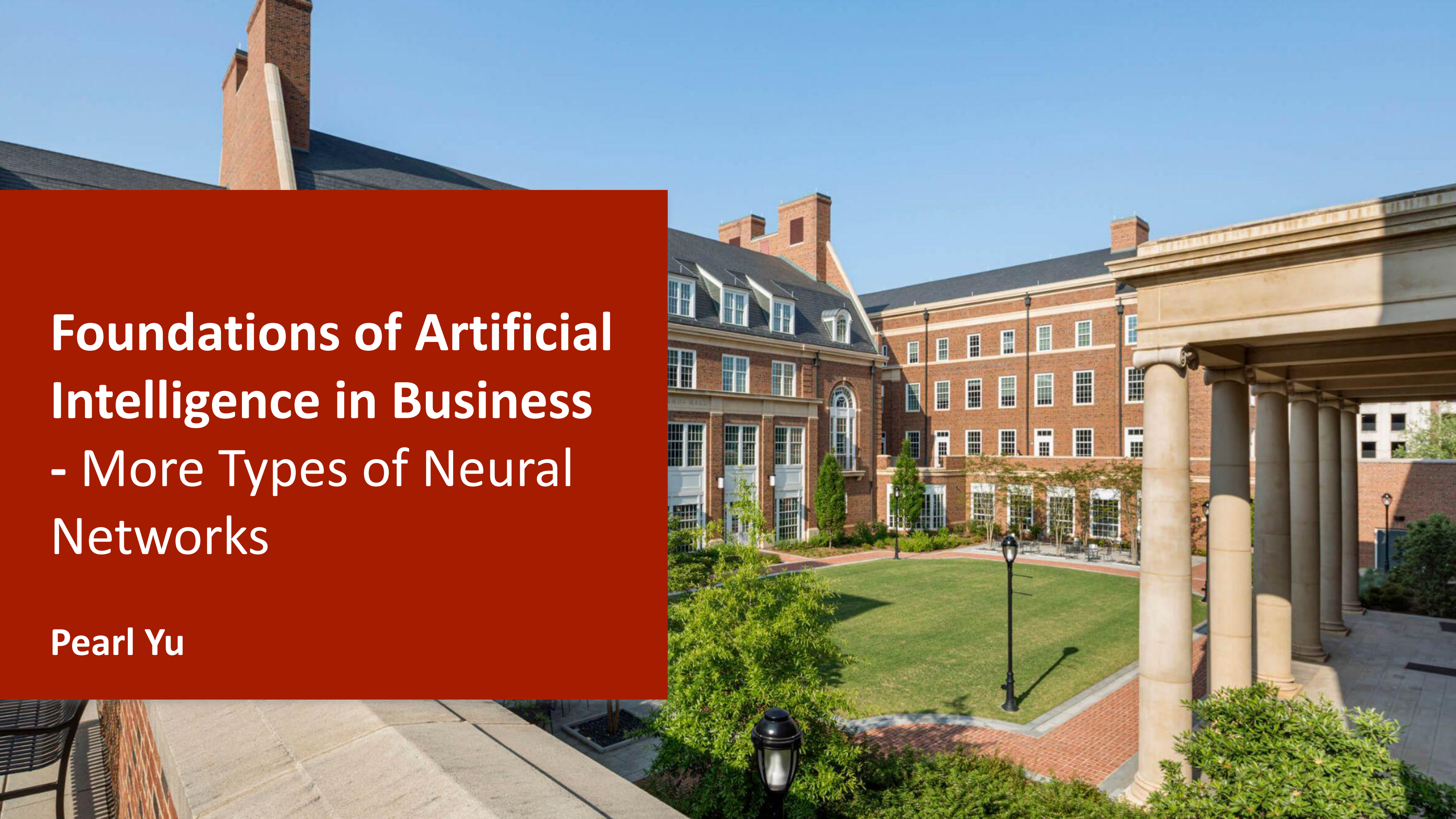


Foundations of Artificial Intelligence in Business
- More Types of Neural Networks

Pearl Yu

Learning Goals

- Understand how to represent images digitally.

- Understand the idea of convolution.

- Have a general idea of the overall architecture of Convolutional Neural Networks.

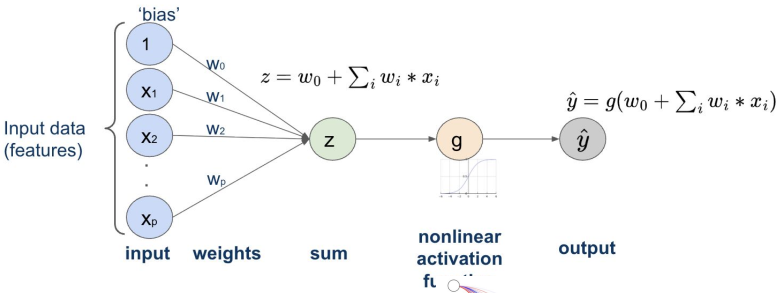
  (We want to deepen our understanding of how different neural network structures are for different tasks and have different strengths.)

Our learning map of perception AI

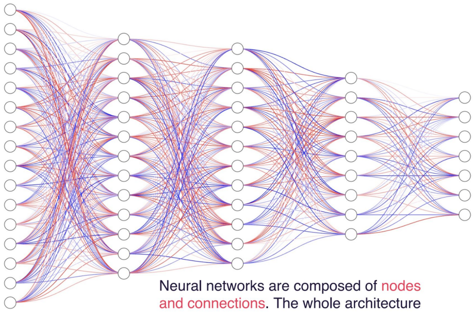
Basics of Neural Networks and Deep Learning

DONE.

Summary

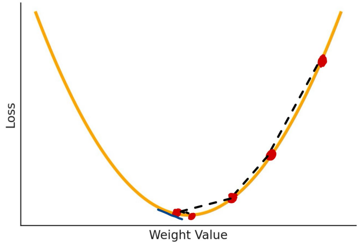


- Define the model
    - Basic unit: Perceptron
        - The weights: how important the node(variable) is.
        - The non-linear activation function.
    - Architecture of neural networks
        - Stacking layers of perceptron



- Training:
    - Weight initialization
    - Back propagation - loops till convergence
        - Forward pass
        - Backward pass
            - Apply the chain rule to calculate the gradients
            - Update weights using gradient descent

$$\frac{\partial Loss}{\partial w_1'} = \frac{\partial Loss}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z_1} * \frac{\partial z_1}{\partial w_1'}$$



- Evaluation
    - Regularization techniques to avoid overfitting
        - Dropout/ Early stopping

Our learning map of perception AI

Basics of Neural Networks and Deep Learning

DONE.

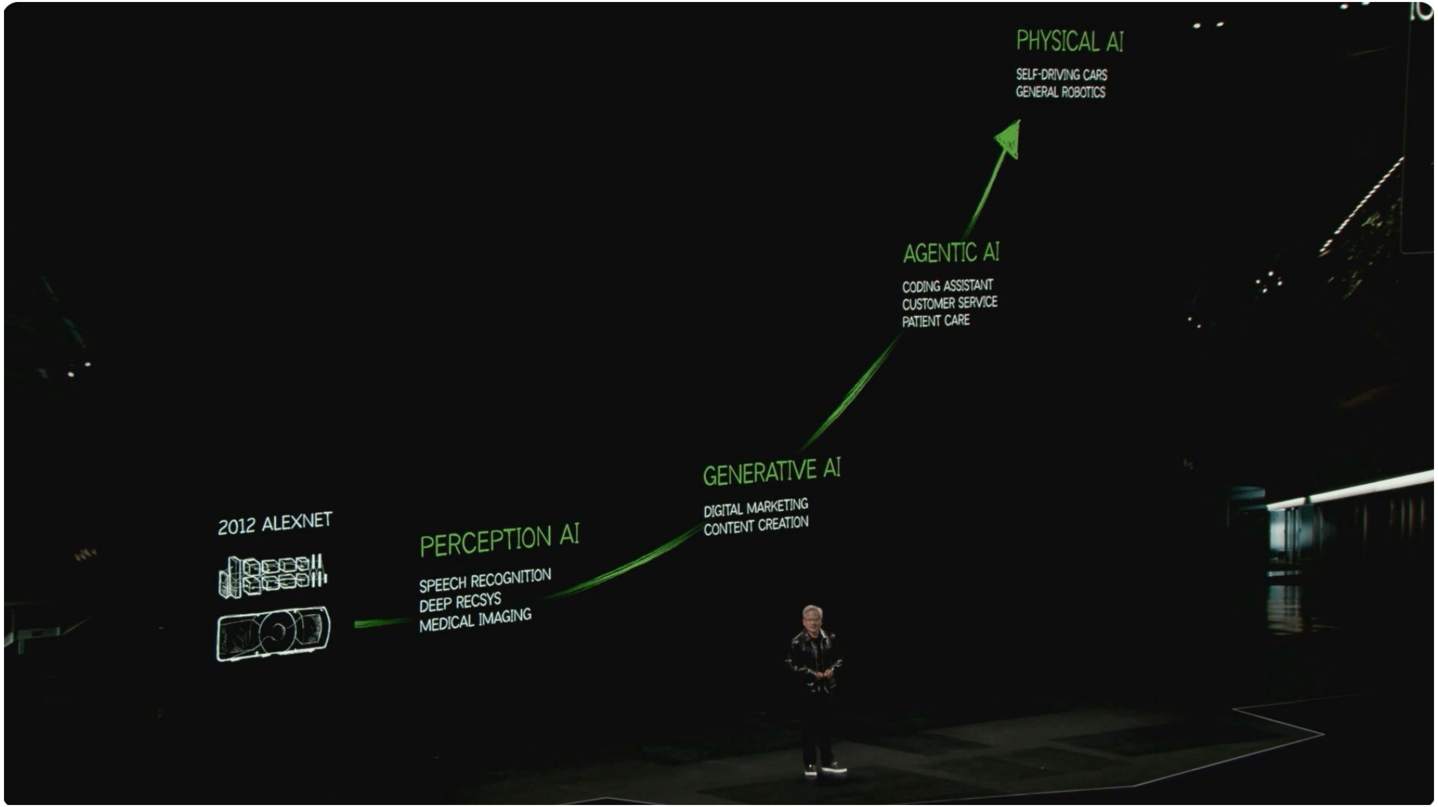
How are **images/ languages** (networks, protein structures) **represented in numbers** so NN can take them as inputs?

For different types of inputs, some neural **network architectures** would perform better.

images

languages

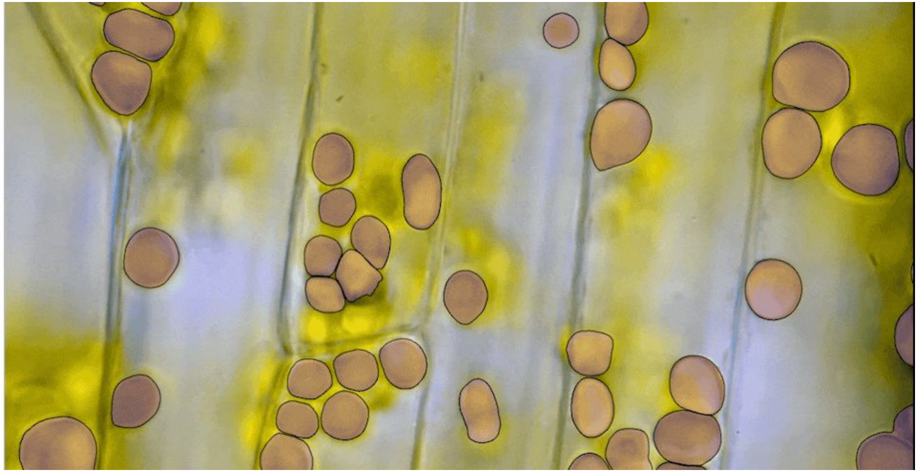
State of the World



The Computer Vision Revolution

- Goal: Discover from images what is present in the world, where things are, what actions are taking place, to predict and anticipate events in the world.

To see what humans can see

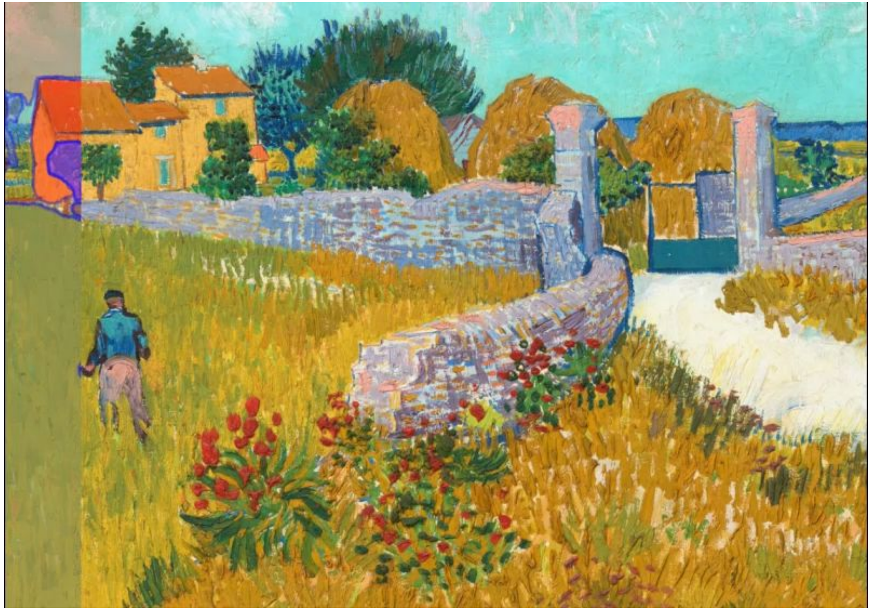


To see what humans don’t see

https://segment-anything.com/

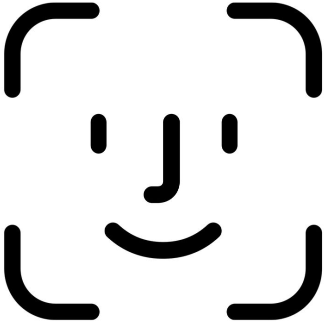
Computer Vision

- Segmentation & Object Identification



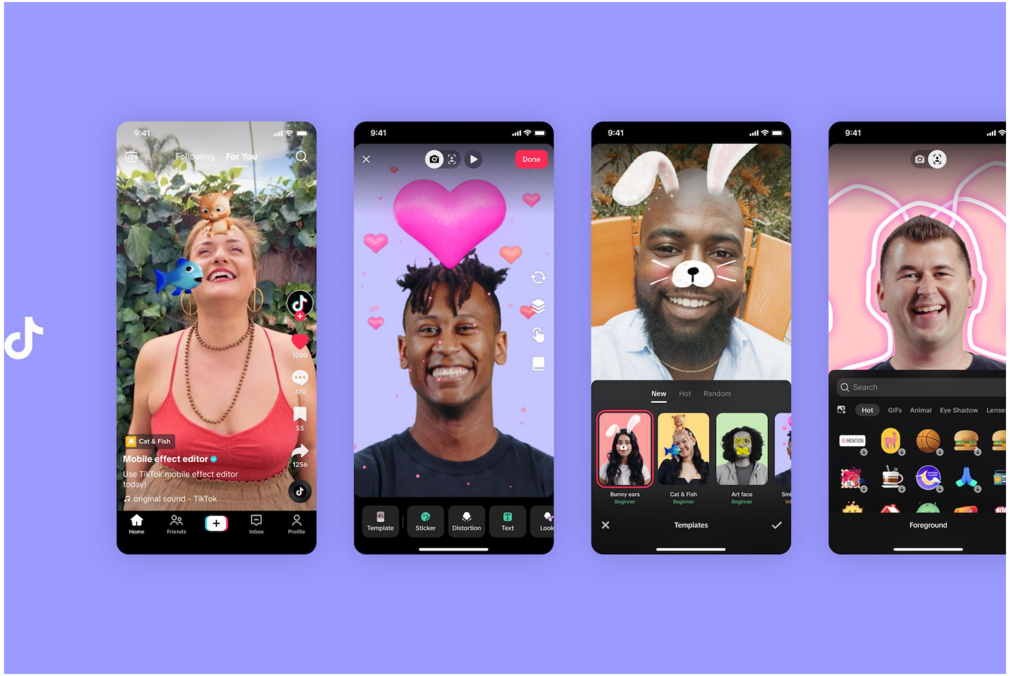
Computer Vision

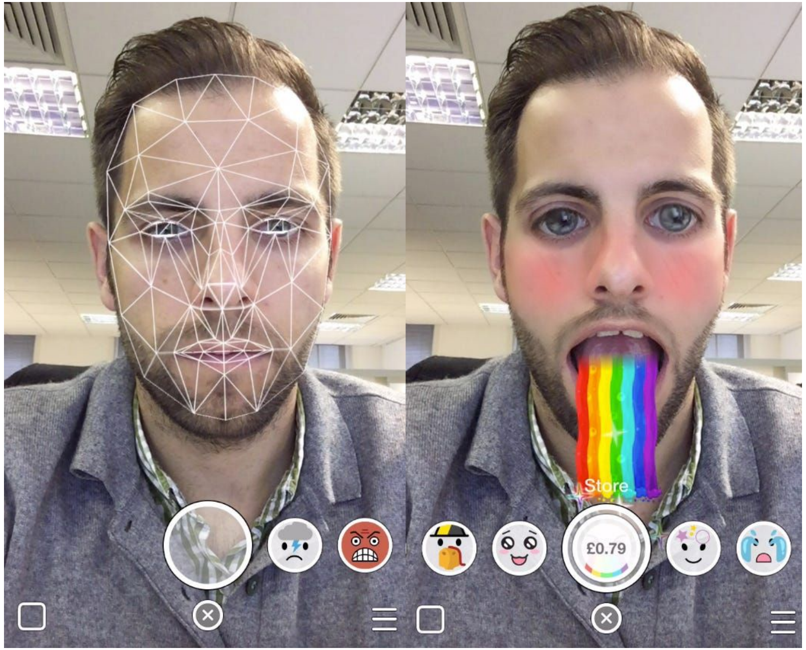
- Facial Detection and Recognition



Computer Vision

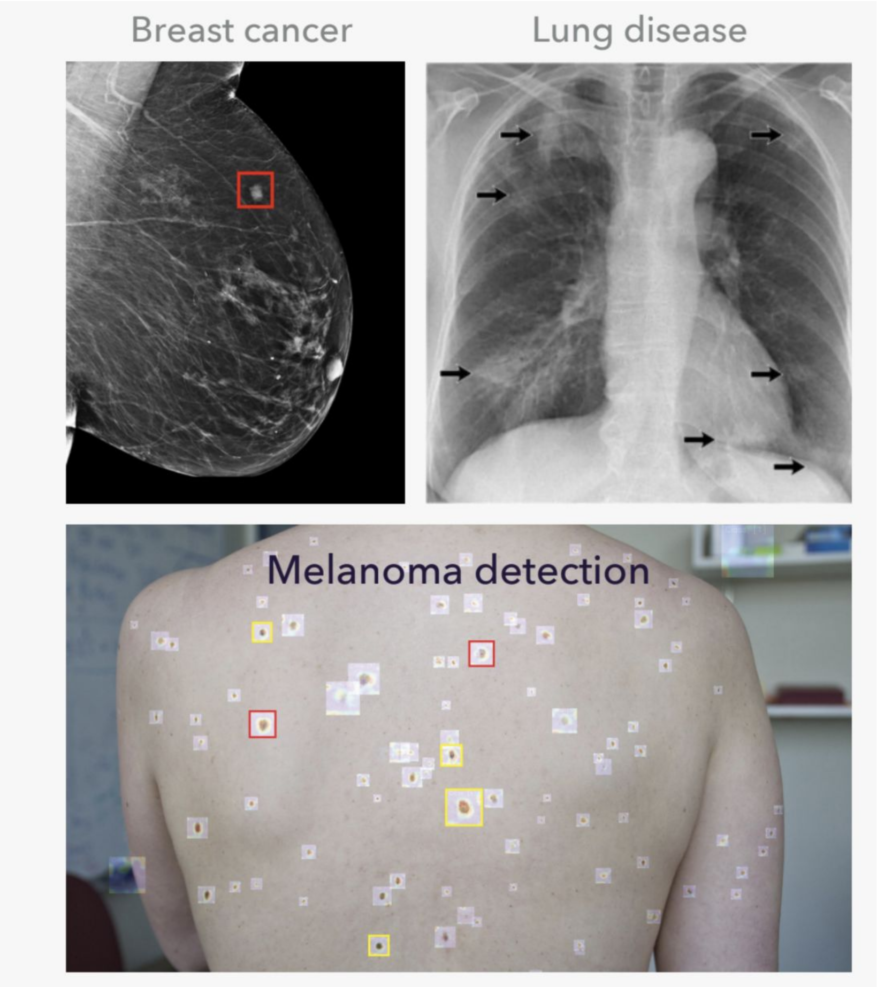
- Facial Detection and Recognition





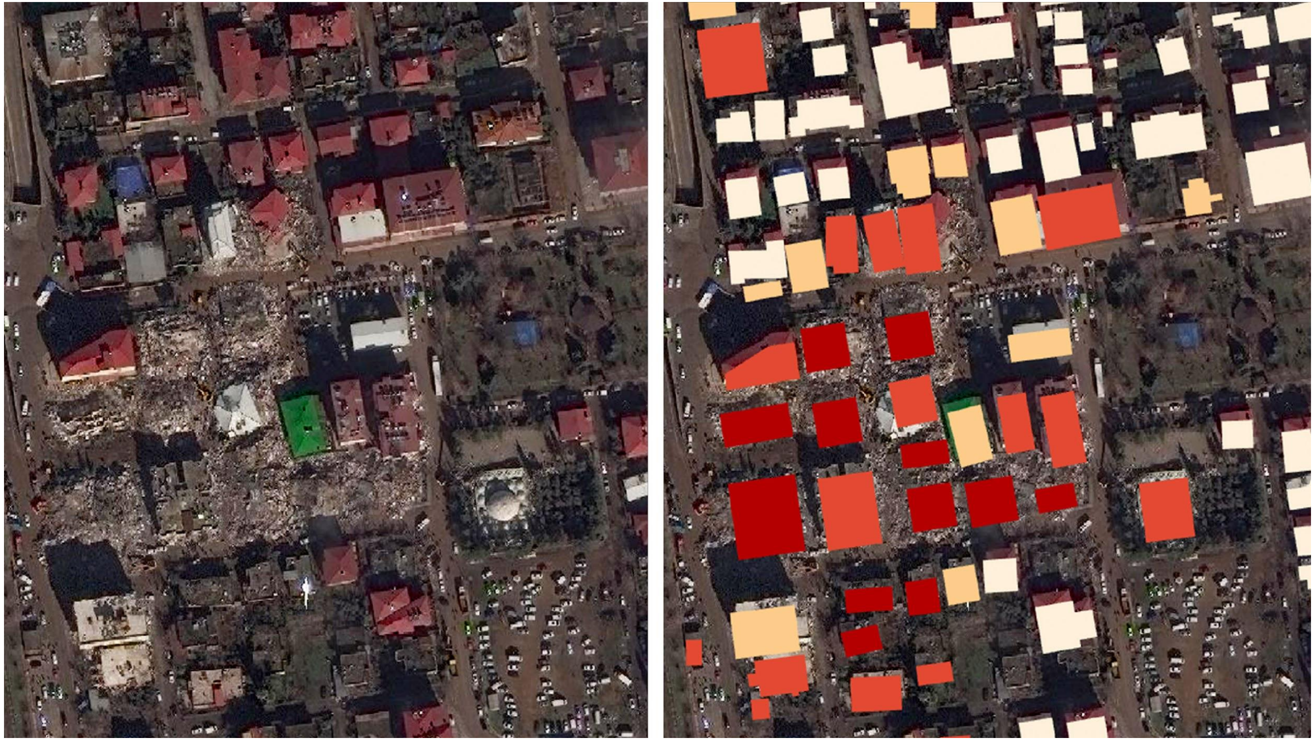
Computer Vision

- Biology, Healthcare



Computer Vision

- Satellite Images -> Natural Disaster Response



Computer Vision

- Autonomous Driving

Computer Vision

- Robotics

Computer Vision

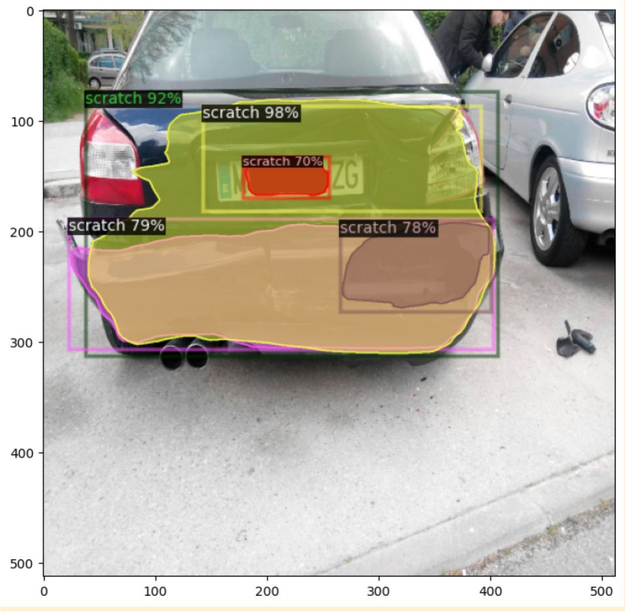
- Accessibility

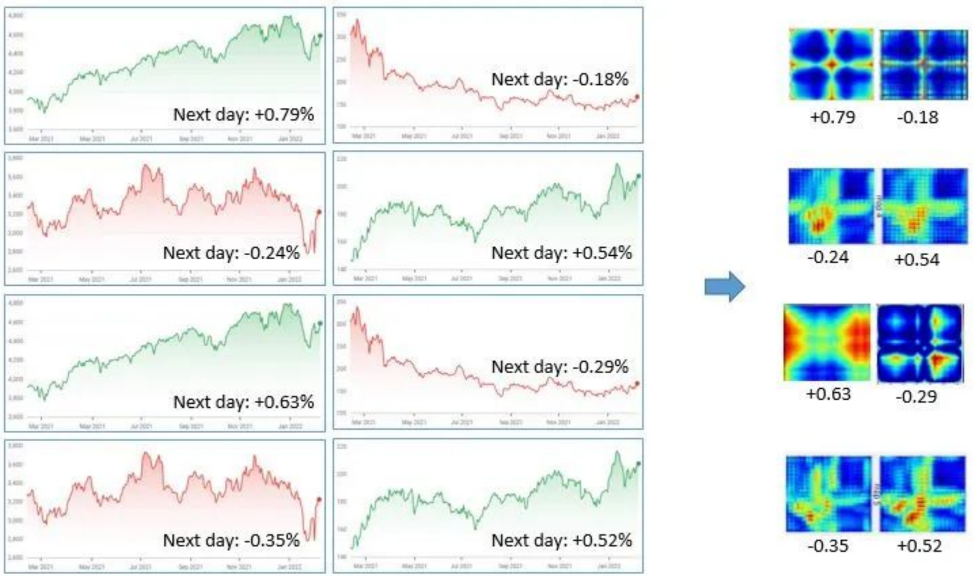
Computer Vision

- Future possibilities

Computer Vision

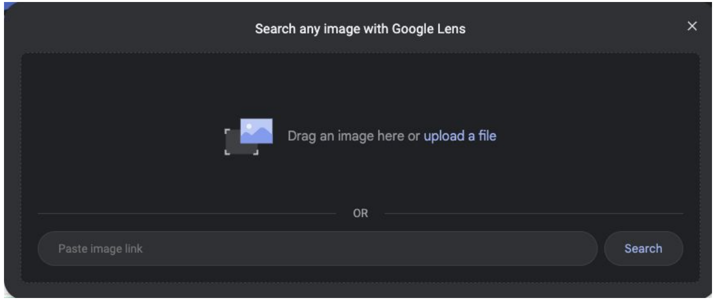
- In Business
  - Fraud detection (Altered images, insurance claim photos), Stock price forecasting

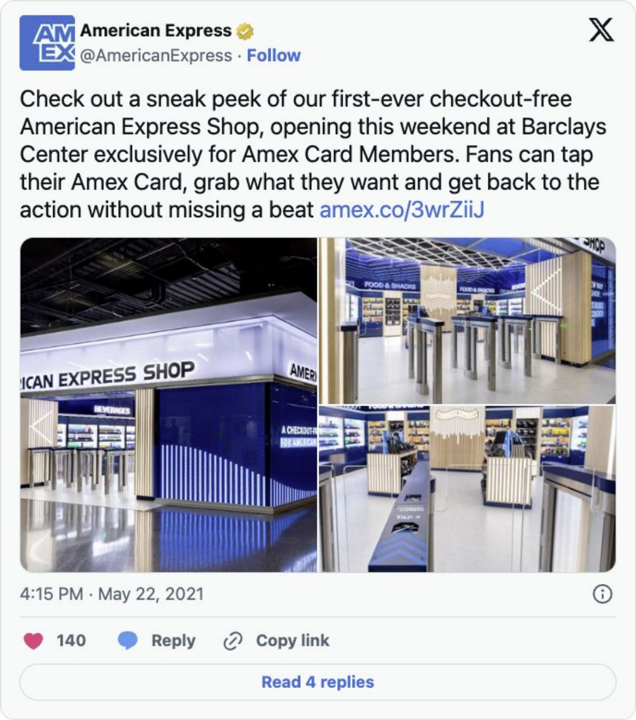




Computer Vision

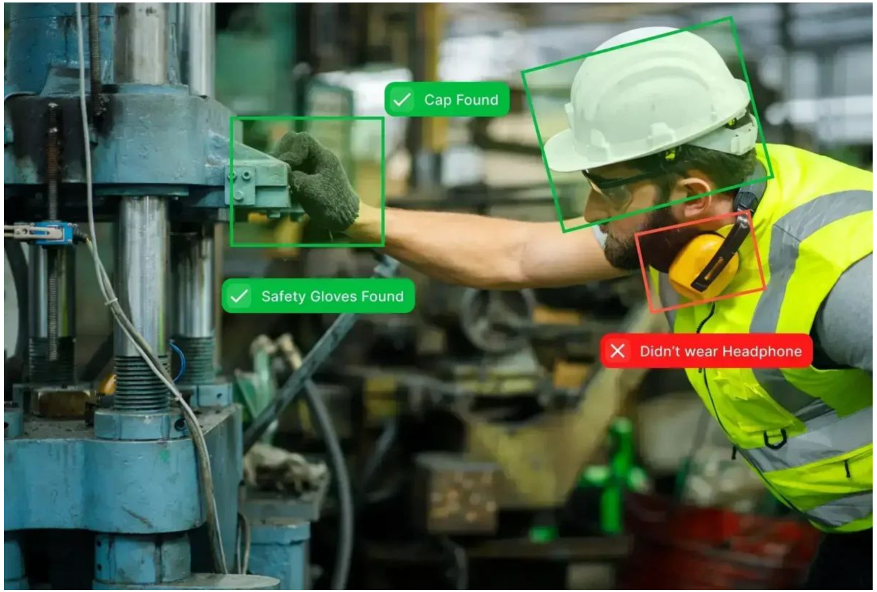
- In Business
  - Retail: Product image search, Checkout free stores





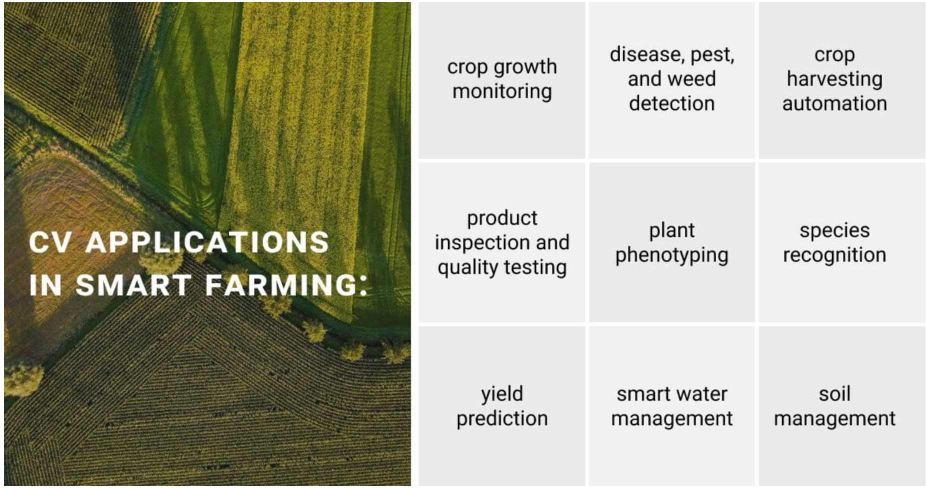
Computer Vision

- In Business
  - Manufacturing: Safety check, quality control



Computer Vision

- In Business
    - Smart farming



Representing Images Digitally

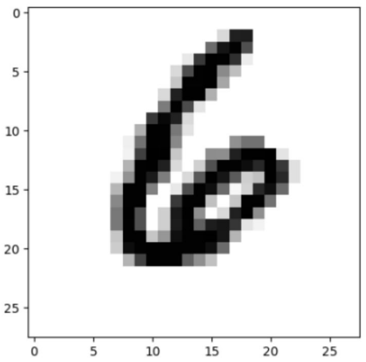
- How does computers see images?

The famous MINST dataset

- Collection of well-centered handwritten digits.
- Image Dimensions: 28x28 pixels, grayscale, i.e., with pixel values ranging from 0 (white) to 255 (black).
- Label Range: 0 to 9 (10 classes).
- Usage: Benchmark for machine learning models, especially in image recognition tasks.

Representing Images Digitally

- How does computers see images?



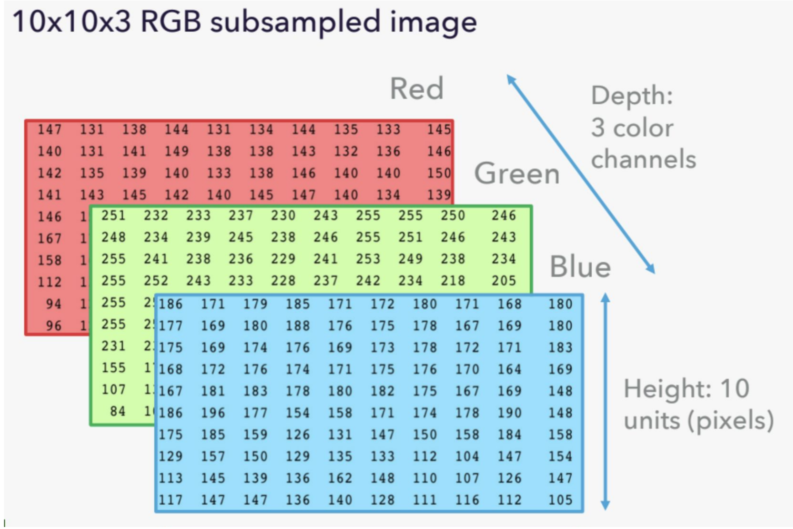

```
[[  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  38 222 225   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 147 234 252 176   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0  23 197 253 252 208  19   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0  38 178 252 253 117  65   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0  57 252 252 253  89   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0  38 222 253 253  79   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0 131 252 179  27   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0 198 246 220  37   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0  79 253 252 135  28   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0  16 140 253 252 118   0   0   0 111 140 140   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0  13 191 255 253  56   0   0 114 113 222 253 253 255  27   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0  76 252 253 223  37   0  48 174 252 252 242 214 253 199  31   0   0   0   0   0]
 [  0   0   0   0   0   0   0  13 109 252 228 130   0  38 165 253 233 164  49  63 253 214  31   0   0   0   0   0]
 [  0   0   0   0   0   0   0  73 252 252 126   0  23 178 252 240 148   7  44 215 240 148   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0 119 252 252   0   0 197 252 252  63   0  57 252 252 140   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0 135 253 174   0  48 229 253 112   0  38 222 253 112   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0 135 252 173   0  48 227 252 158 226 234 201  27  12   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0  57 252 252  57 104 240 252 252 253 233  74   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0  51 242 252 253 252 252 252 252 240 148   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0  75 189 253 252 252 157 112  63   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   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   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]
 [  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   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   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]]
```

A grayscale image is a matrix of pixels.

- The light intensity of each pixel is a number between 0 and 255.
- As the number increases from 0 to 255, the pixel goes from white through gray to black.

Representing Images Digitally

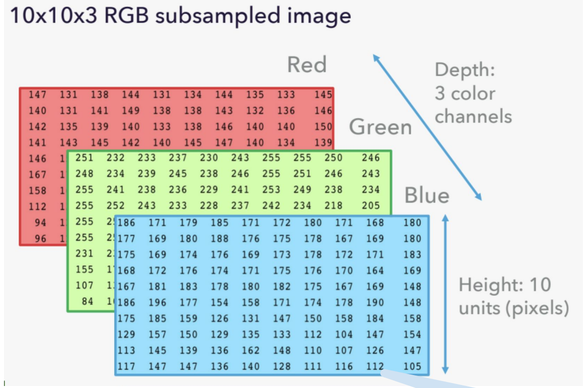
- How about color images?
  3 matrices of numbers,
  corresponding to the Red, Green and Blue "channels" respectively.



Representing Images Digitally

- How about color images?
  3 matrices of numbers,
  corresponding to the Red, Green and Blue “channels” respectively.

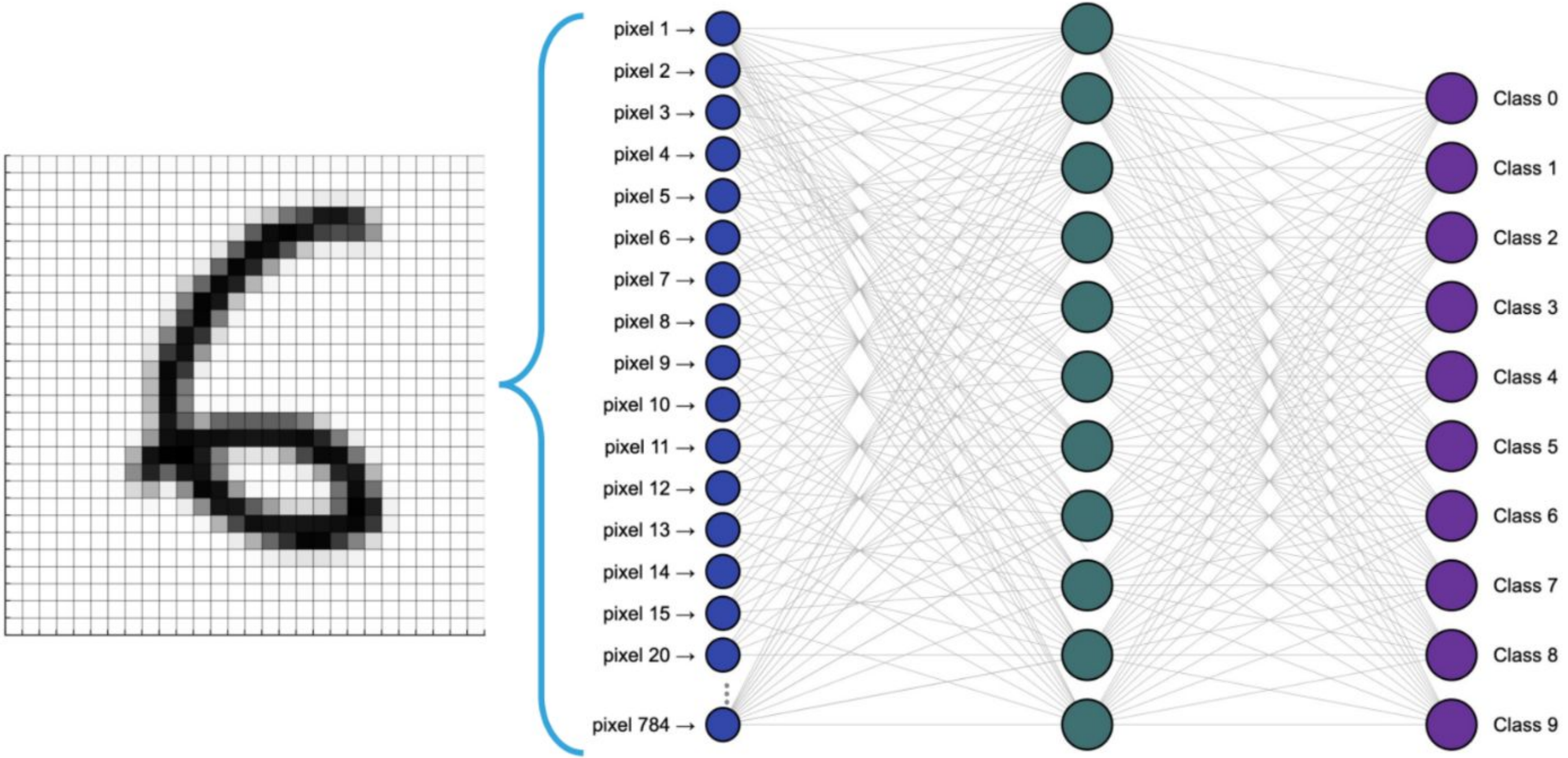
Each pixel of a color image is represented by three intensities (not one.



Each light intensity is still a number between 0 and 255.

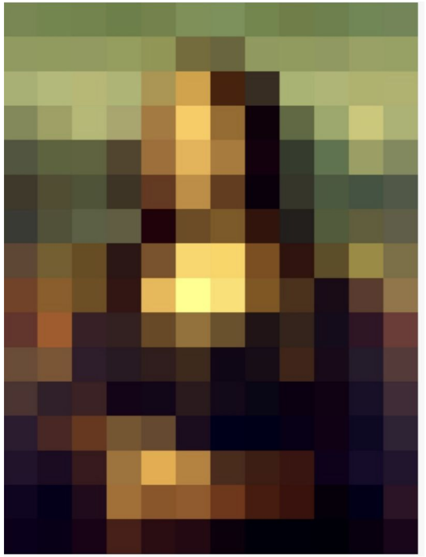
Taking Images as Inputs

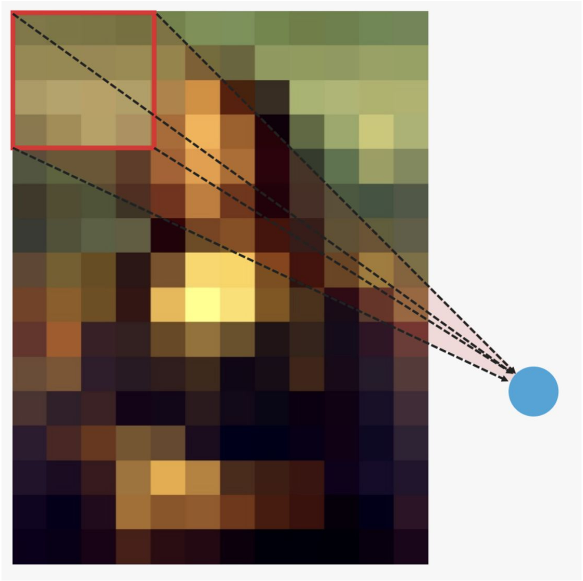
- Flattening the image to feed into a neural network **LOSES** spatial structures.

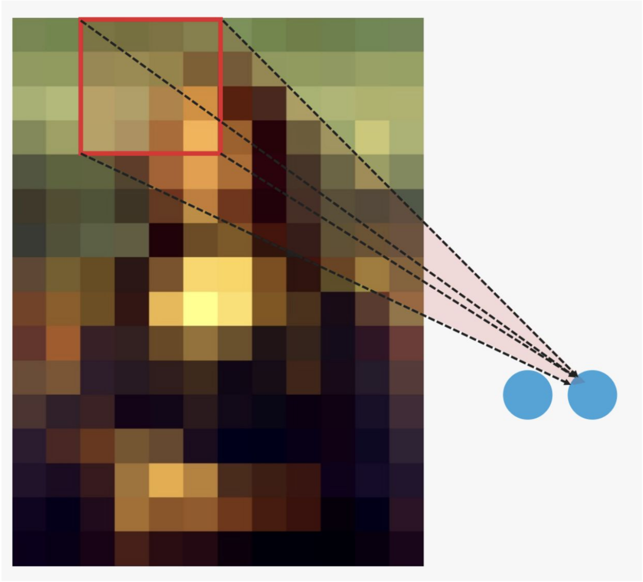


Taking Images as Inputs

- The idea: Let a neuron only connects to a small area
- We use a sliding window to process all pixels successively.

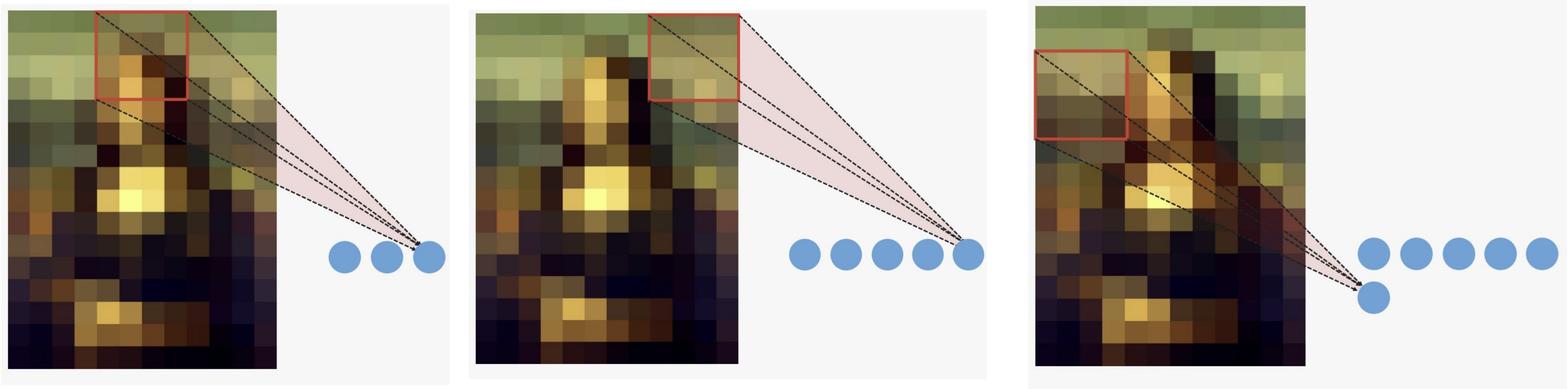






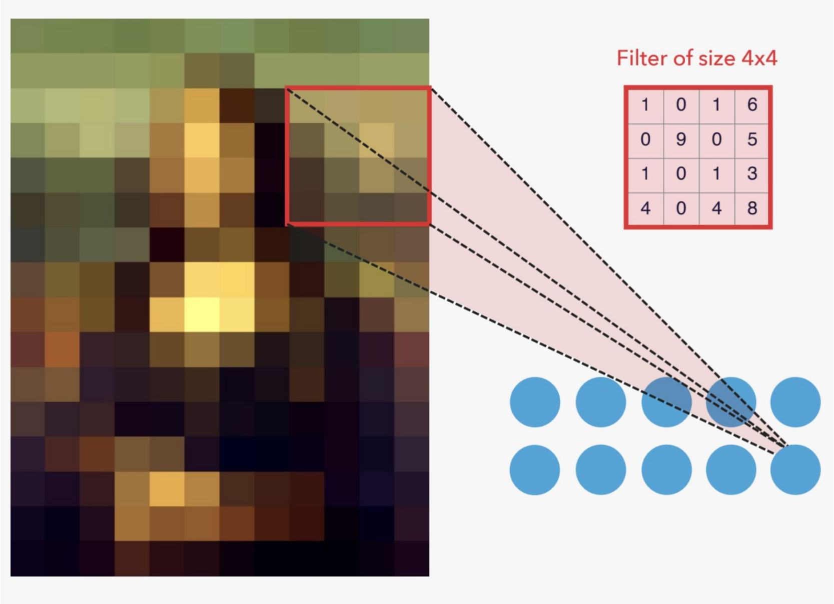
Taking Images as Inputs

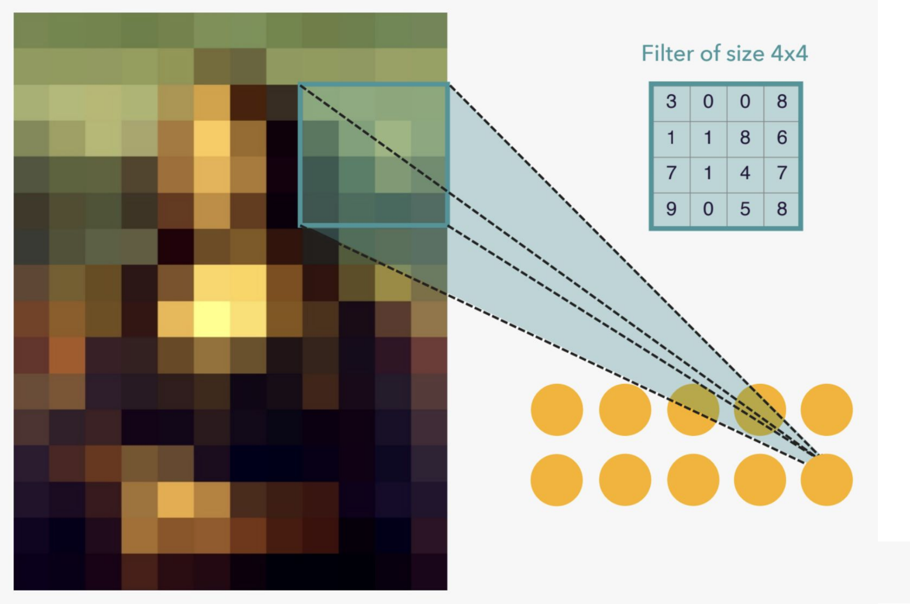
- The idea: Let a neuron only connects to a small area
- We use a sliding window to process all pixels successively.



Convolution

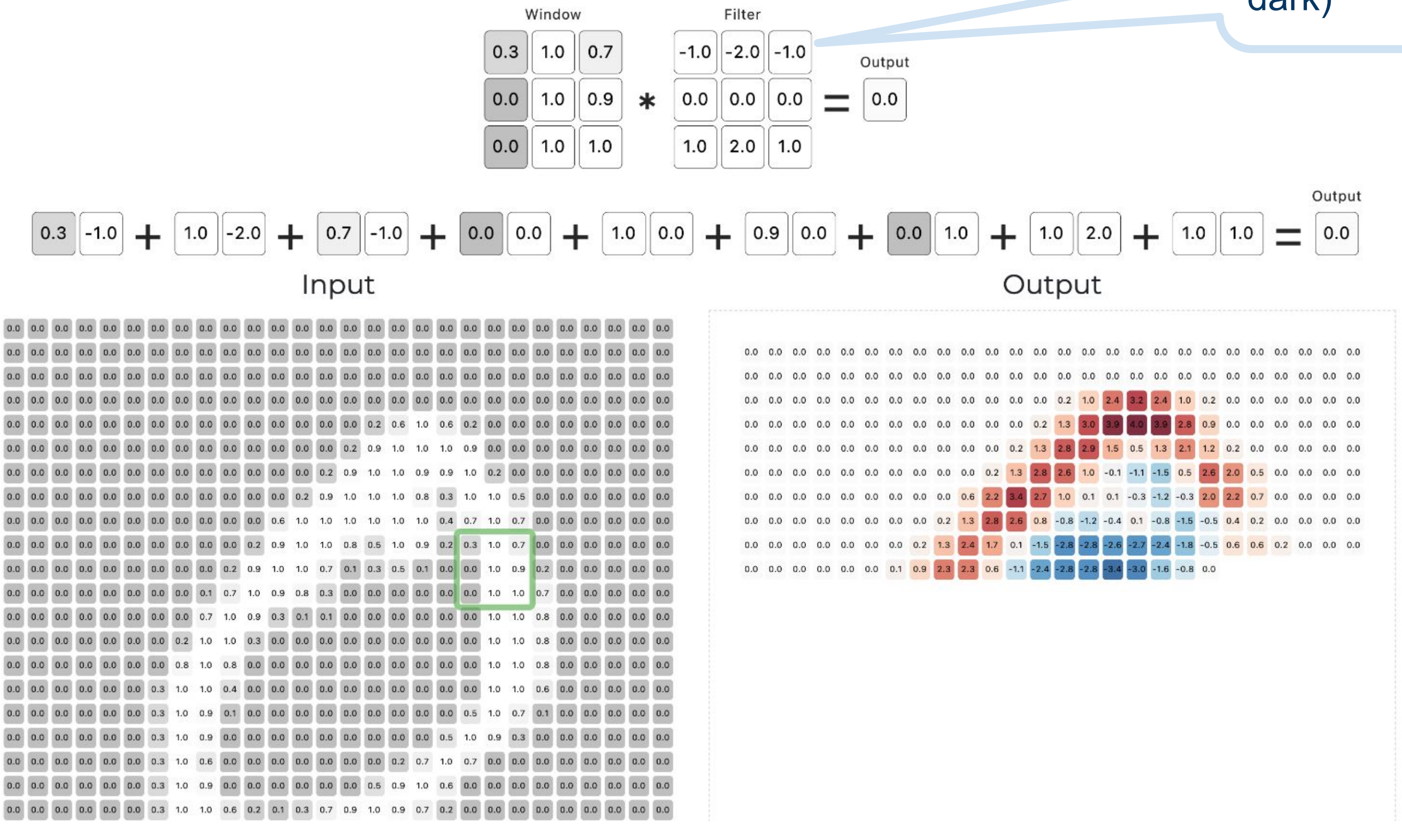
- How can we weigh the values in the patch to detect specific features?
- Apply a set of weights – a filter – to extract local features.
- Spatially share parameters of each filter (features that matter in one part of the input should matter elsewhere).
- Use multiple filters to extract different features.





How a filter works to detect patterns

Detecting top edge.
(colors from light to dark)



A filter is a set of *shared* **weights**.

(How important the pixels to the output.)

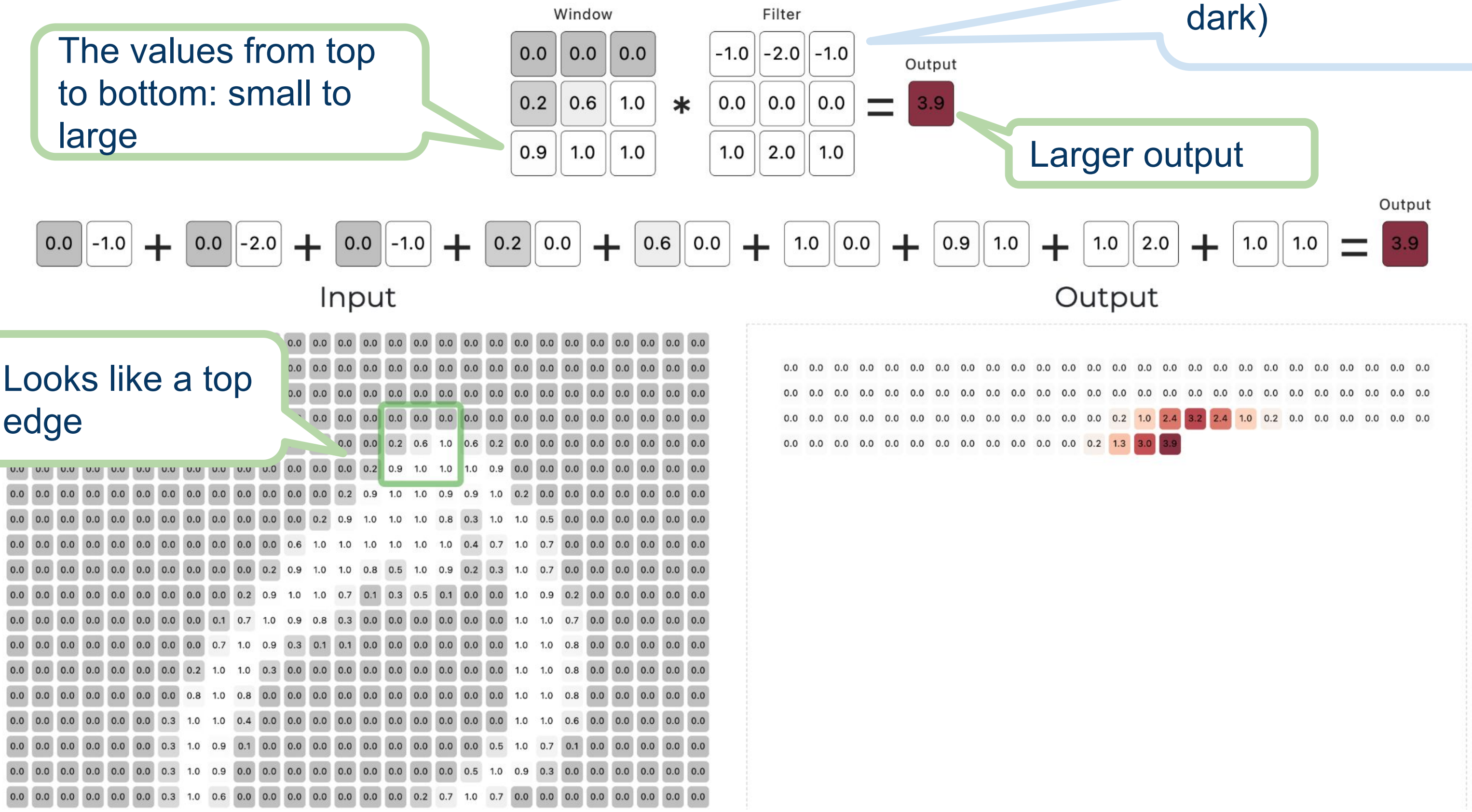
How a filter works to detect patterns

The values from top to bottom: small to large

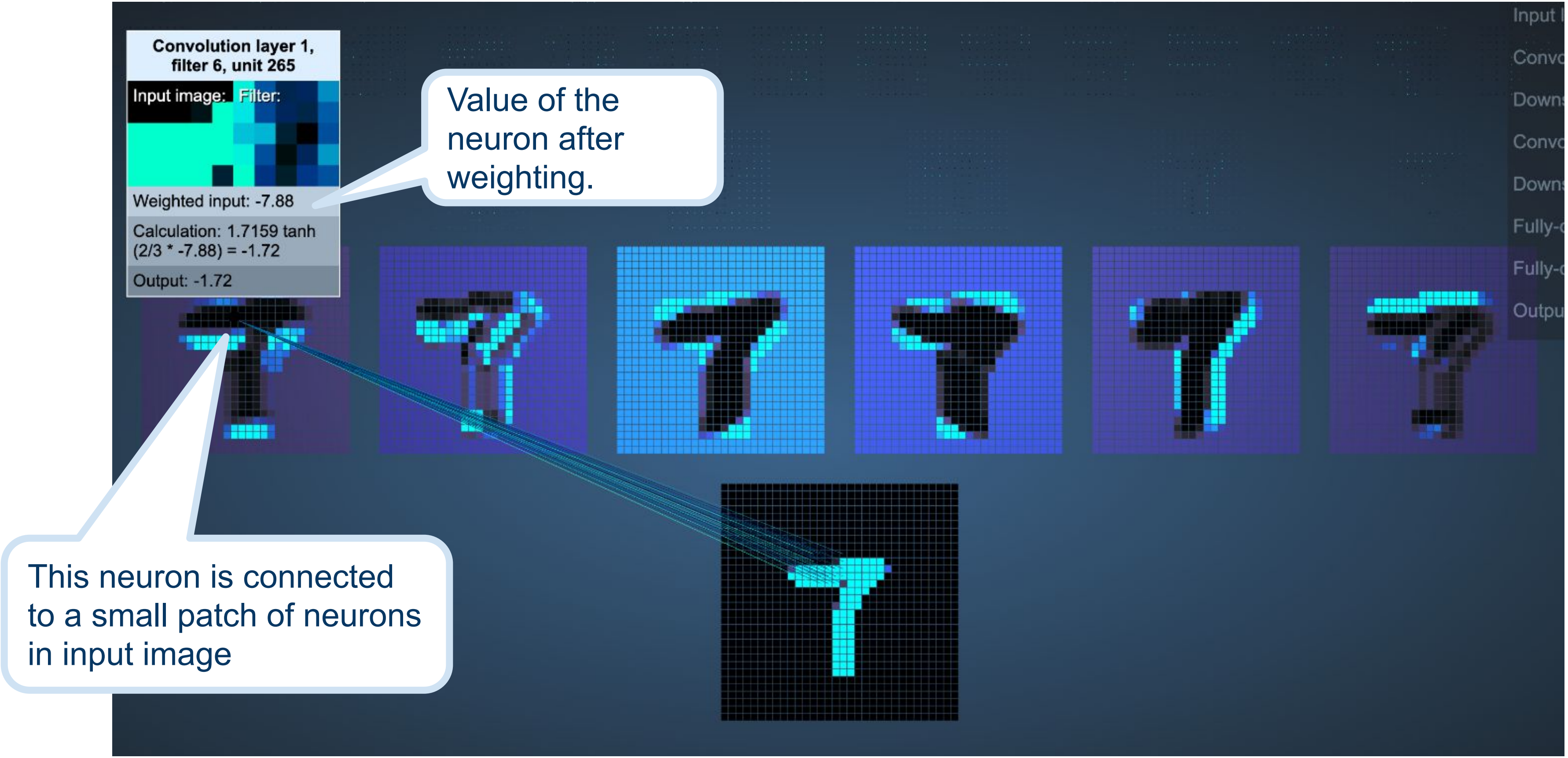
Detecting top edge. (colors from light to dark)

Larger output

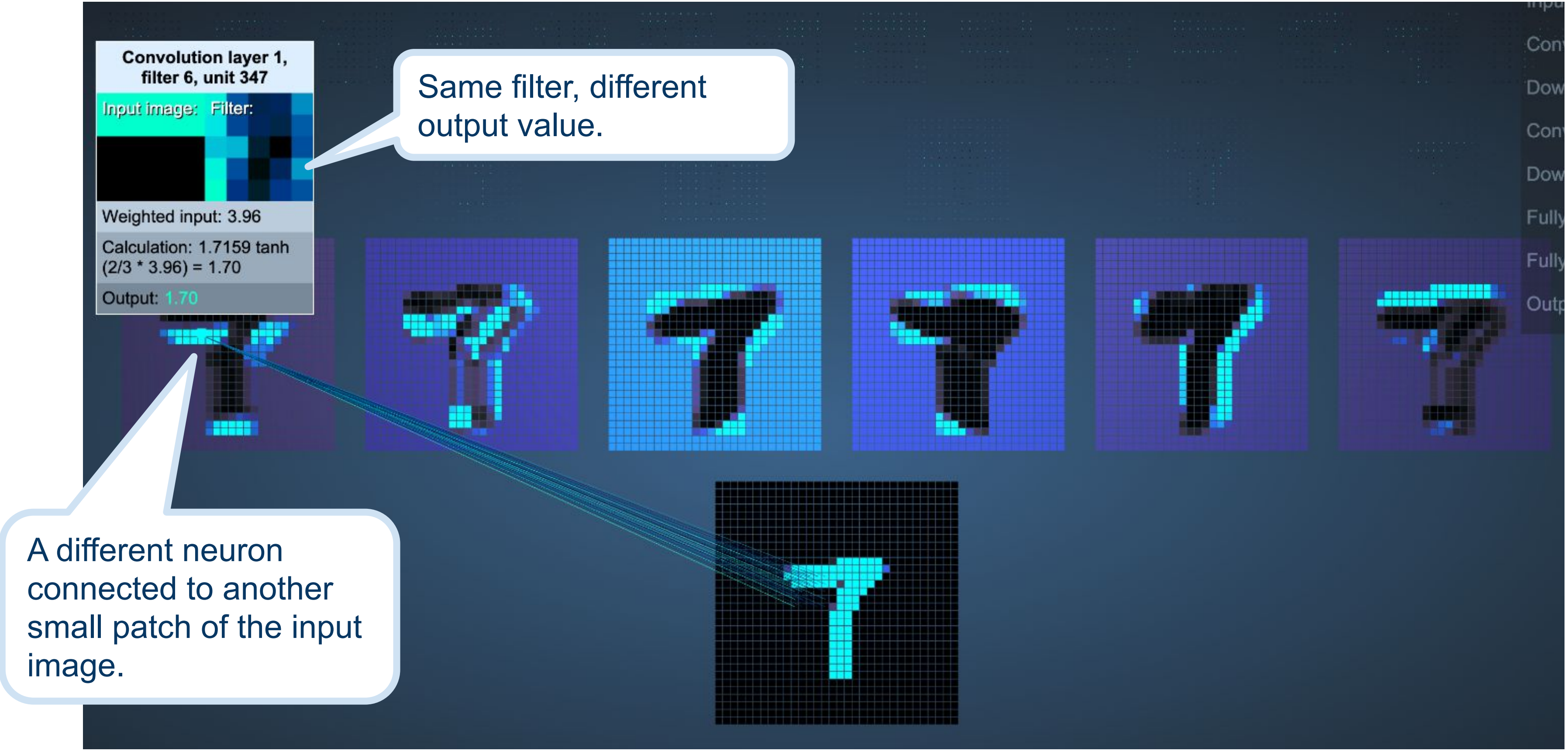
Looks like a top edge



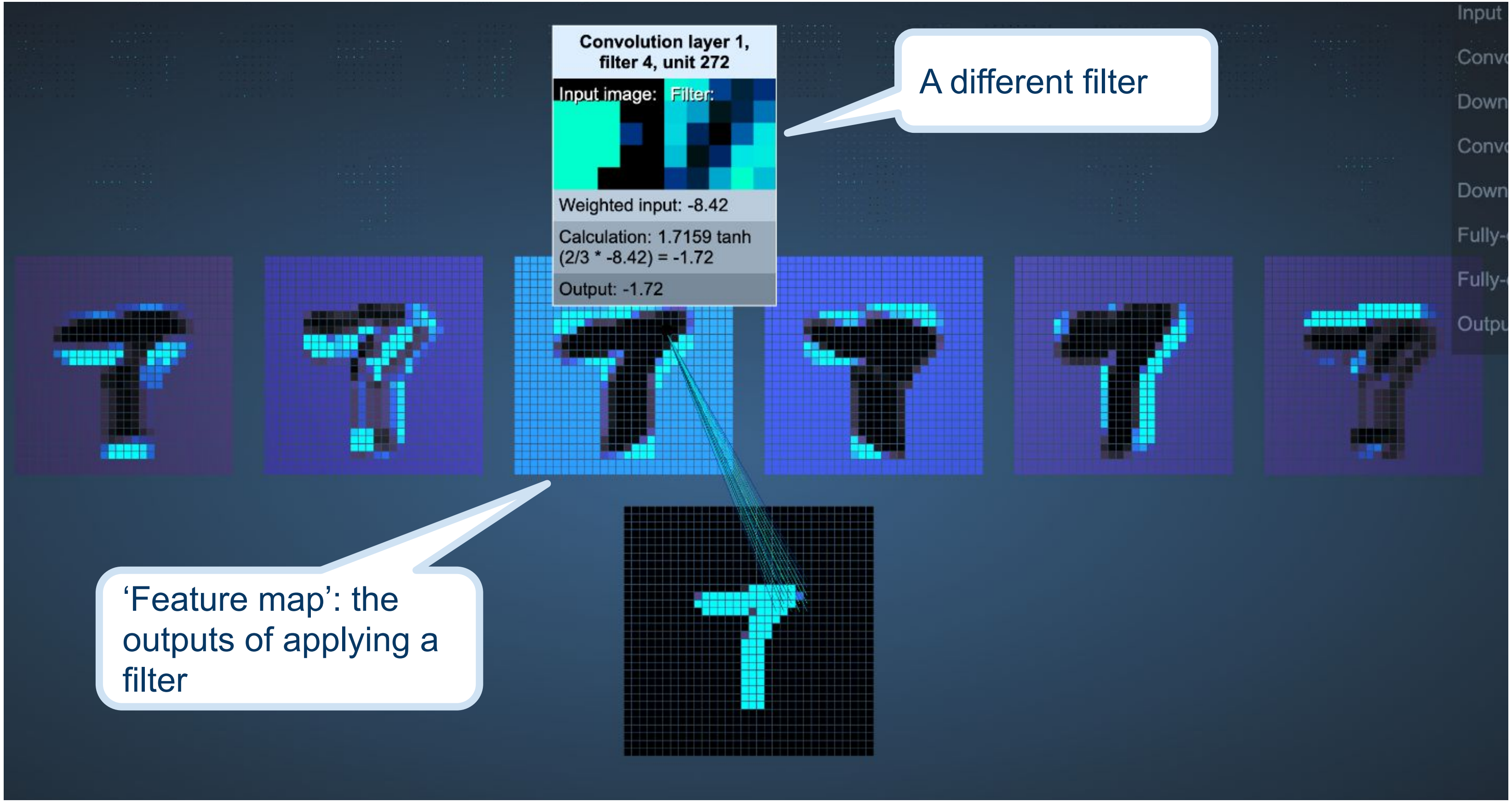
How a convolution layer looks like with multiple filters



How a convolution layer looks like with multiple filters

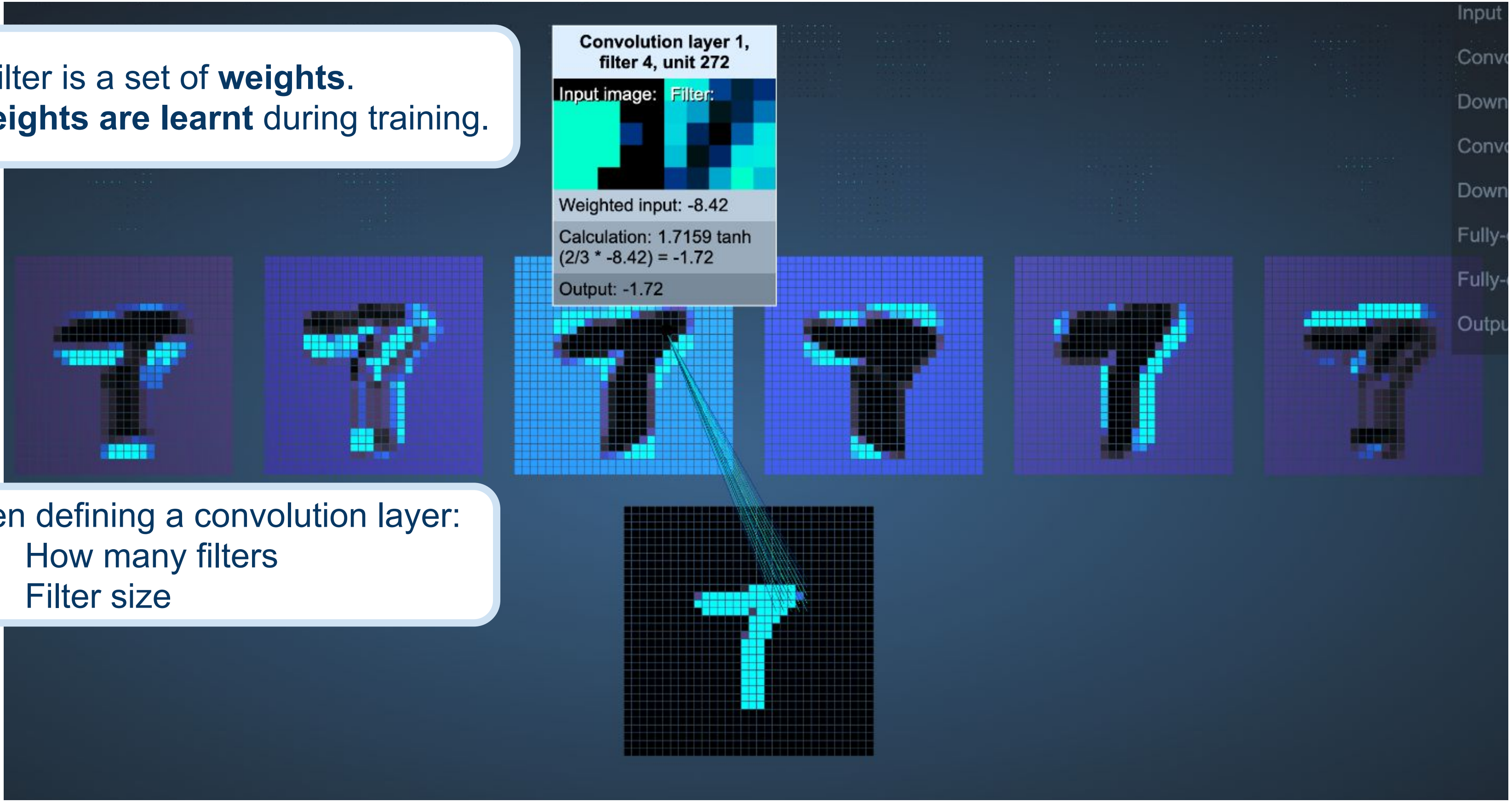


How a convolution layer looks like with multiple filters



How a convolution layer looks like with multiple filters

A filter is a set of **weights**.
Weights are learnt during training.



When defining a convolution layer:

- How many filters
- Filter size

How a convolution layer looks like with multiple filters

A convolutional layer

Input image is 32 by 32, 3 color channels

```
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(32, 32, 3)))
```

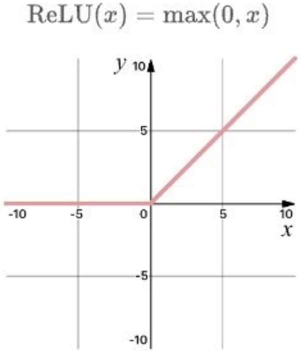
32 filters

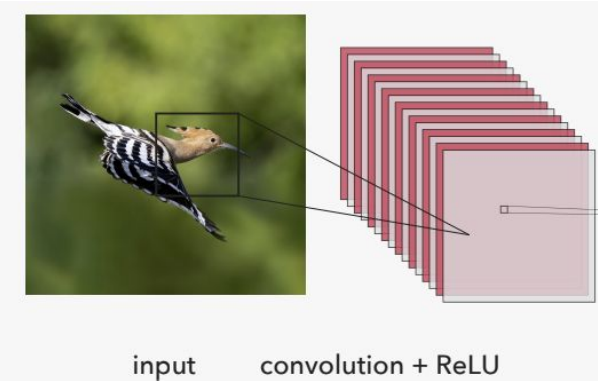
Each is 3 by 3

Activation function

Activation Function

- Add in non-linearity





Visualizing the Convolutions

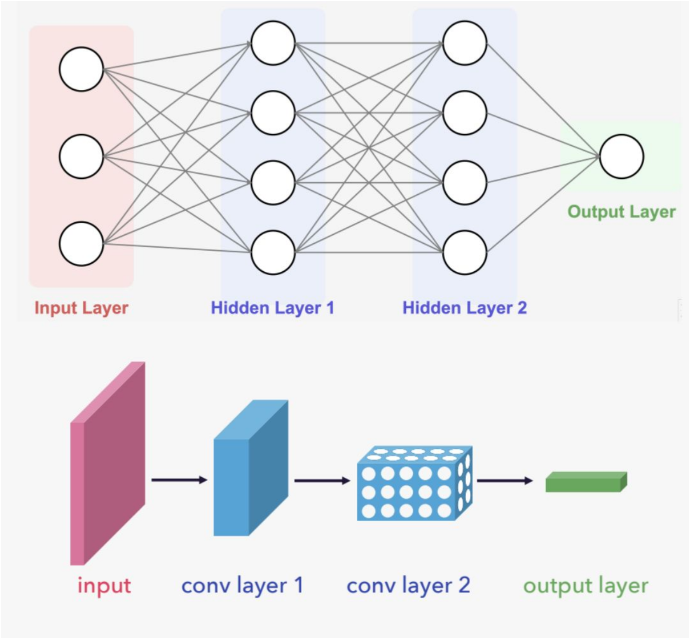
- Filter demo:
  https://deeplizard.com/resource/pavq7noze2

- Multiple filters demo:
  https://adamharley.com/nn_vis/cnn/2d.html

Comparing traditional v.s. Convolutional

- Neuron Connections:
  - Traditional neuron: first hidden layer connected to every pixel of the input image
  - Convolutional filter: **neurons are selectively connected to a small region** of the input image.

- **Filter Application:** Moving the filter across the image is equivalent to **applying a filter to each specific local region**, but crucially, this is done with **shared weights**, promoting feature detection consistency across the image.



Benefits of Convolutional Filters

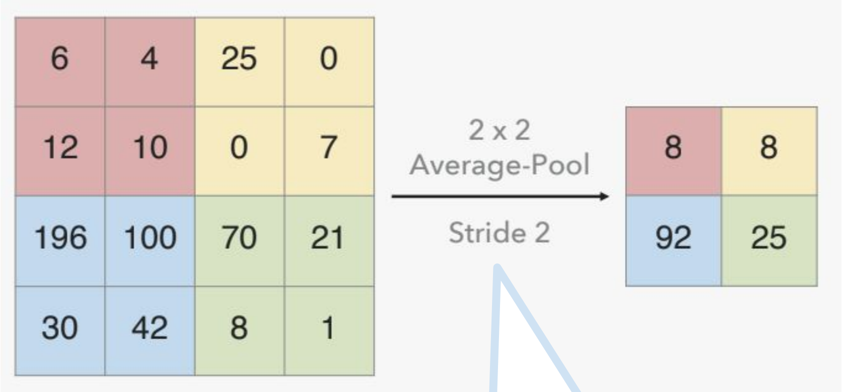
- **Far fewer parameters**, lowering model complexity.

- **Preserve local adjacency** – Maintain the spatial hierarchy of input data, ensuring neighboring pixels remain connected for effective pattern and feature detection.

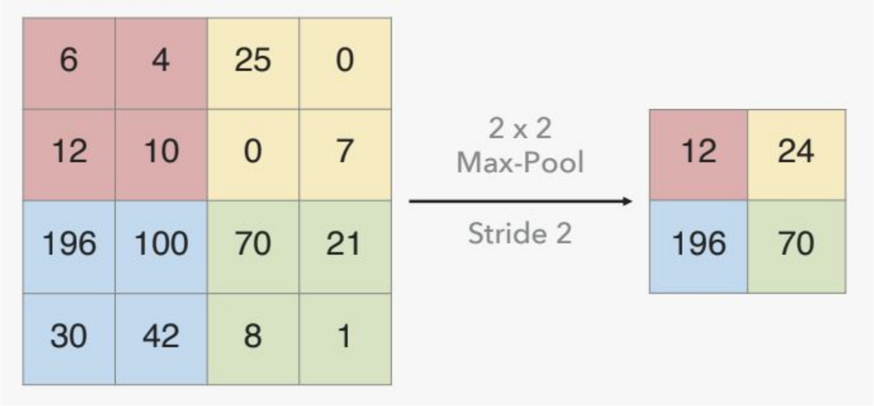
- **Translation invariance** – Learned features can be recognized anywhere in the input image, making models more efficient and robust to variations in object placement.

Pooling Layer

- Downsampling to reduce dimensionality (complexity) while keeping important info
- Basically the pooling layers summarize the features generated by the convolutional layers

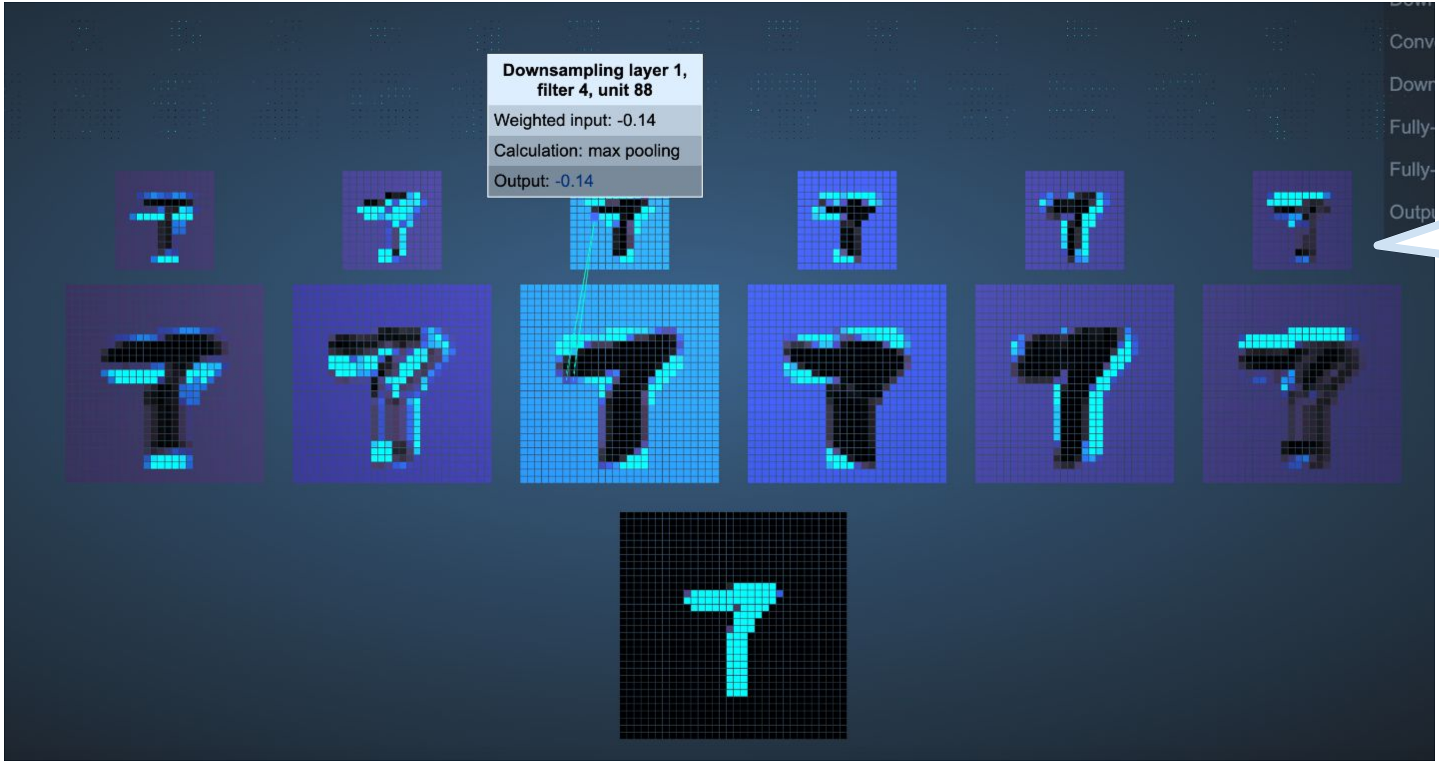


Slide a small window across the feature map



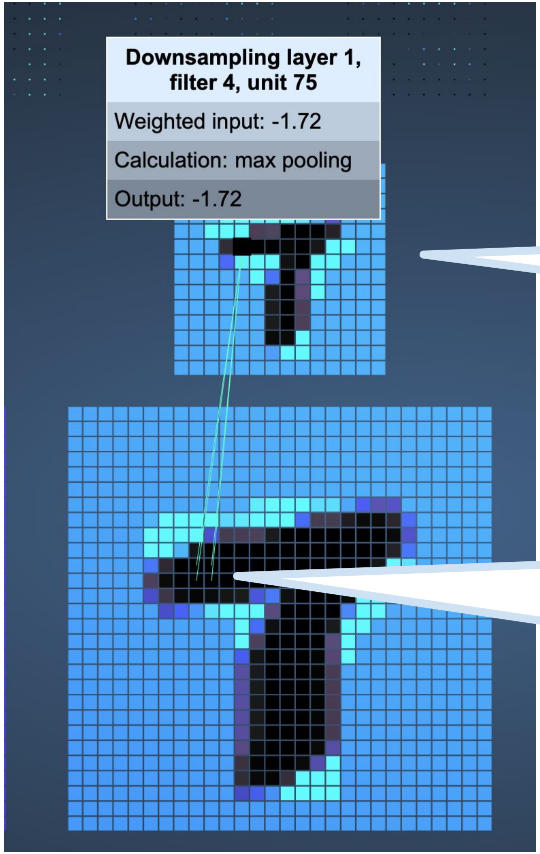
Max pooling keeps the strongest signal. (the most ‘activated’ feature)

Pooling Layer



The pooling layer

Pooling Layer



Now we have a smaller feature map

The pooling layer **doesn't have weights to learn**. It's just taking the max number of a small patch when it's max pooling.

Pooling Layer

```
model.add(layers.MaxPooling2D((2, 2)))
```

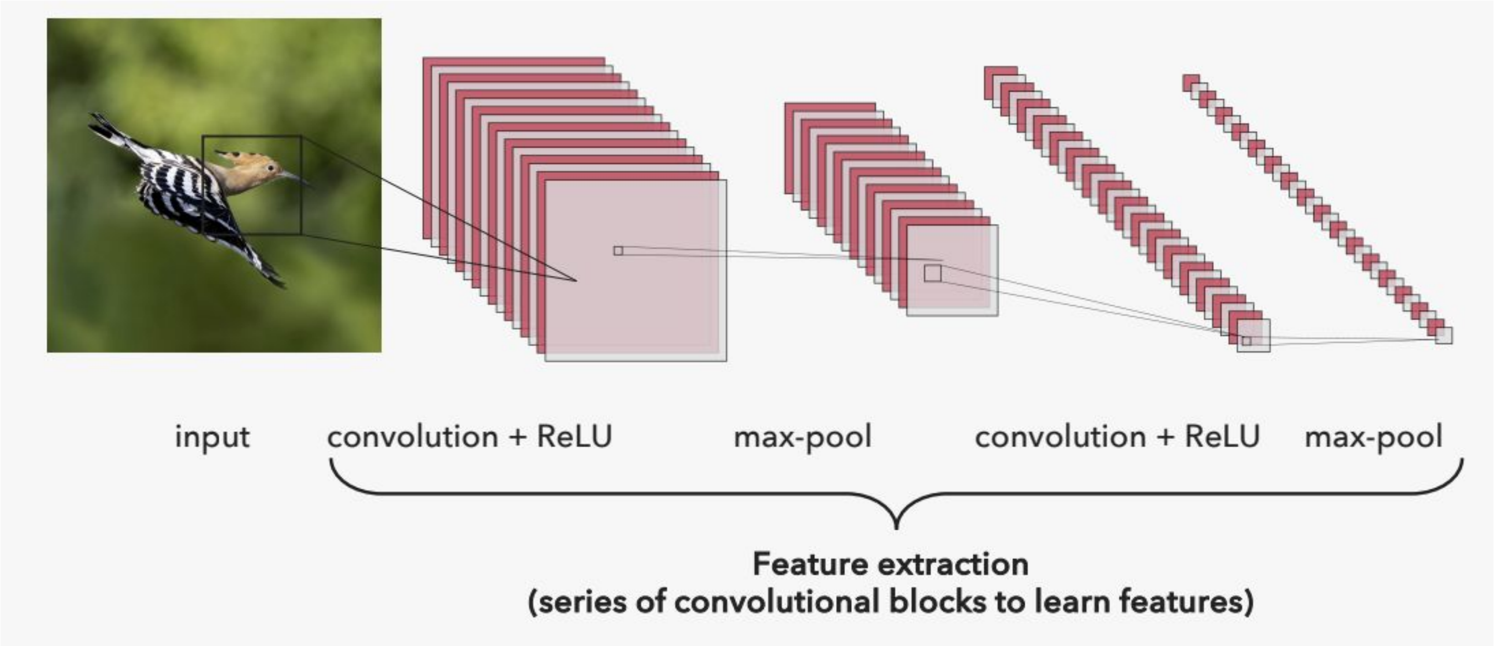
Max pooling layer

Pooling window is 2 pixels tall, 2 pixels wide.

(Looks at 2 X 2 blocks of the feature map at a time)

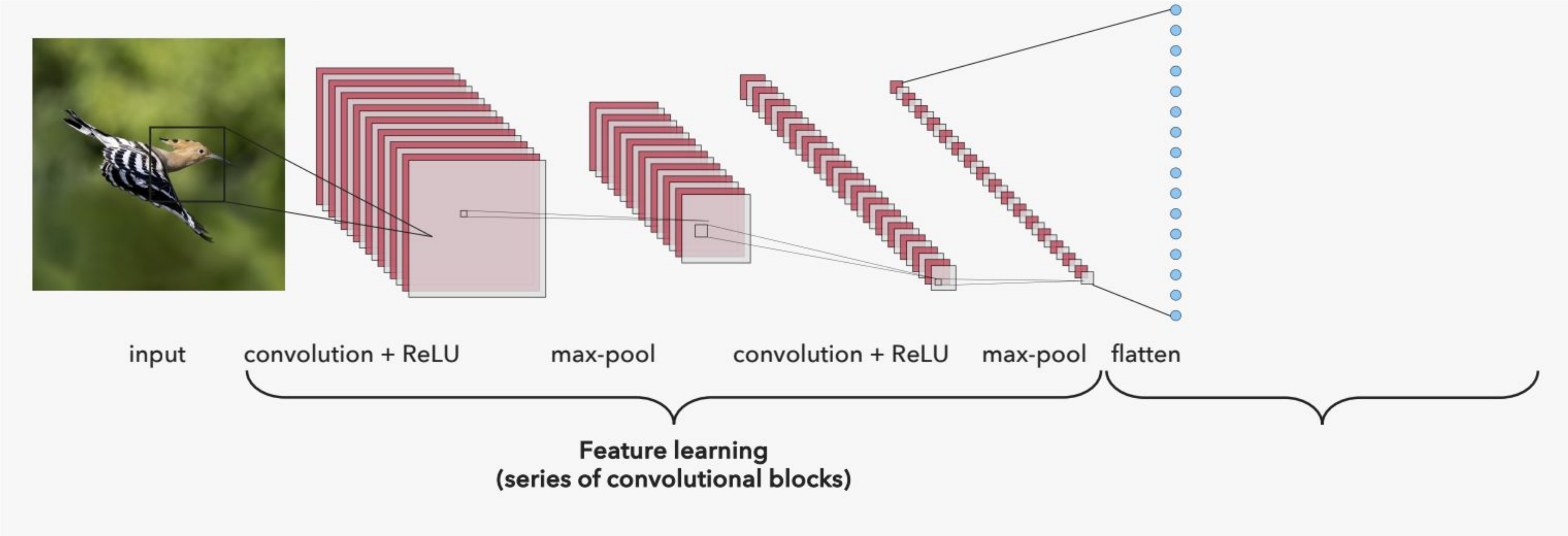
Feature Extraction

- The convolutional blocks output high-level features of the input.



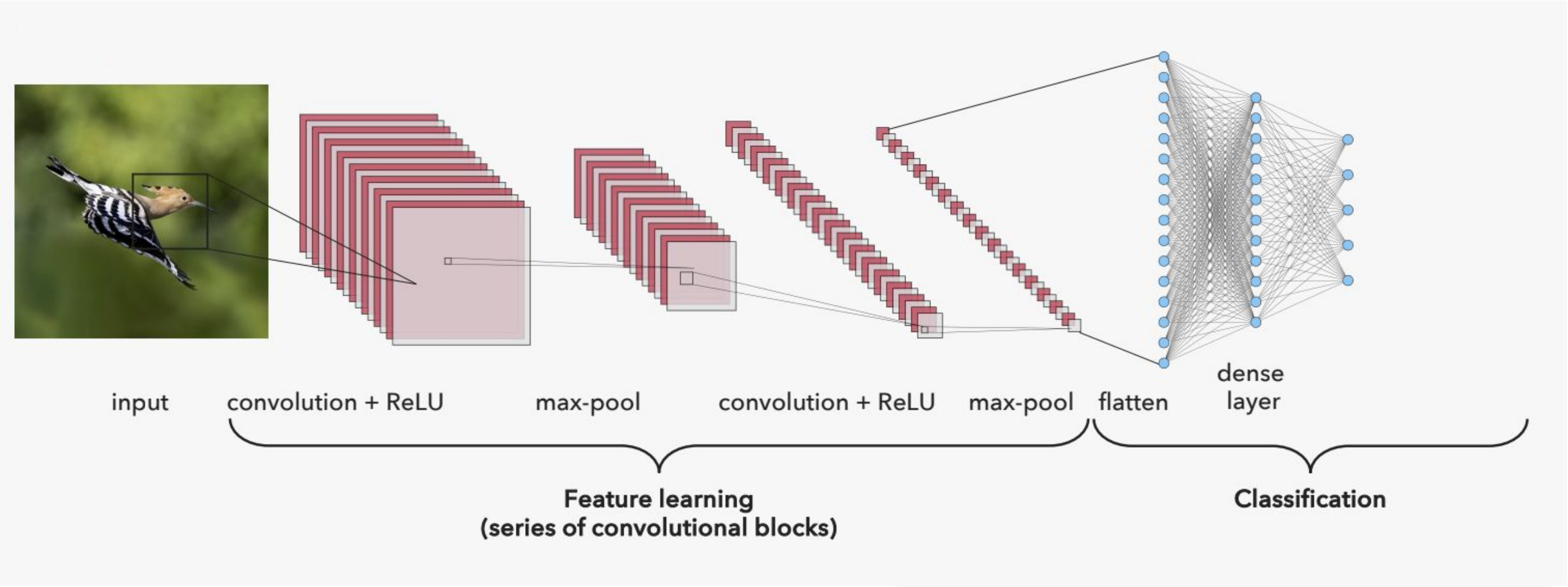
To predict class probabilities

- **Flatten** into one long vector (A flattened layer)



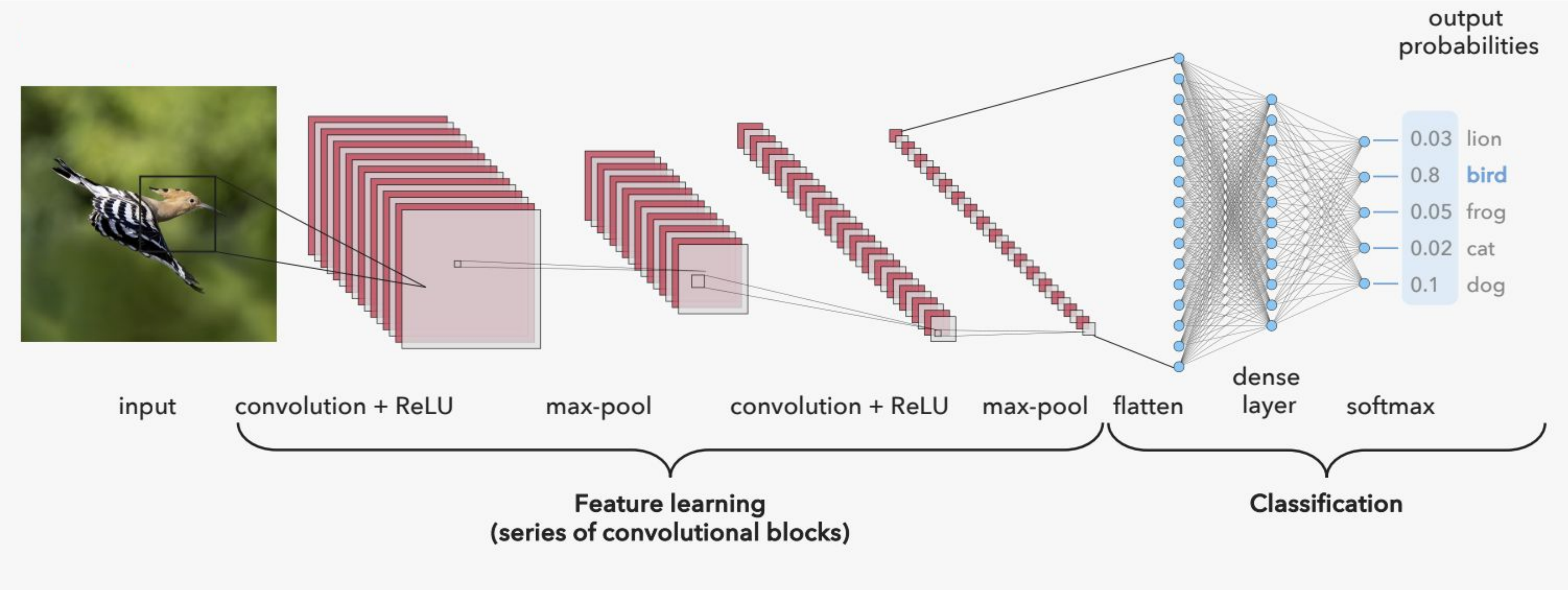
To predict class probabilities

- **Flatten** into one long vector (A flattened layer)
- The **fully connected layers** use these features to classify the input image.

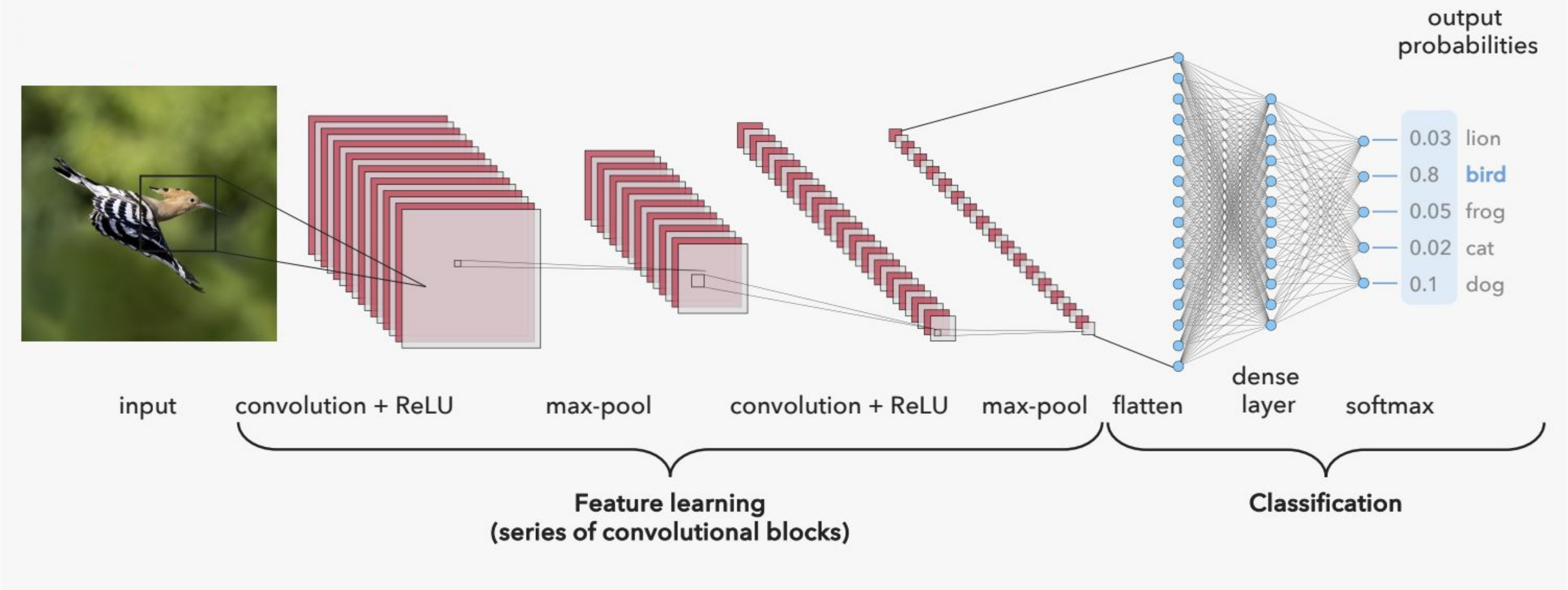


To predict class probabilities

- **Flatten** into one long vector (A flattened layer)
- The **fully connected layers** use these features to classify the input image.
- The **softmax** outputs the probabilities the image belongs to each class.



A Typical CNN



- A few convolutional blocks
- Flattening layer
- A few fully-connected layers
- Output layer

Trained with back propagation and gradient descent

Visualizing the Architecture

- Filter demo: https://deeplizard.com/resource/pavq7noze2
- Multiple filters demo: https://adamharley.com/nn_vis/cnn/2d.html