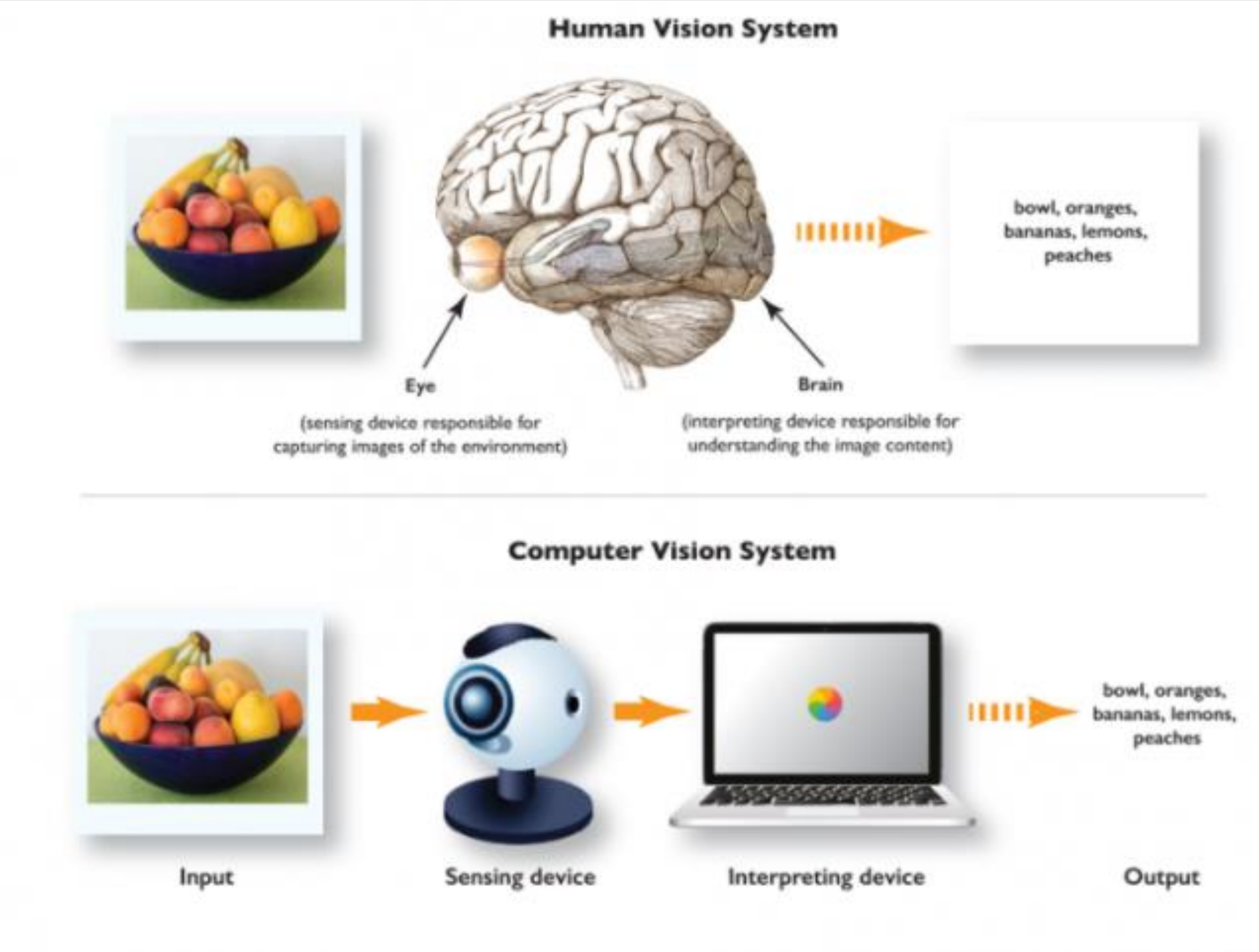
MODULE 6: INTRODUCTION TO DL FOR COMPUTER VISION

BA713 - Machine Learning & AI

CONTENTS

COMPUTER VISION

- Field of Computer Science
- Create digital systems that can process, analyze and gain insights from visual data
- Similar manner as human brain
- Teach a computer to process an image at pixel value and understand it
- **Object classification**
- **Object identification**
- **Object tracking**

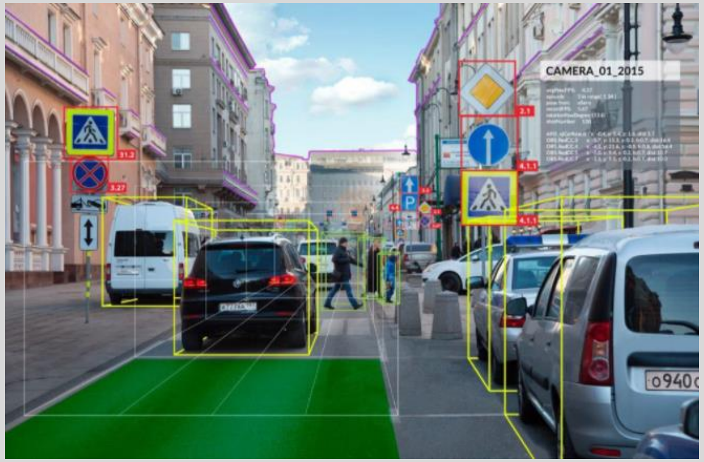


Source: https://xd.adobe.com/ideas/principles/emerging-technology/what-is-computer-vision-how-does-it-work/

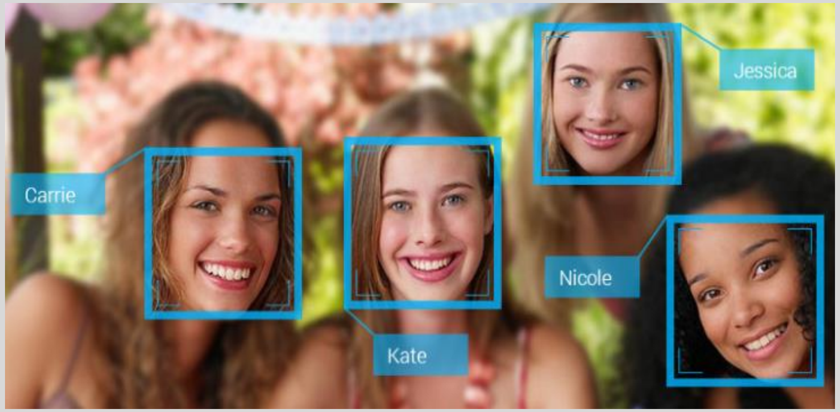
SOME APPLICATIONS OF DL IN COMPUTER VISION

Robotic Applications

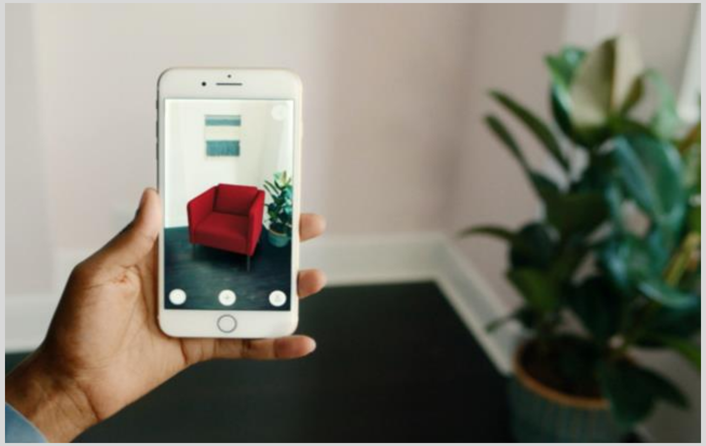
Healthcare Applications



Self Driving Cars



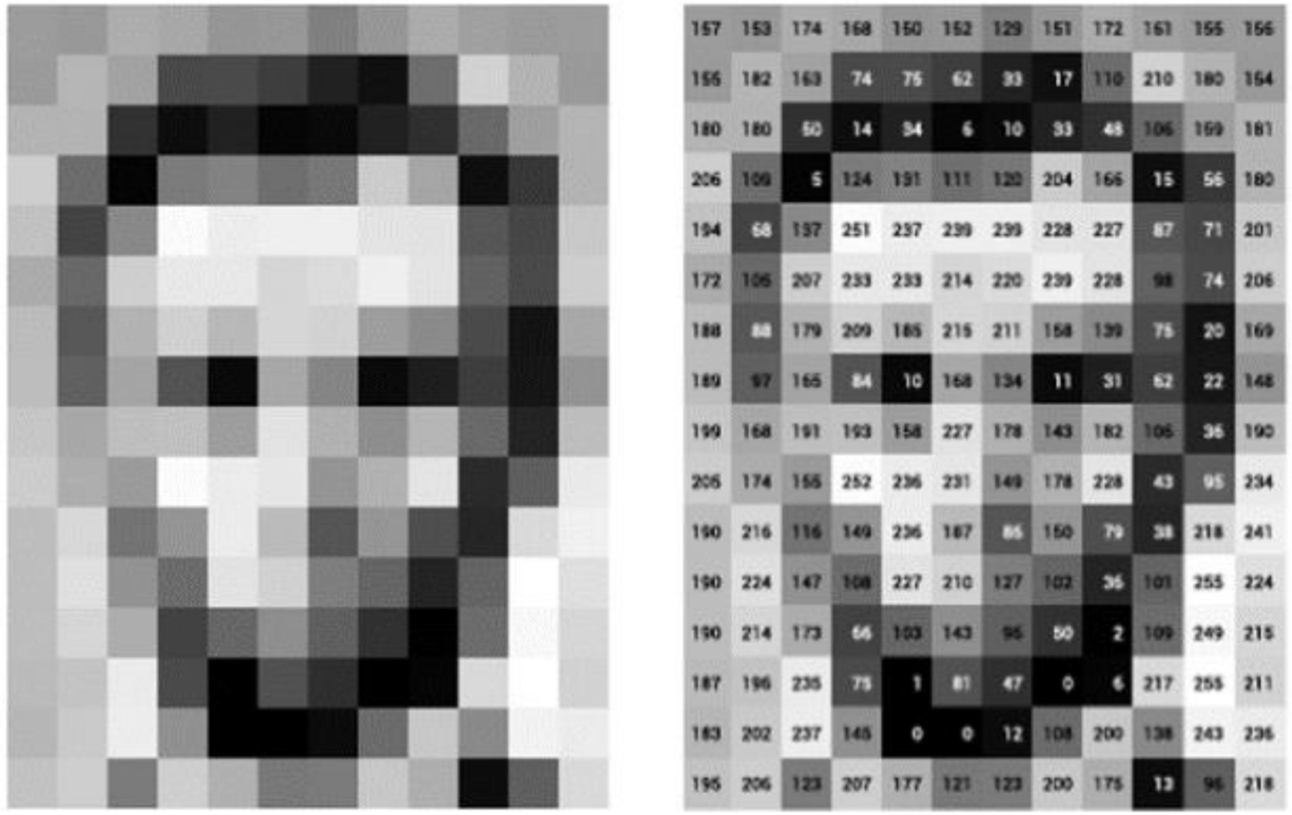
Facial Recognition



Augmented Reality

WORKING OF COMPUTER VISION

- Pattern Recognition
- Series of pixel values for each color→ matrix of numbers between 0 to 255

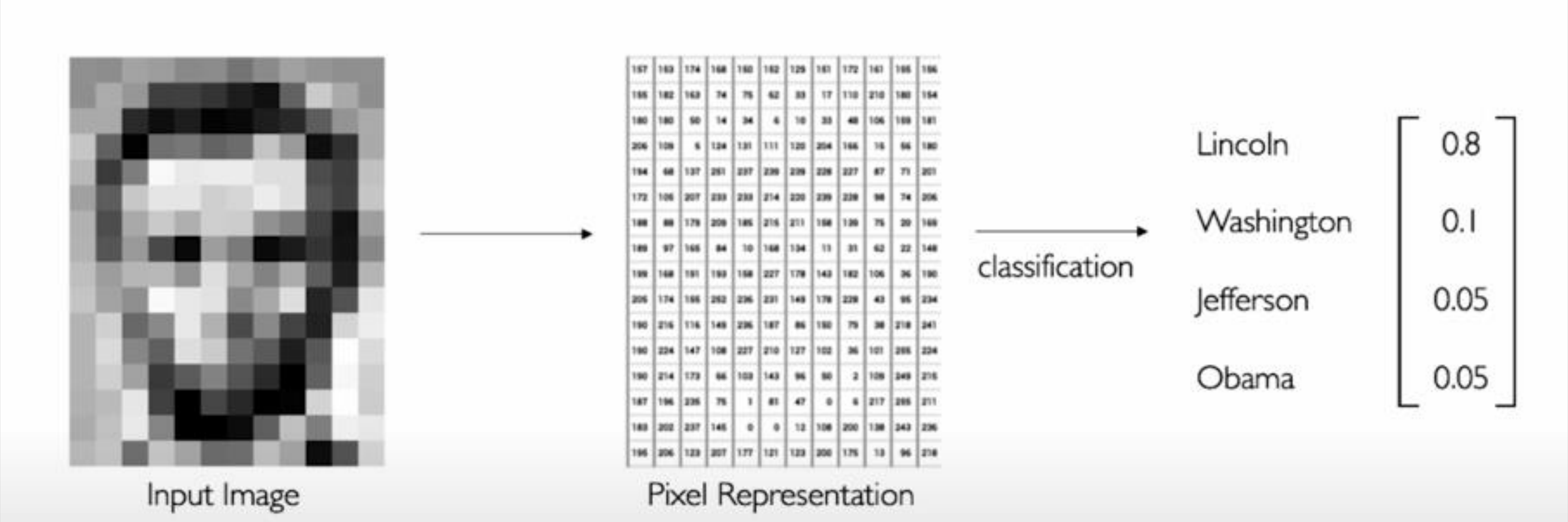


157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

Source: https://xd.adobe.com/ideas/principles/emerging-technology/what-is-computer-vision-how-does-it-work/

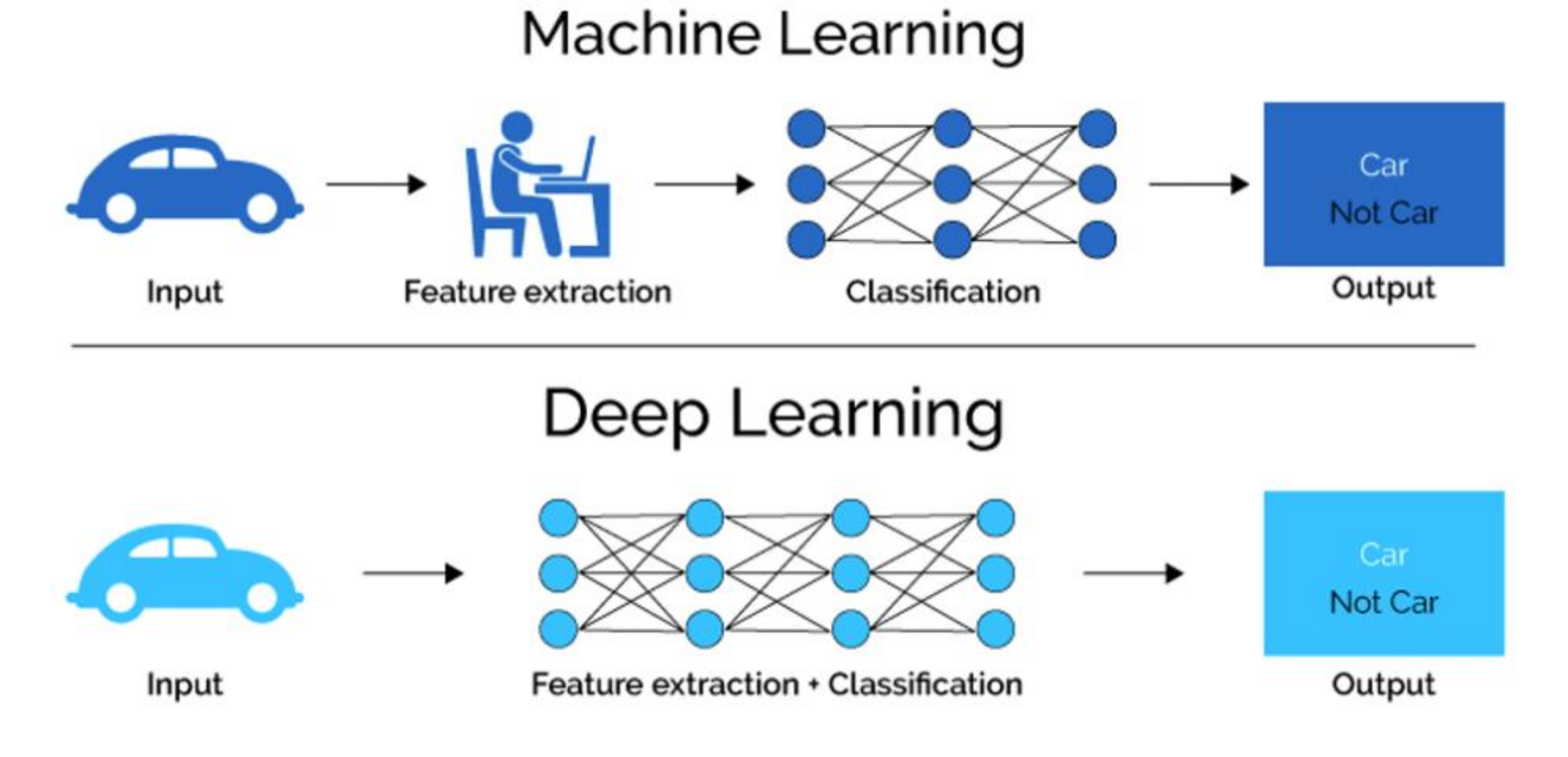
COMPUTER VISION TASKS

- Classification→ Probability of belonging to a particular class
- Regression→ target value is continuous



Source: http://introtodeeplearning.com/

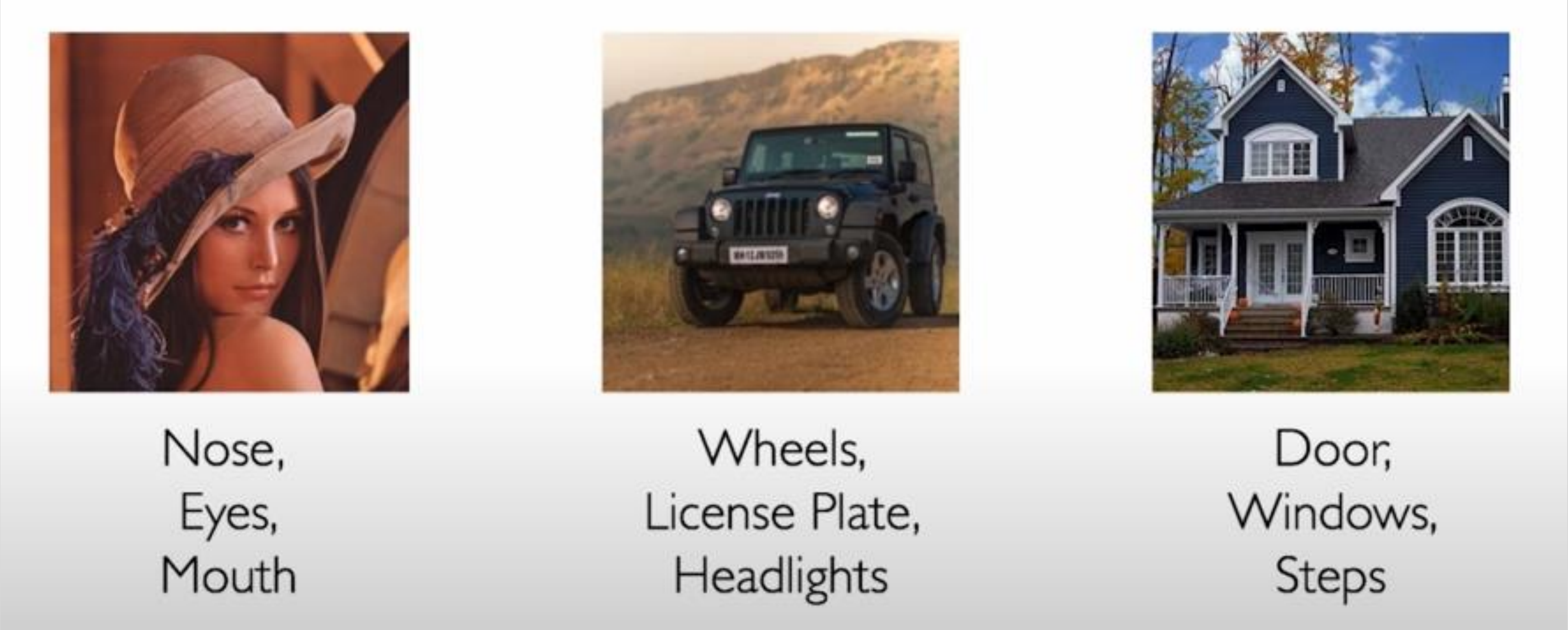
LEARNING FEATURES



Source: https://xd.adobe.com/ideas/principles/emerging-technology/what-is-computer-vision-how-does-it-work/

HIGH LEVEL FEATURES

- Identify key high-level features

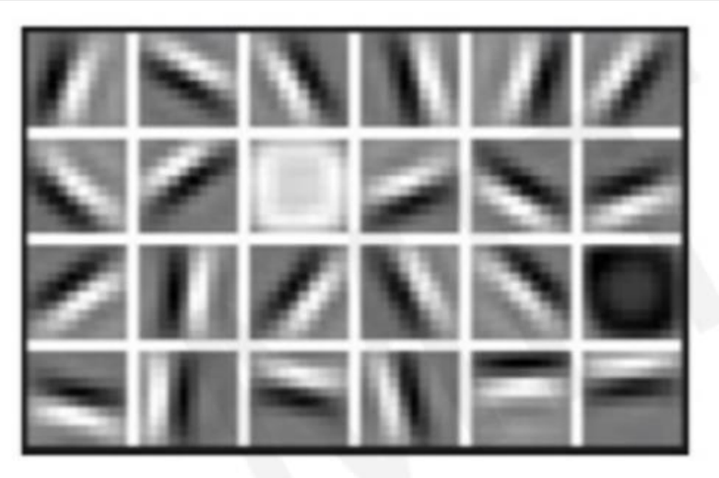


Source: introtodeeplearning.com

LEARNING FEATURE REPRESENTATION

- Increasing amount of data
- Increased features
- Time consuming→ impractical
- Computer Vision→ Deep learning→ learn a hierarchy of features

Low level features



Lines and Edges

Mid level features

Eyes and nose and ears

High level features

Facial Structure

Source: introtodeeplearning.com

CONVOLUTIONAL NEURAL NETWORK (CNN or ConvNet)

- Deep Learning Algorithm
- Takes image input, assigns importance to various objects in the image to differentiate them
- Ability to learn characteristics
- Analogous to connectivity pattern in Human brain→ ***Visual Cortex***
- Individual neurons respond to stimuli only in a restricted region of the visual field known as the ***Receptive Field***
- collection of such fields overlap to cover the entire visual area

- 1998→ ***Yann LeCun* et al.**
- **LeNet-5** Architecture→ MNIST Dataset

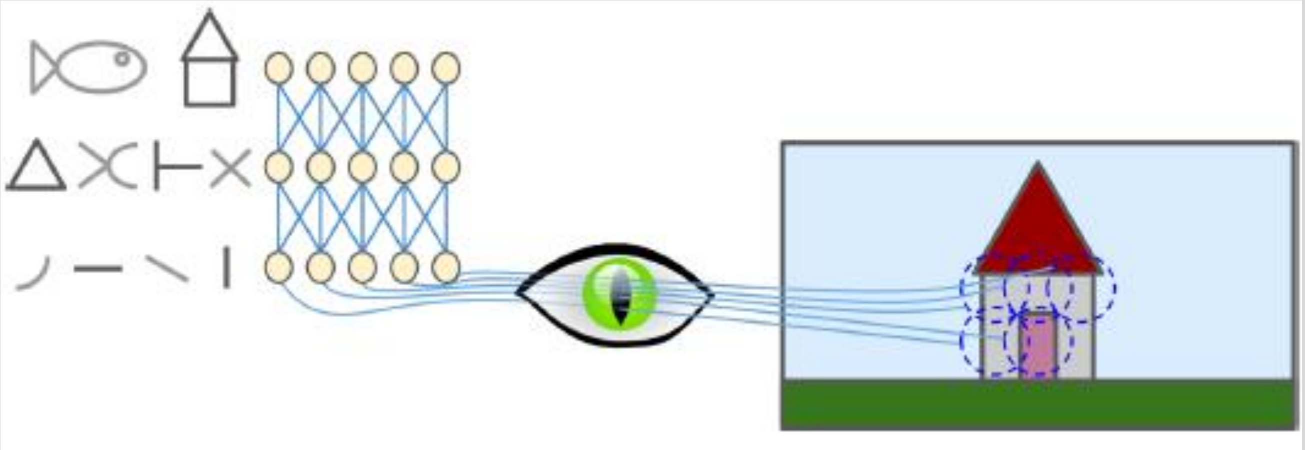
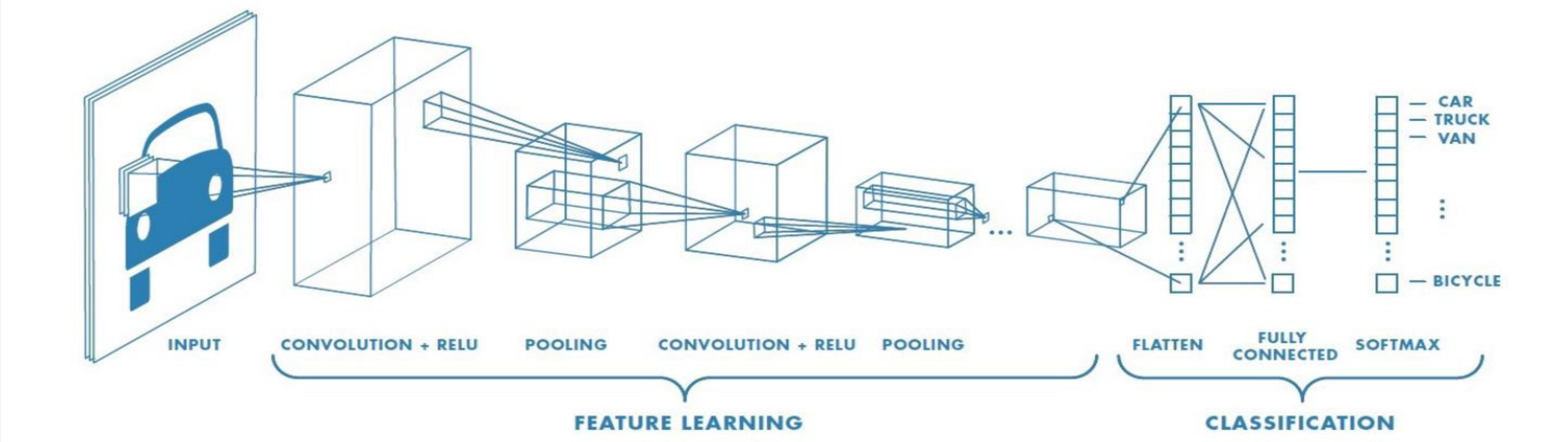


Figure: Biological neurons in the visual cortex respond to specific patterns in small regions of the visual field called receptive fields; as the visual signal makes its way through consecutive brain modules, neurons respond to more complex patterns in larger receptive fields.

Source: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

CNN (ConvNet) ARCHITECTURE



Source: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

BUILDING BLOCKS OF CNN

- CNN can identify 3 colors channels (red, green and blue→RGB) as well as other color spaces
- Complex image 8K (7680x4320)
- Reduce into simpler form without losing important features
- Scalable to huge datasets

- **Convolutional layer**
- **Pooling layer**
- **Fully-connected (FC) layer**

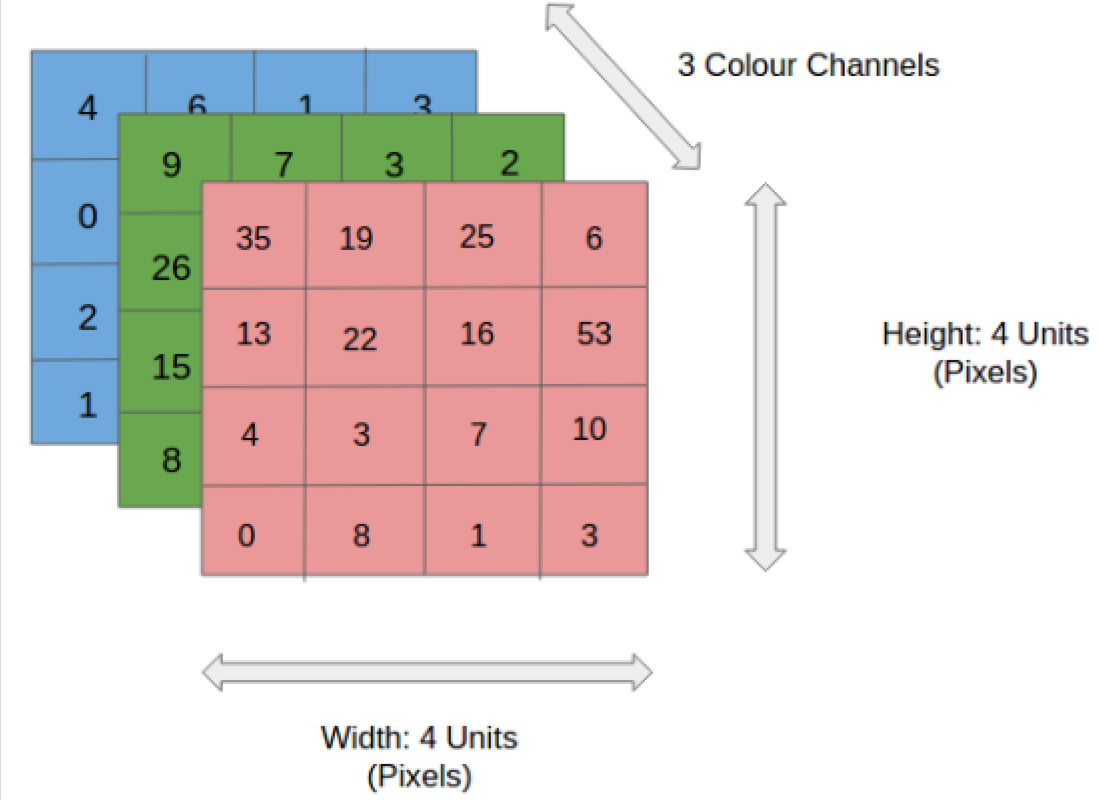


Figure: 4x4x3 RGB Image

Source: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

CONVOLUTIONAL LAYER - THE KERNEL

- core building block of a CNN, and it is where most of the computation occurs
- input data, a filter, and a feature map
- Image Dimensions = 5 (Height) x 5 (Breadth) x 1 (Number of channels, e.g., RGB)
- green section resembles our 5x5x1 input image (**I**)
- element involved in carrying out the convolution operation in the first part of a Convolutional Layer is called the **Kernel/Filter**, **K** (Yellow color) → 3x3x1 matrix
- K shifts 9 times (stride=1) performing matrix multiplication between **K** and portion **P** of the image **I**
- **Stride:** amount movement between applications of the filter to the input image, default value is 1→ one unit at a time

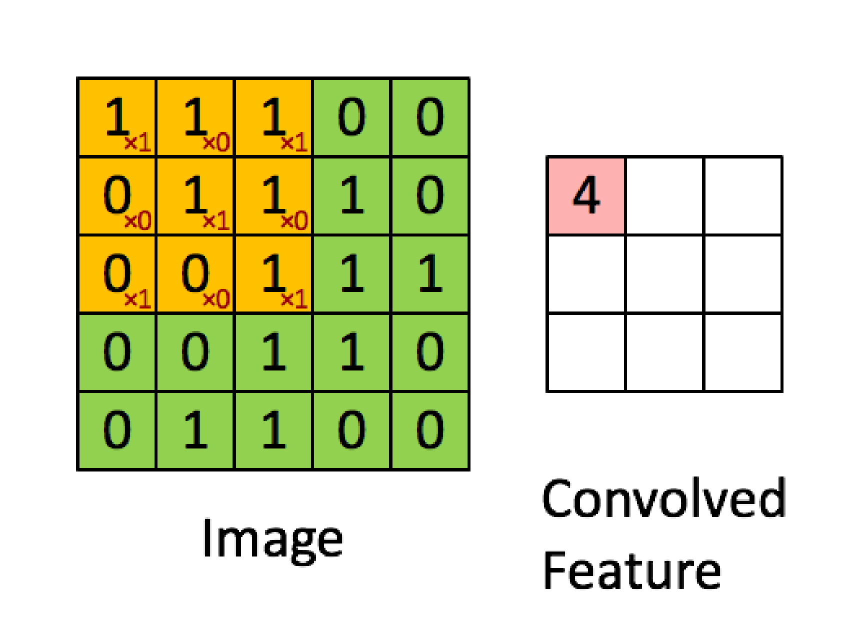


Figure: Convoluting a 5x5x1 image with a 3x3x1 kernel to get a 3x3x1 convolved feature

```
Kernel/Filter, K =

1  0  1
0  1  0
1  0  1
```

Source: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

CONVOLUTIONAL LAYER - THE KERNEL

- The filter moves to the right with a certain Stride Value till it parses the complete width
- First ConvLayer is responsible for capturing the Low-Level features such as edges, color, gradient orientation, etc
- More layers→ higher level features
- understanding of images in the dataset

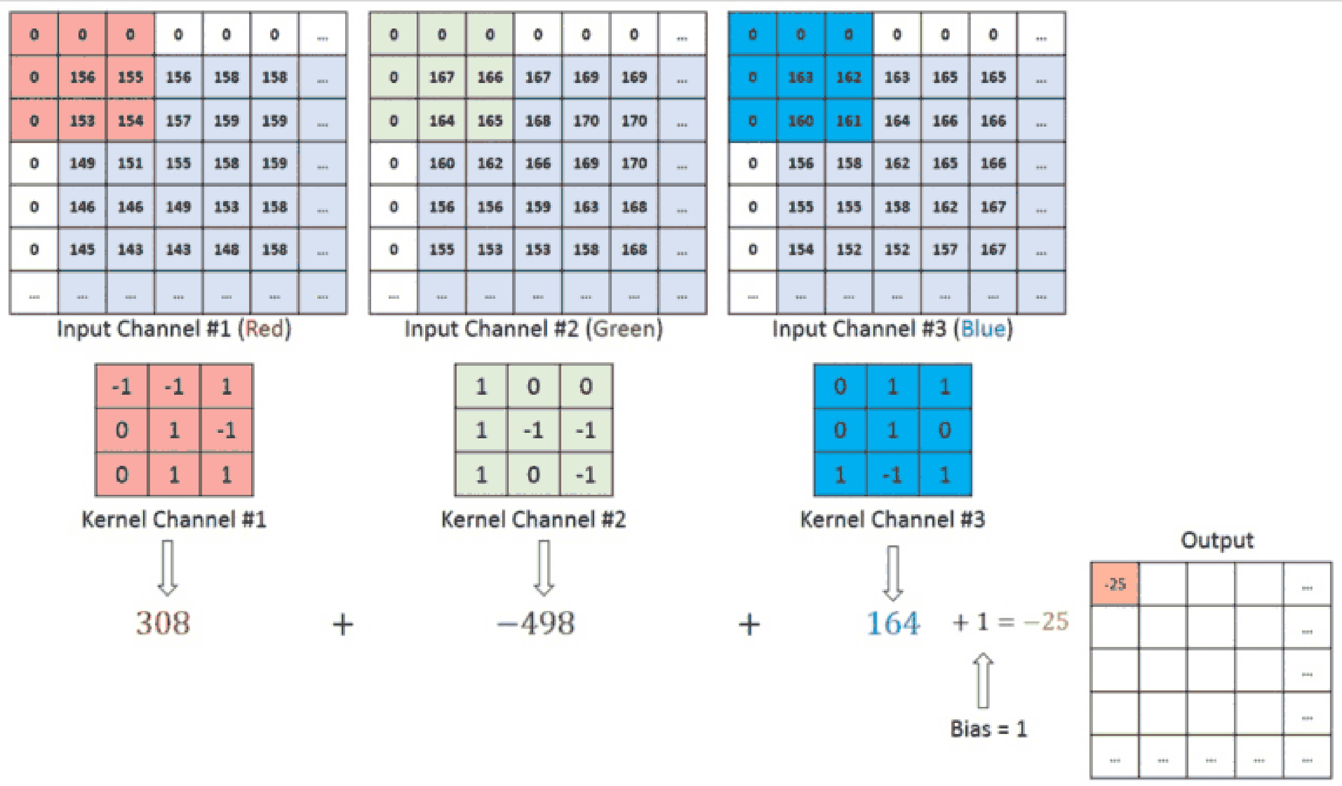


Figure: Convolution operation on a MxNx3 image matrix with a 3x3x3 Kernel

Source: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

PARAMETER SHARING IN CNN

- control the number of parameters

Parameters whose value is set before training:

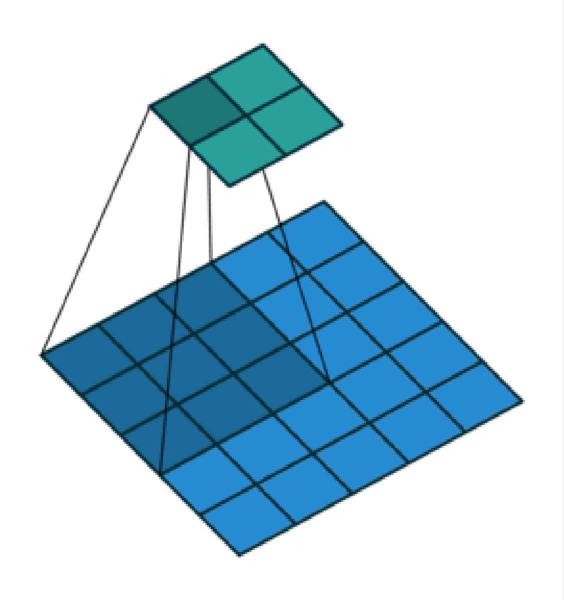
- **Number of filters:** affects the depth of the output
- **Stride:** distance, or number of pixels, that the kernel moves over the input matrix
- **Zero-padding:** is usually used when the filters do not fit the input image. This sets all elements that fall outside of the input matrix to zero, producing a larger or equally sized output.

Note: After each convolution operation, a CNN applies a Rectified Linear Unit (ReLU) transformation to the feature map, introducing nonlinearity to the model.

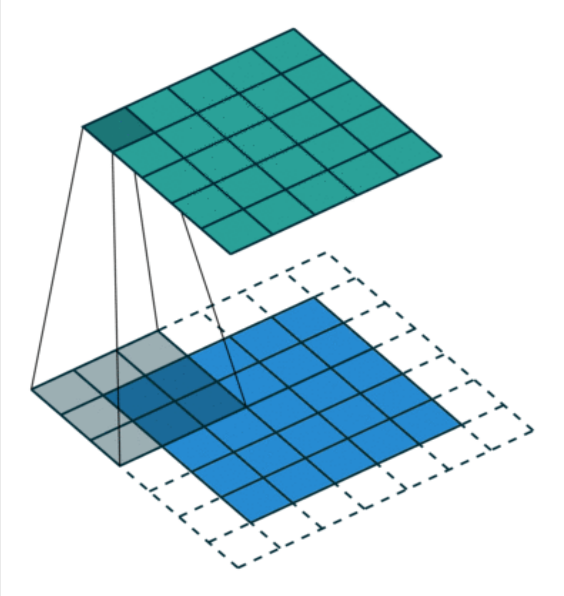
Source: https://www.ibm.com/cloud/learn/convolutional-neural-networks

TYPES OF PADDING

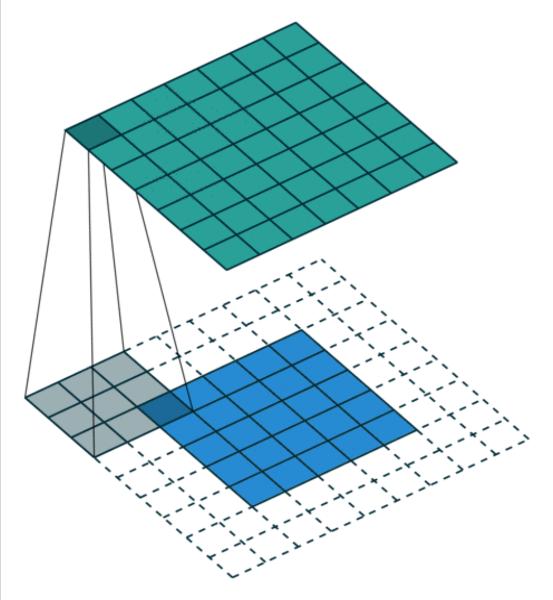
- ✓ **Valid/No padding**→ last convolution is dropped if dimensions do not align
- ✓ **Same padding**→ ensures that the output layer has the same size as the input layer
- ✓ **Full padding**→ increases the size of the output by adding zeros to the border of the input



No padding



Same padding, 5x5x1 image is padded with 0s to create a 6x6x1 image

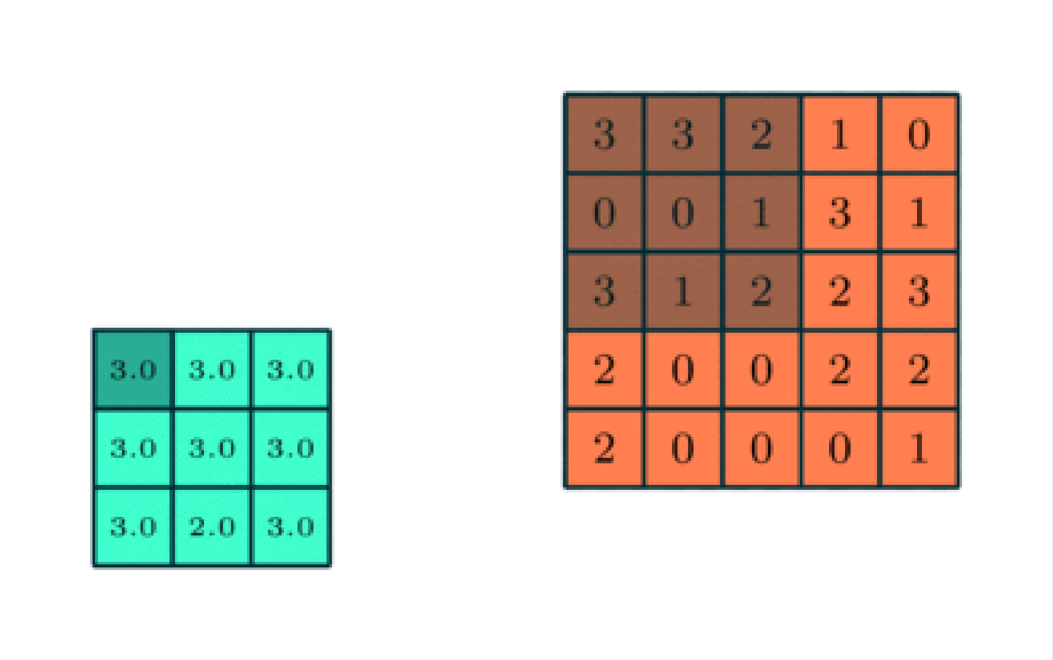


Full padding

Source: https://github.com/vdumoulin/conv_arithmetic

POOLING LAYER

- responsible for reducing the spatial size of the Convolved Feature
- sweeps a filter across the entire input
- decrease the computational power required to process the data
- **Downsampling** → reduces dimensions
- kernel applies an aggregation function to the values→ populating the output array
- extracting dominant features
- reduce complexity and overfitting
- improve efficiency

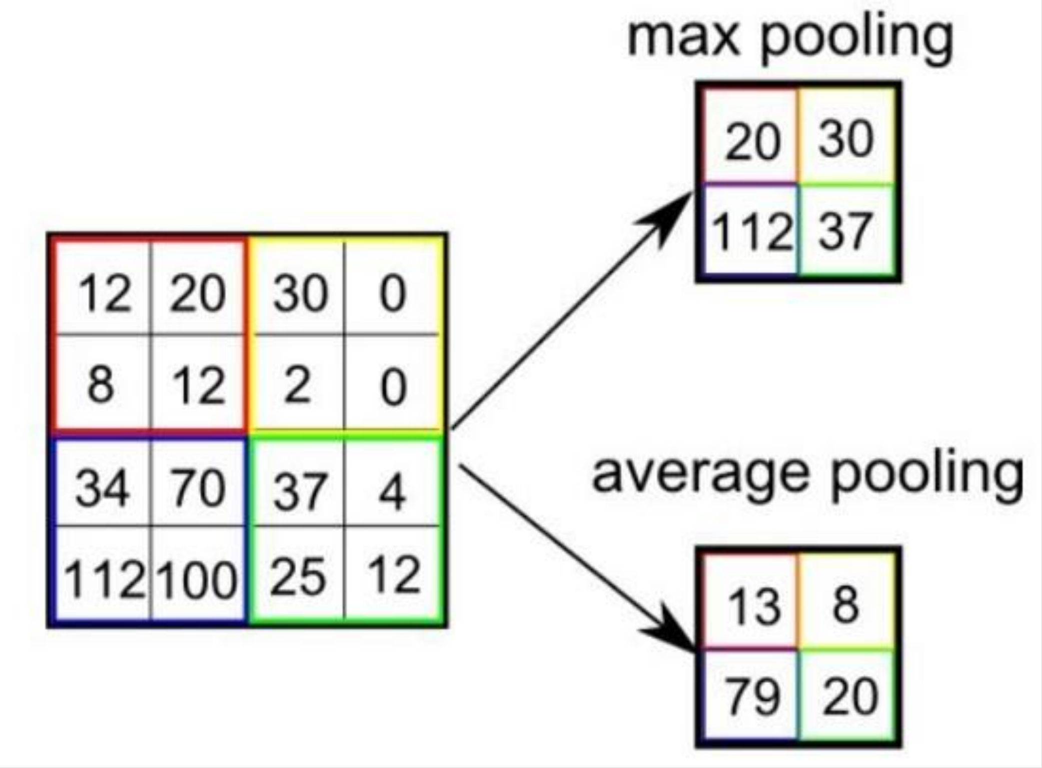


3x3 pooling over 5x5 convolved feature

Source: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

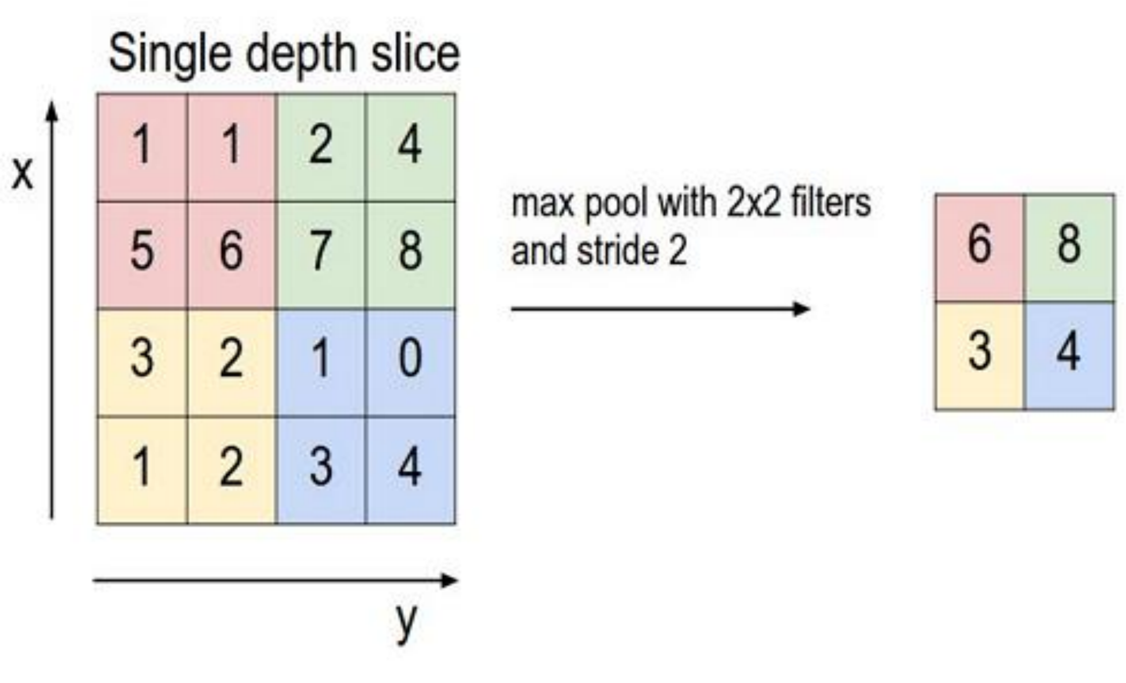
TYPES OF POOLING

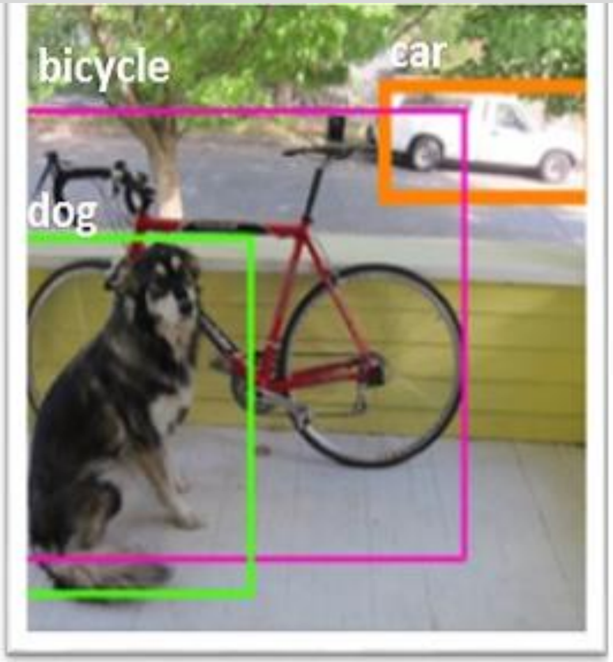
- ✓ **Max Pooling**→ returns the maximum value from the portion of the image covered by the Kernel→ discards the noisy activations
- ✓ **Average Pooling**→ returns the average of all the values from the portion of the image covered by the Kernel→ dimension reduction

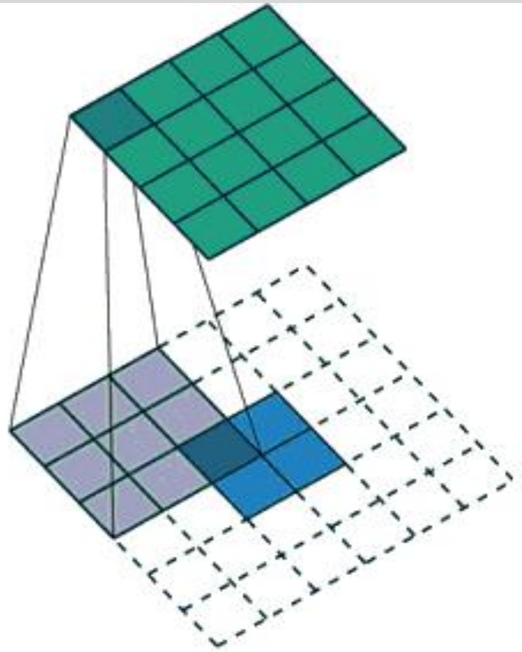


Source: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

AN EXAMPLE OF MAX POOLING

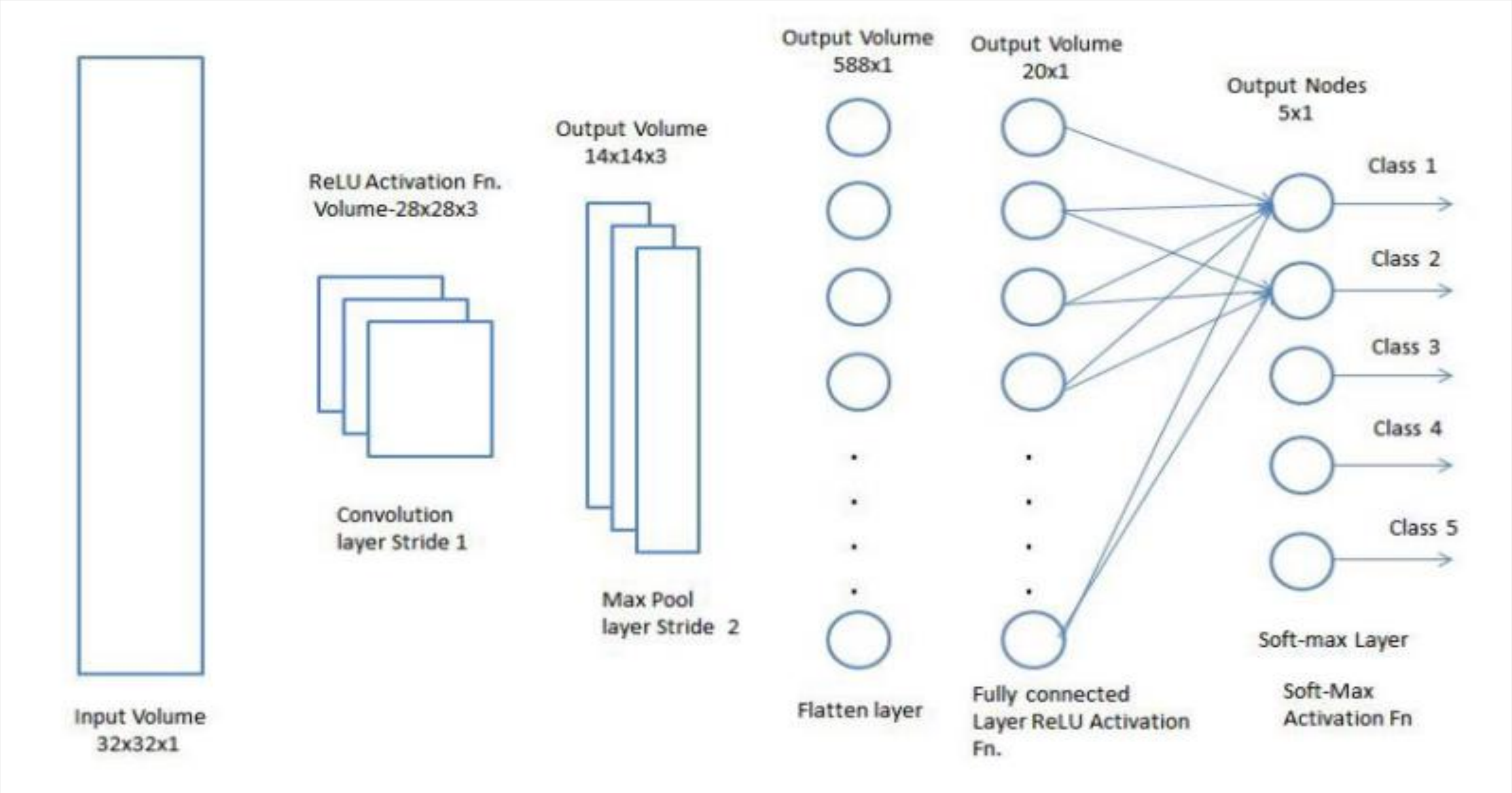






Source: https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks-Part-2/

FULLY CONNECTED (FC) LAYER FOR CLASSIFICATION



Source: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

FULLY CONNECTED (FC) LAYER

- each node in the output layer connects directly to a node in the previous layer→ FC layer
- classification based on the features extracted through the previous layers and their different filters
- Learn non-linear combinations of high-level features from convolutional and pooling layers
- Convolutional and pooling layers→ ReLU function
- Flatten the input before feeding in the Feed-forward Neural Network
- Backpropagation of error→ each iteration→ Gradient Descent→ improve classification results
- Low-level features→ convolutional layers and classification at output layer
- FC layers usually leverage a softmax activation function to classify different classes

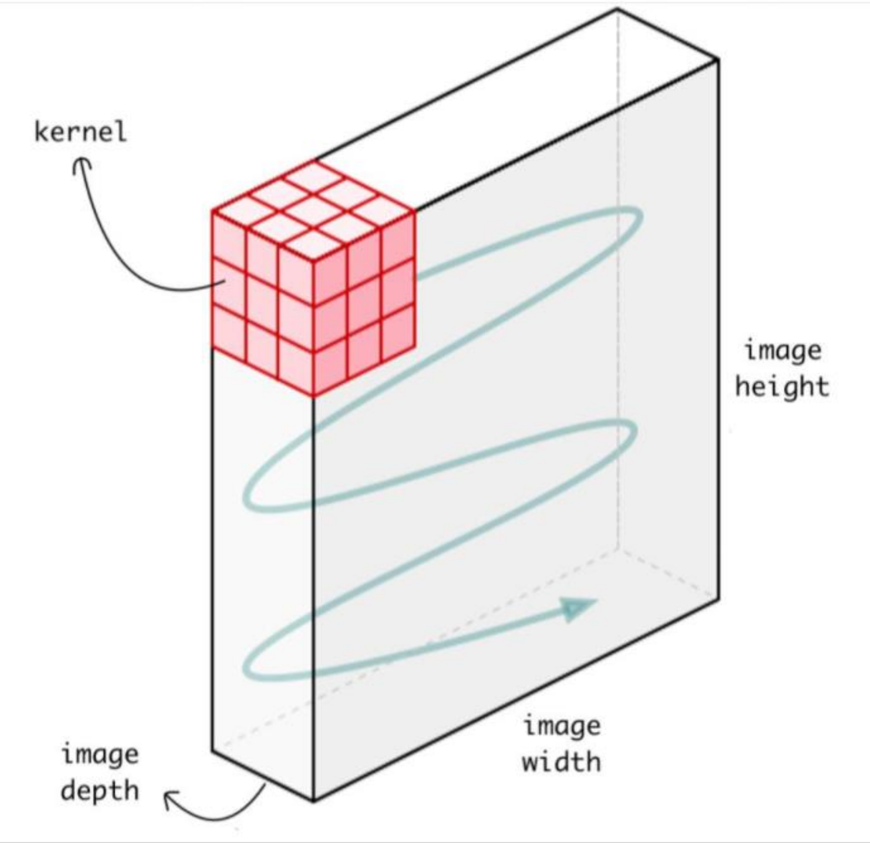
Source: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

1D, 2D and 3D Convolution

1D CNN
- kernel moves in one direction
- The input and output data of 1D CNN is 2-dimensional
- Time series data

2D CNN
- kernel moves in 2 directions
- input and output data of 2D CNN is 3-dimensional
- image data

3D CNN
- the kernel moves in 3 directions
- input and output data is 4-dimensional
- Usually used for 3D image data (MRI, CT scan)



Source: https://www.programmersought.com/article/63014851723/

1D, 2D and 3D Convolution Examples

1D CNN

```
import keras

from keras.layers import Conv1D

model = keras.models.Sequential()

model.add(Conv1D(1, kernel_size=5, input_shape = (120, 3)))

model.summary()
```

2D CNN

```
import keras

from keras.layers import Conv2D

model = keras.models.Sequential()

model.add(Conv2D(1, kernel_size=(3,3), input_shape = (128, 128, 3)))

model.summary()
```

3D CNN

```
import keras

from keras.layers import Conv3D

model = keras.models.Sequential()

model.add(Conv3D(1, kernel_size=(3,3,3), input_shape = (128, 128, 128, 3)))

model.summary()
```

Source: https://www.programmersought.com/article/63014851723/

DIFFERENT CNN ARCHITECTURES

- **LeNet-5**→Yann LeCun in **1998**
- **AlexNet**→ Alex Krizhevsky in **2012**
- **GoogLeNet** → Christian Szegedy et al. from Google Research in **2014**
- **VGGNet** → Karen Simonyan and Andrew Zisserman from the Visual Geometry Group (VGG) research lab at Oxford University in **2014**
- **Residual Network (or ResNet)** → **2015,** Kaiming He et al.
- **Xception**→ variant of GoogLeNet architecture, **2016**, François Chollet
- **Squeeze-and-Excitation Network (SENet)**→ **2017,** Jie Hu et al.
- **You Only Look Once (YOLO)**→ object detection→ **2018** (YOLOv2)→ Joseph Redmon et al.

Source: Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow

CNN ARCHITECTURE EXAMPLES

LeNet-5 architecture

Layer	Type	Maps	Size	Kernel size	Stride	Activation
Out	Fully connected	–	10	–	–	RBF
F6	Fully connected	–	84	–	–	tanh
C5	Convolution	120	1 × 1	5 × 5	1	tanh
S4	Avg pooling	16	5 × 5	2 × 2	2	tanh
C3	Convolution	16	10 × 10	5 × 5	1	tanh
S2	Avg pooling	6	14 × 14	2 × 2	2	tanh
C1	Convolution	6	28 × 28	5 × 5	1	tanh
In	Input	1	32 × 32	–	–	–

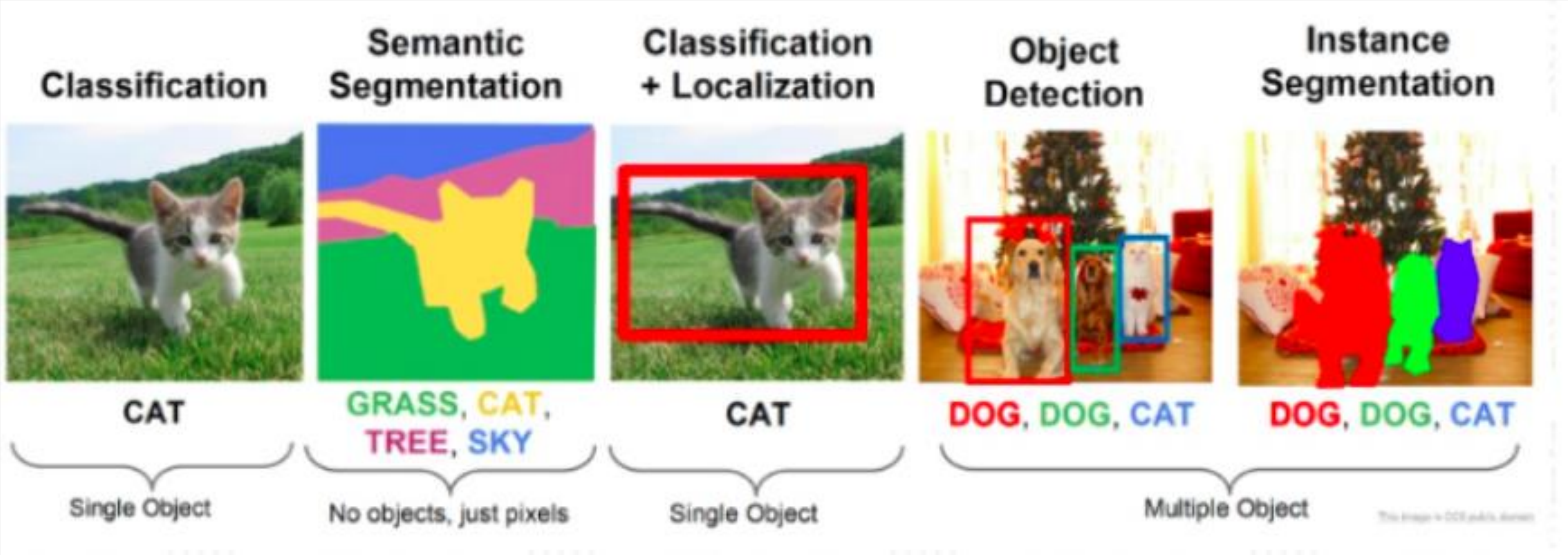
AlexNet architecture

Layer	Type	Maps	Size	Kernel size	Stride	Padding	Activation
Out	Fully connected	–	1,000	–	–	–	Softmax
F10	Fully connected	–	4,096	–	–	–	ReLU
F9	Fully connected	–	4,096	–	–	–	ReLU
S8	Max pooling	256	6 × 6	3 × 3	2	valid	–
C7	Convolution	256	13 × 13	3 × 3	1	same	ReLU
C6	Convolution	384	13 × 13	3 × 3	1	same	ReLU
C5	Convolution	384	13 × 13	3 × 3	1	same	ReLU
S4	Max pooling	256	13 × 13	3 × 3	2	valid	–
C3	Convolution	256	27 × 27	5 × 5	1	same	ReLU
S2	Max pooling	96	27 × 27	3 × 3	2	valid	–
C1	Convolution	96	55 × 55	11 × 11	4	valid	ReLU
In	Input	3 (RGB)	227 × 227	–	–	–	–

Source: Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow

APPLICATIONS OF CNN

- Classification
- Classification and Localization
- Object Detection
- Semantic Segmentation
- Instance Segmentation



Source: https://www.r-bloggers.com/2020/10/platypus-r-package-for-object-detection-and-image-segmentation/

CNN HANDS-ON EXERCISE

THANKS!

Do you have any questions?