Image analysis

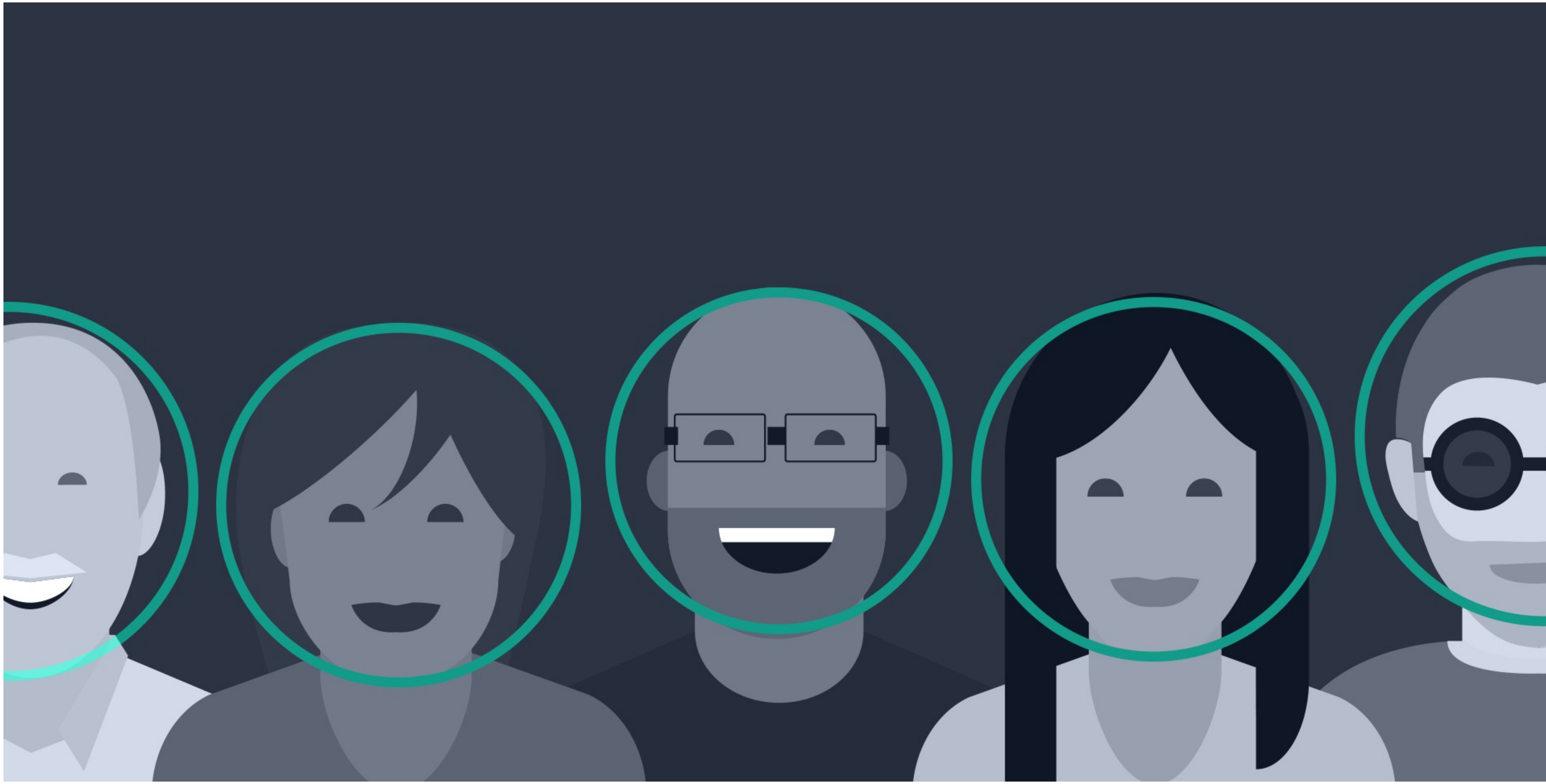
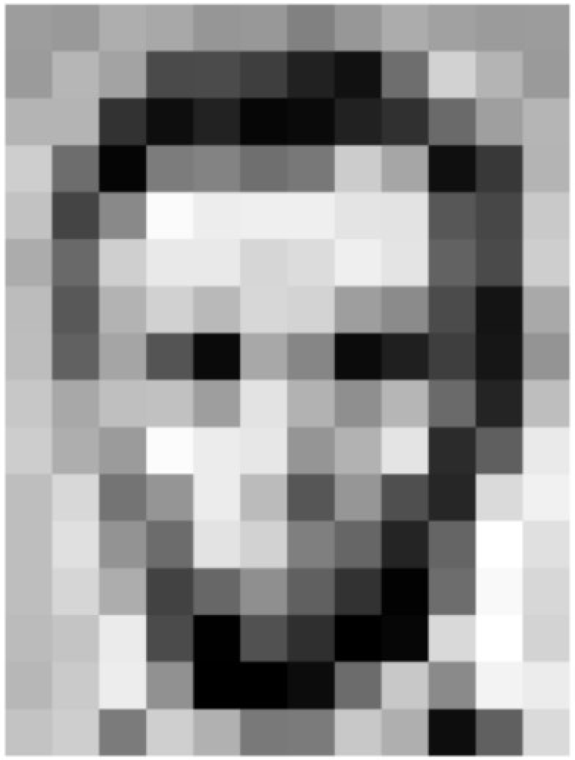
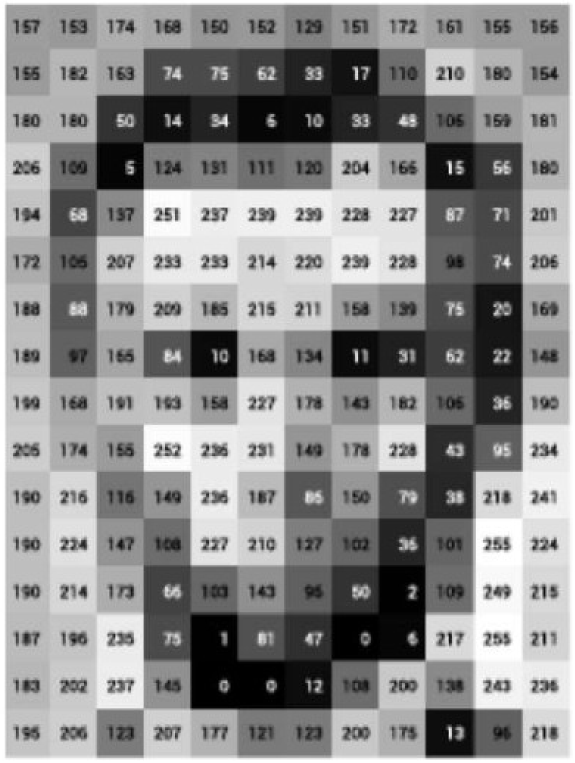
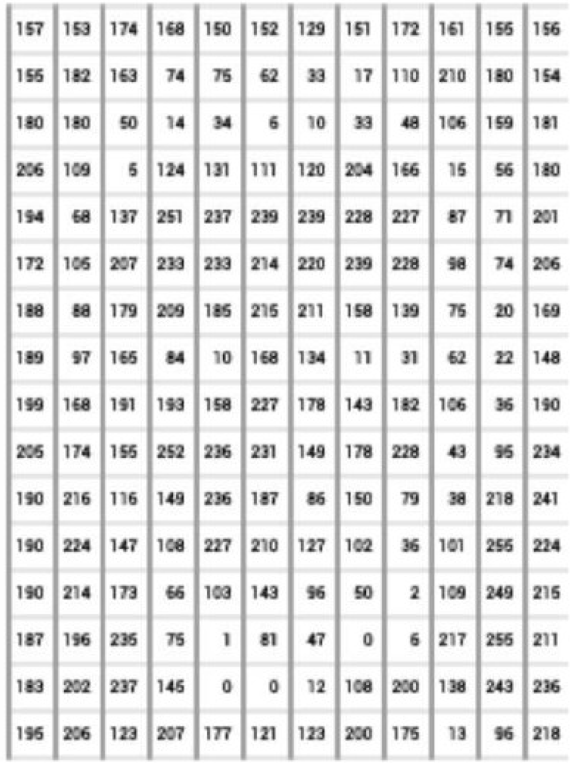


Image are matrices







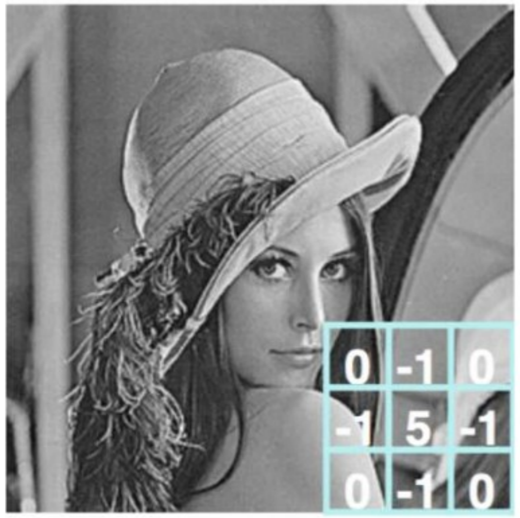
157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	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

Pixels are numbers

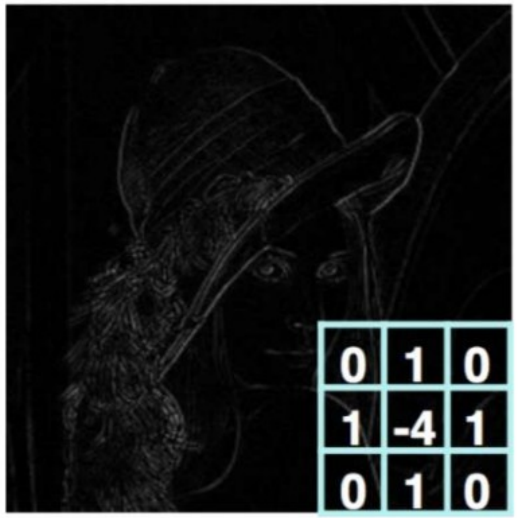
- Normalize Pixel Values [0..255]→[0,1]
- Center Pixel Values: scale pixel values to have a zero mean.
  - ImageNet [0.485, 0.456, 0.406]
- Pixel Standardization: scale pixel values to have a zero mean and unit variance.

Filters are operations

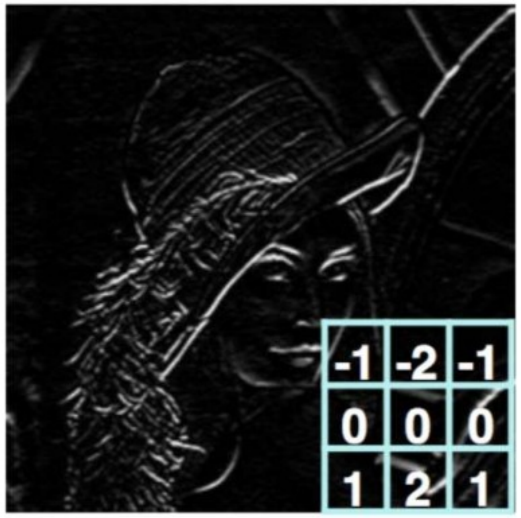
Original



Sharpen

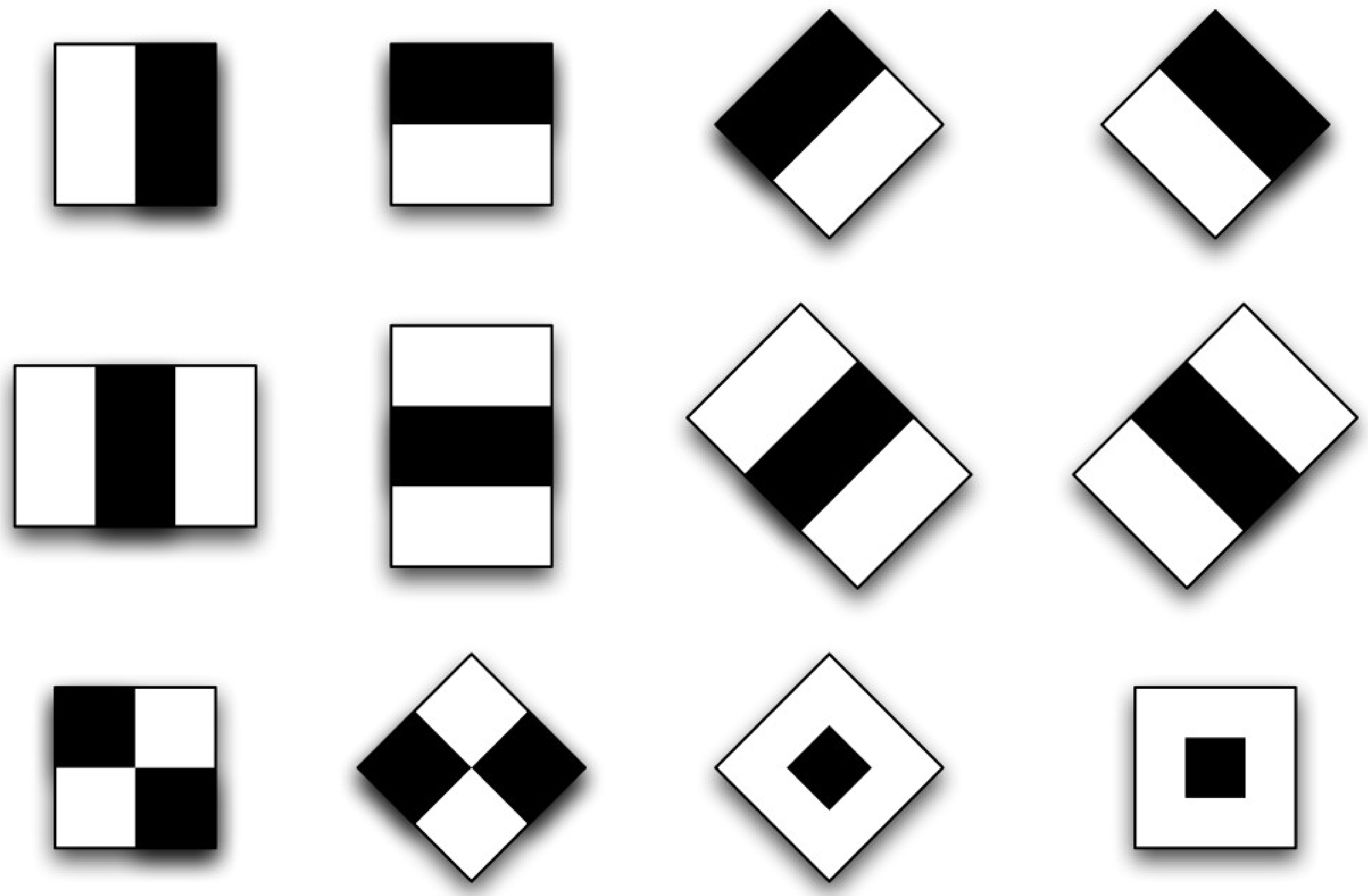


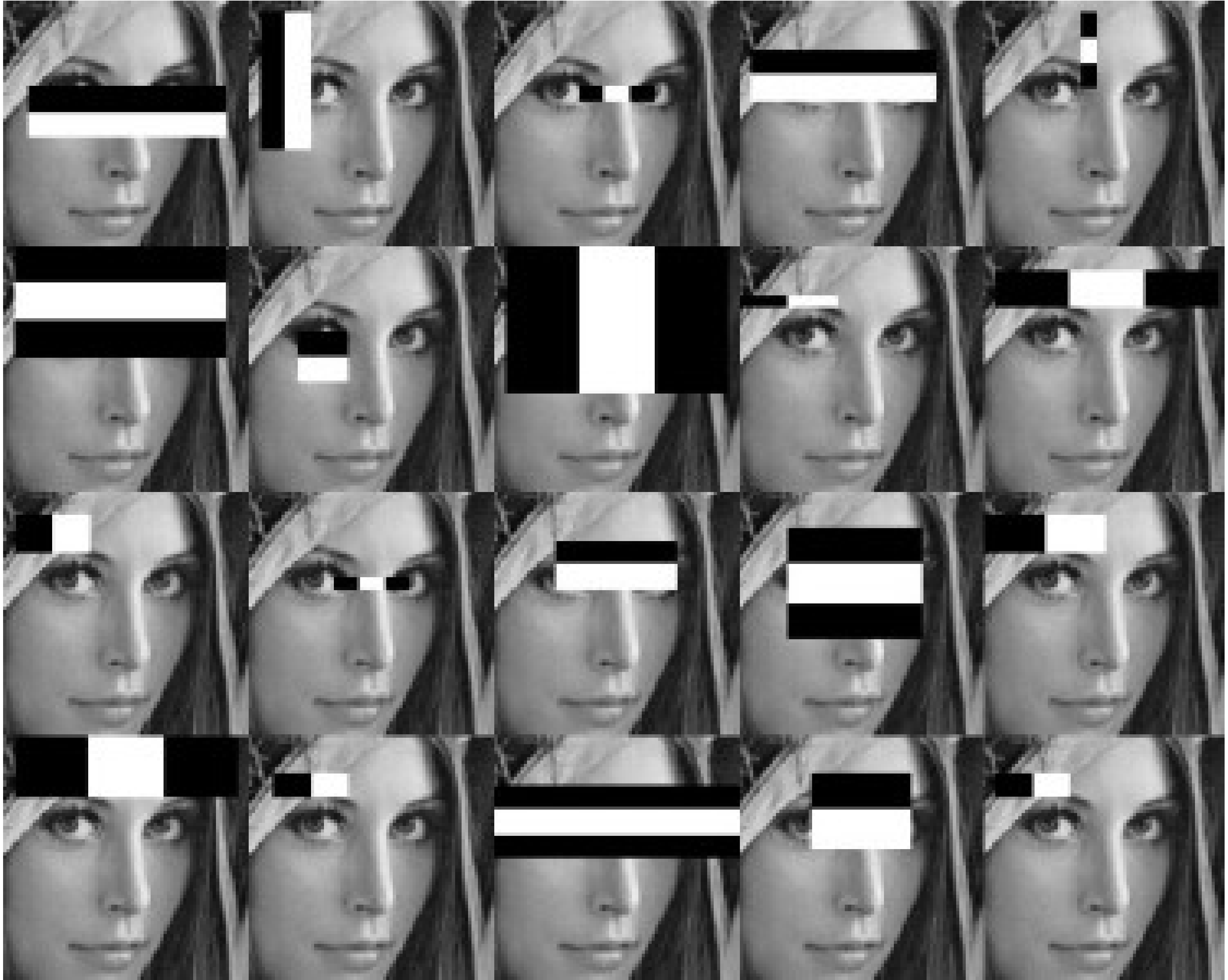
Edge Detect

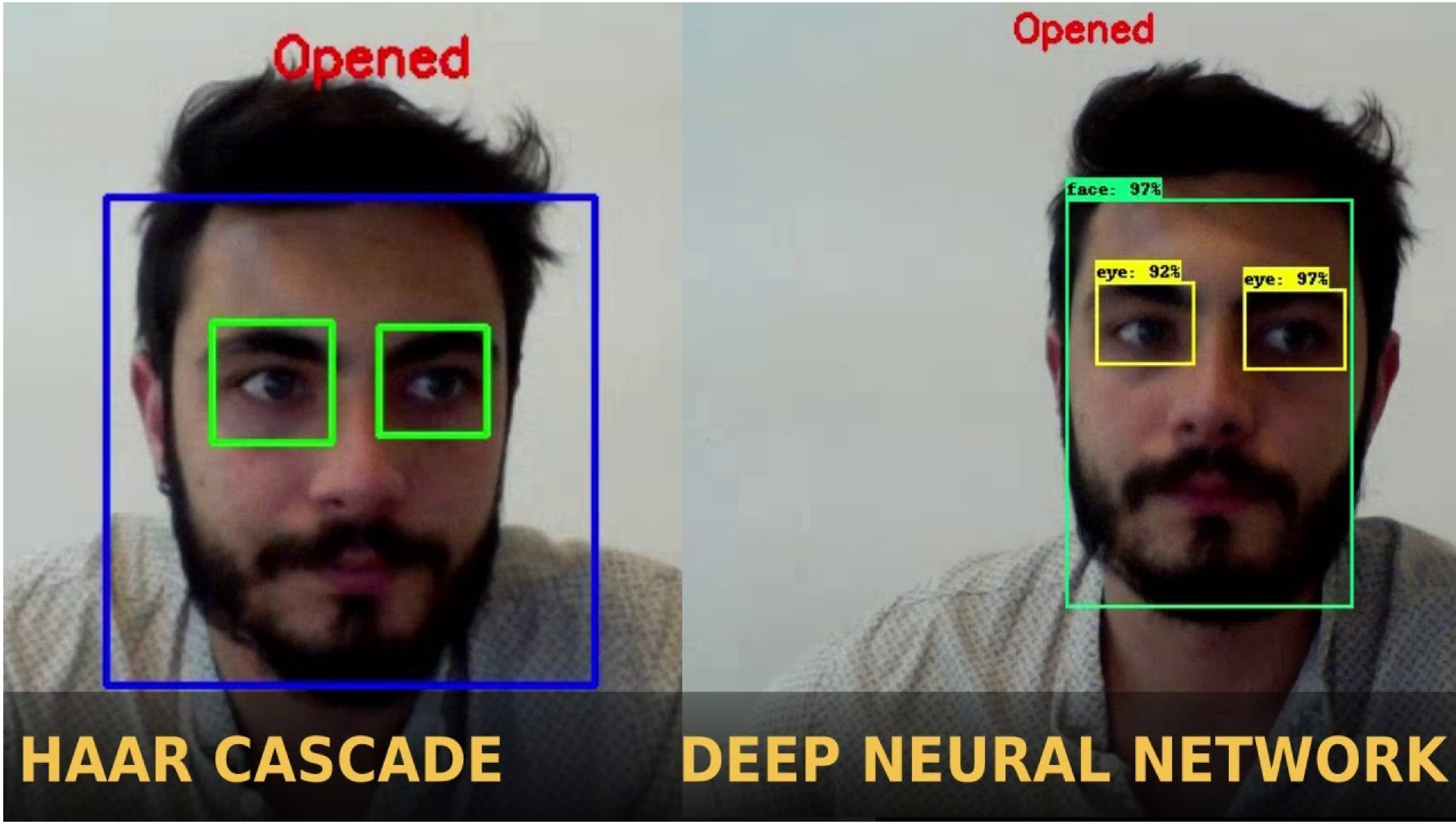


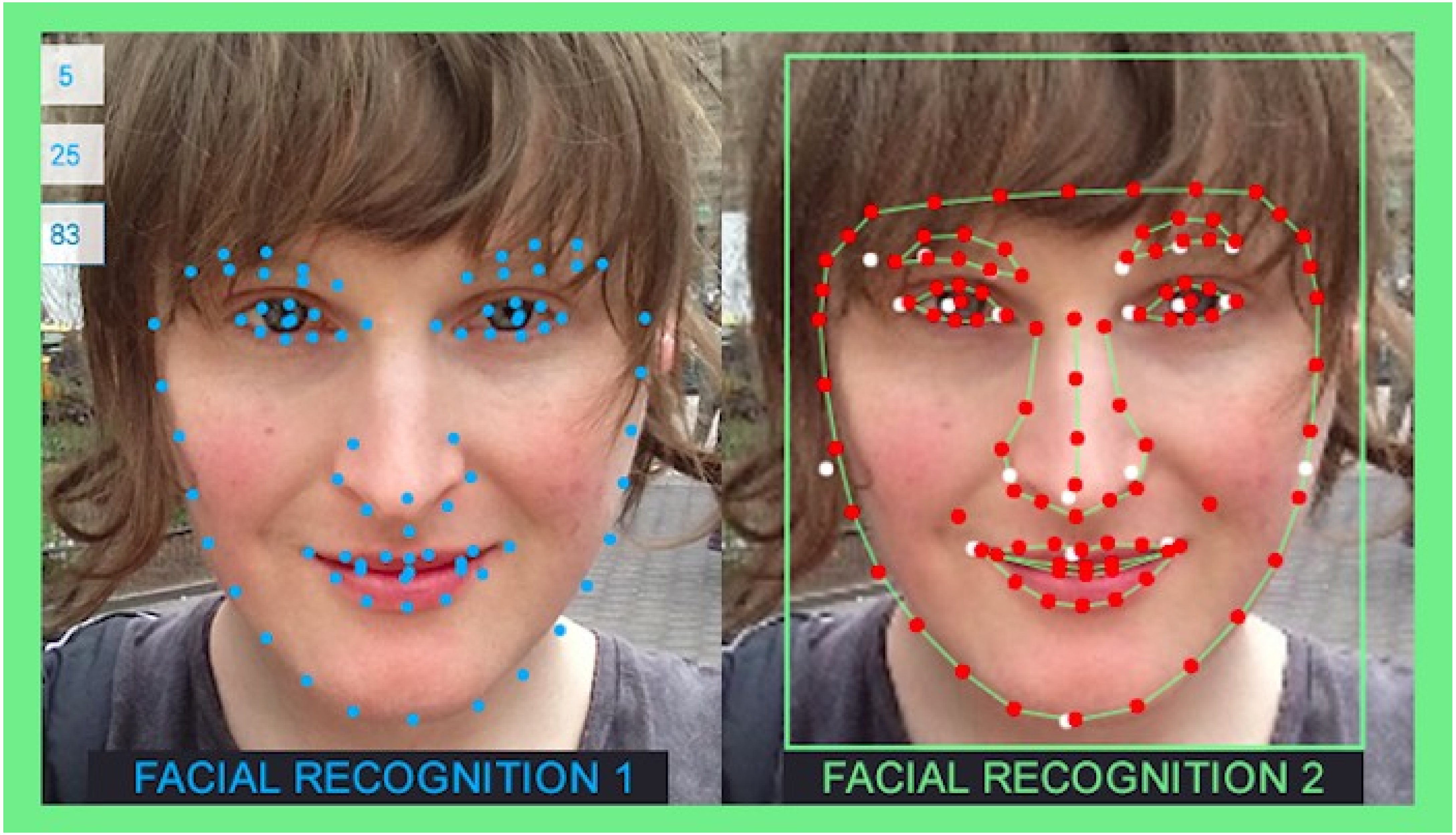
"Strong" Edge Detect

Detectors are masks






Opened
Opened
face: 97%
eye: 92%
eye: 97%
HAAR CASCADE
DEEP NEURAL NETWORK


5
25
83
FACIAL RECOGNITION 1
FACIAL RECOGNITION 2

pyimagesearch
You can master Computer Vision, Deep Learning, and OpenCV

About FAQ Contact

Get Started Topics Books and Courses OpenCV Install Guides Blog

You can master Computer Vision, Deep Learning, and OpenCV.

I've taken some of my best material from the past 5 years running PyImageSearch and designed a fully personalized, 17-lesson crash course on how to learn Computer Vision, Deep Learning, and OpenCV. **Get instant access now.**

START YOUR FIRST LESSON

https://www.pyimagesearch.com/

Deep Learning for Image analysis

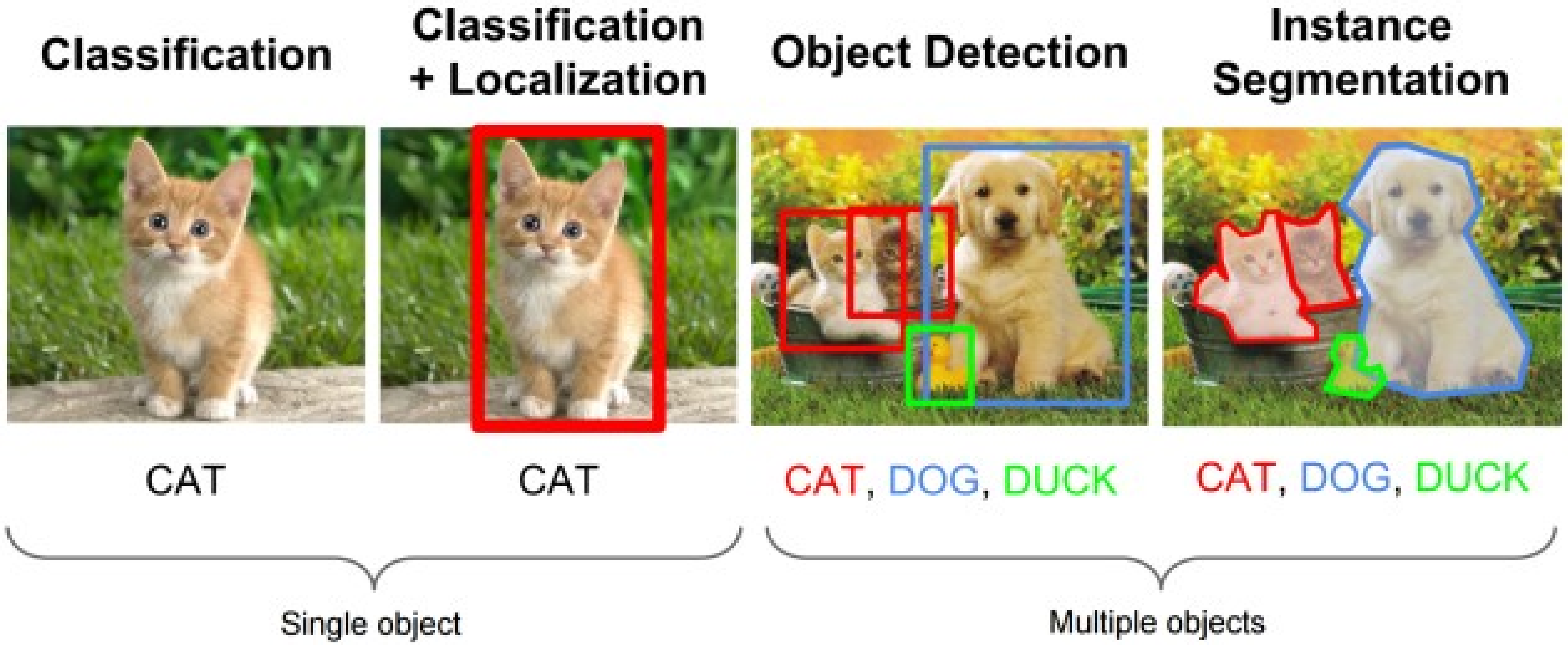
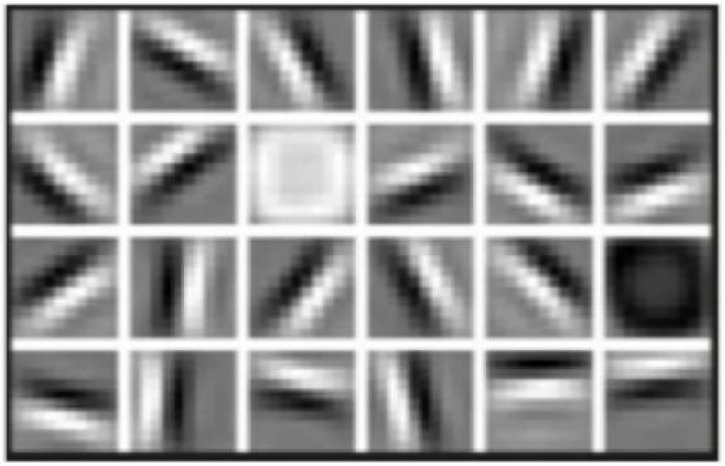
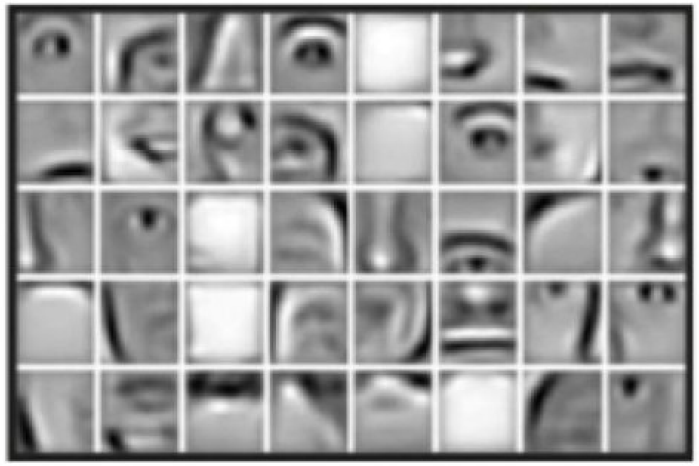
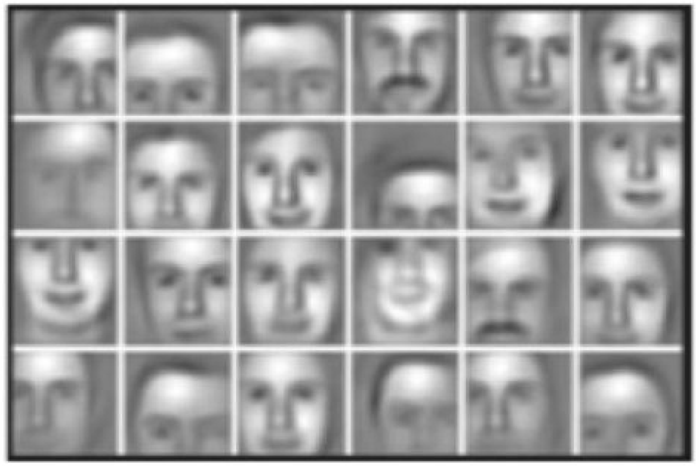

Classification
Classification + Localization
Object Detection
Instance Segmentation
CAT
CAT
CAT, DOG, DUCK
CAT, DOG, DUCK
Single object
Multiple objects

Image features

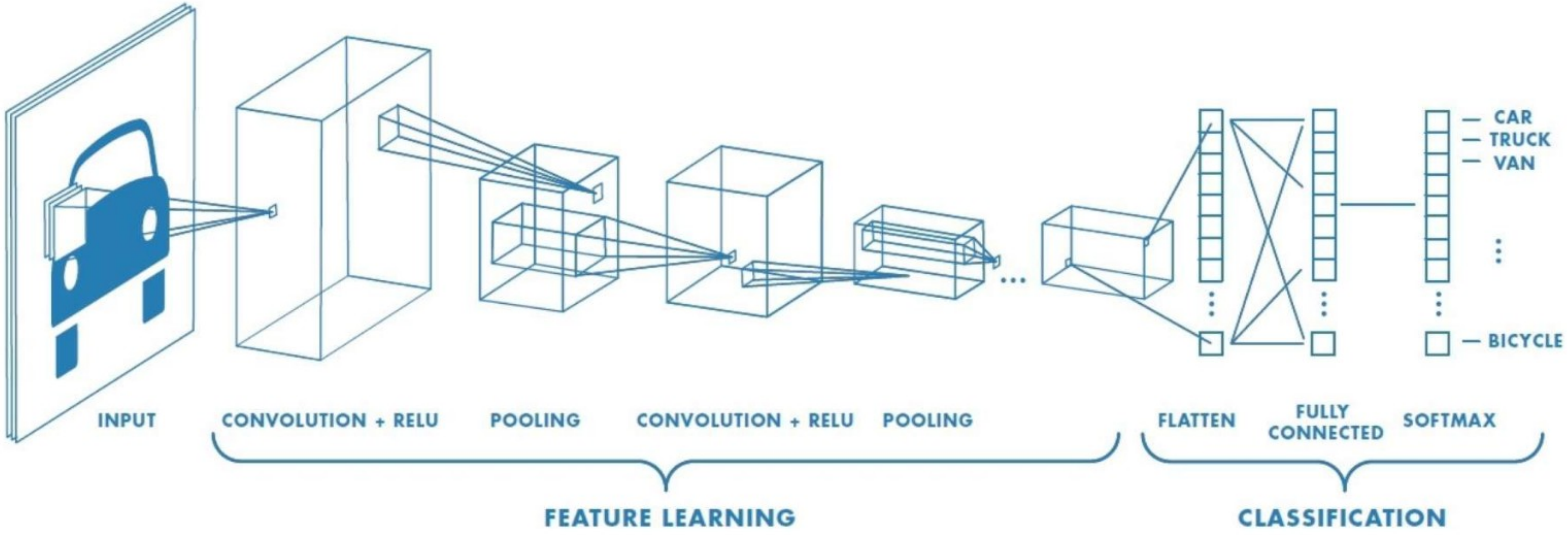






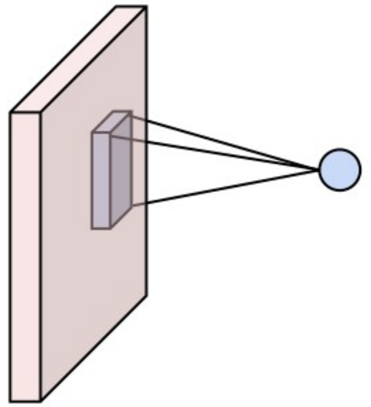
Can we learn the underlying features
directly from data?

Convolutional networks

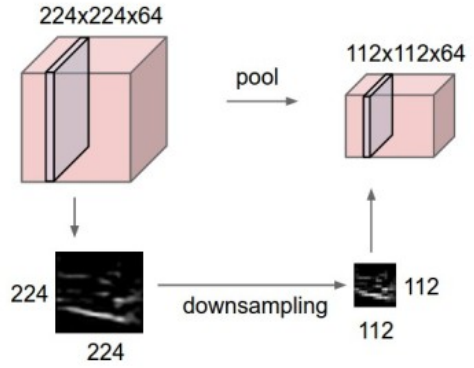


Convolutional networks

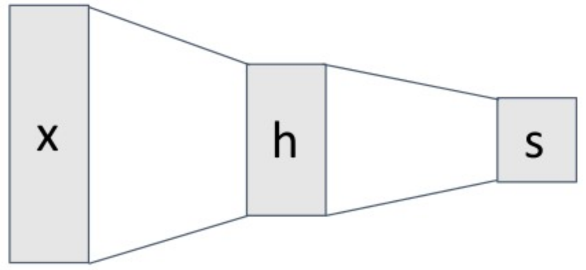
Convolution Layers



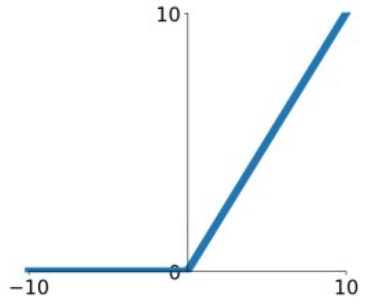
Pooling Layers



Fully-Connected Layers

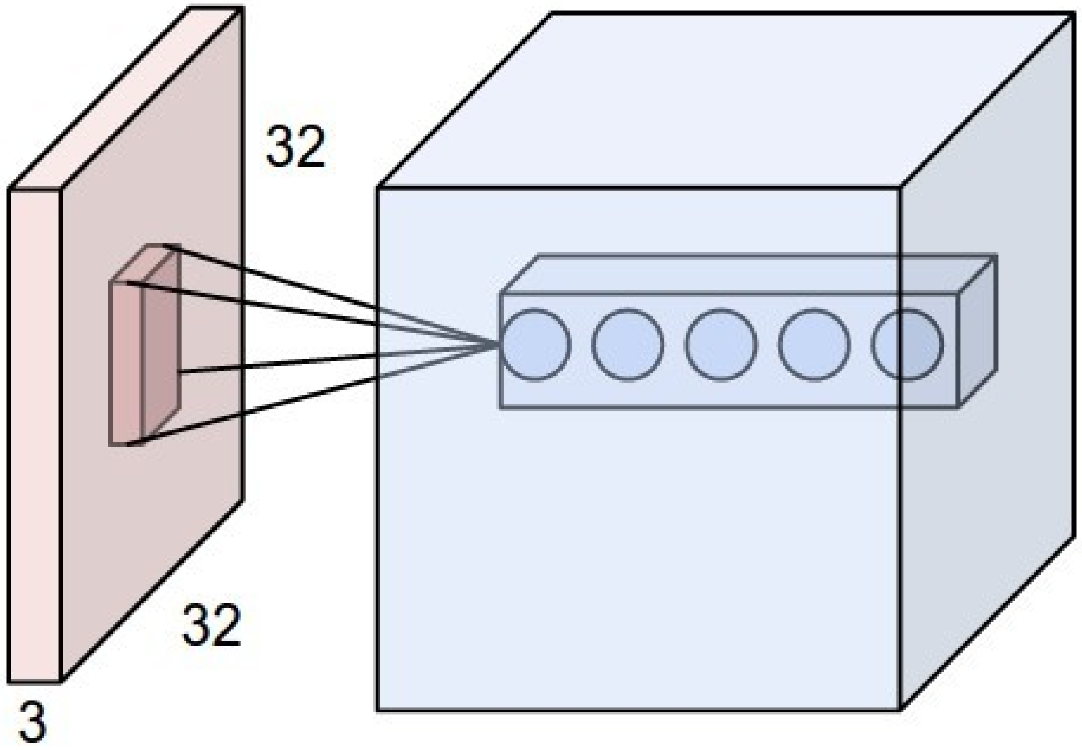


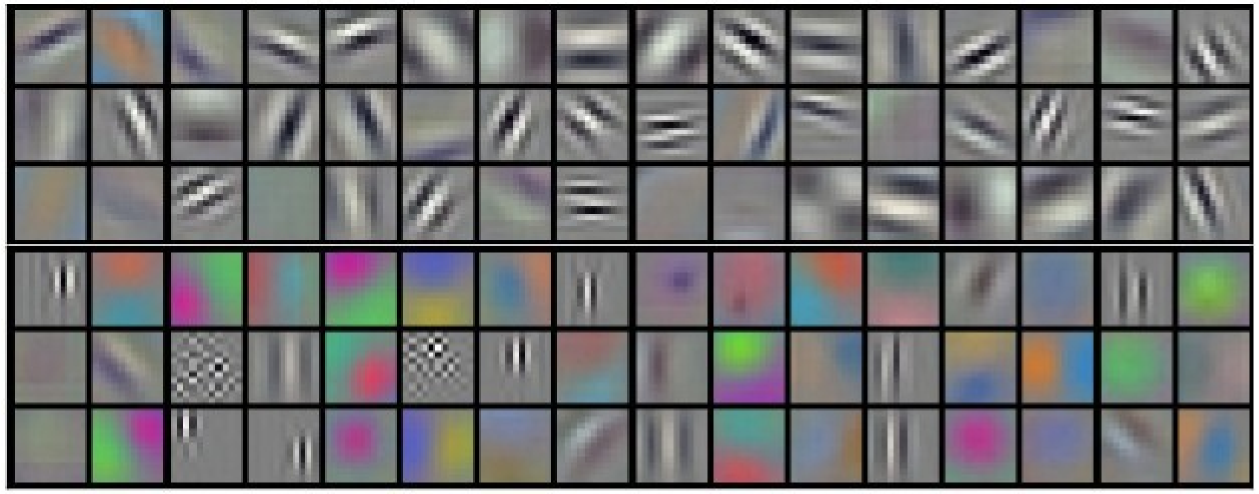
Activation Function

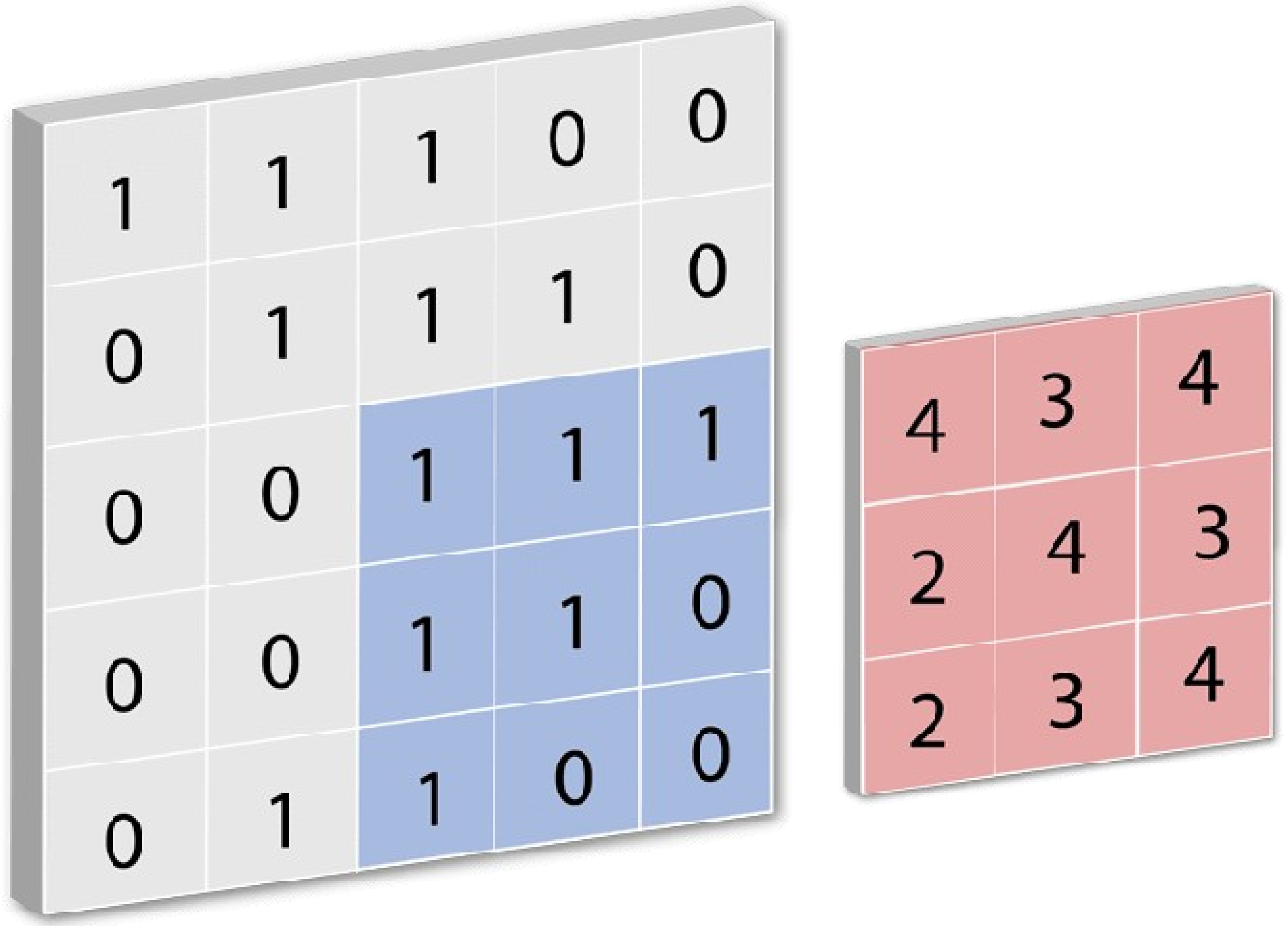


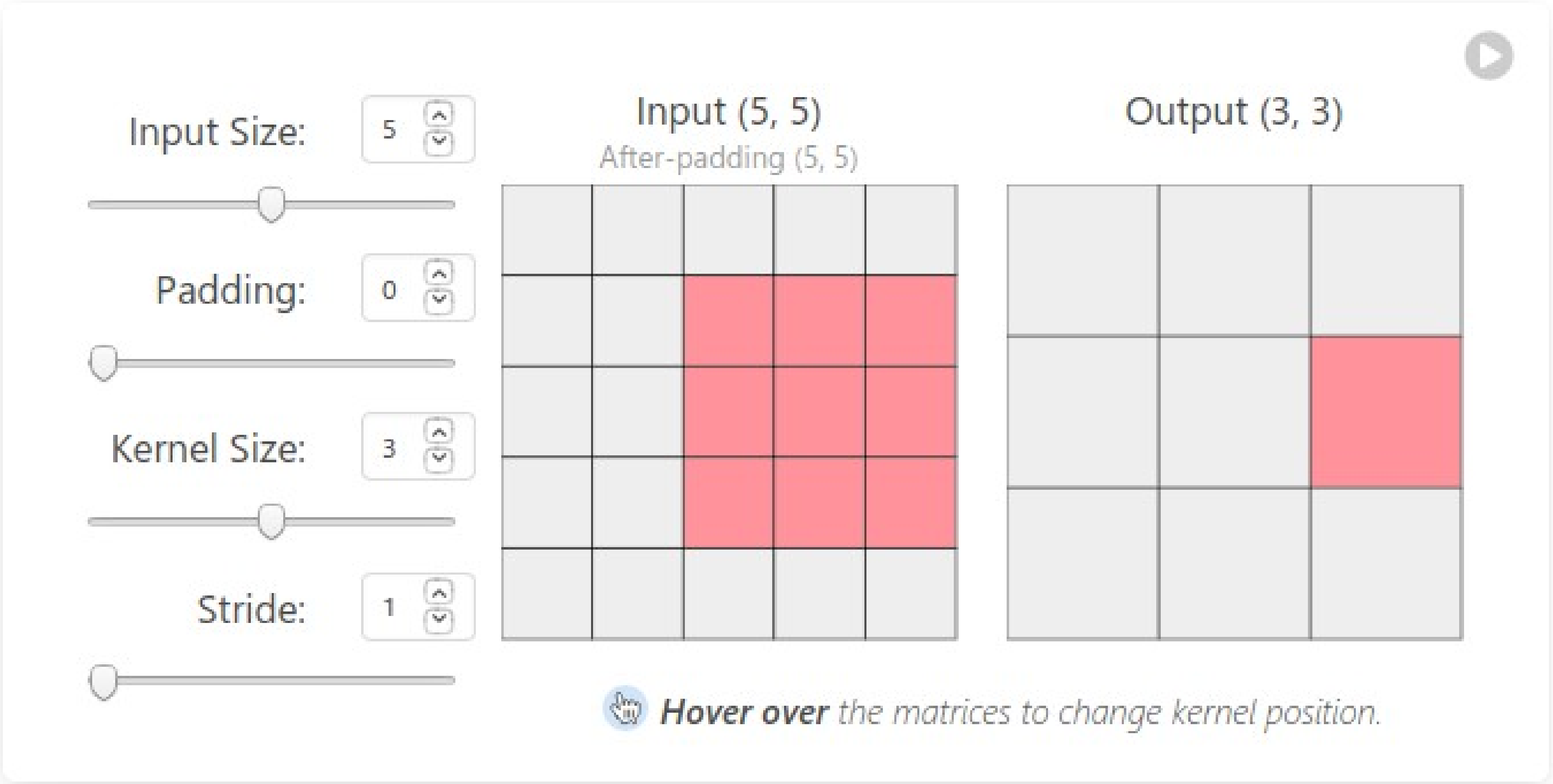
Normalization

$$\hat{x}_{i,j} = \frac{x_{i,j} - \mu_j}{\sqrt{\sigma_j^2 + \varepsilon}}$$


32
32
3




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
4 3 4
2 4 3
2 3 4

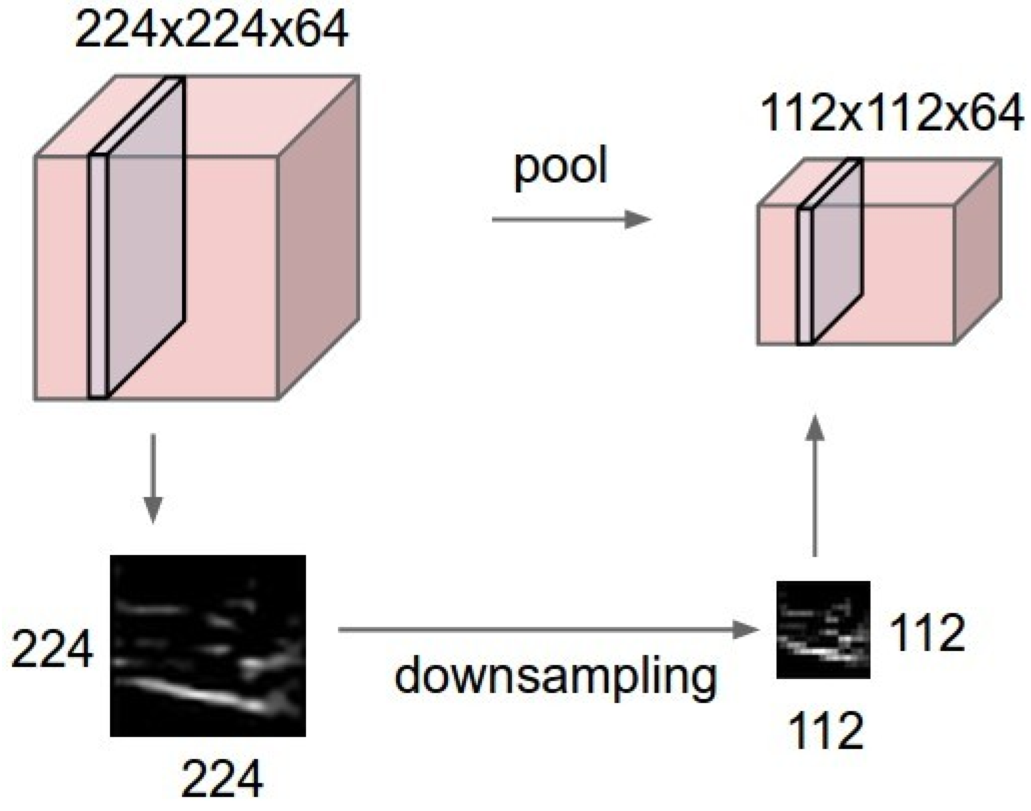

Input Size: 5

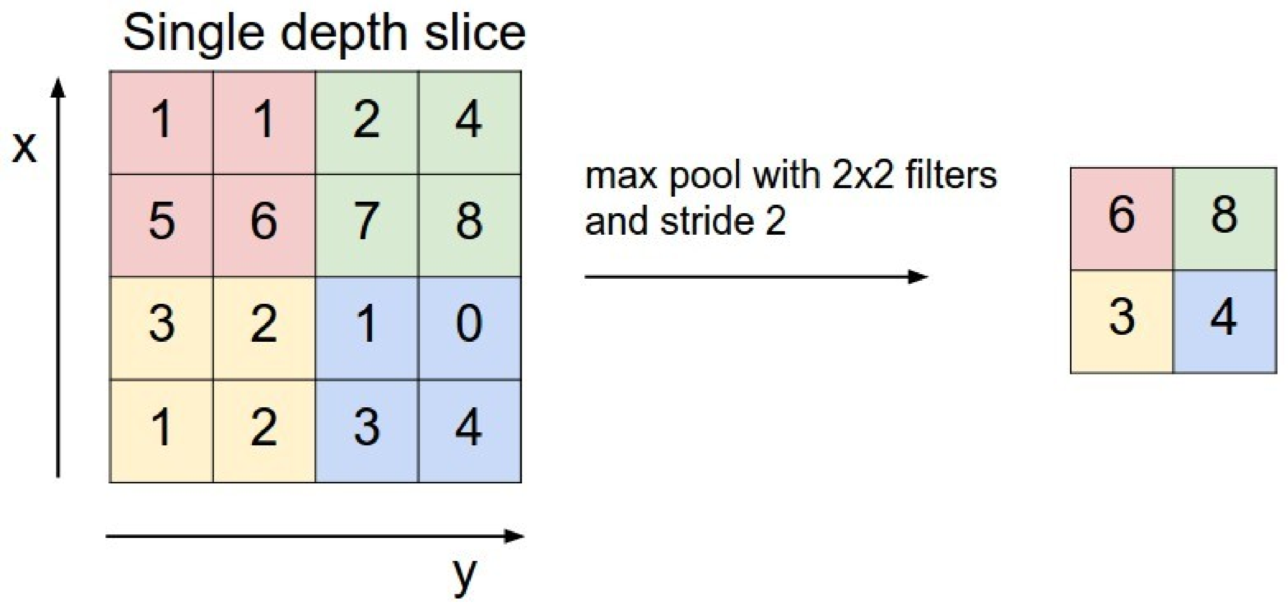
Padding: 0

Kernel Size: 3

Stride: 1

***Hover over** the matrices to change kernel position.*

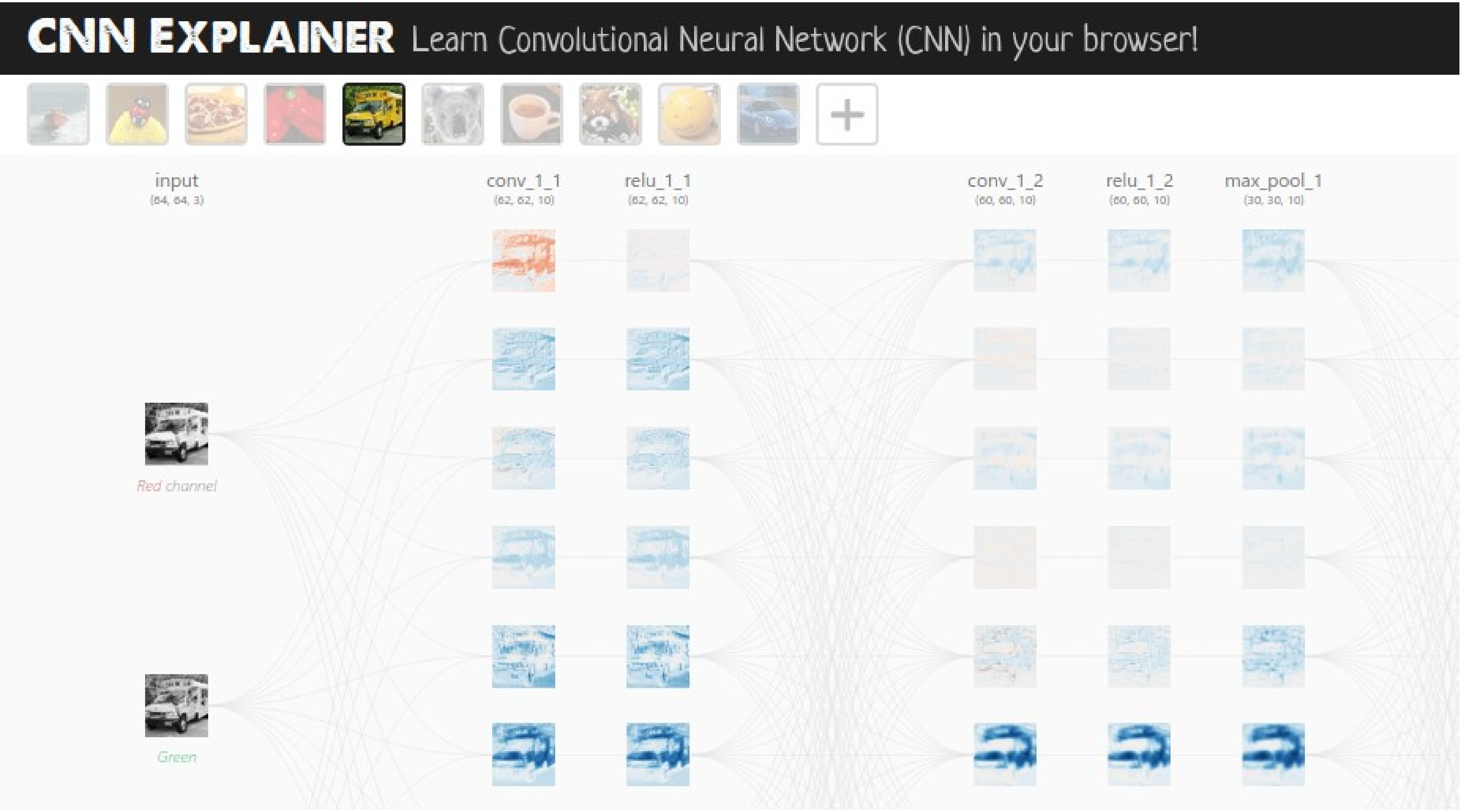

224x224x64
pool
112x112x64
224
224
downsampling
112
112


Single depth slice
x
y
1 1 2 4
5 6 7 8
3 2 1 0
1 2 3 4
max pool with 2x2 filters
and stride 2
6 8
3 4

Keras CNN

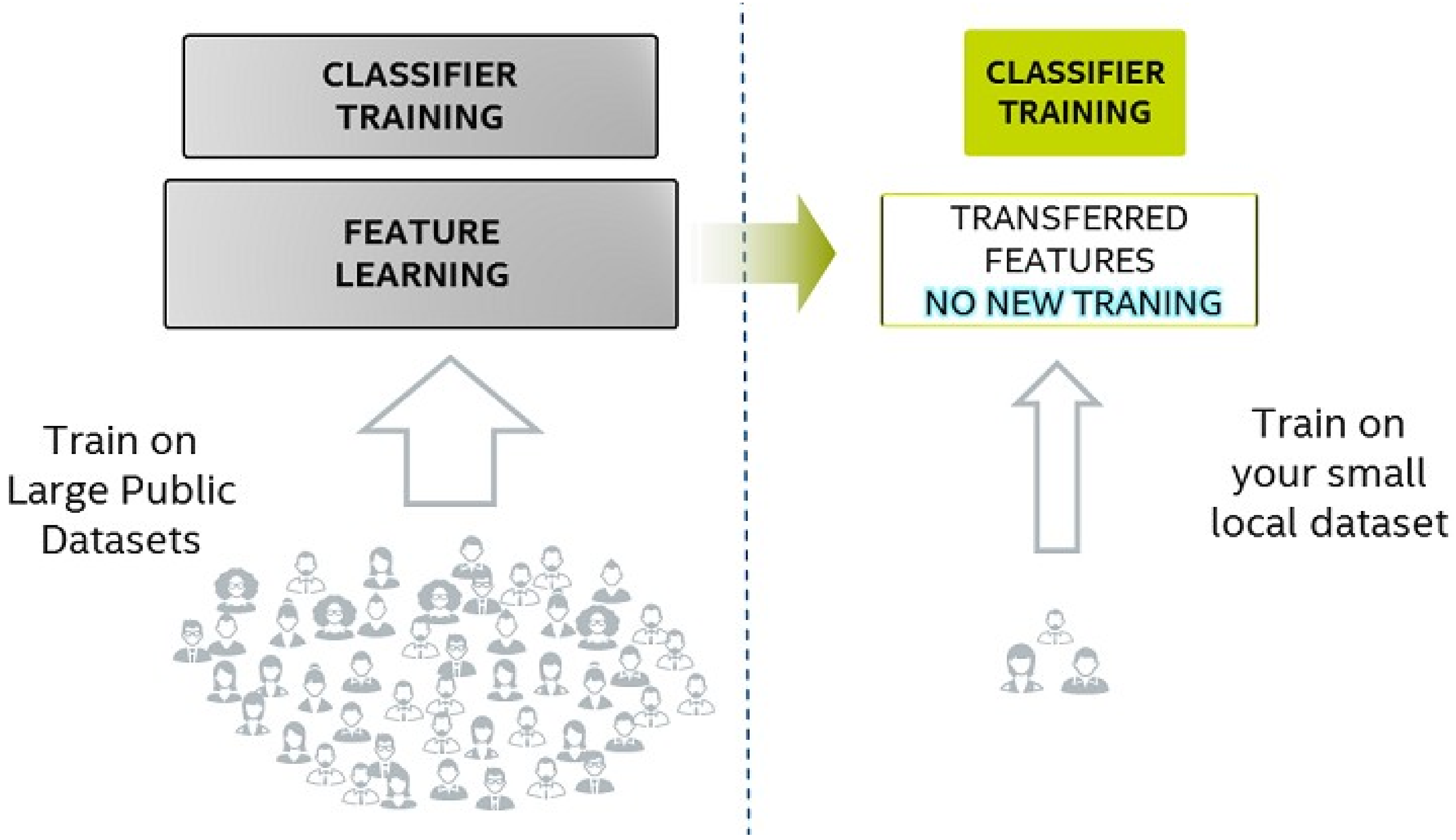
```
from keras import layers
from keras import models

model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
```



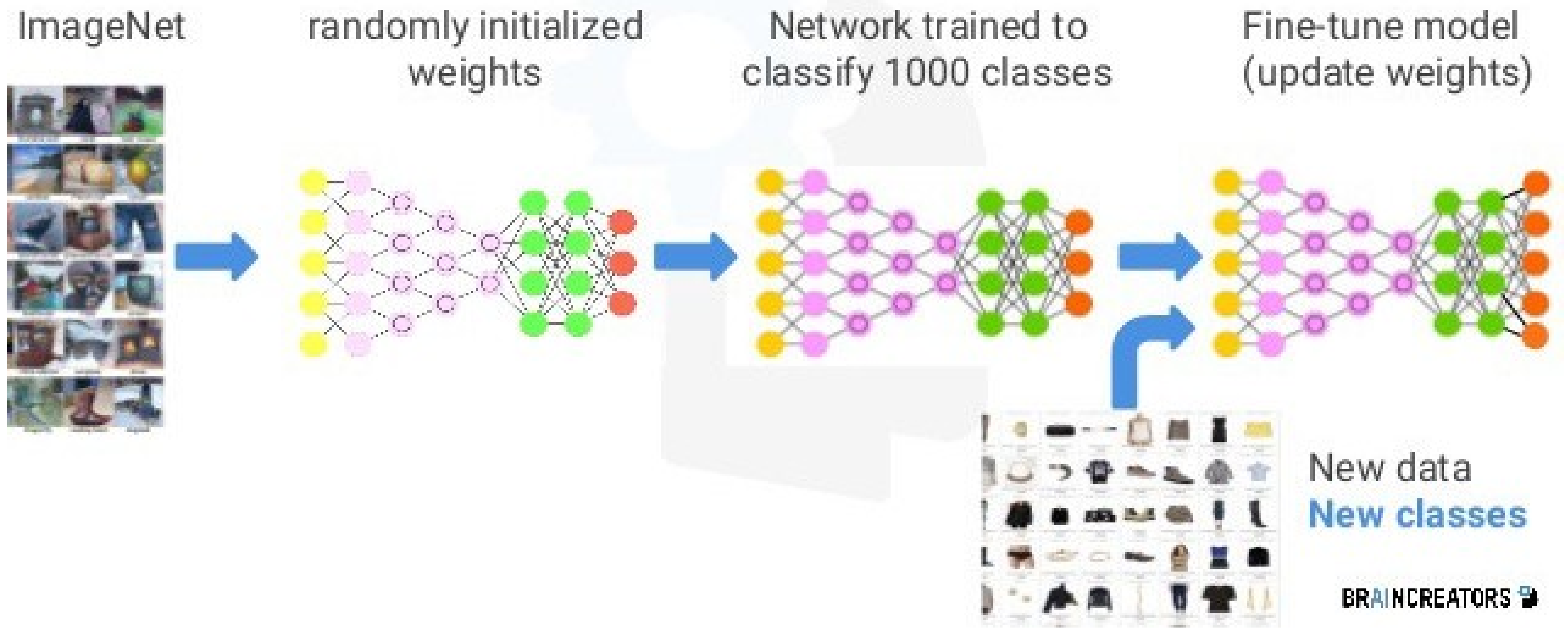
https://poloclub.github.io/cnn-explainer/

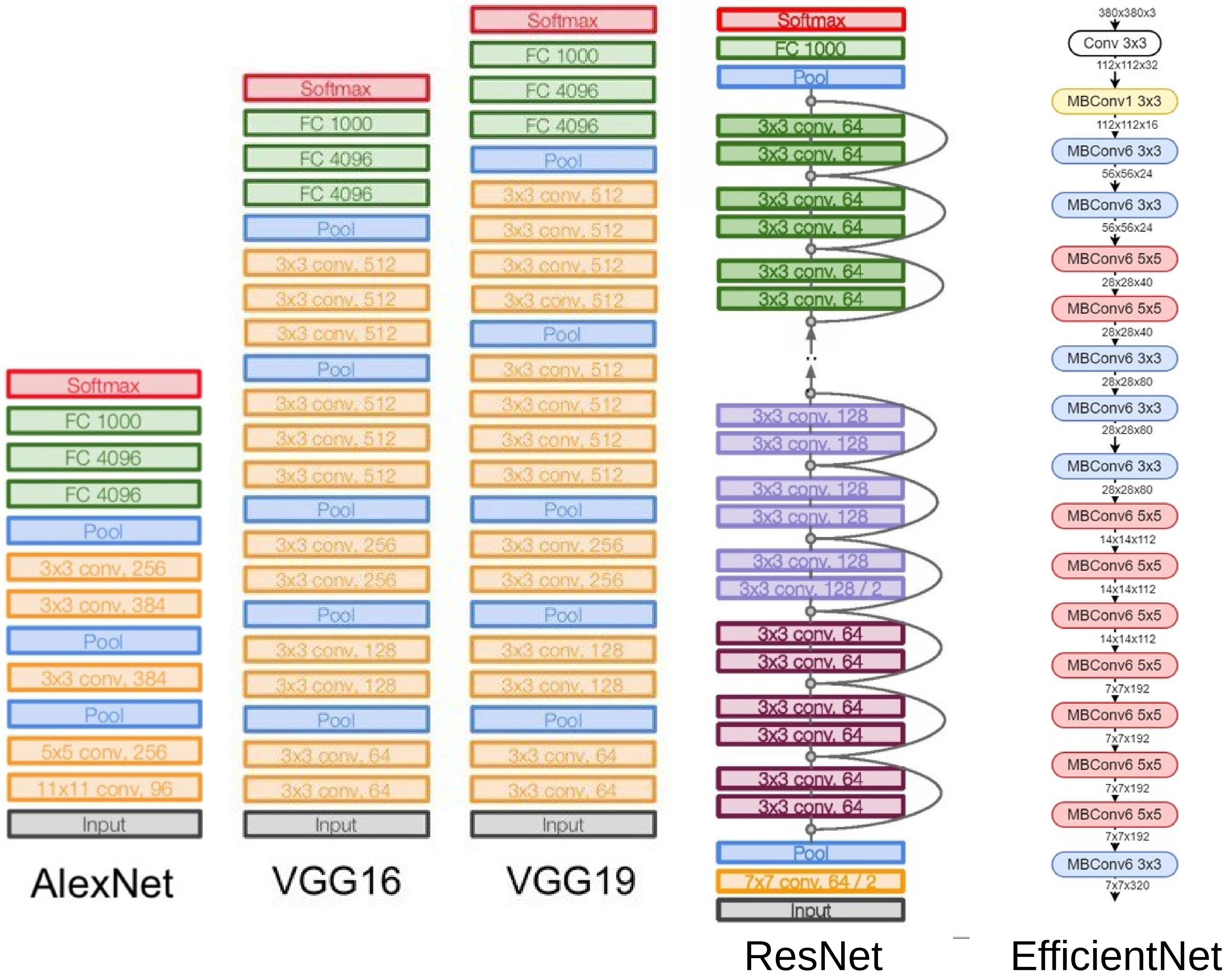
Transfer learning



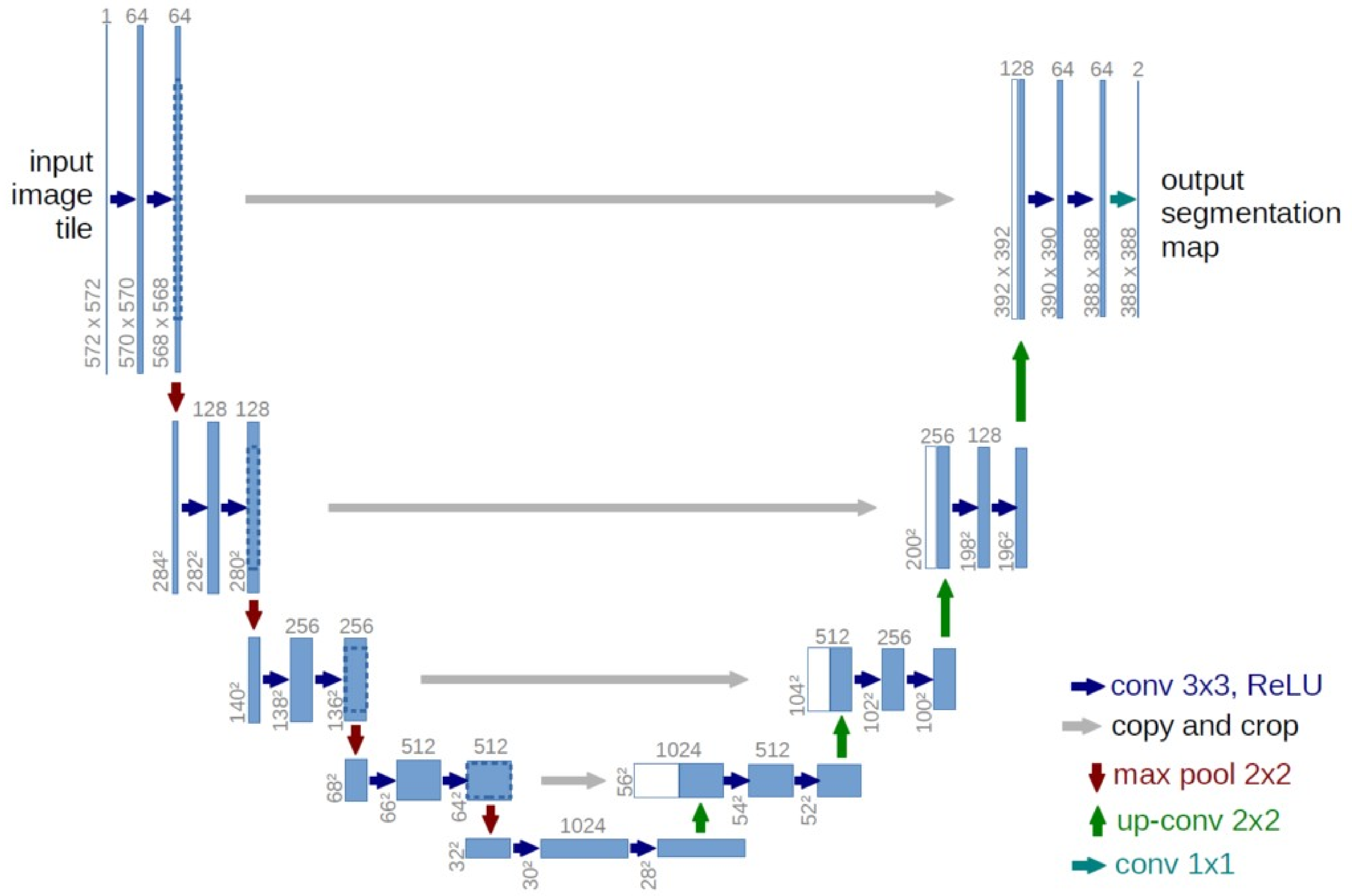
IMAGENET

Transfer learning


ImageNet
randomly initialized weights
Network trained to classify 1000 classes
Fine-tune model (update weights)
New data
New classes
BRAINCREATORS

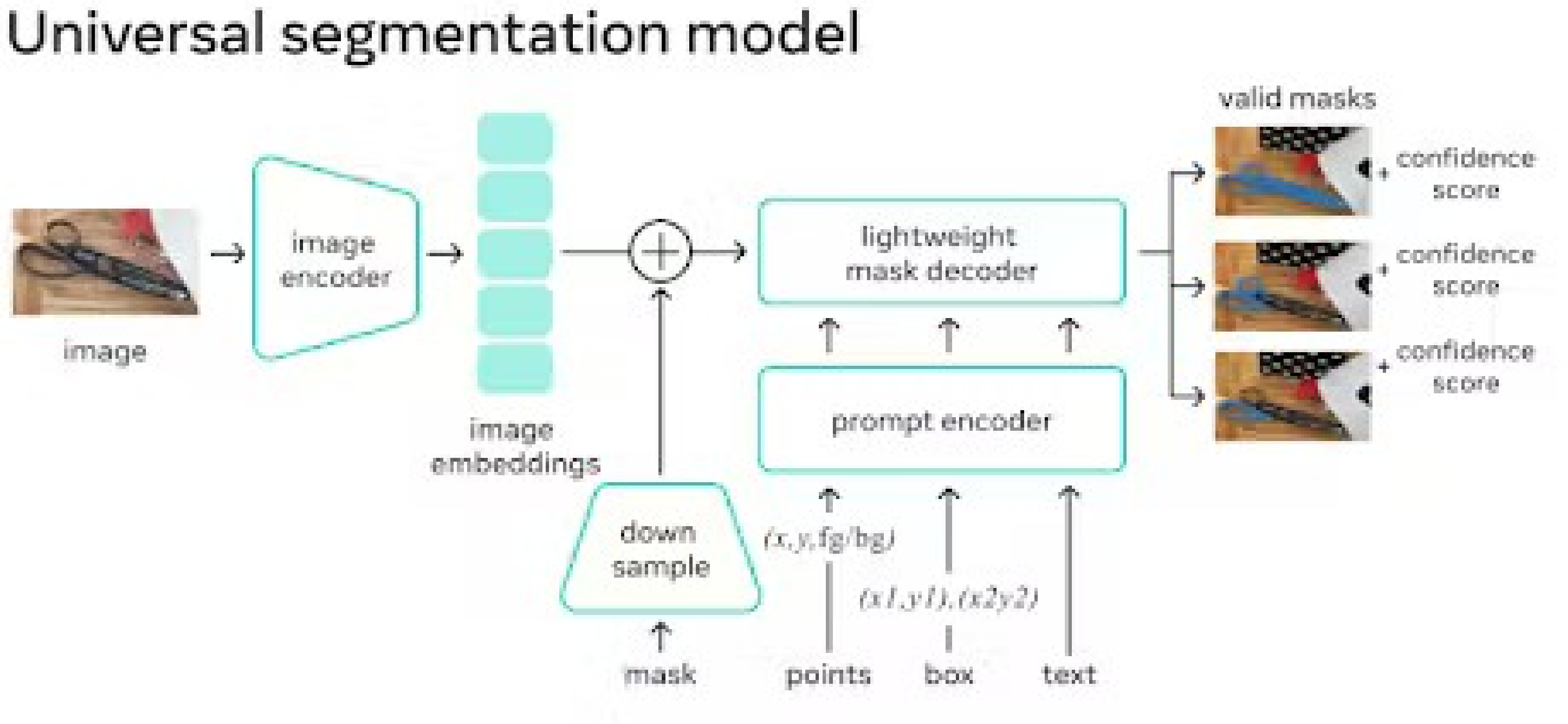

Softmax
FC 1000
FC 4096
FC 4096
Pool
3x3 conv, 256
3x3 conv, 384
Pool
3x3 conv, 384
Pool
5x5 conv, 256
11x11 conv, 96
Input
AlexNet
Softmax
FC 1000
FC 4096
FC 4096
Pool
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
Pool
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
Pool
3x3 conv, 256
3x3 conv, 256
Pool
3x3 conv, 128
3x3 conv, 128
Pool
3x3 conv, 64
3x3 conv, 64
Input
VGG16
Softmax
FC 1000
FC 4096
FC 4096
Pool
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
Pool
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
Pool
3x3 conv, 256
3x3 conv, 256
Pool
3x3 conv, 128
3x3 conv, 128
Pool
3x3 conv, 64
3x3 conv, 64
Input
VGG19
Softmax
FC 1000
Pool
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 128
3x3 conv, 128
3x3 conv, 128
3x3 conv, 128
3x3 conv, 128
3x3 conv, 128 / 2
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
Pool
7x7 conv, 64 / 2
Input
ResNet
380x380x3
Conv 3x3
112x112x32
MBConv1 3x3
112x112x16
MBConv6 3x3
56x56x24
MBConv6 3x3
56x56x24
MBConv6 5x5
28x28x40
MBConv6 5x5
28x28x40
MBConv6 3x3
28x28x80
MBConv6 3x3
28x28x80
MBConv6 3x3
28x28x80
MBConv6 5x5
14x14x112
MBConv6 5x5
14x14x112
MBConv6 5x5
14x14x112
MBConv6 5x5
7x7x192
MBConv6 5x5
7x7x192
MBConv6 5x5
7x7x192
MBConv6 5x5
7x7x192
MBConv6 3x3
7x7x320
EfficientNet

U-Net



1
64
64
input image tile
572 x 572
570 x 570
568 x 568
128
128
284²
282²
280²
256
256
140²
138²
136²
512
512
68²
66²
64²
1024
32²
30²
28²
1024
512
56²
54²
52²
512
256
104²
102²
100²
256
128
200²
198²
196²
128
64
64
2
392 x 392
390 x 390
388 x 388
388 x 388
output segmentation map
conv 3x3, ReLU
copy and crop
max pool 2x2
up-conv 2x2
conv 1x1

Segment anything



https://samgeo.gishub.org/