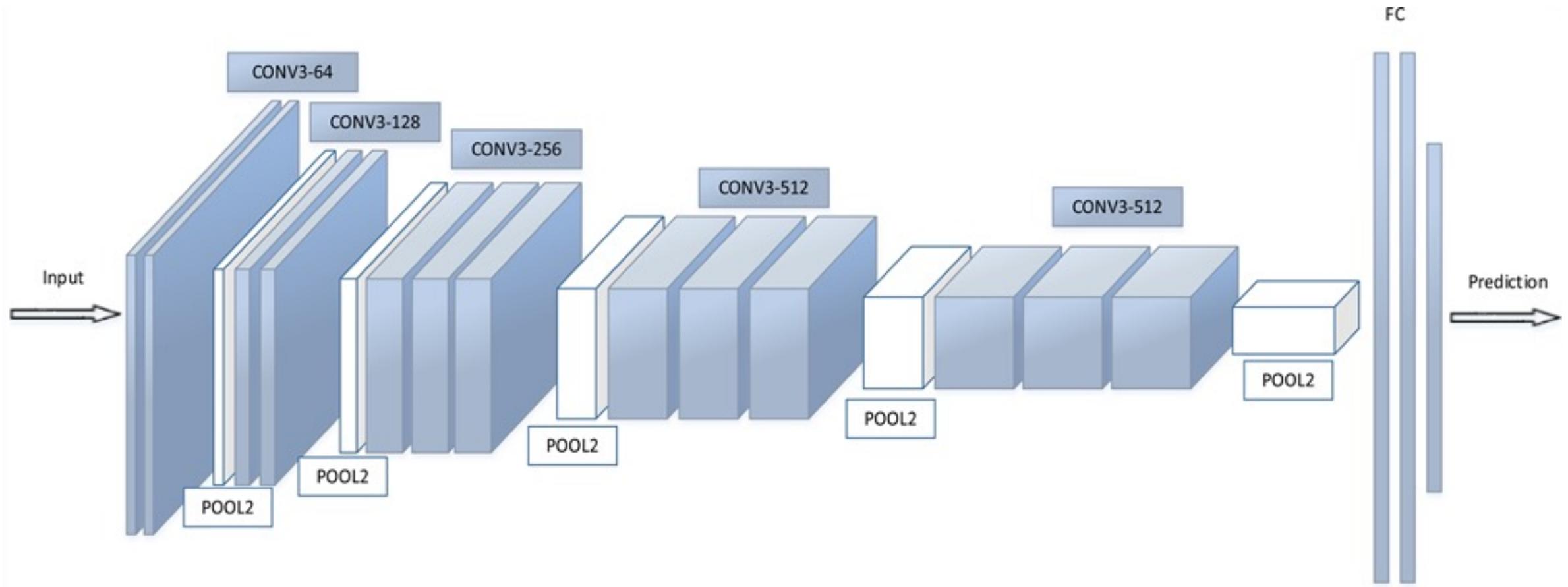
CNN for Classification: Prediction



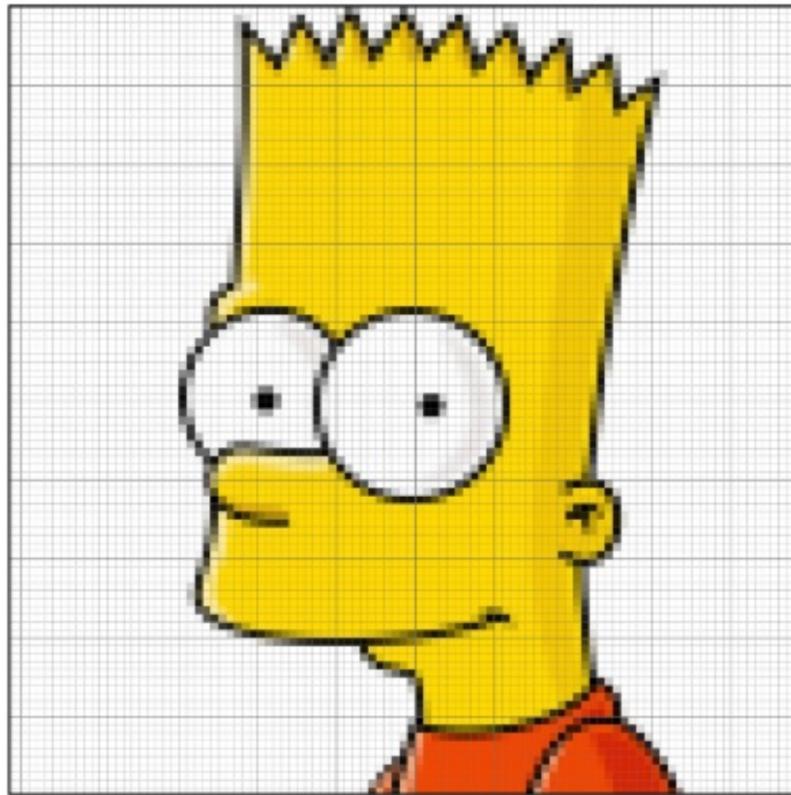
- CONV and POOL layers output high-level features of input
- Fully connected layer uses these features for classifying input image
- Express output as **probability** of image belonging to a particular class

$$\text{softmax}(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}}$$

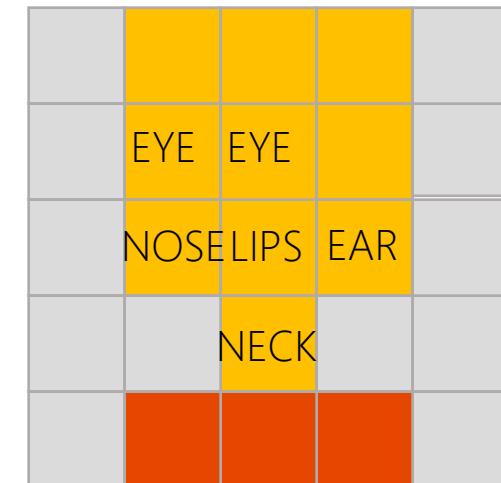
VGG Networks



Abstraction of an Image

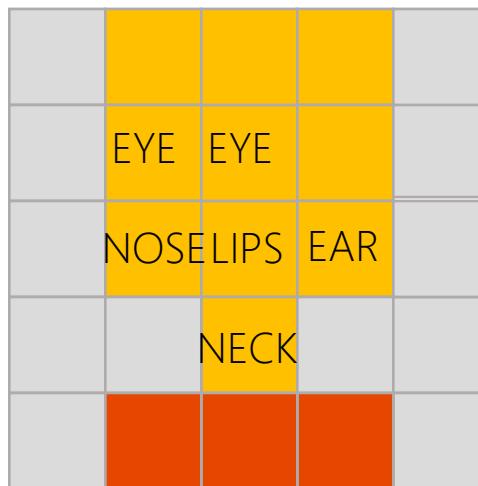


Conv2D + MaxPooling
+ Conv2D + MaxPooling
+ Conv2D + MaxPooling
+ ...



Abstraction of an Image

- Final decision is made by the fully-connected layers:



"Has two eyes, a nose, lips, an ear, and a neck
and wears t-shirts"



Cat	X
Dog	X
Person	O
Orange	X