# Convolutional Neural Network

—

CNNs

COGNIBOT

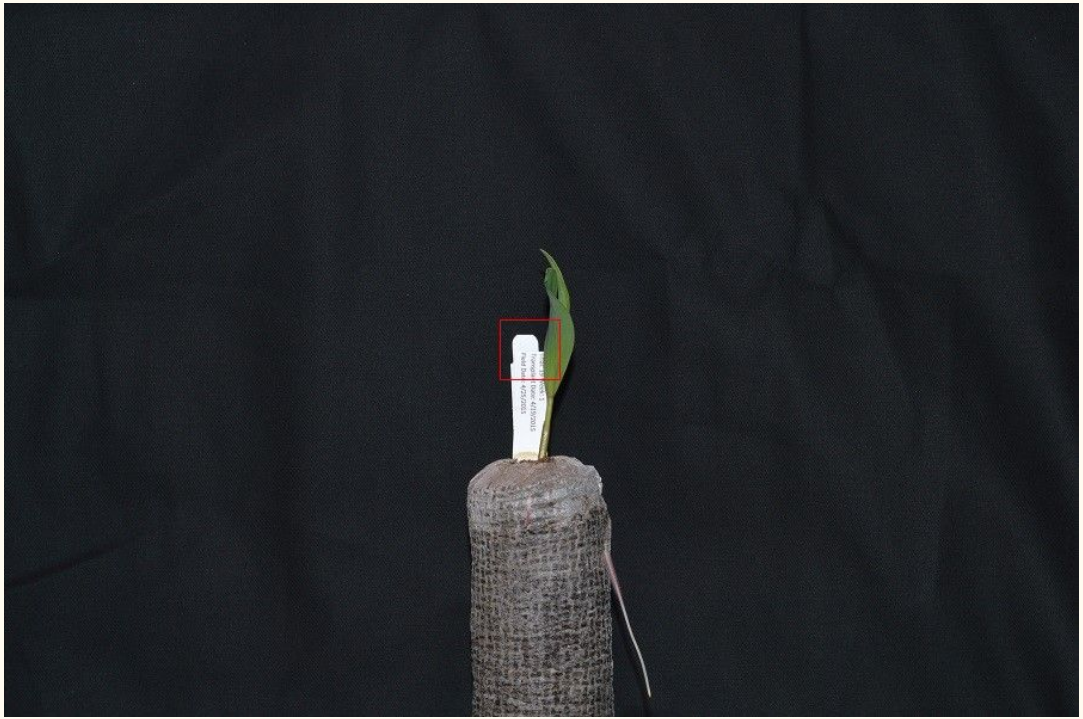
*AI meets Industry*

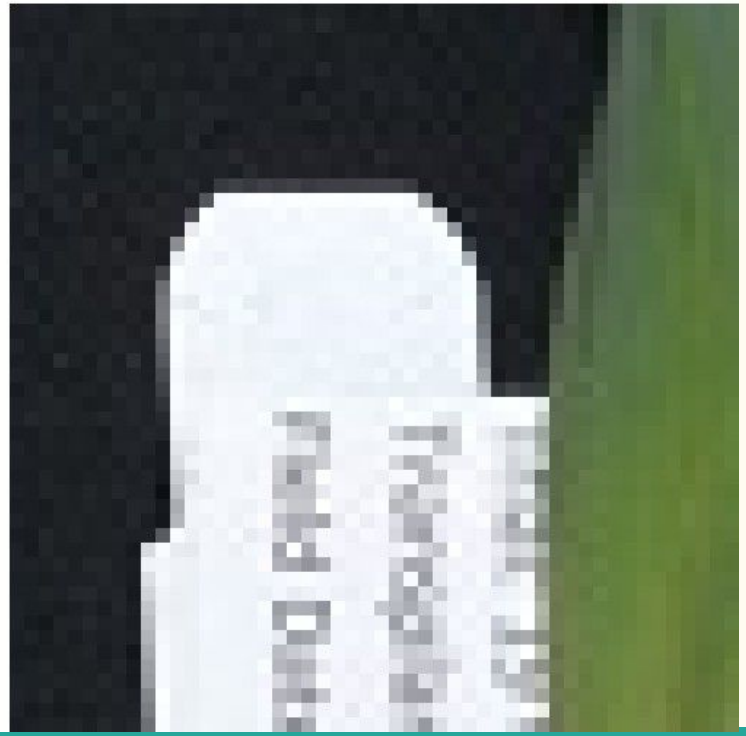
# Digital Images

COGNIBOT

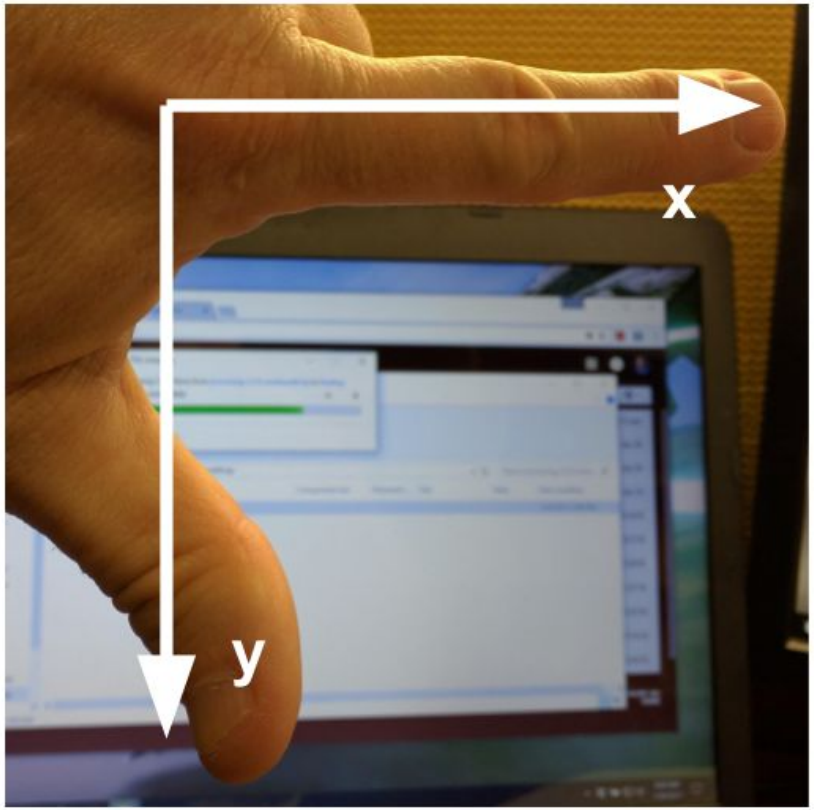
*AI meets Industry*

- Images are stored as rectangular arrays of hundreds, thousands, or millions of discrete “picture elements,” otherwise known as pixels. Each pixel can be thought of as a single square point of colored light.





# Coordinate system in images




| | [0] | [1] | [2] | [3] | [4] |
|---|---|---|---|---|---|
| [0] | 0 | 0 | 0 | 0 | 0 |
| [1] | 0 | 0 | 0 | 0 | 0 |
| [2] | 0 | 0 | 0 | 0 | 0 |
| [3] | 0 | 0 | 0 | 0 | 0 |
| [4] | 0 | 0 | 0 | 0 | 0 |

```
matrix = new int[5][5];
```

# Color system

| Color name | RGB triplet | Color |
|---|---|---|
| Red | (255, 0, 0) | |
| Lime | (0, 255, 0) | |
| Blue | (0, 0, 255) | |
| White | (255, 255, 255) | |
| Black | (0, 0, 0) | |
| Gray | (128, 128, 128) | |
| Fuchsia | (255, 0, 255) | |
| Yellow | (255, 255, 0) | |
| Aqua | (0, 255, 255) | |
| Silver | (192, 192, 192) | |
| Maroon | (128, 0, 0) | |
| Olive | (128, 128, 0) | |
| Green | (0, 128, 0) | |
| Teal | (0, 128, 128) | |
| Navy | (0, 0, 128) | |
| Purple | (128, 0, 128) | |

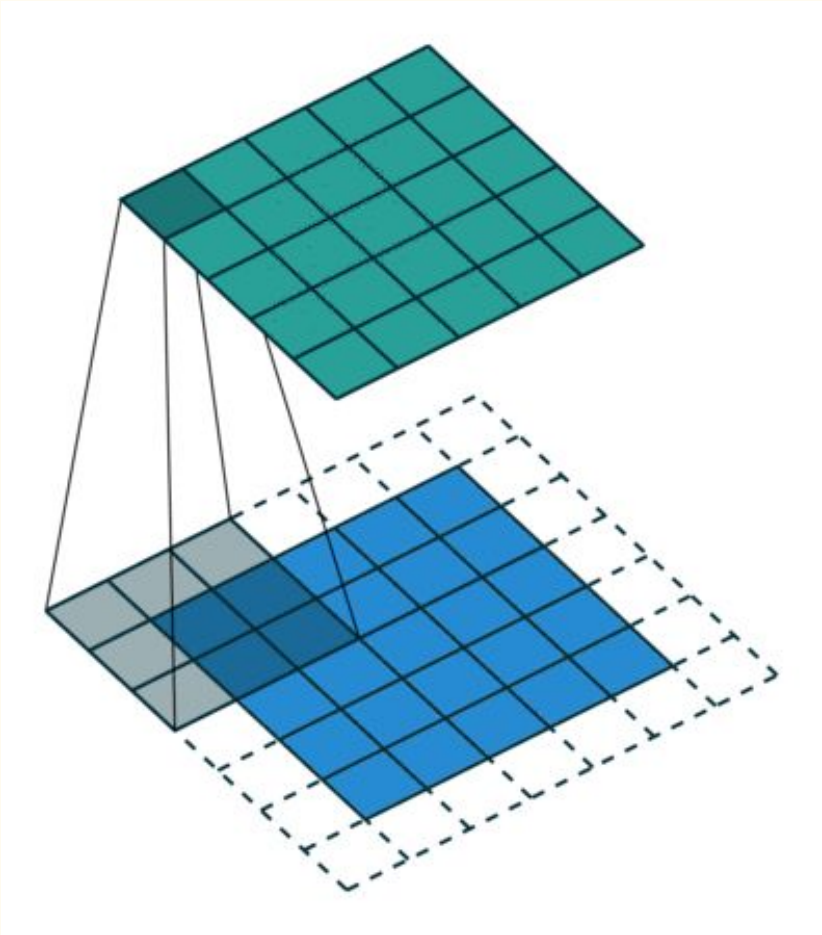
COGNIBOT
AI meets Industry

# Image Convolutions

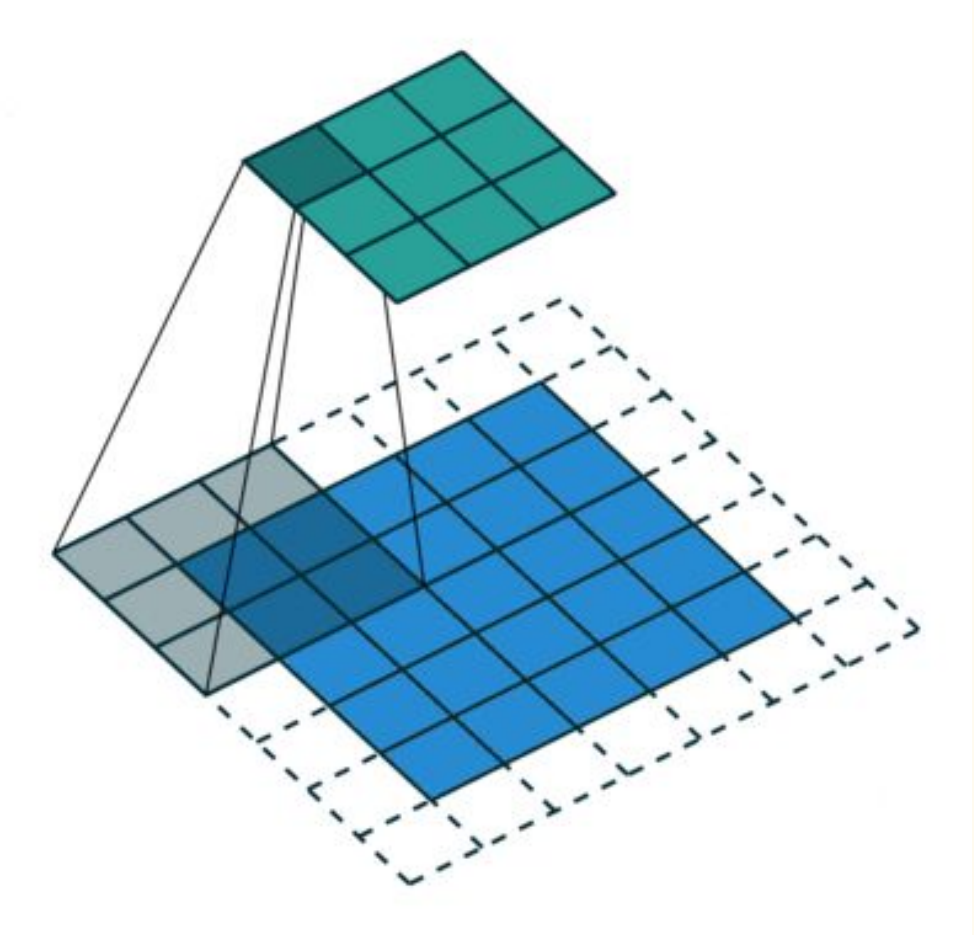
See https://setosa.io/ev/image-kernels/

COGNIBOT

AI meets Industry



Stride length = 1



Stride length = 2

# CNNs

COGNIBOT

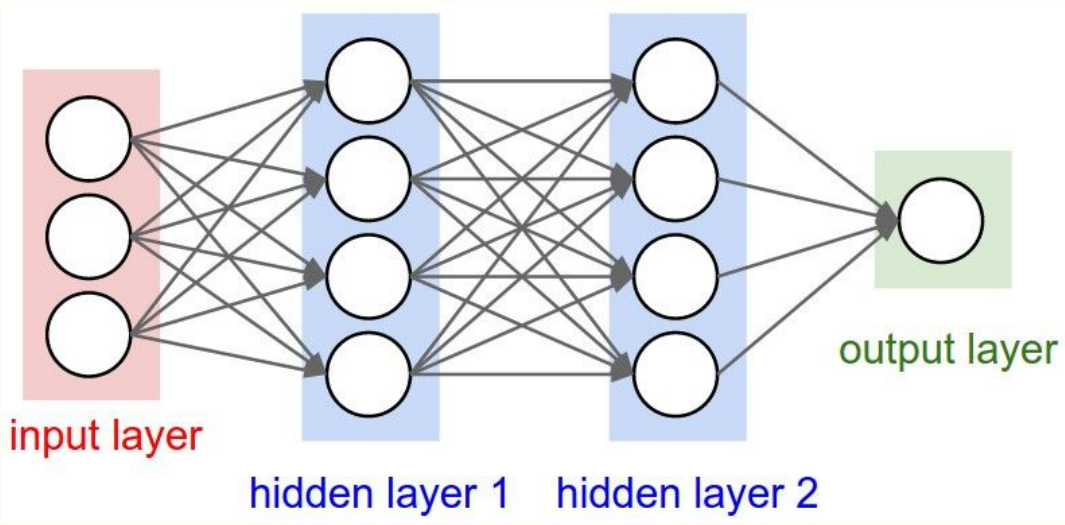
*AI meets Industry*

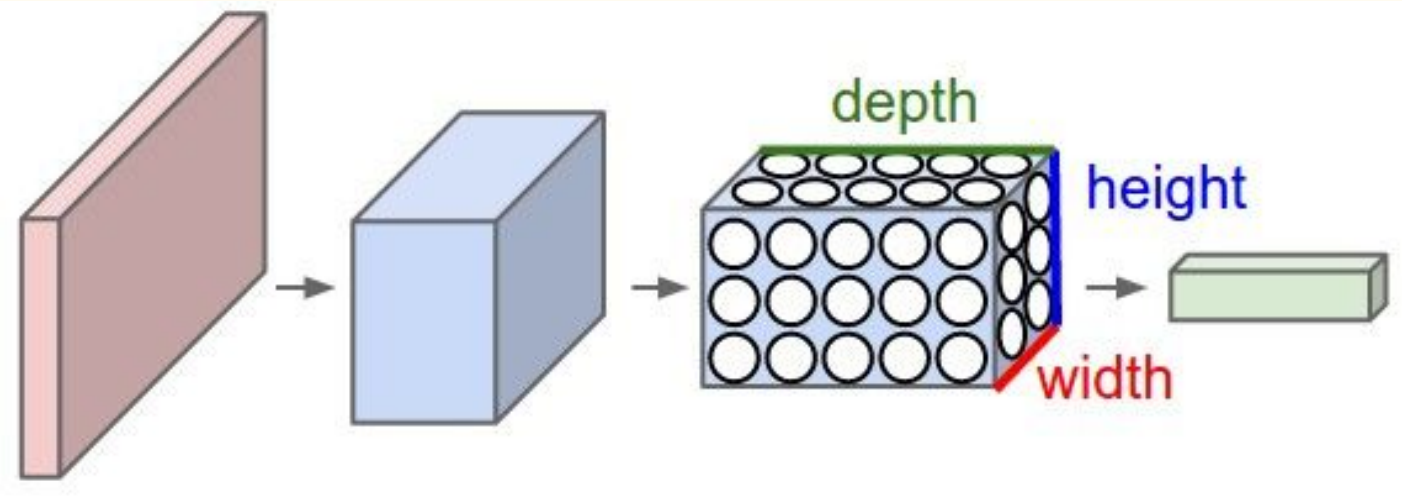
# What are CNN's?

- Convolutional Neural Networks are very similar to ordinary Neural Networks from the previous chapter: they are made up of neurons that have learnable weights and biases.
- The whole network still expresses a single differentiable score function: from the raw image pixels on one end to class scores at the other.
- ConvNet architectures make the explicit assumption that the inputs are images, which allows us to encode certain properties into the architecture.

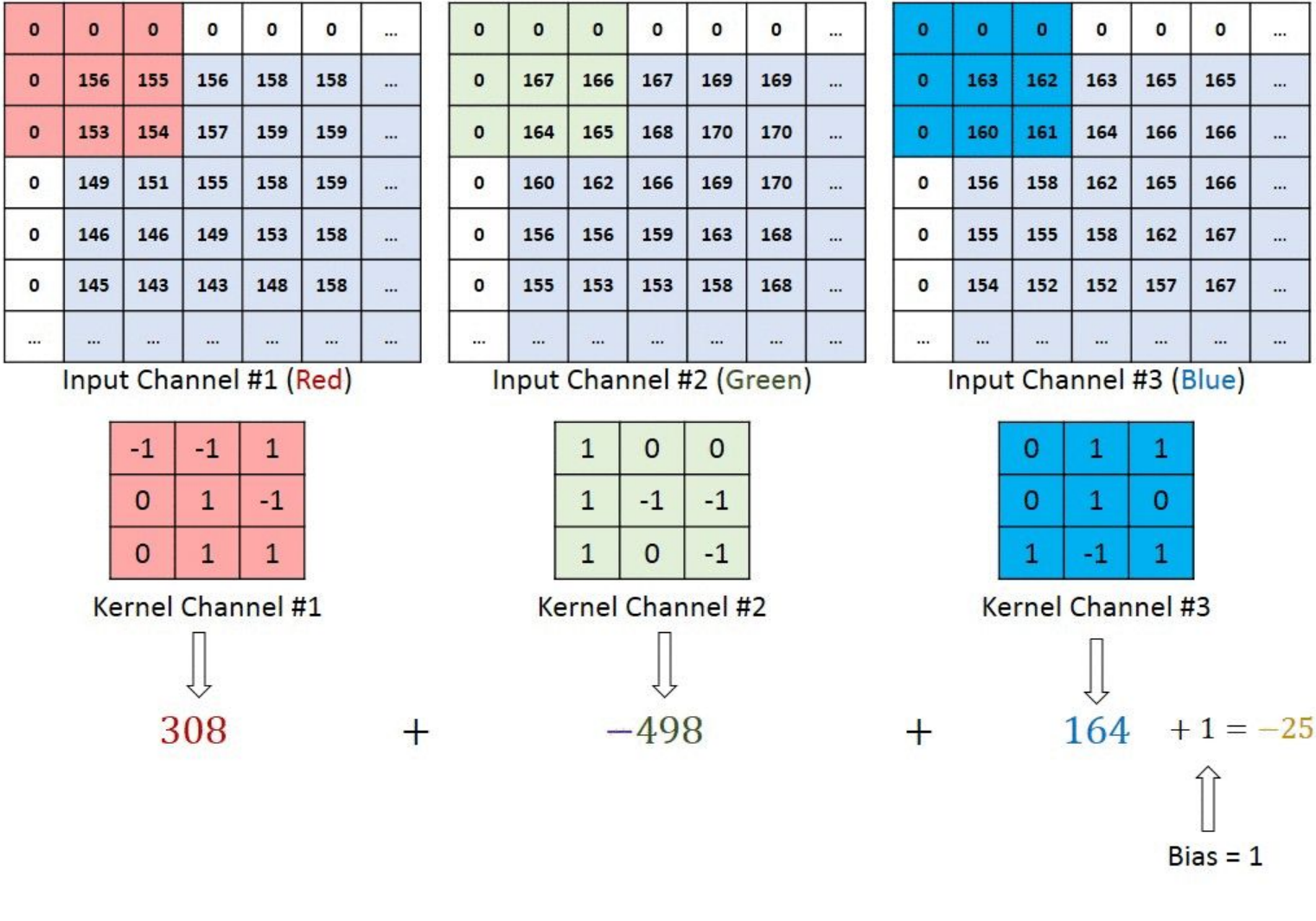
# Why CNN?

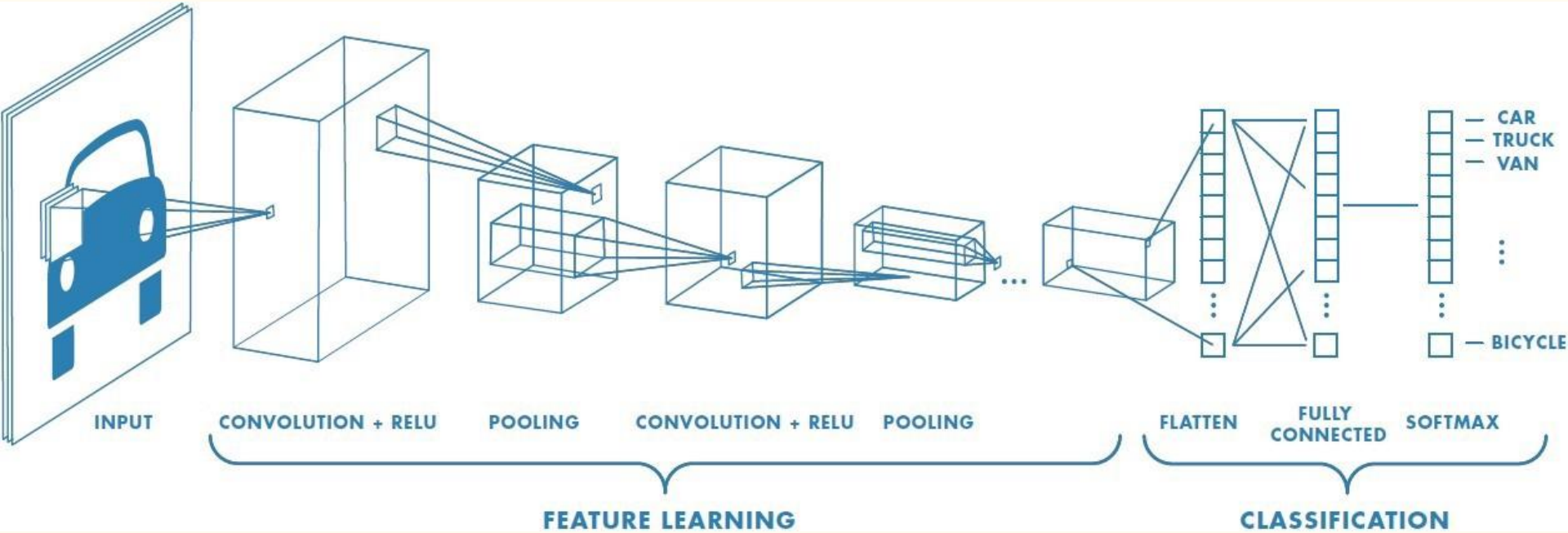
- Regular Neural Nets don't scale well to full images.
- For example, an image of more respectable size, e.g. 200x200x3, would lead to neurons that have 200*200*3 = 120,000 weights.
- Unlike a regular Neural Network, the layers of a ConvNet have neurons arranged in 3 dimensions: width, height, depth.
- The neurons in a layer will only be connected to a small region of the layer before it, instead of all of the neurons in a fully-connected manner.



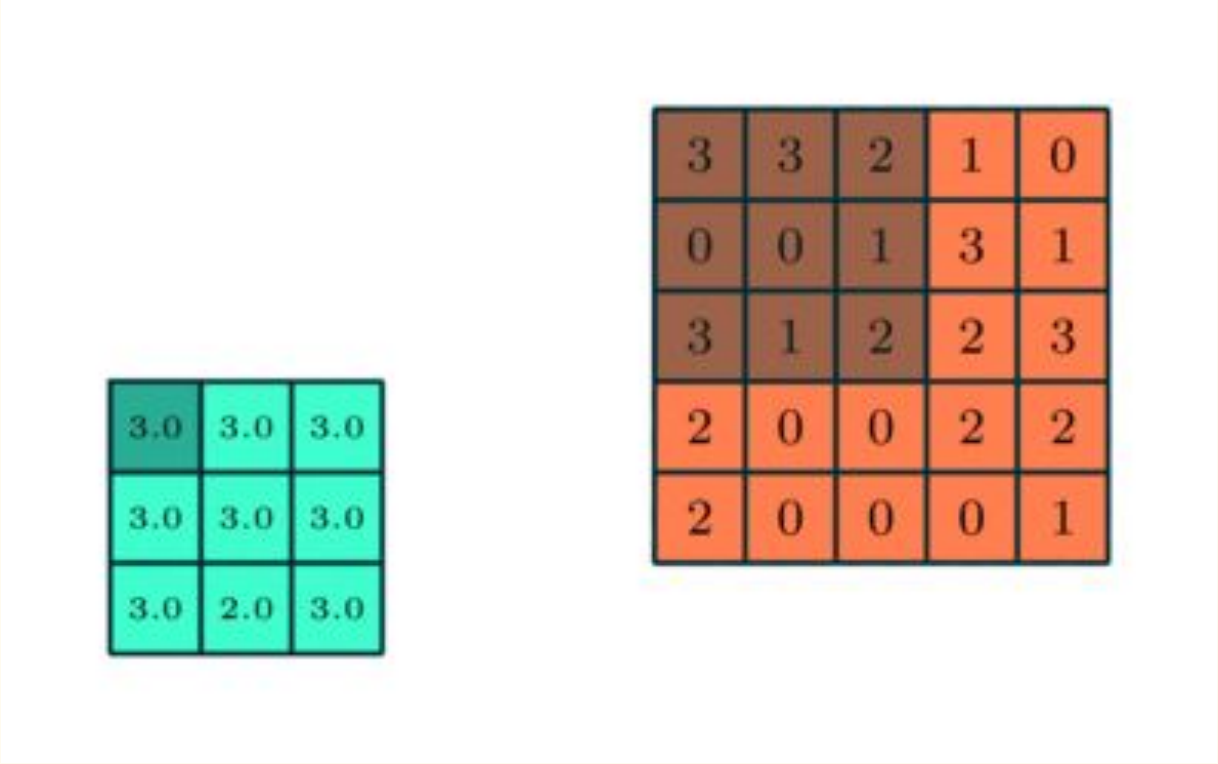

OGNIBOT
AI meets Industry


0 0 0 0 0 0 ...
0 156 155 156 158 158 ...
0 153 154 157 159 159 ...
0 149 151 155 158 159 ...
0 146 146 149 153 158 ...
0 145 143 143 148 158 ...
... ... ... ... ... ... ...
Input Channel #1 (Red)
0 0 0 0 0 0 ...
0 167 166 167 169 169 ...
0 164 165 168 170 170 ...
0 160 162 166 169 170 ...
0 156 156 159 163 168 ...
0 155 153 153 158 168 ...
... ... ... ... ... ... ...
Input Channel #2 (Green)
0 0 0 0 0 0 ...
0 163 162 163 165 165 ...
0 160 161 164 166 166 ...
0 156 158 162 165 166 ...
0 155 155 158 162 167 ...
0 154 152 152 157 167 ...
... ... ... ... ... ... ...
Input Channel #3 (Blue)
-1 -1 1
0 1 -1
0 1 1
Kernel Channel #1
1 0 0
1 -1 -1
1 0 -1
Kernel Channel #2
0 1 1
0 1 0
1 -1 1
Kernel Channel #3
308
+
−498
+
164
+ 1 = −25
Bias = 1
Output
-25 ... ... ... ...
... ... ... ... ...
... ... ... ... ...
... ... ... ... ...
... ... ... ... ...

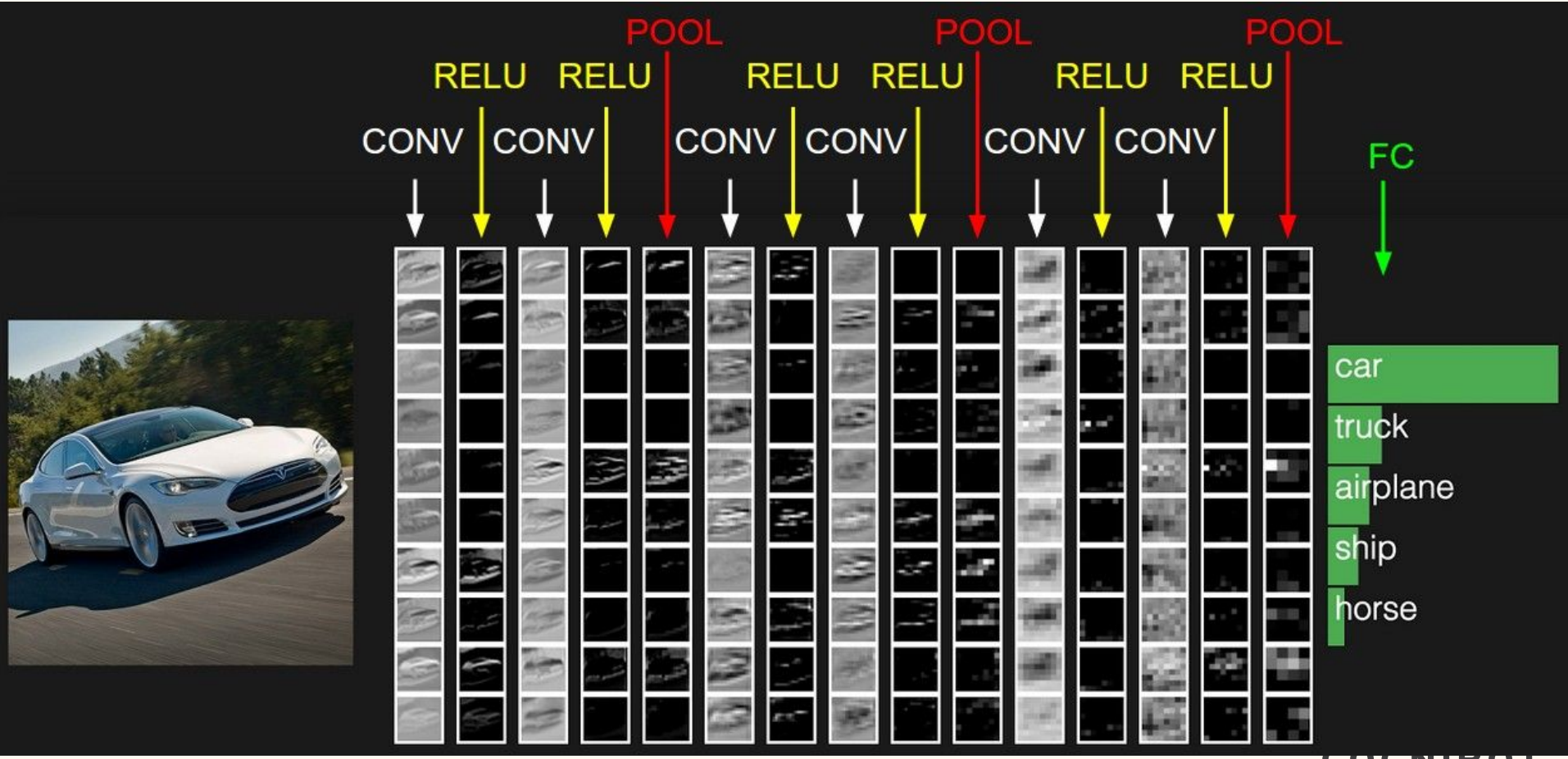



COGNIBOT
*AI meets Industry*

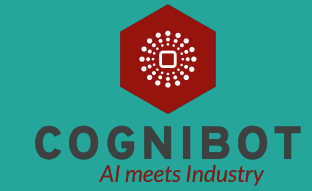
# Pooling layer




COGNIBOT
AI meets Industry


POOL
POOL
POOL
RELU
RELU
RELU
RELU
RELU
RELU
CONV
CONV
CONV
CONV
CONV
CONV
FC
car
truck
airplane
ship
horse

# See Kaggle Notebook



# See

https://cs231n.github.io/

https://keras.io/examples/ (example in class)

COGNIBOT

*AI meets Industry*