DIGITAL TALENT SCHOLARSHIP 2019

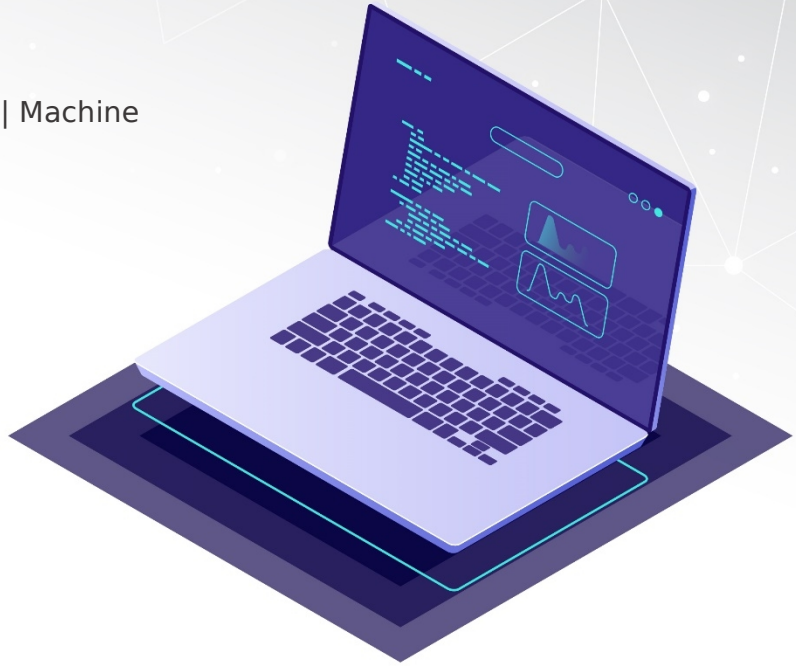
DIGITAL TALENT SCHOLARSHIP

digitalent.kominfo.go.id

Program Fresh Graduate Academy Digital Talent Scholarship 2019 | Machine Learning

Deep Learning: Convolutional Neural Networks

Nama pembicara dengan gelar

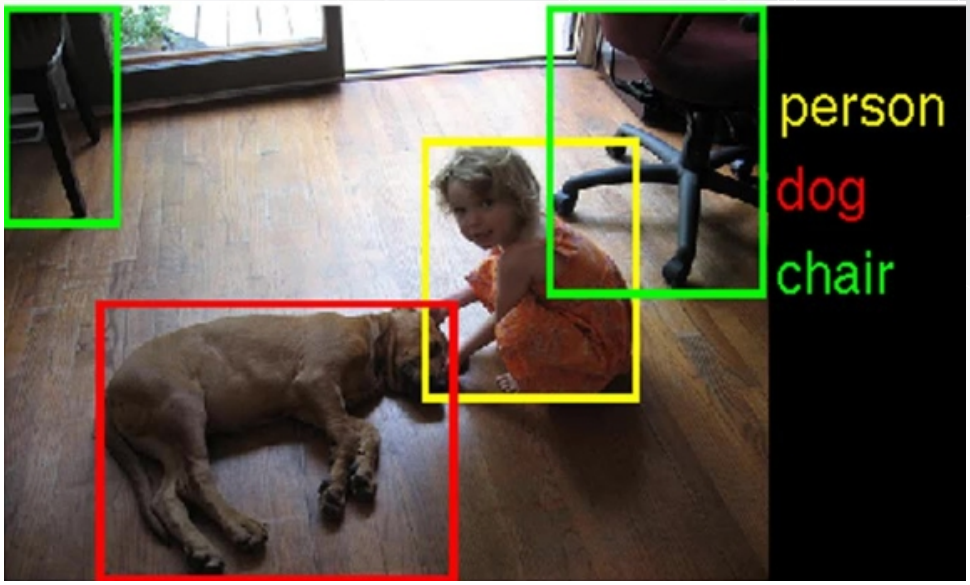


Bagian 1

Introduction to Convolutional Networks

Applications

- Signal and image processing
- Handwritten text/digits recognition
- Natural object classification (photos and videos)
- Segmentation
- Face detection
- Recommender systems
- Speech recognition
- Natural Language Processing



One of the CNN applications: Natural object classification

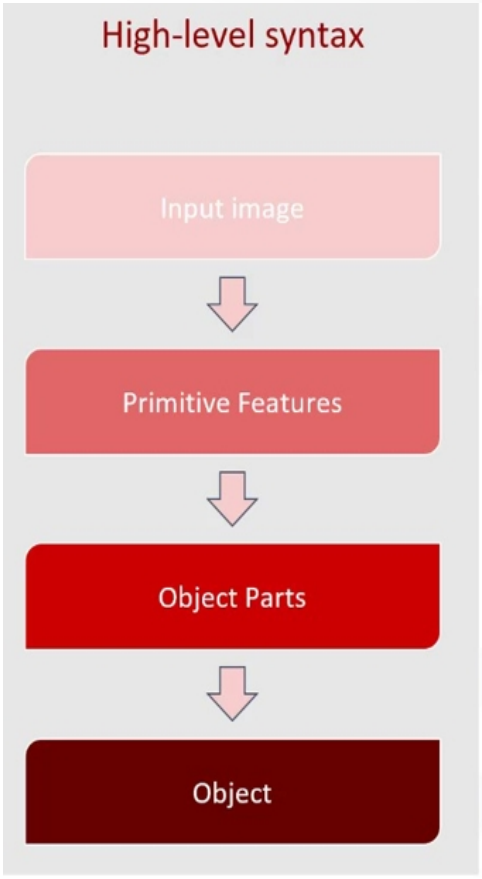
Original goal of CNN

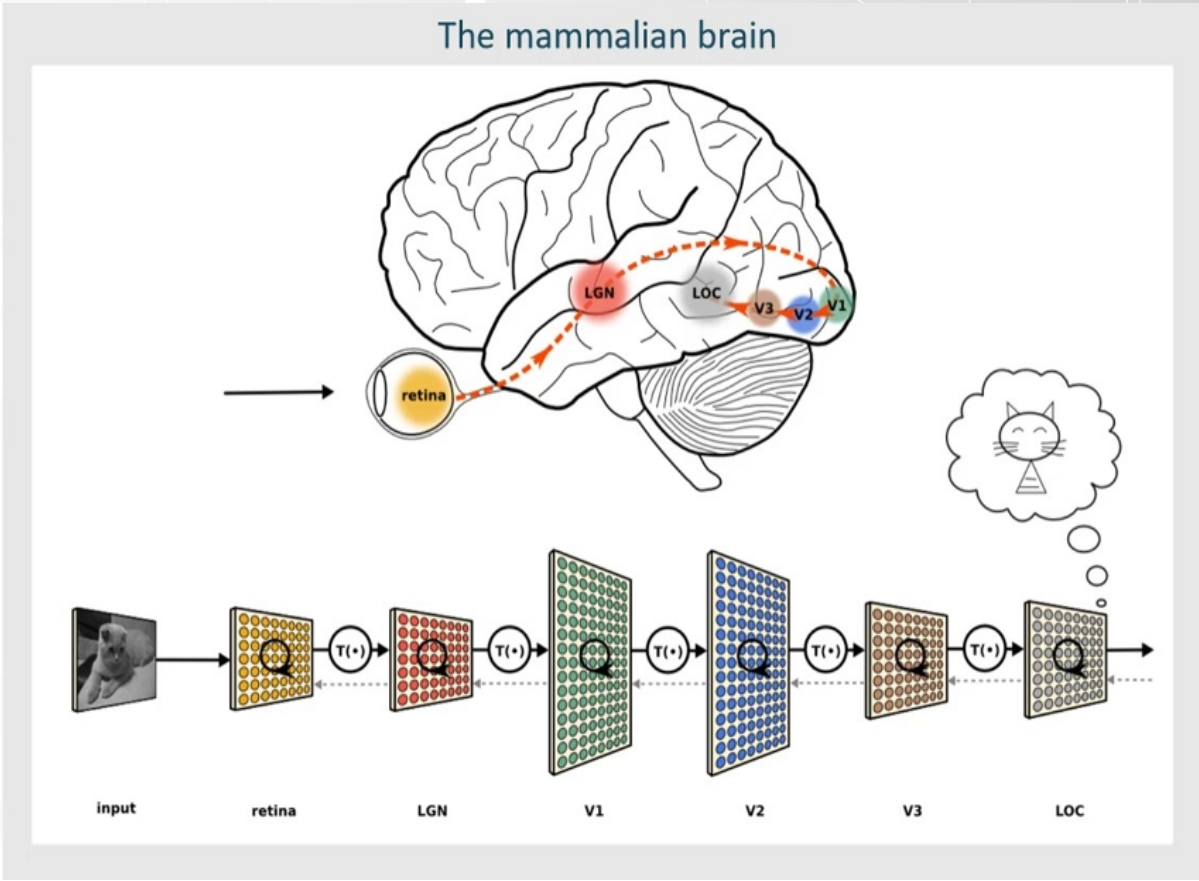
"How to create good representations of the visual world in a way it could be used to support recognition?"

Key features:

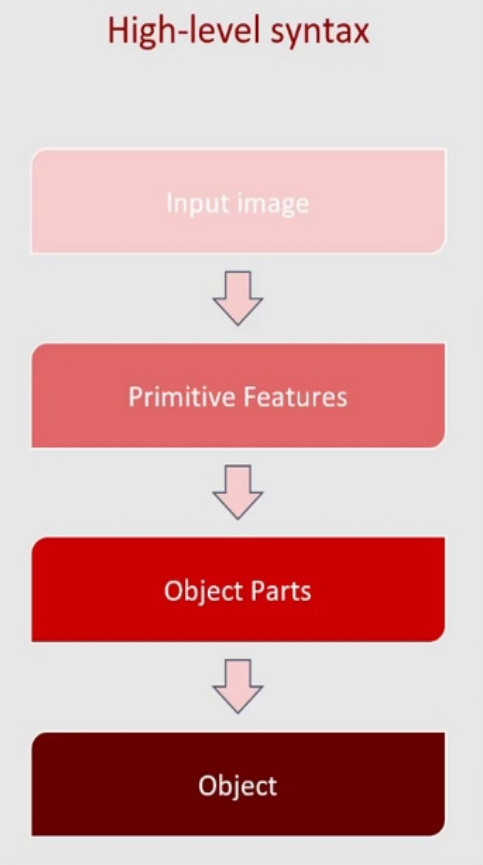
- **Detect and classify** objects into categories
- **Independence** from pose, scale, illumination, conformation and clutter

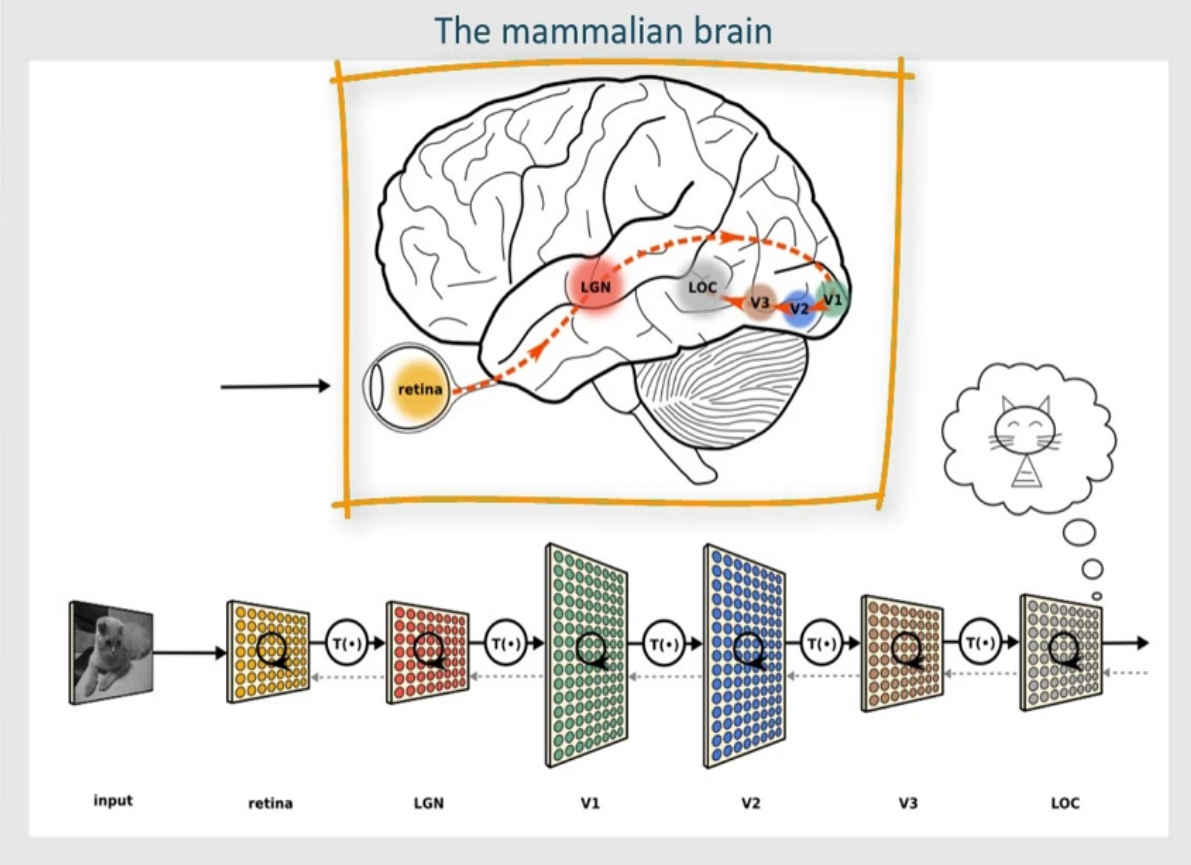
Inheritance from the real world



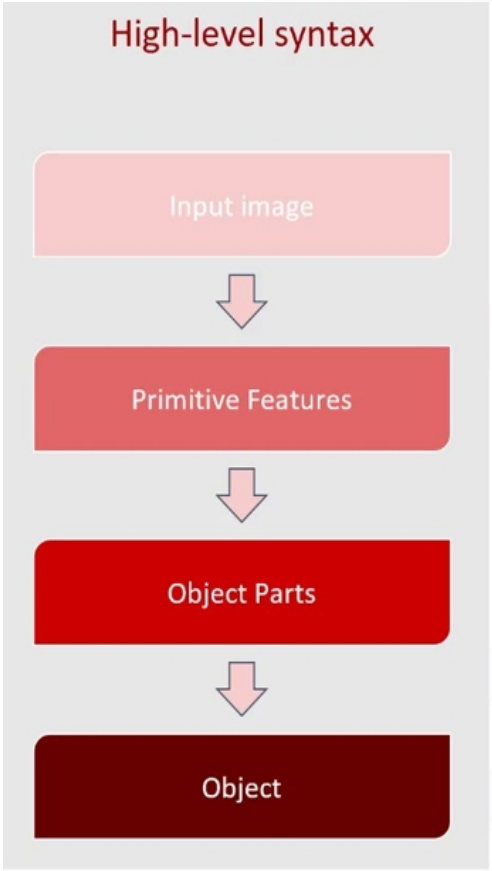


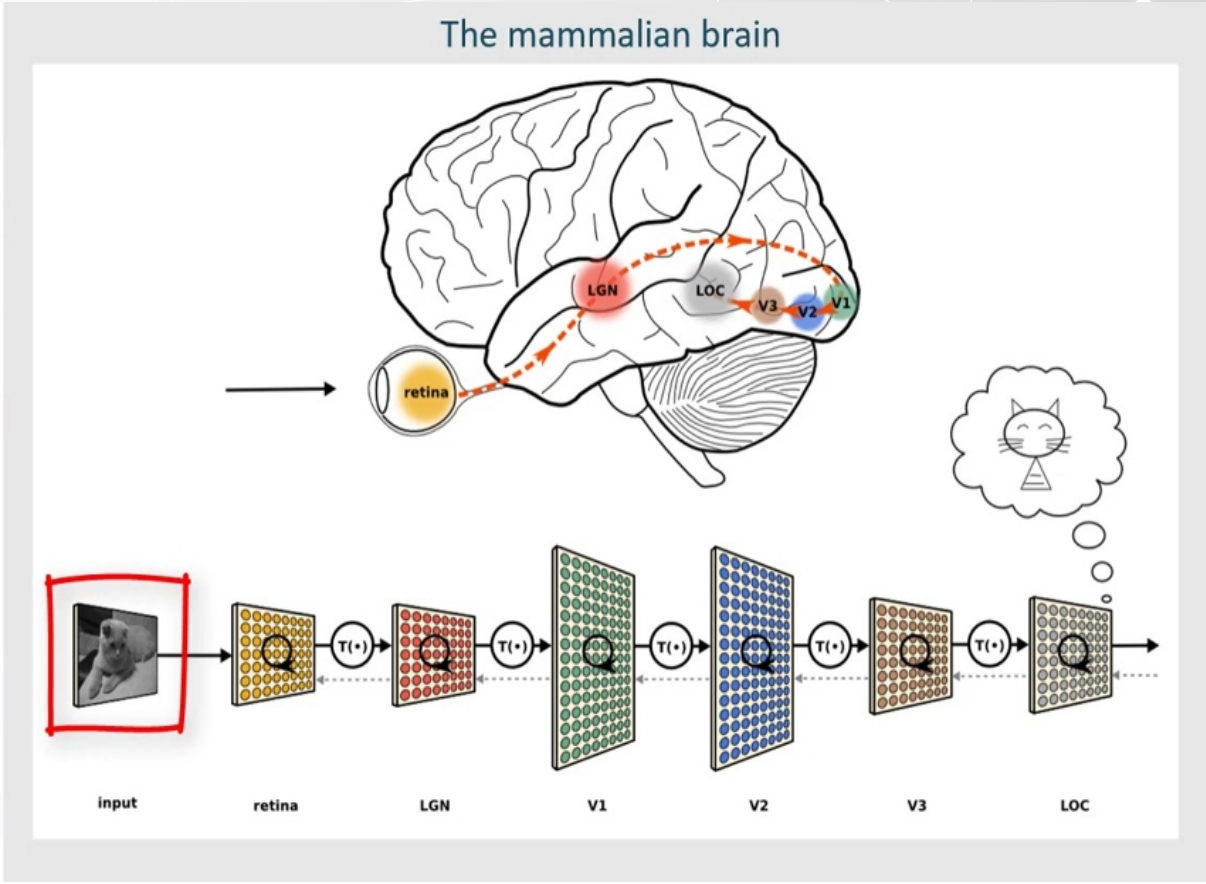
Inheritance from the real world



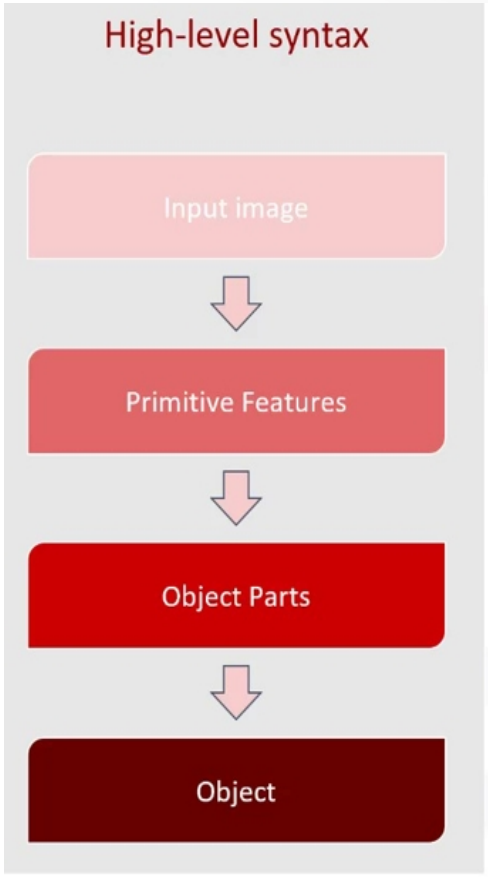


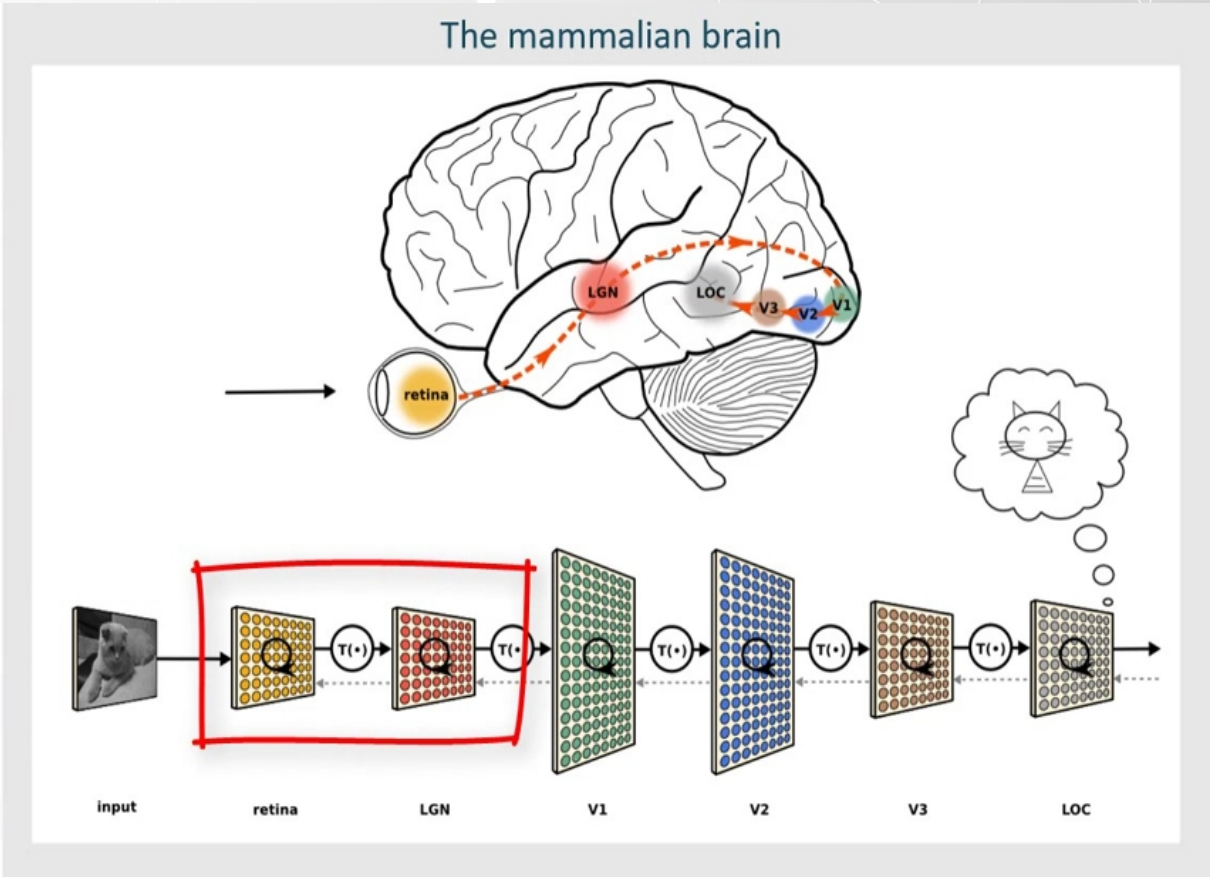
Inheritance from the real world



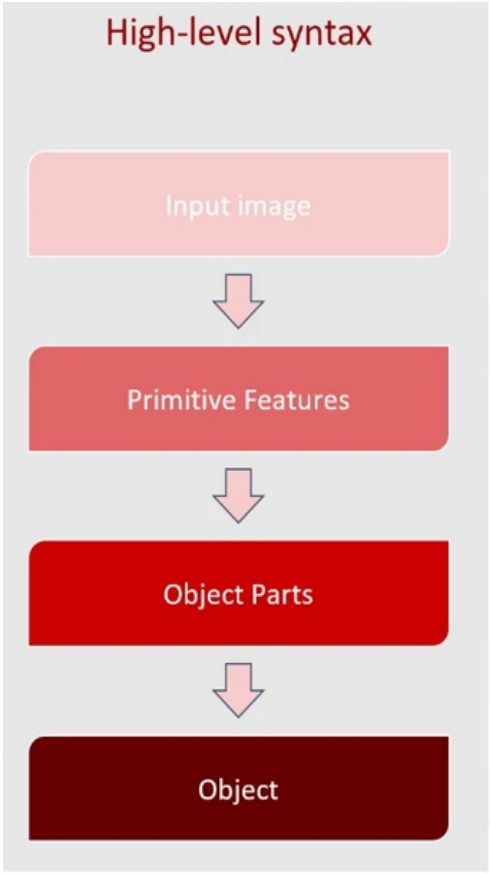


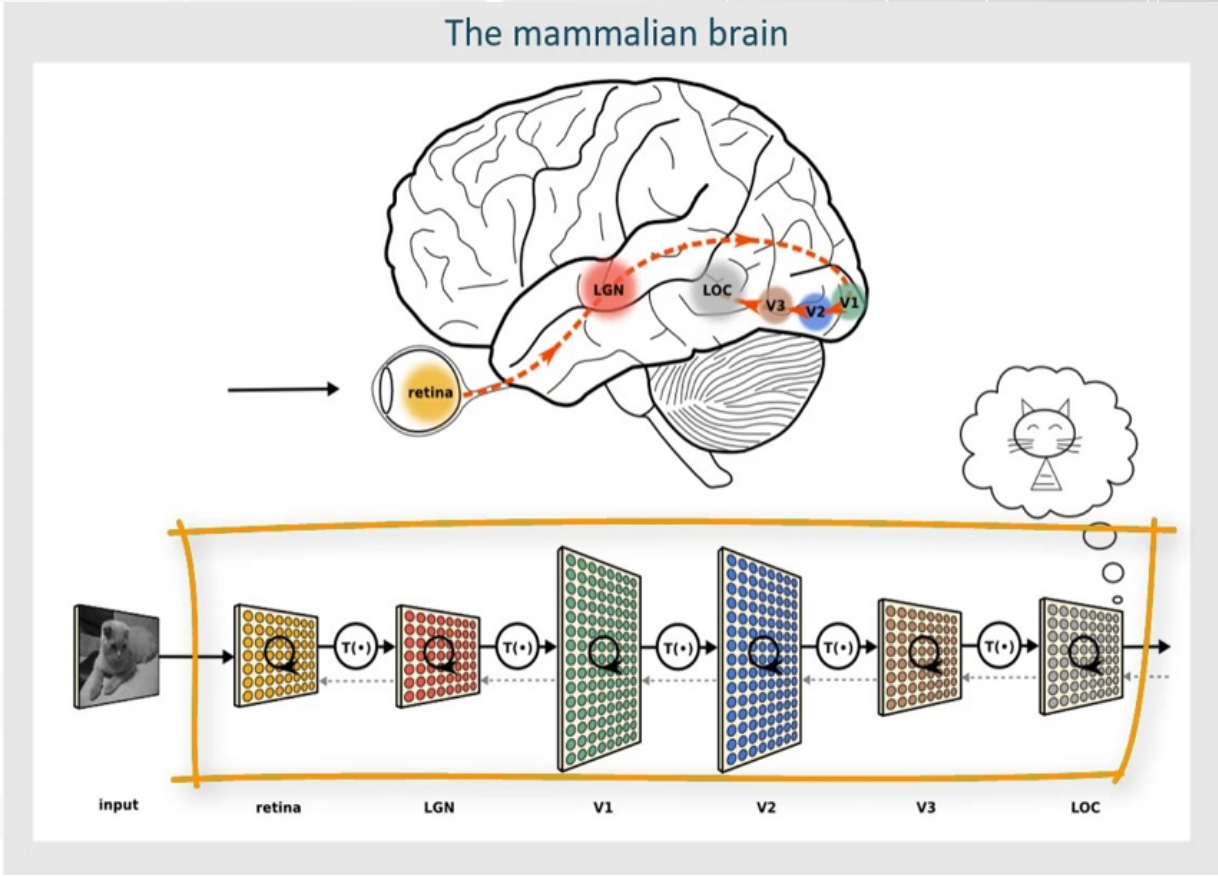
Inheritance from the real world



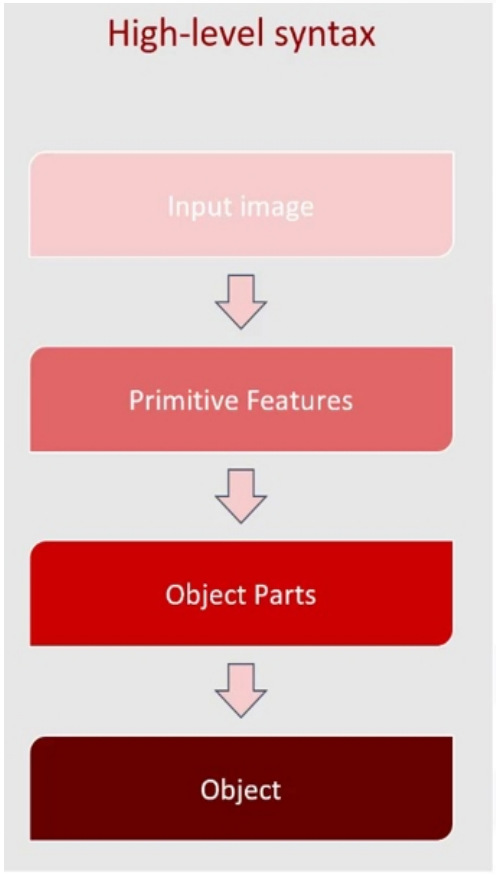


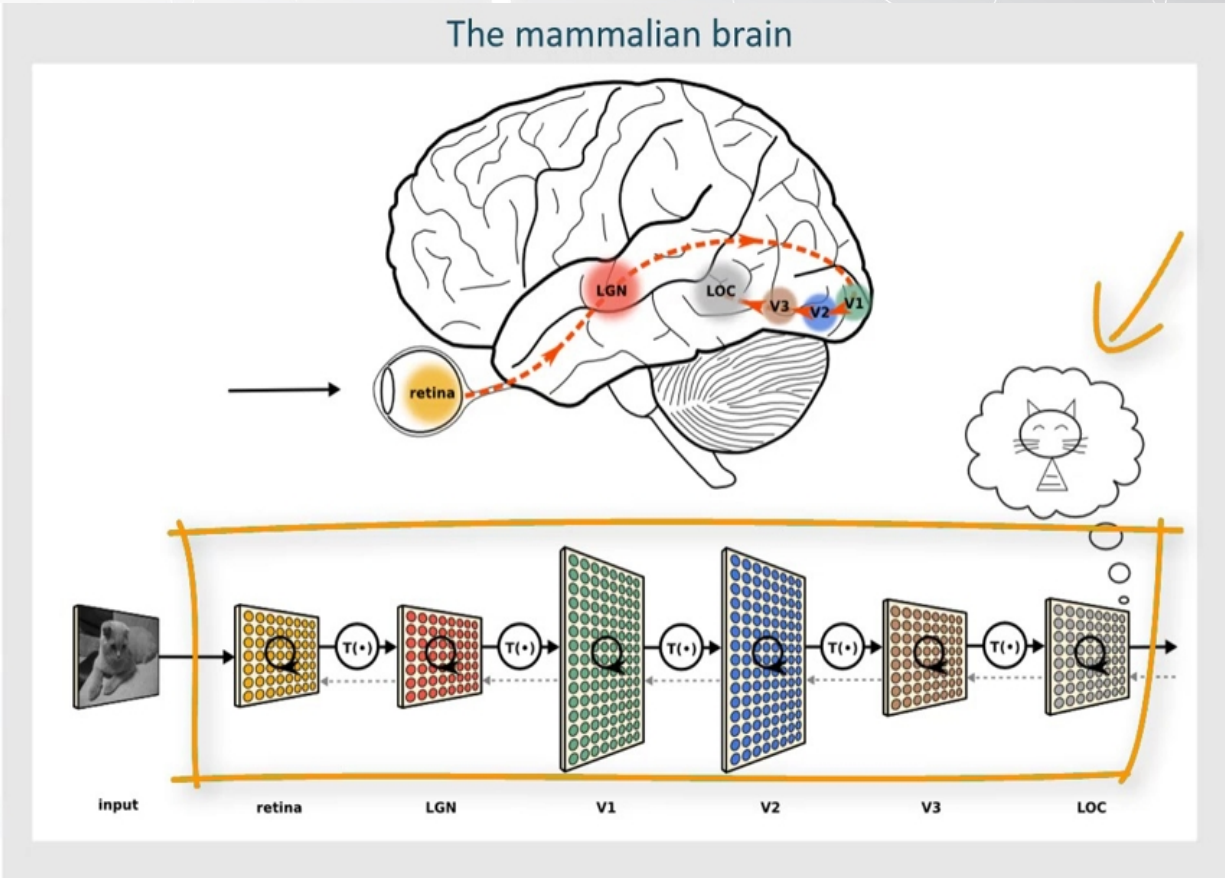
Inheritance from the real world



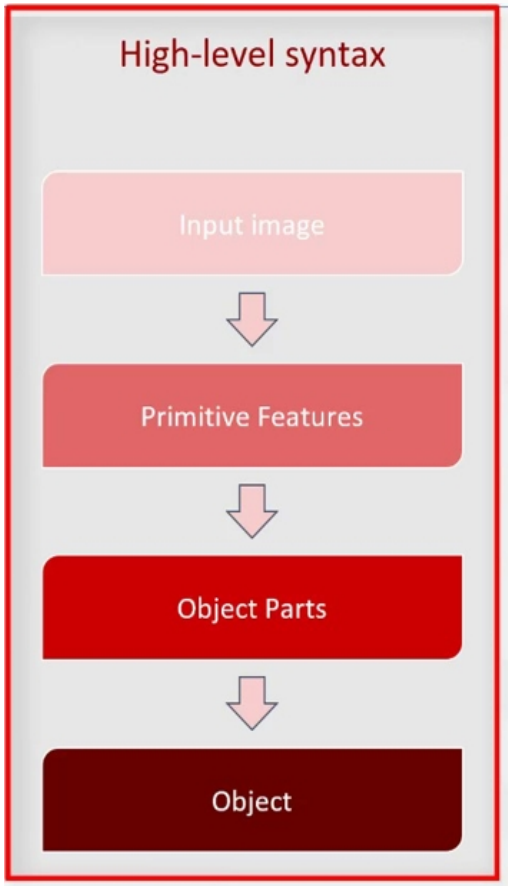


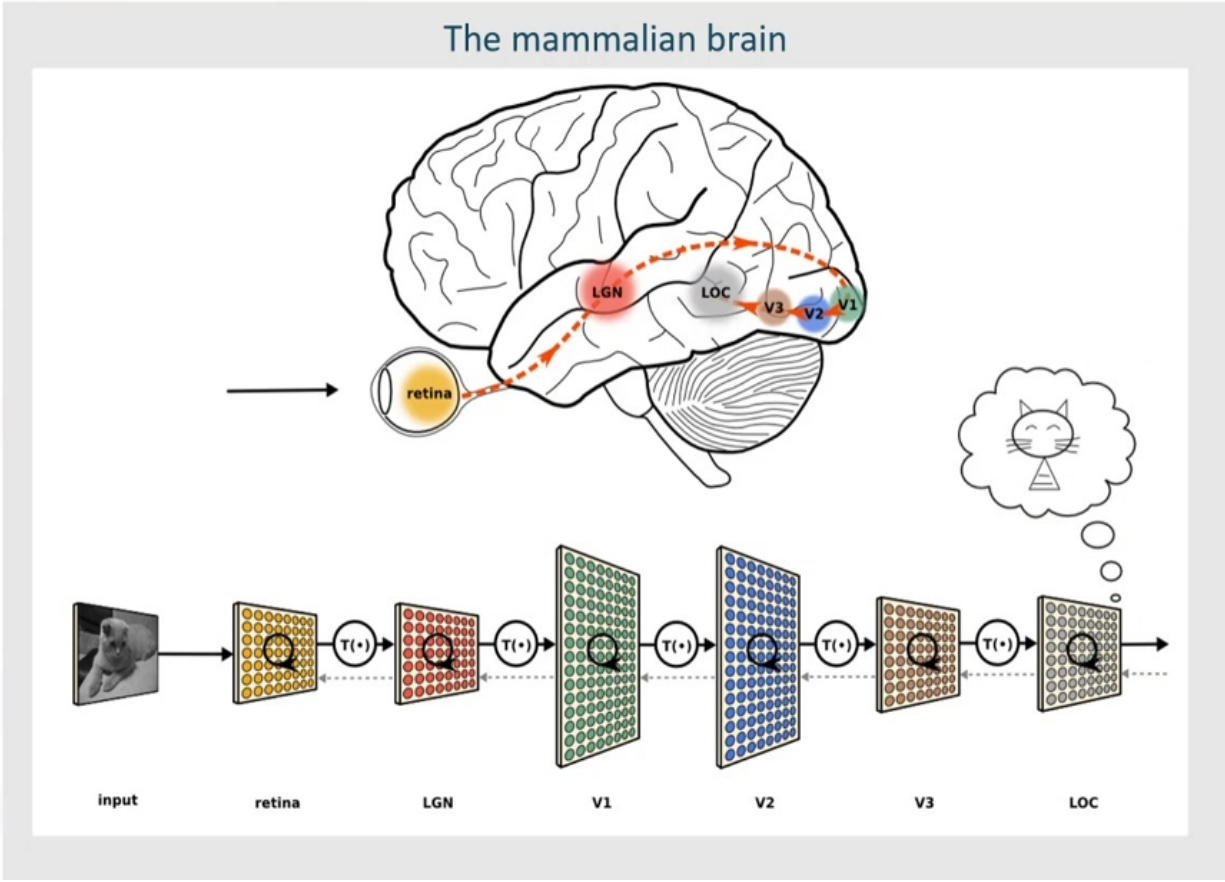
Inheritance from the real world



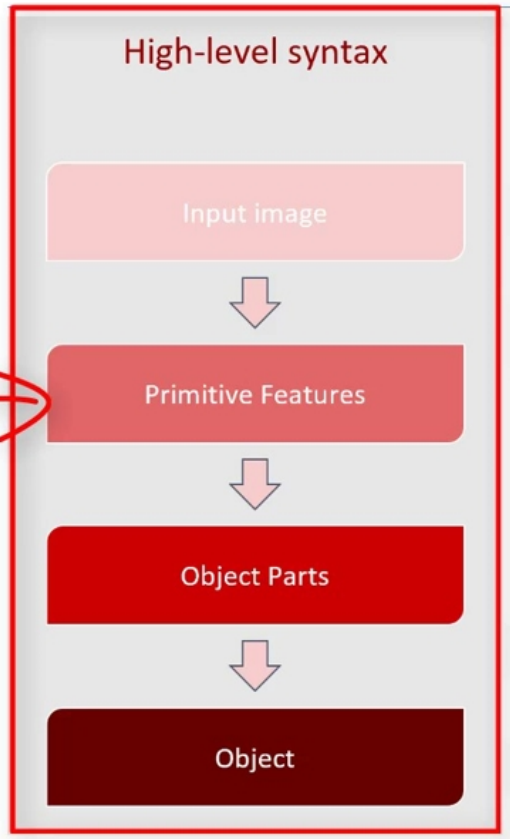


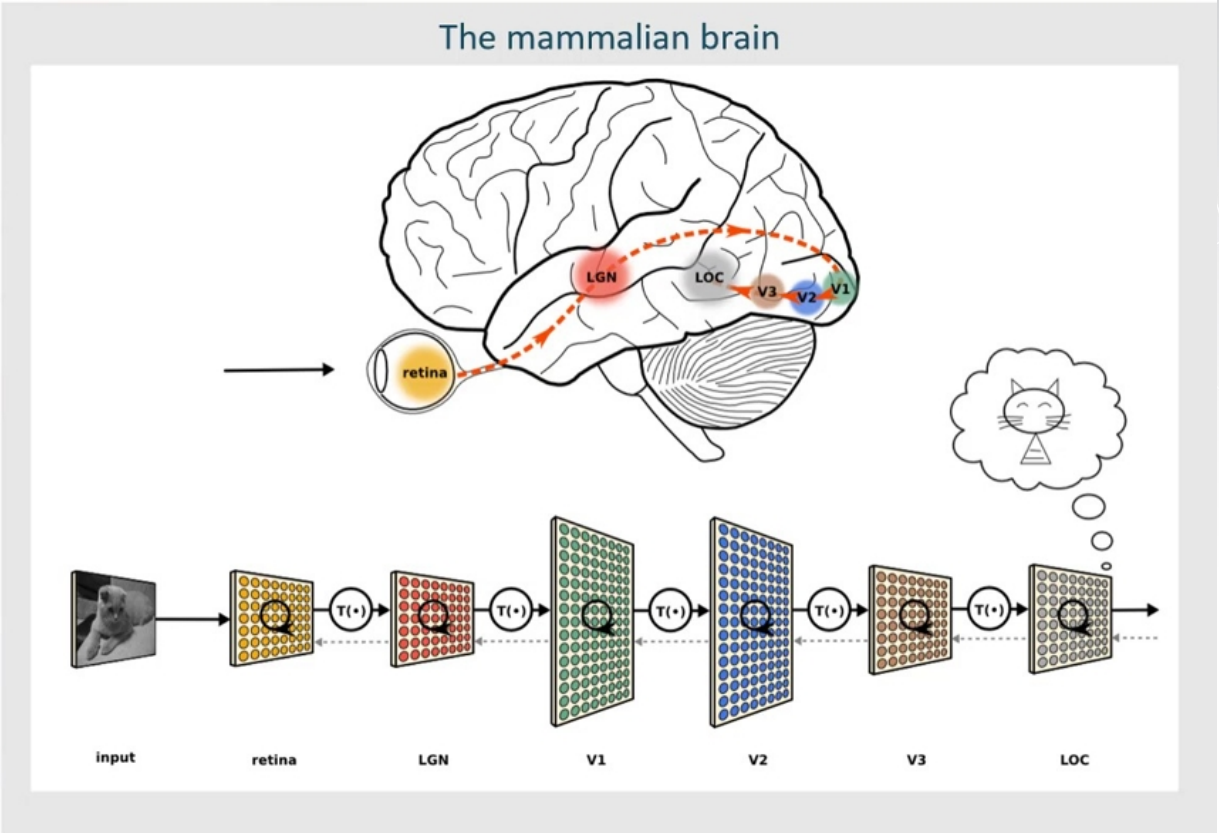
Inheritance from the real world



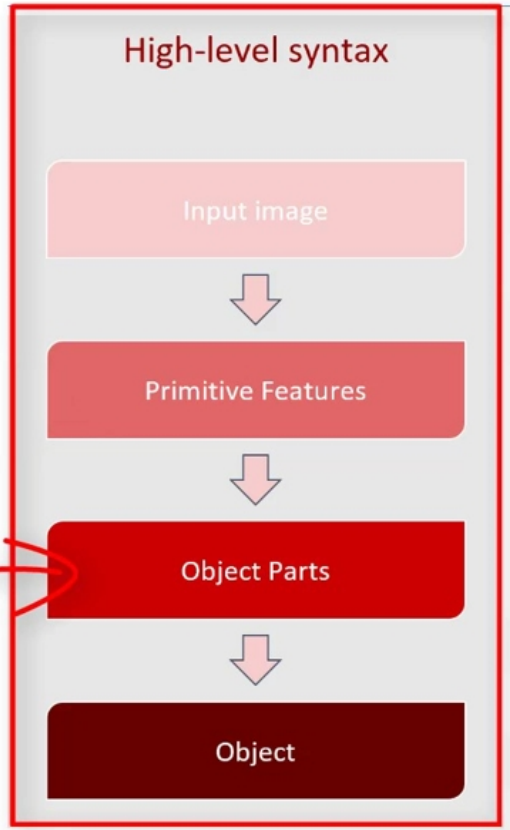


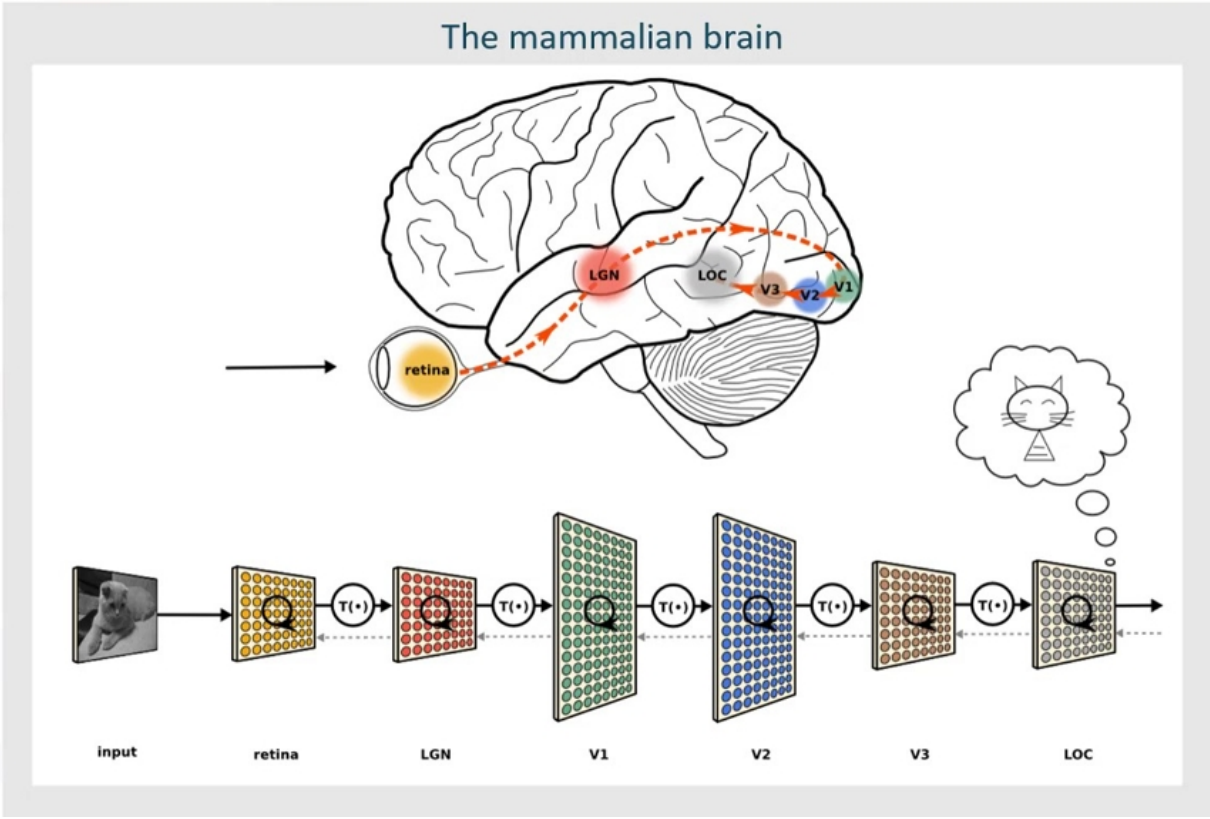
Inheritance from the real world



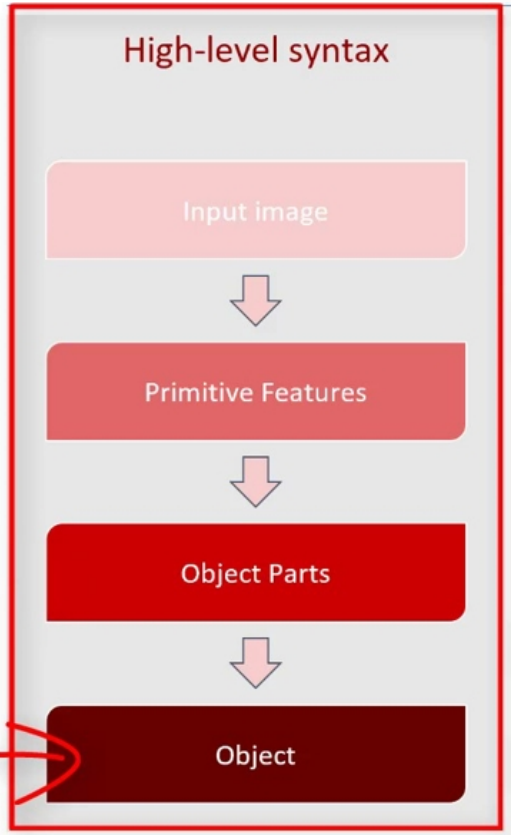


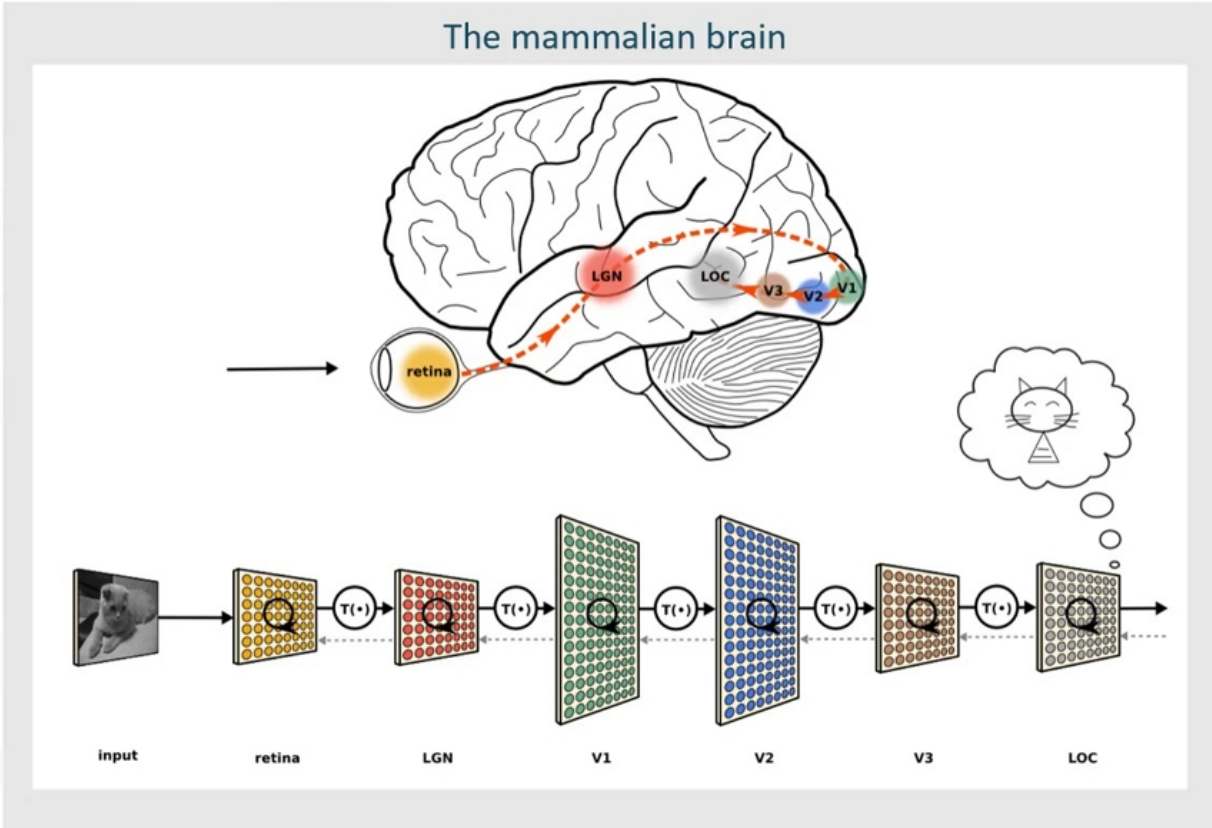
Inheritance from the real world



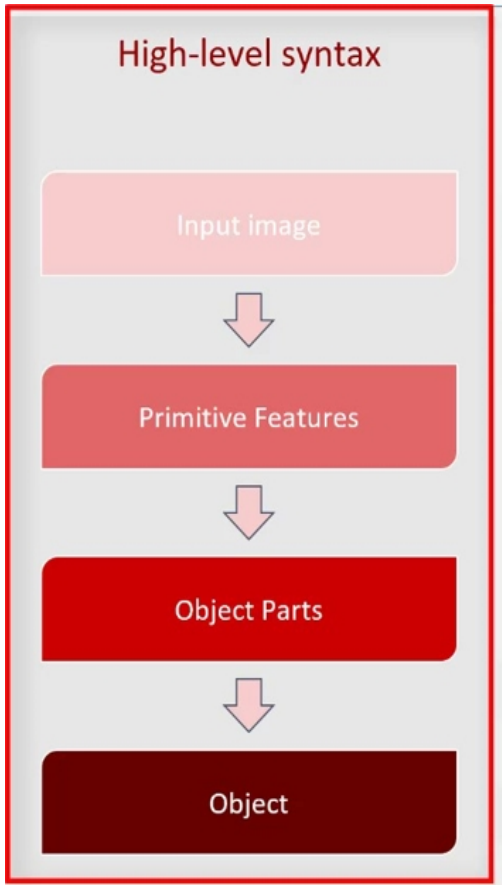


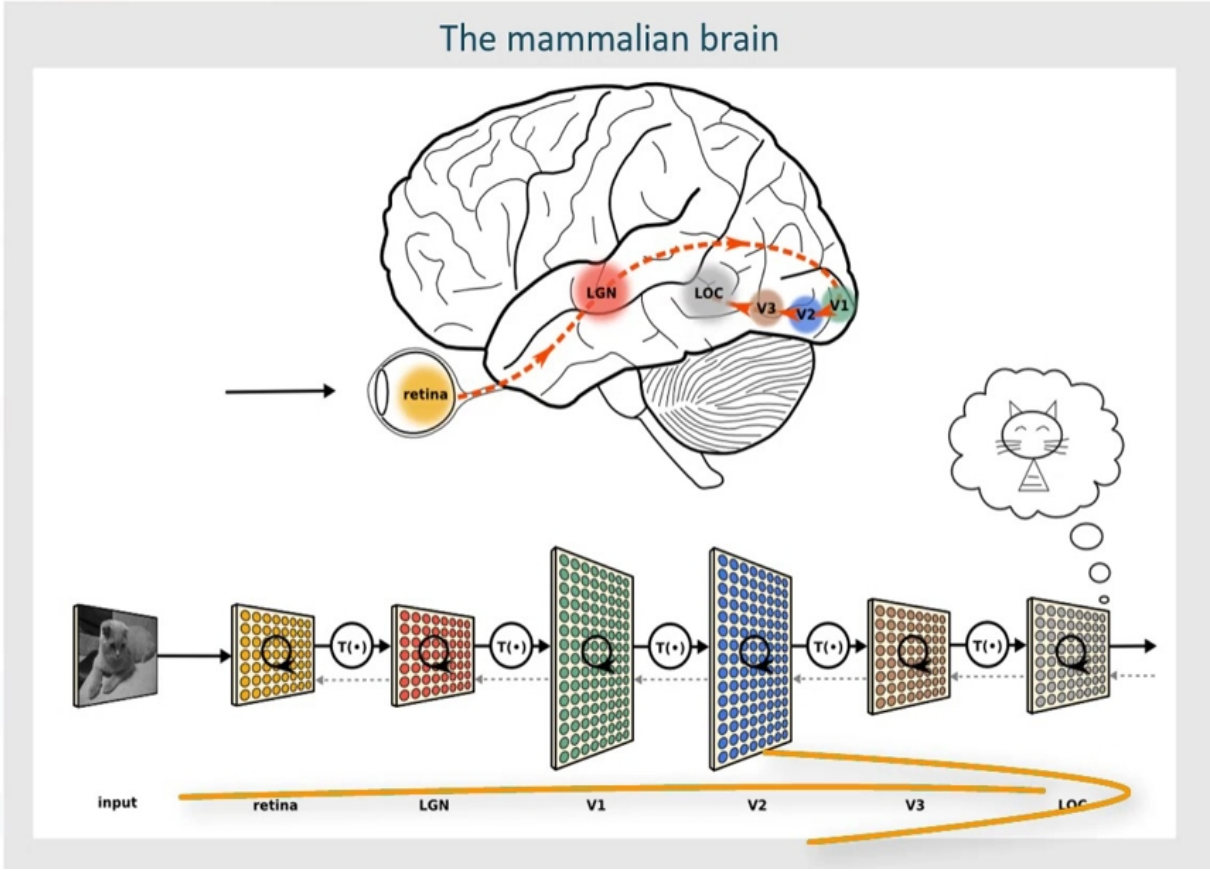
Inheritance from the real world





Inheritance from the real world





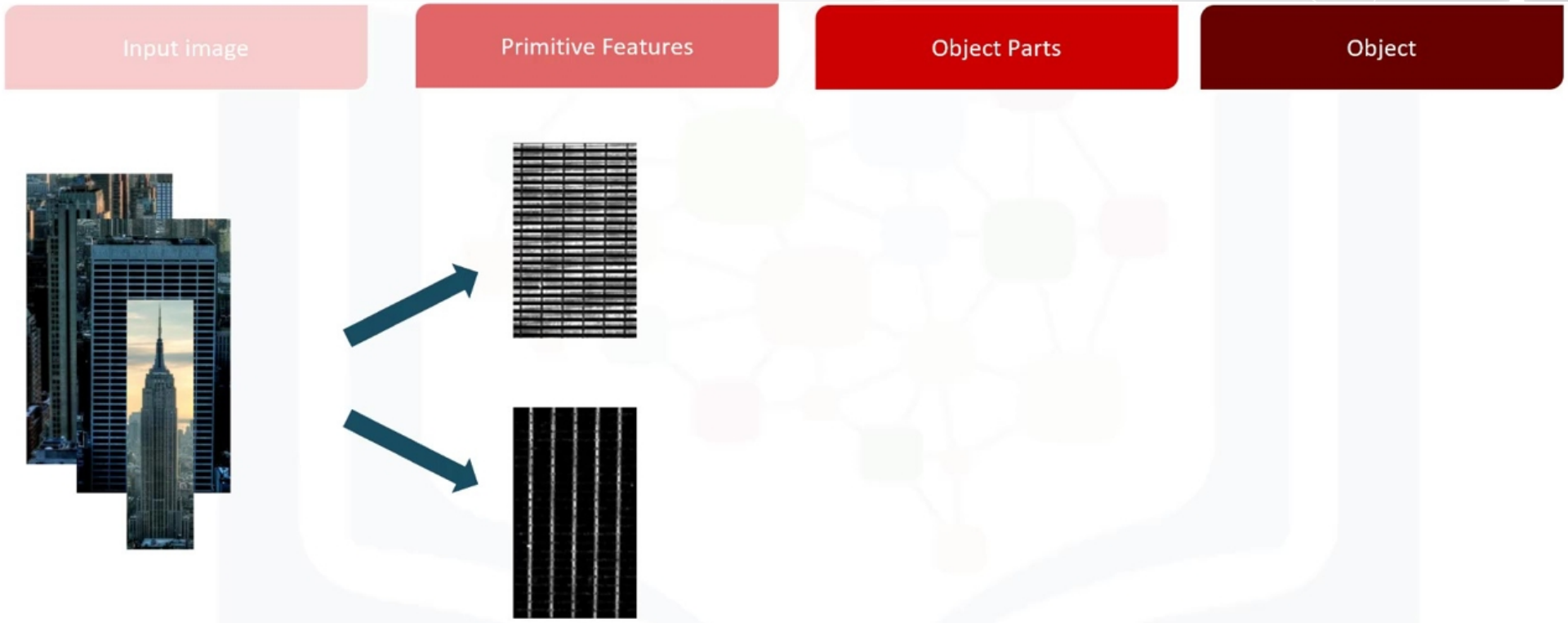
A visual explanation

A visual explanation

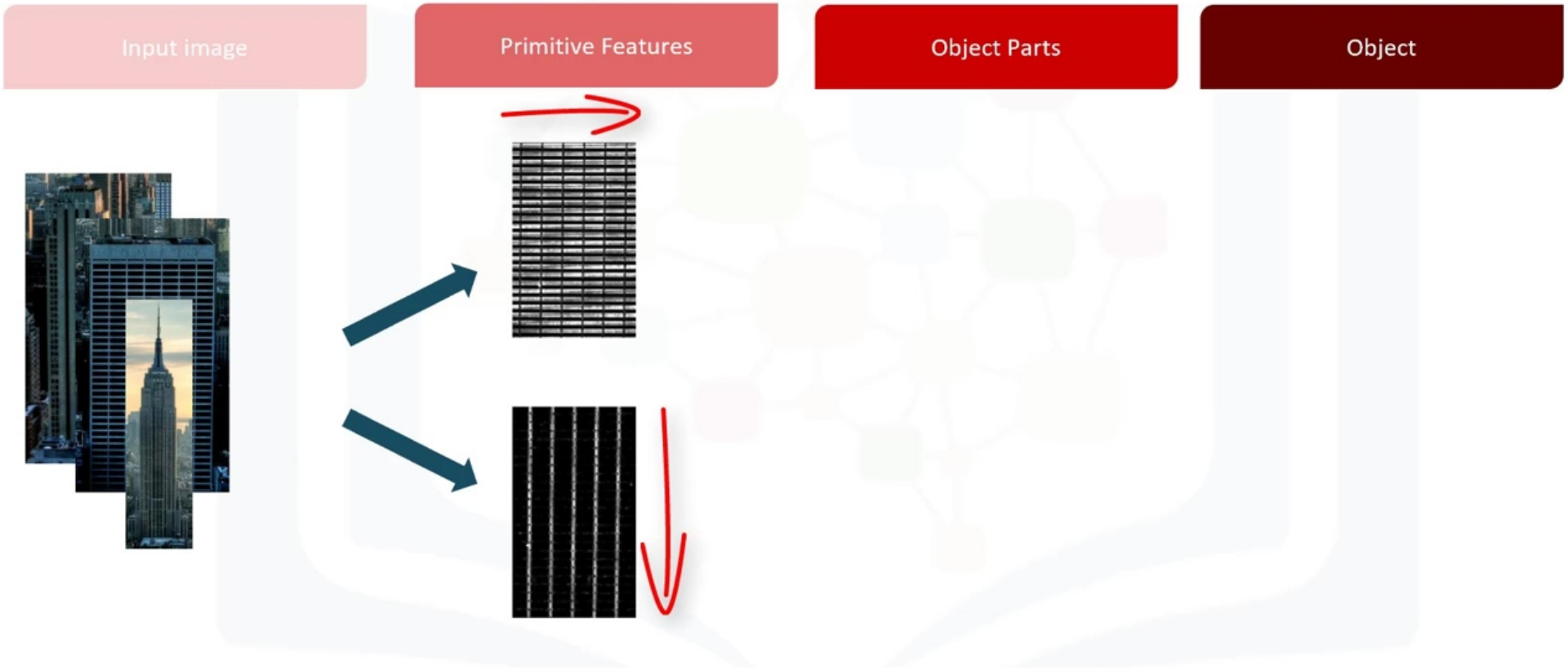
A visual explanation

Input image | Primitive Features | Object Parts | Object

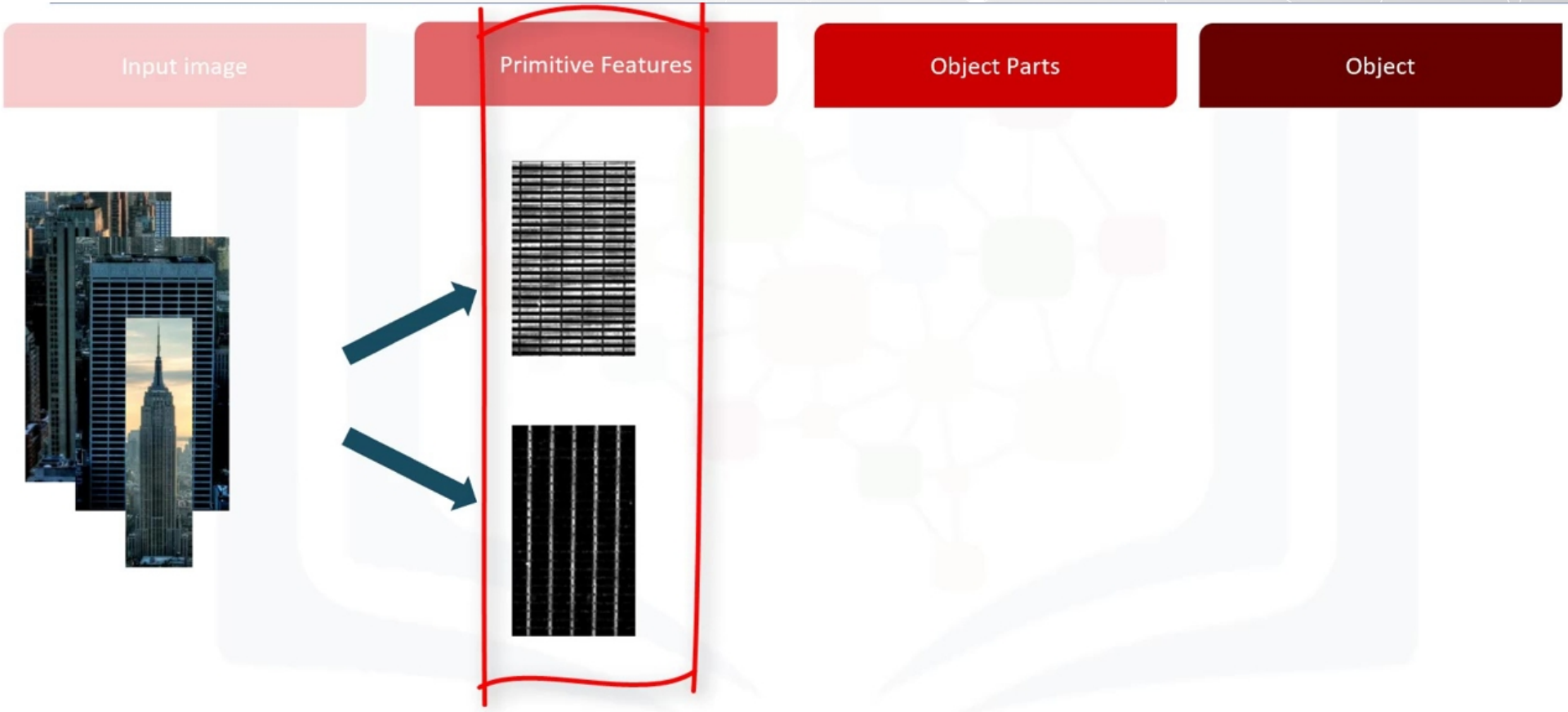
A visual explanation



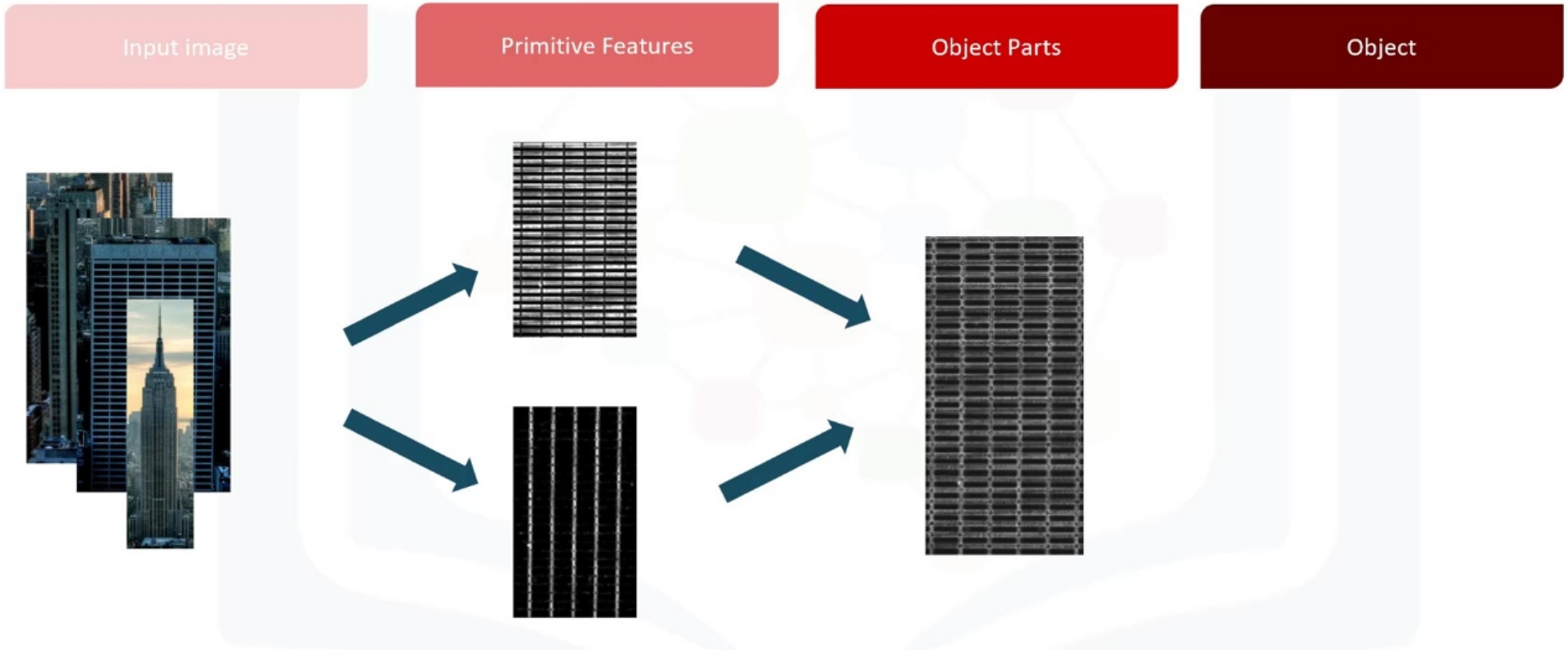
A visual explanation



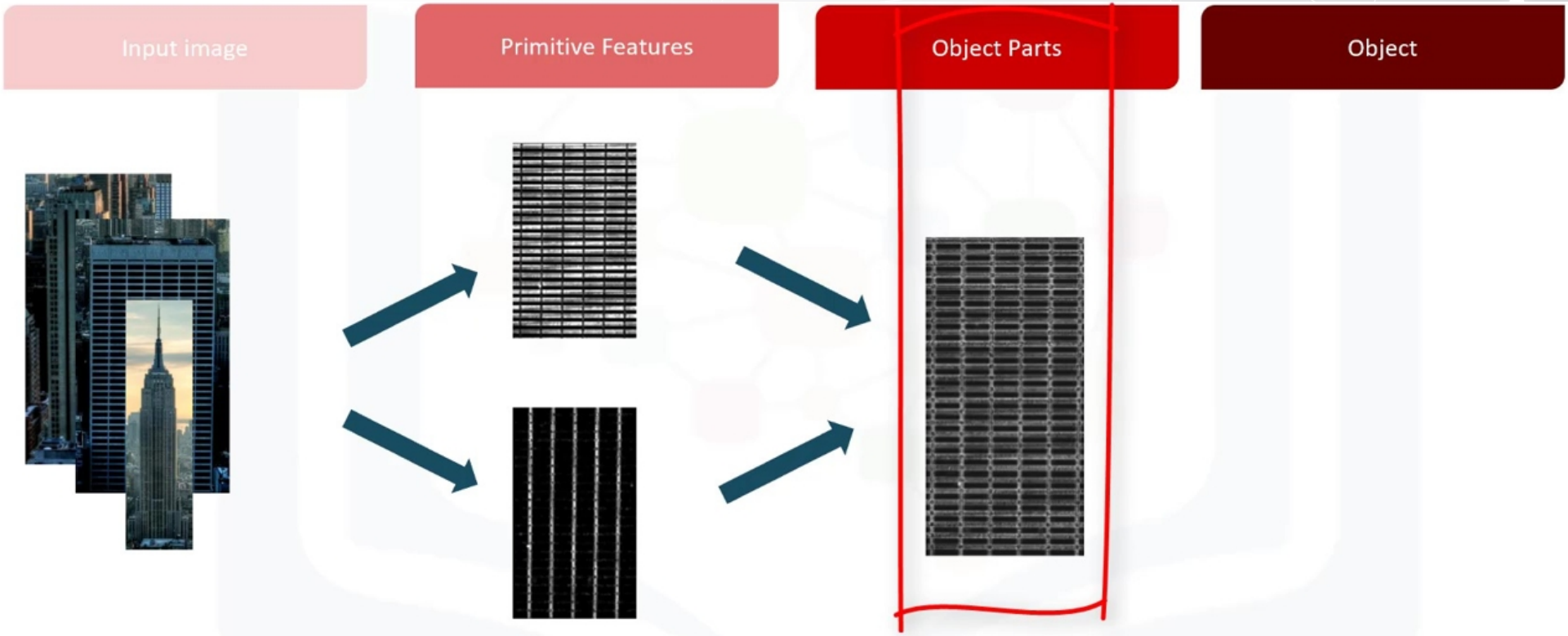
A visual explanation



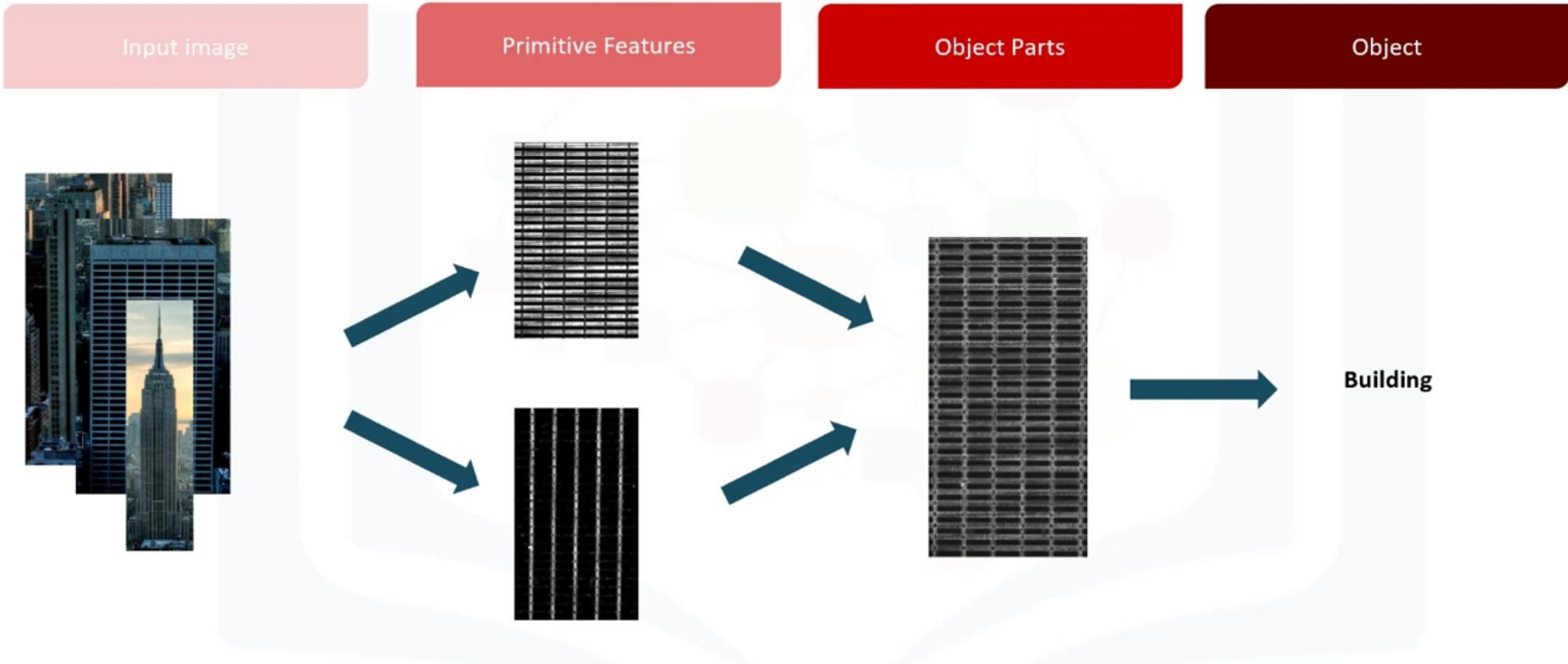
A visual explanation



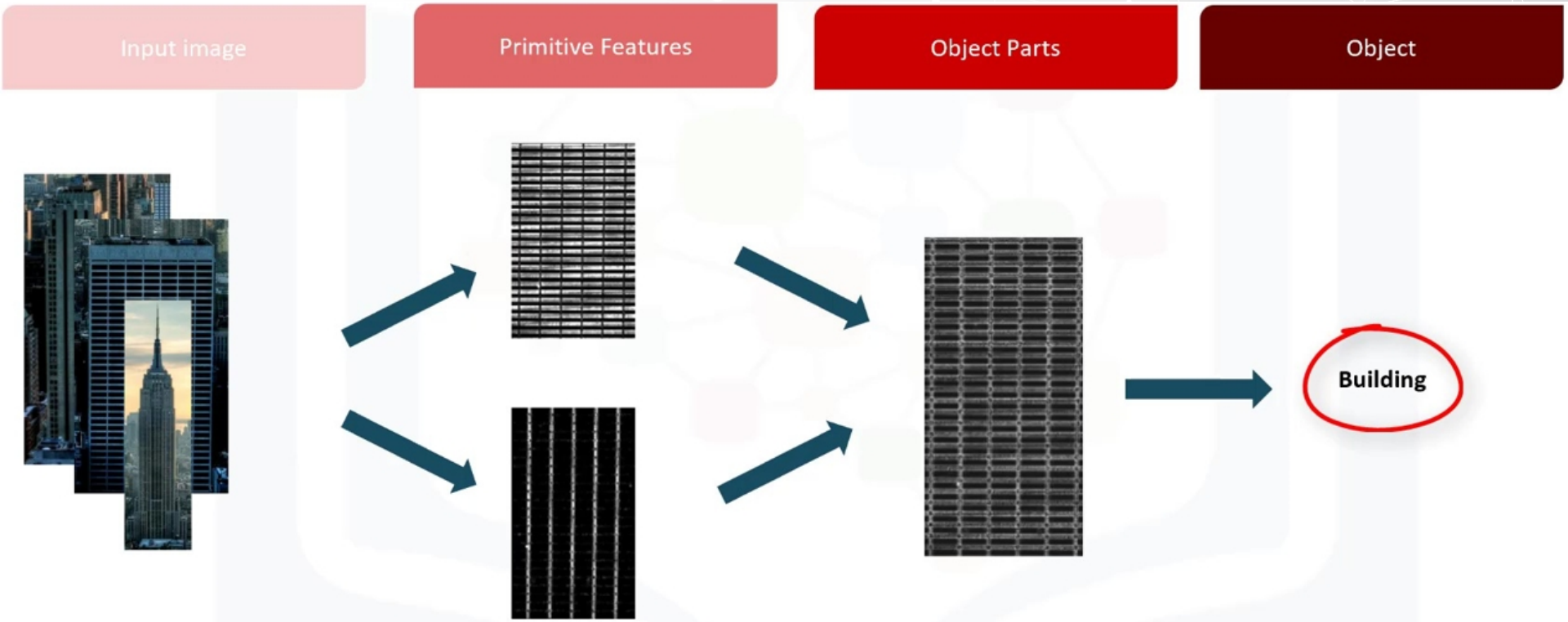
A visual explanation



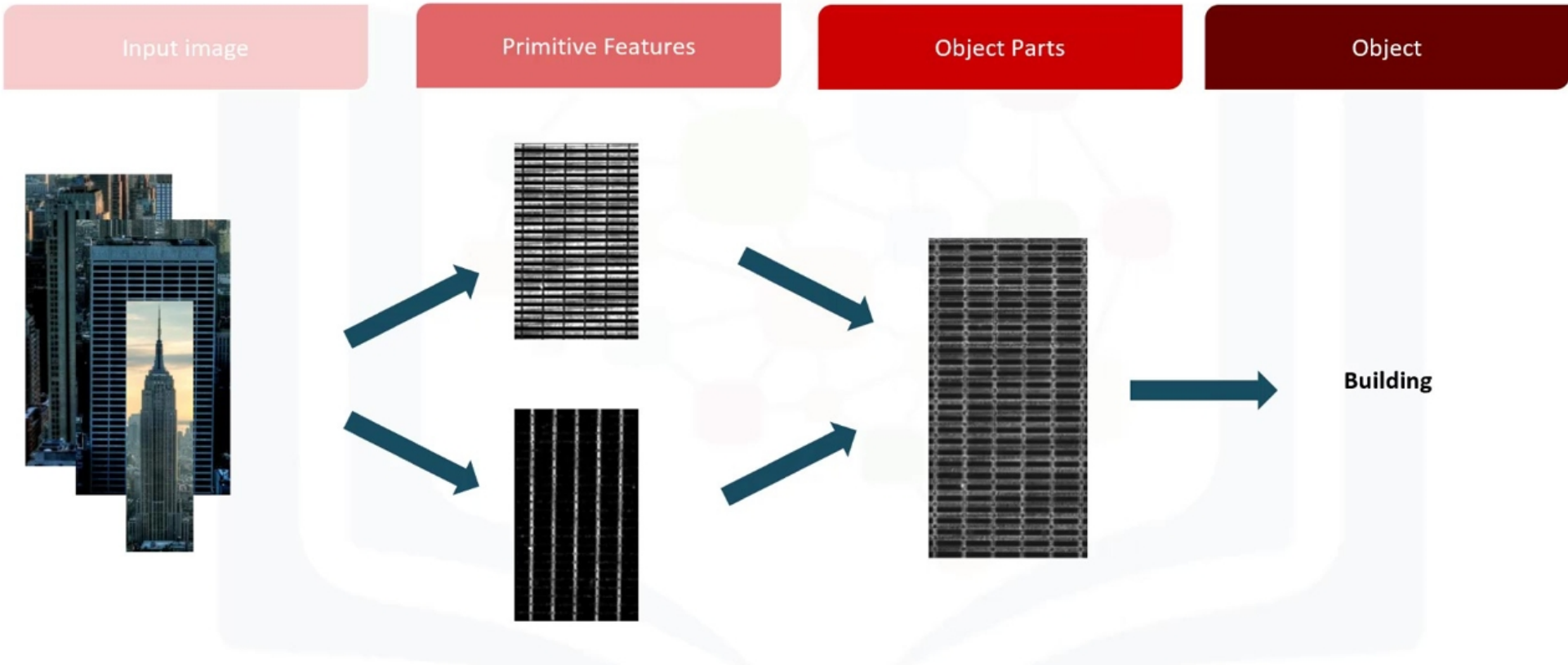
A visual explanation



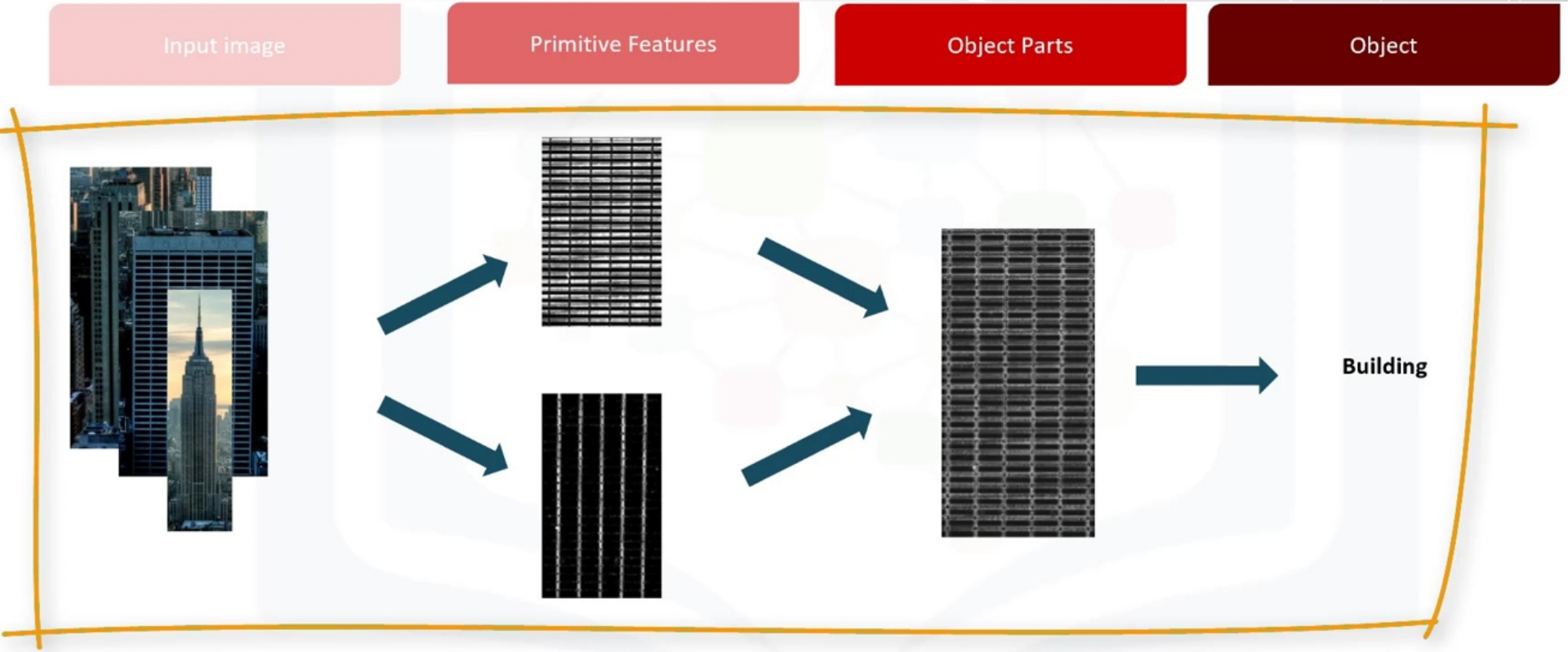
A visual explanation



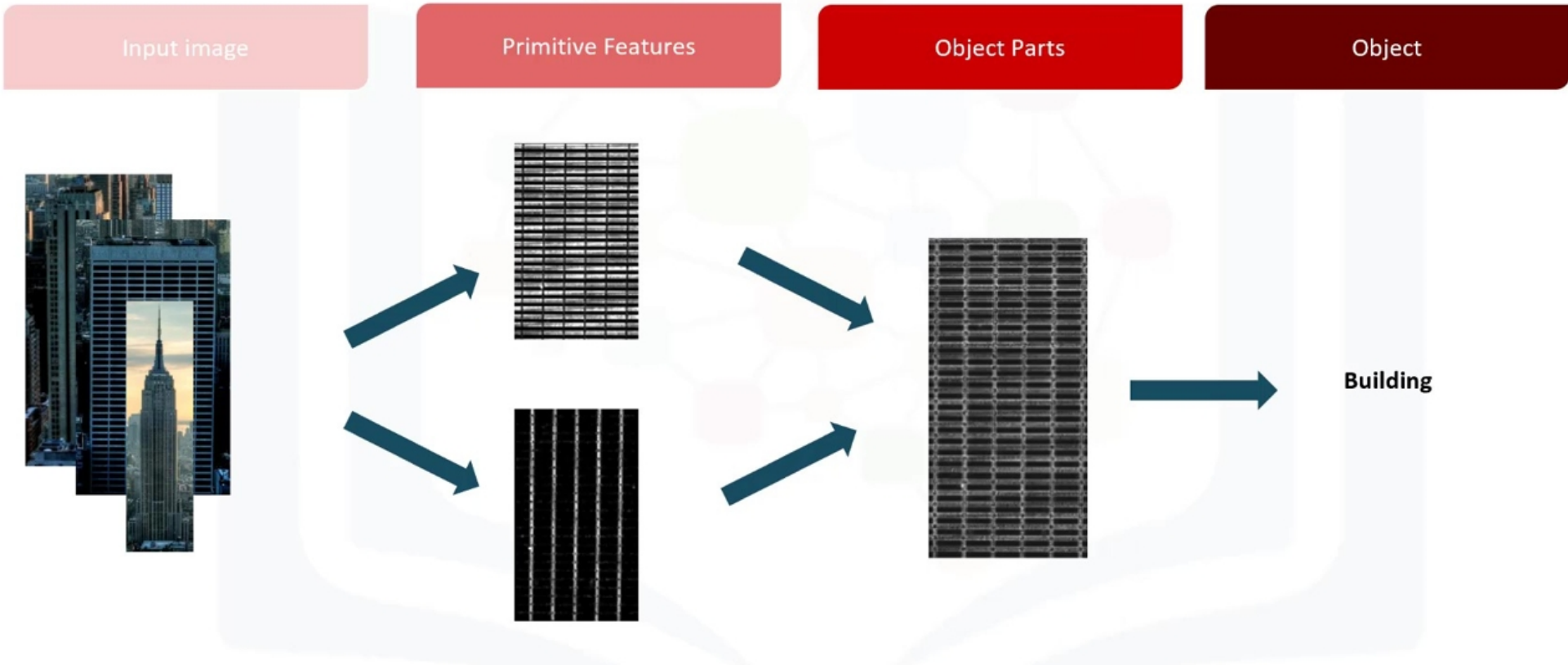
A visual explanation



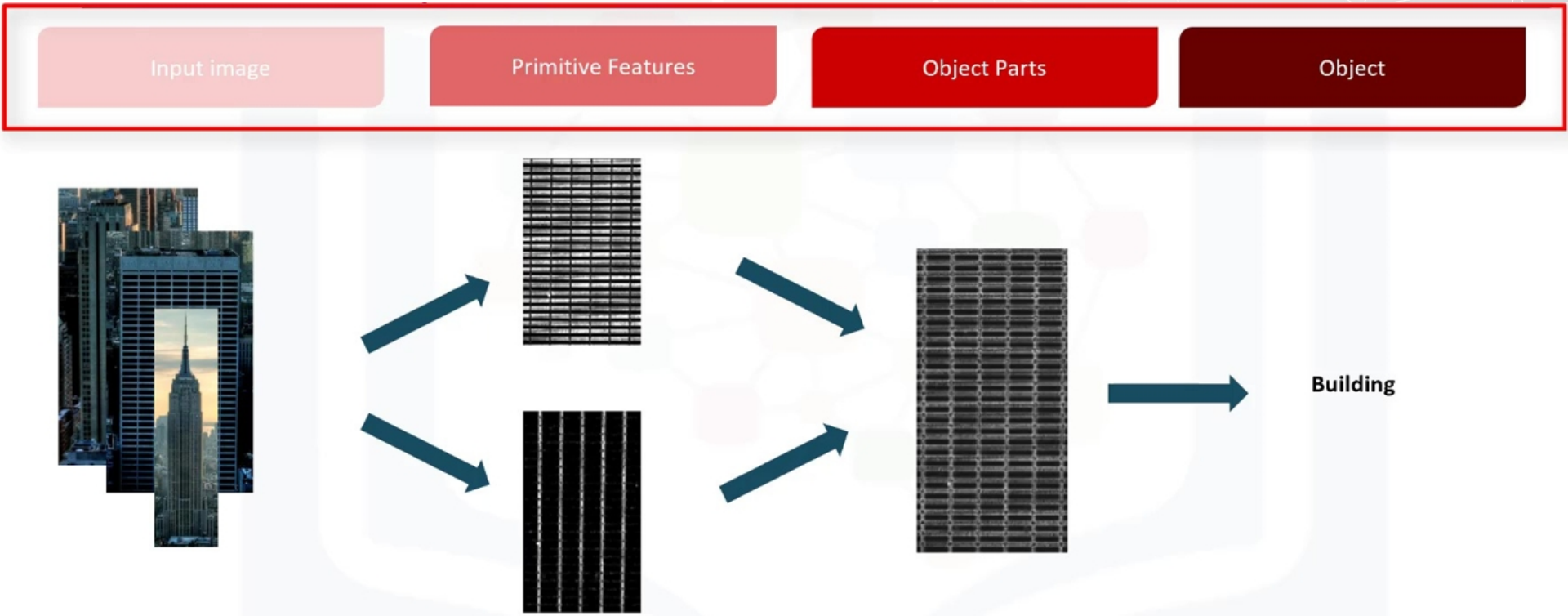
A visual explanation



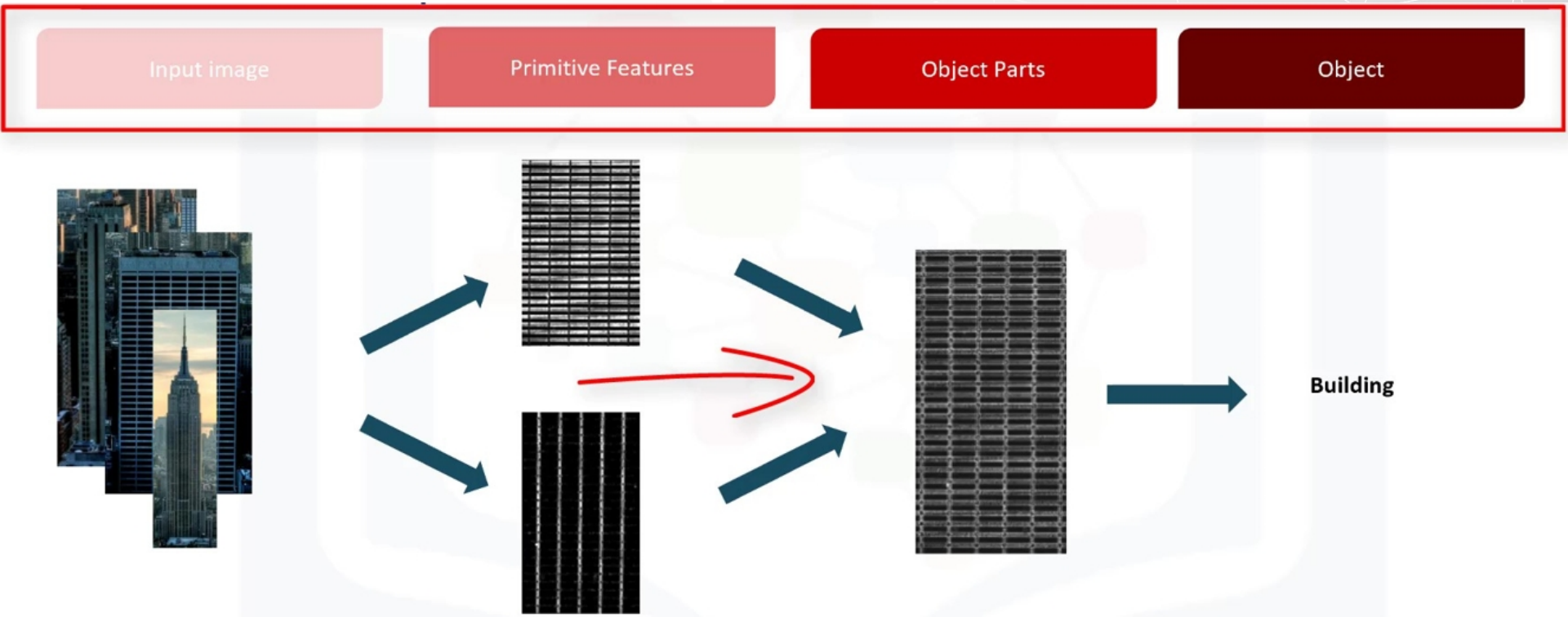
A visual explanation



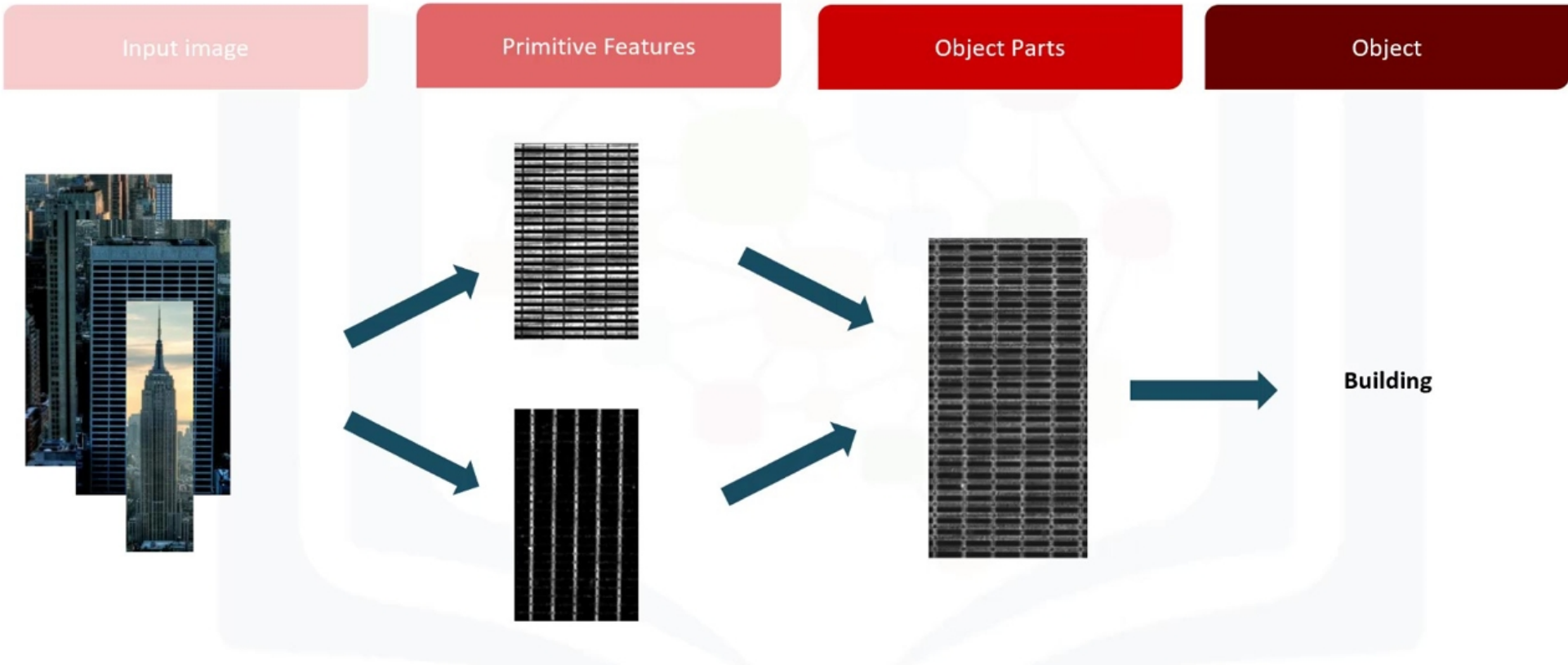
A visual explanation



A visual explanation



A visual explanation

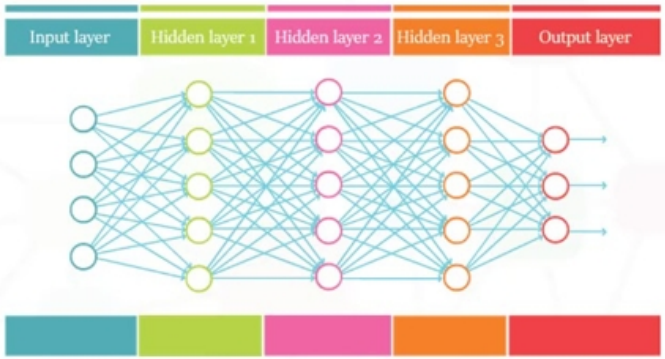


Bagian 2

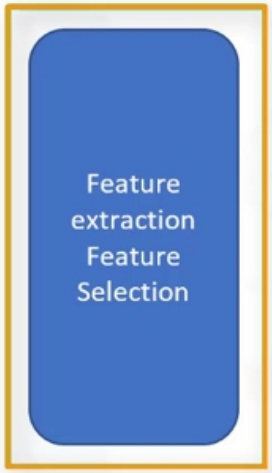
CNN for Classification

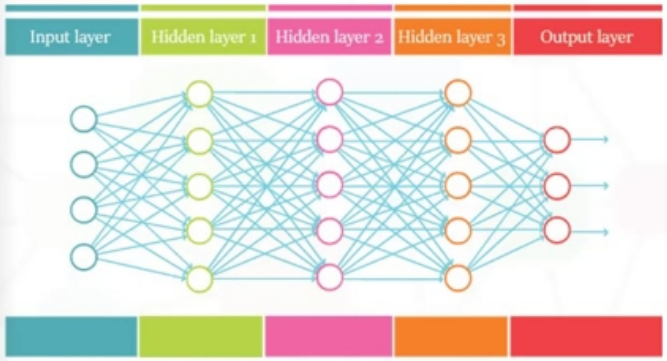
Shallow Neural Networks, why not?

Feature extraction Feature Selection



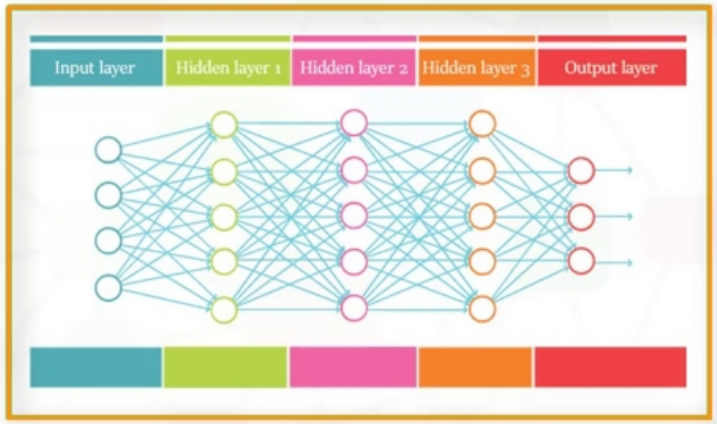
Shallow Neural Networks, why not?





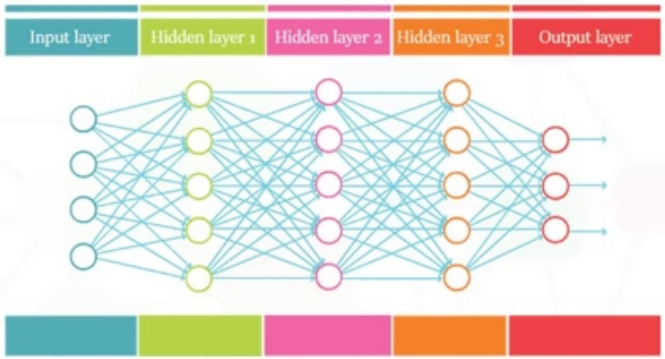
Shallow Neural Networks, why not?

Feature extraction
Feature Selection



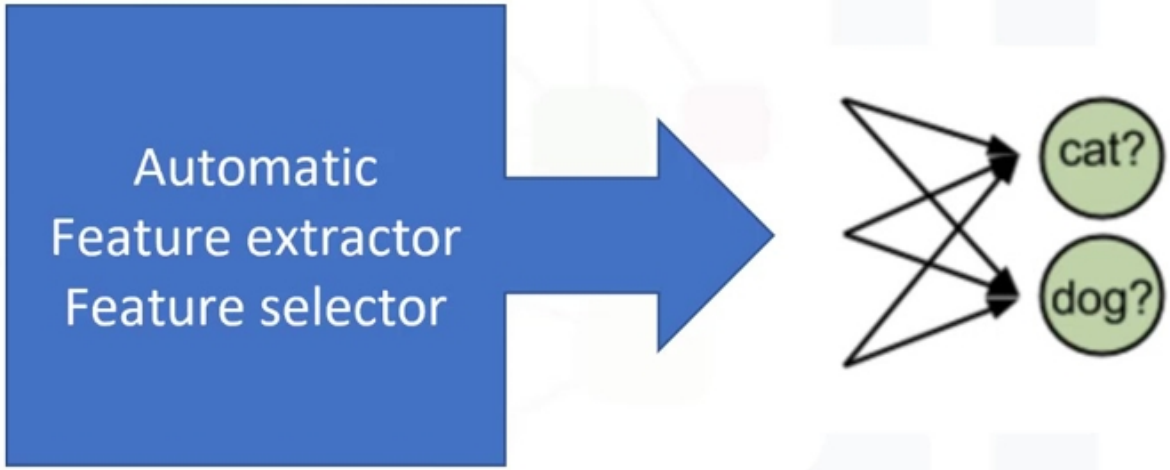
Shallow Neural Networks, why not?

Feature extraction Feature Selection

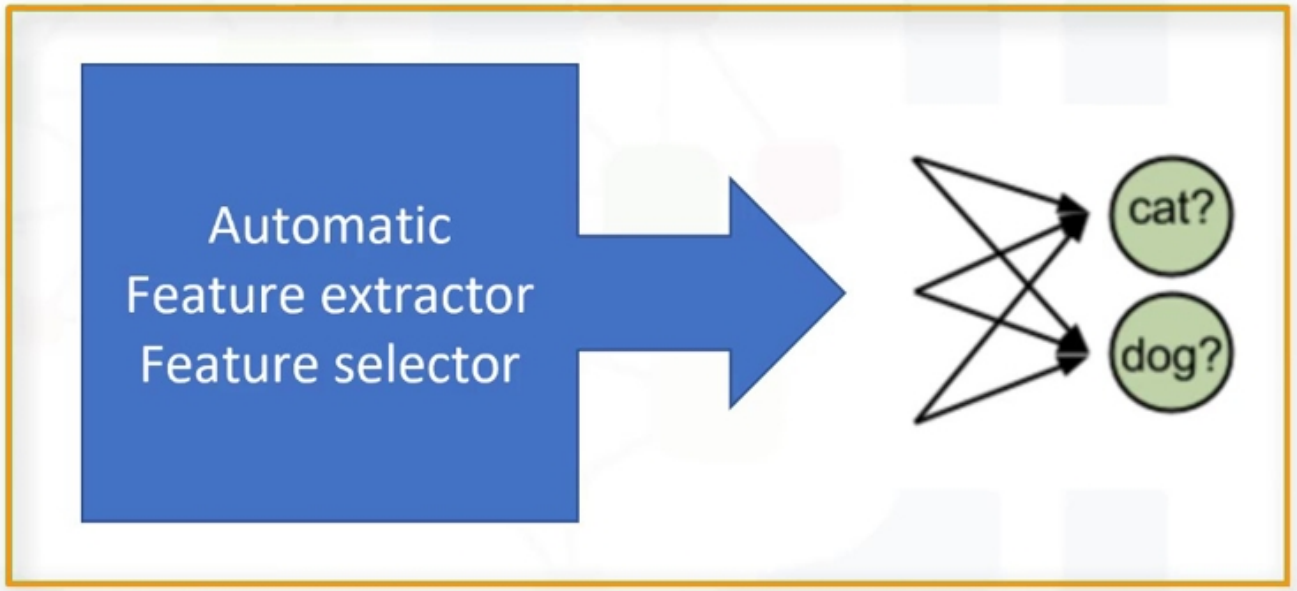


- The process of selecting the best features is hard
- Extending the features to other types of images is not possible

Convolutional Neural Networks (CNNs)

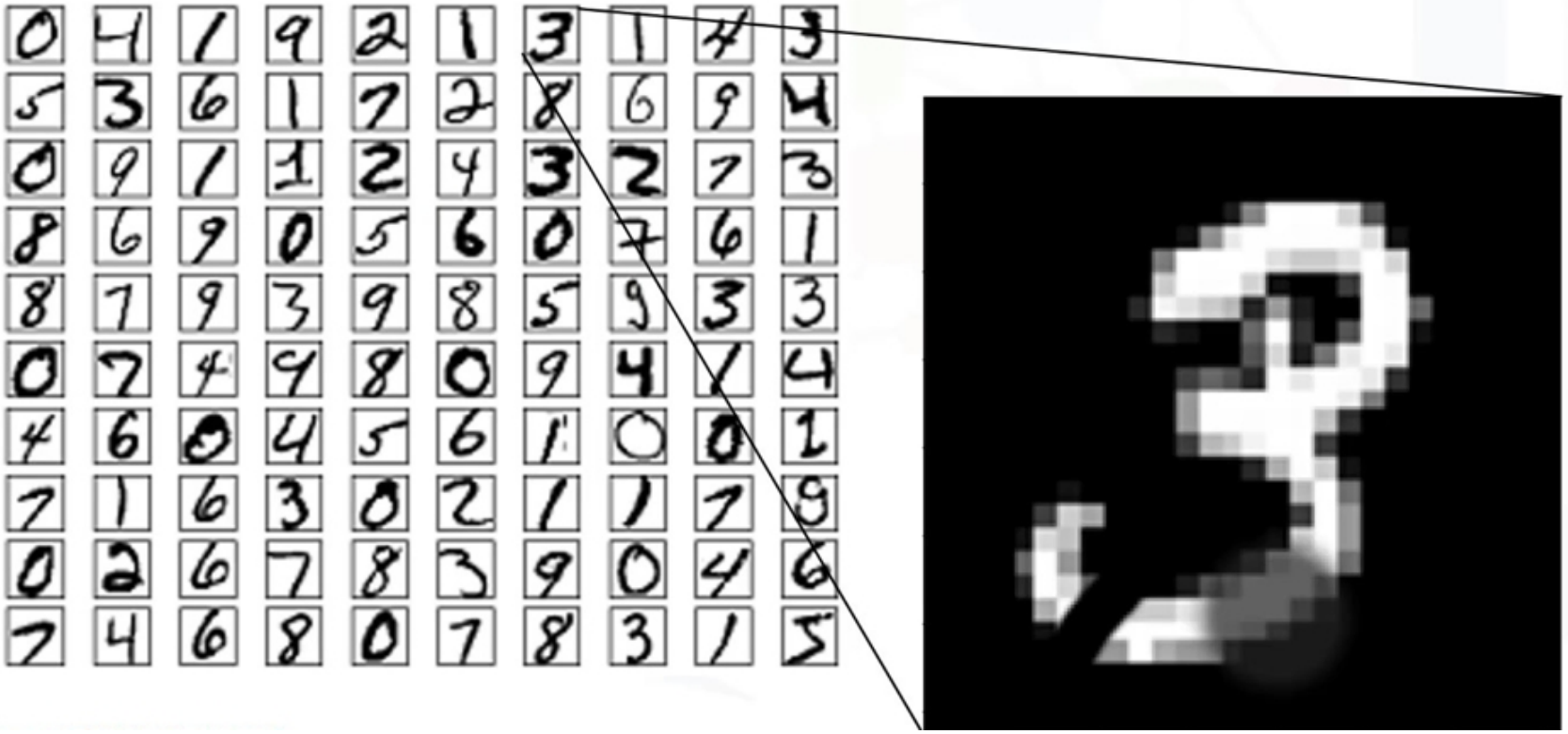


Convolutional Neural Networks (CNNs)



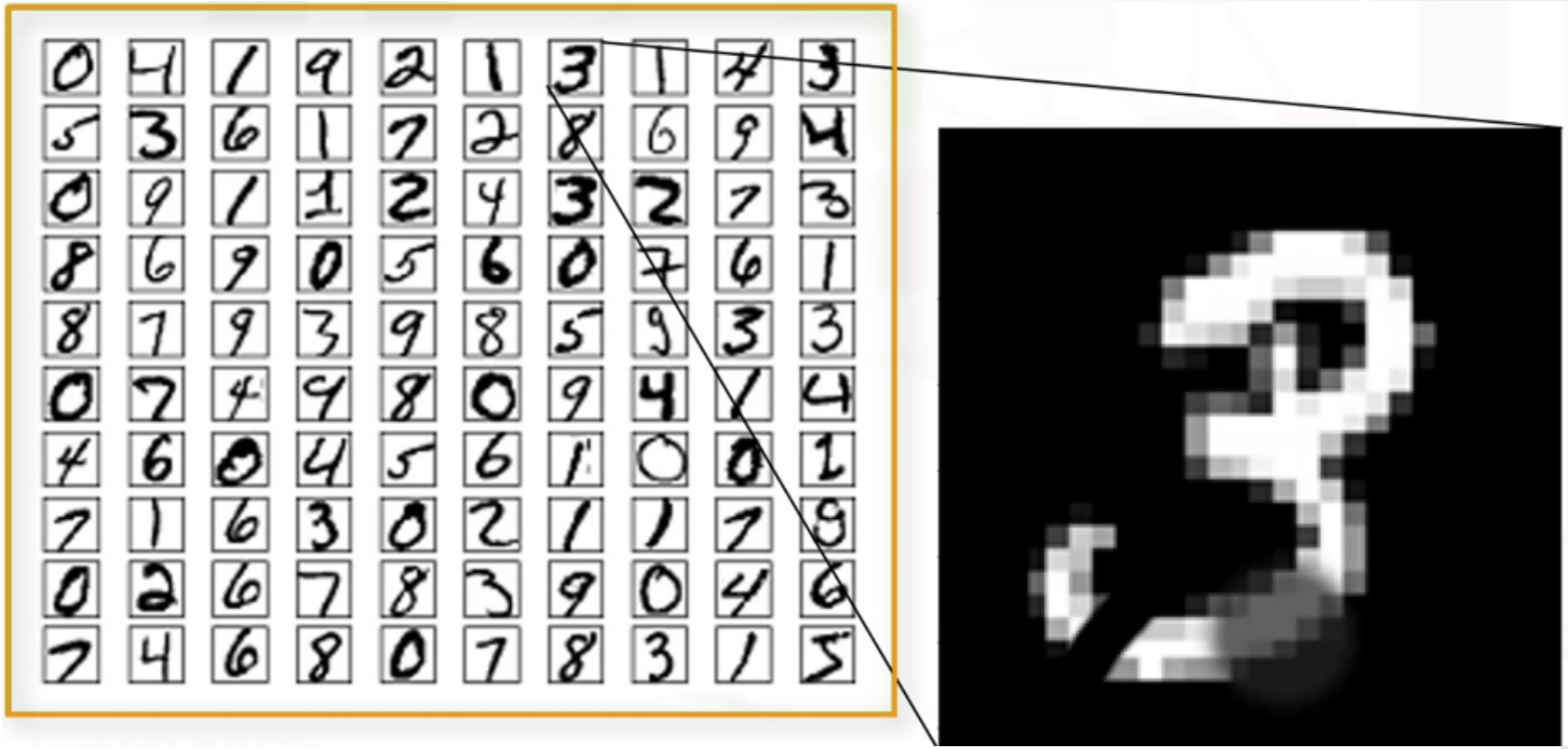
Datasets

- MNIST Dataset: database of handwritten digits

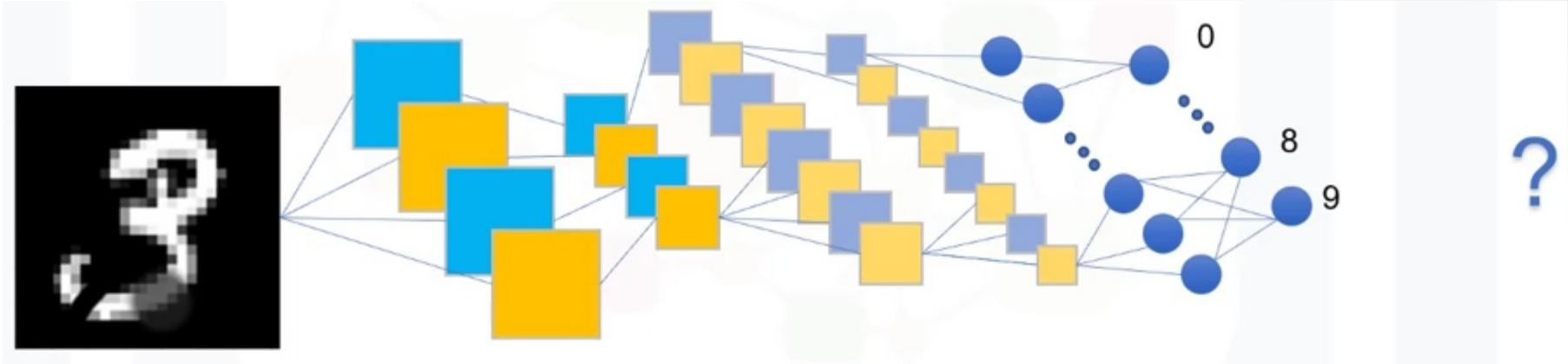


Datasets

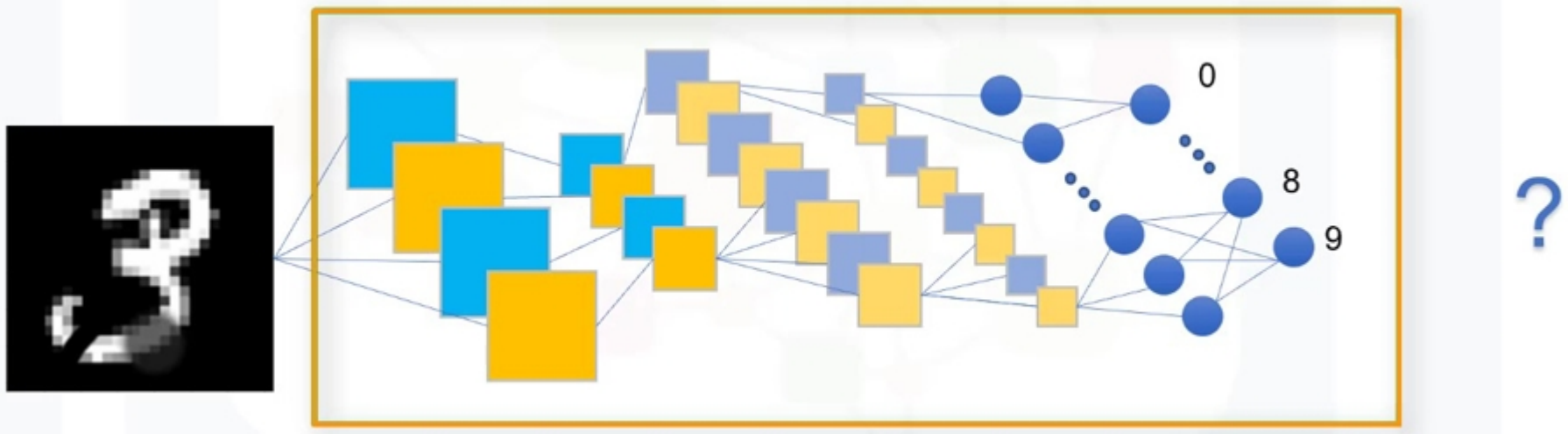
- MNIST Dataset: database of handwritten digits



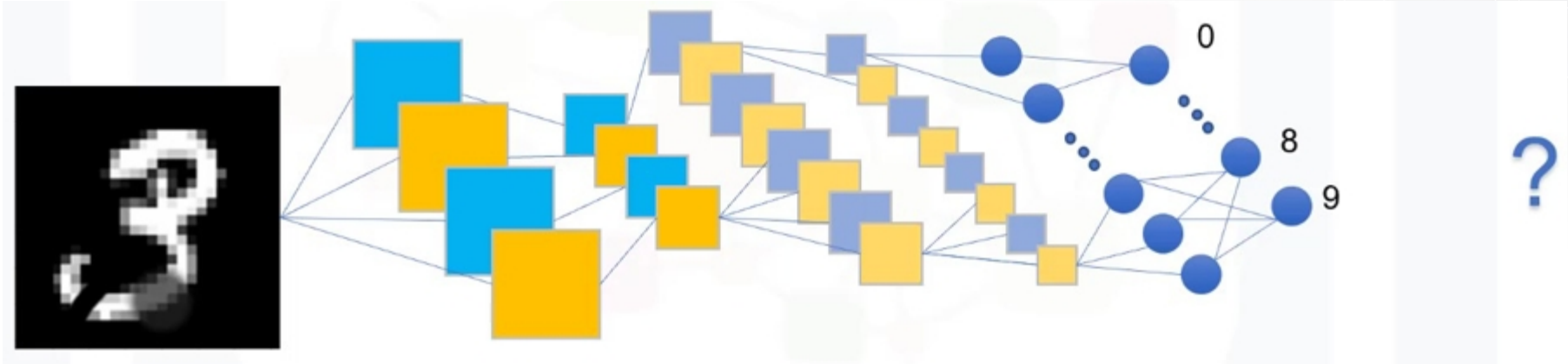
Digit Recognition



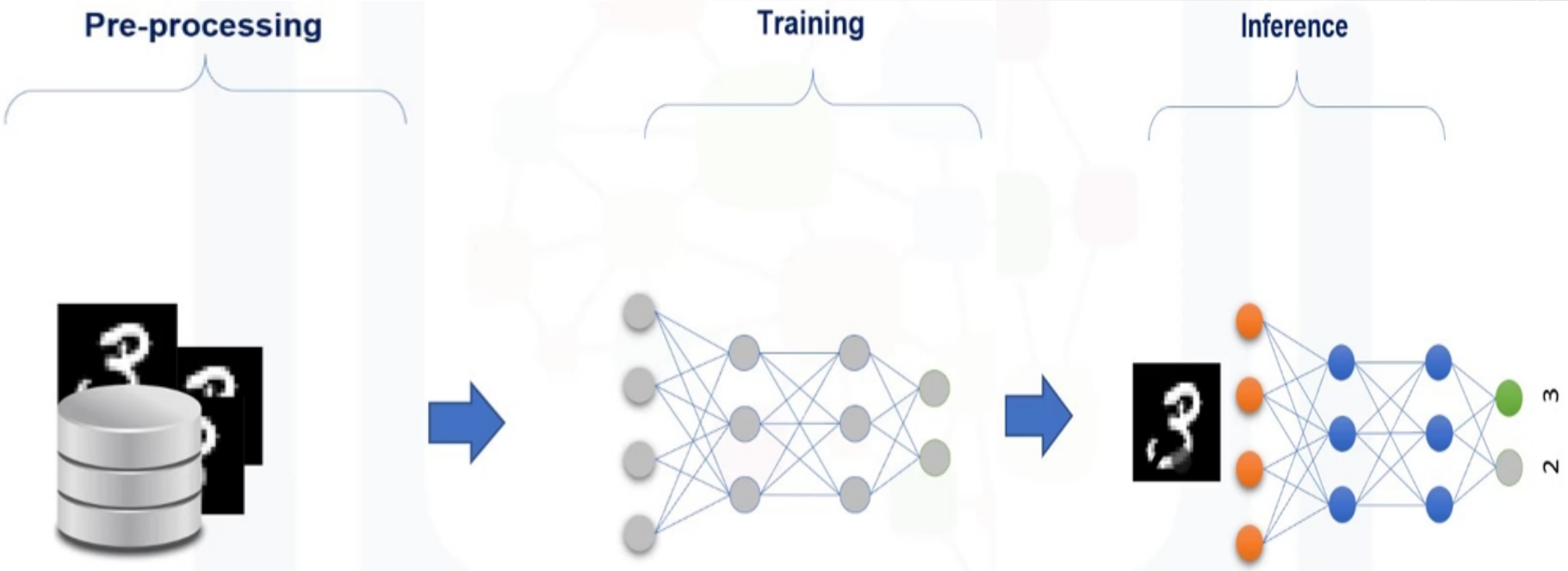
Digit Recognition



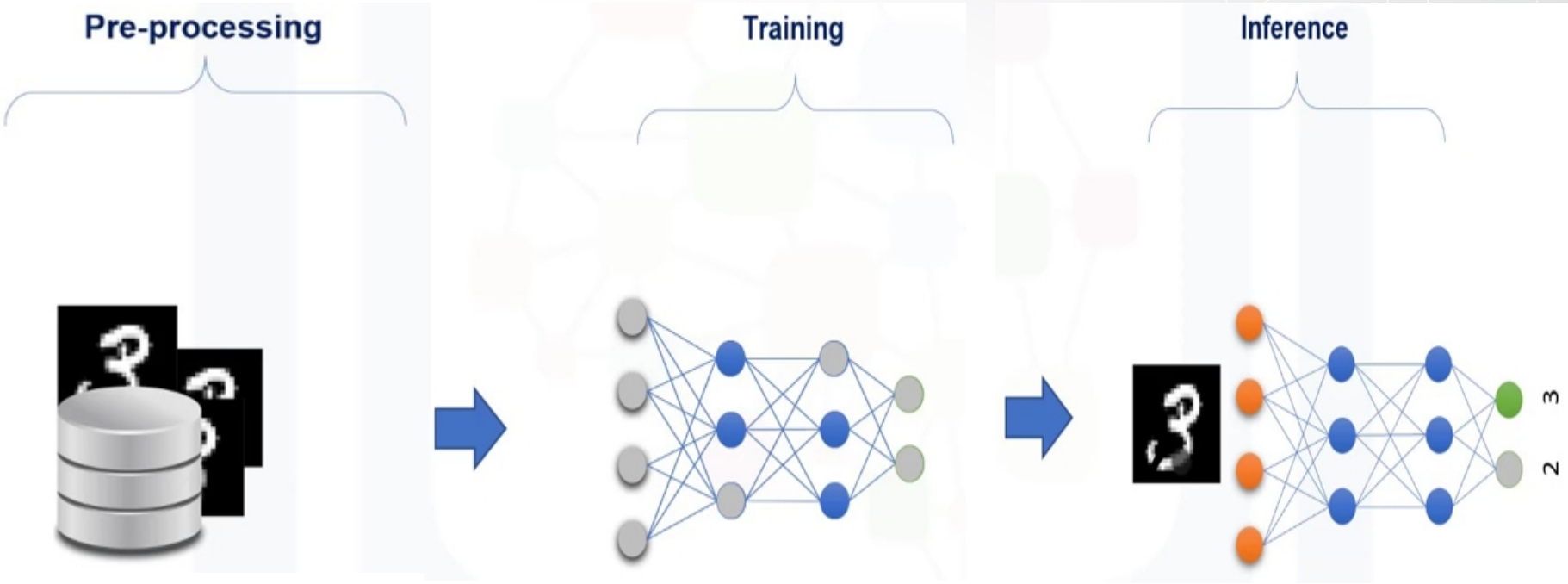
Digit Recognition



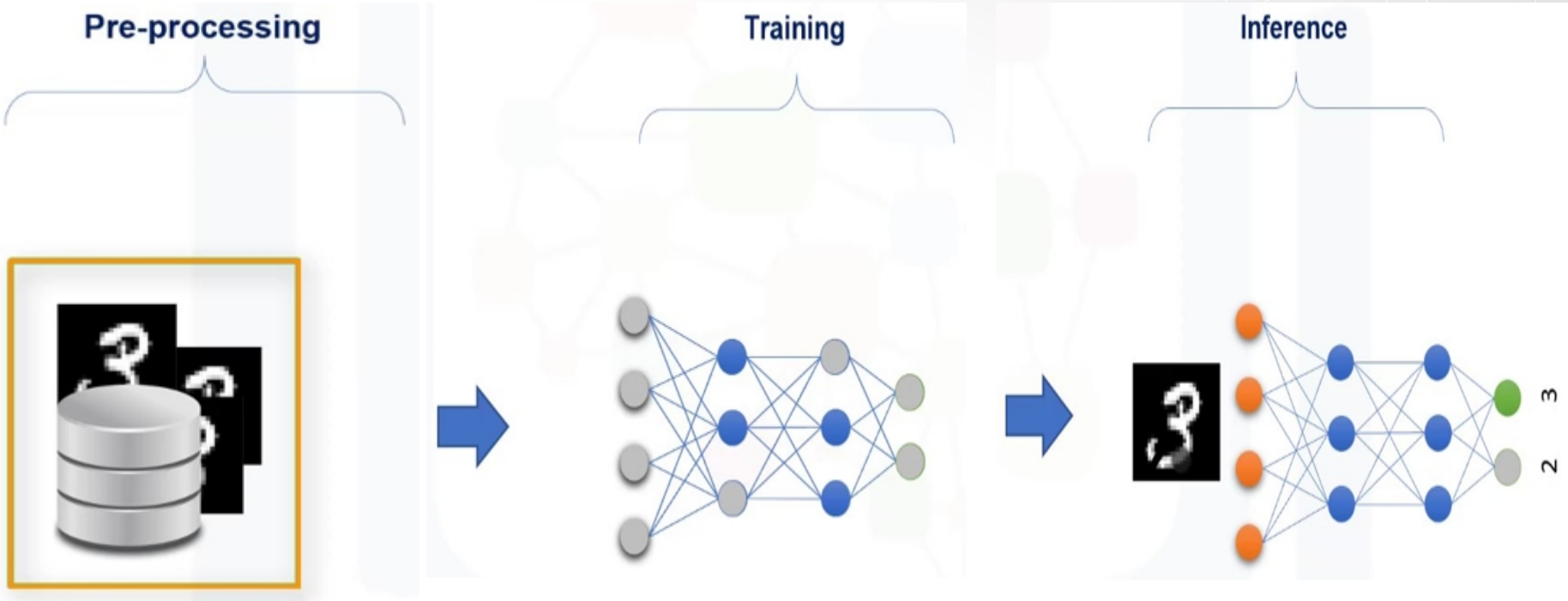
Training and Inference



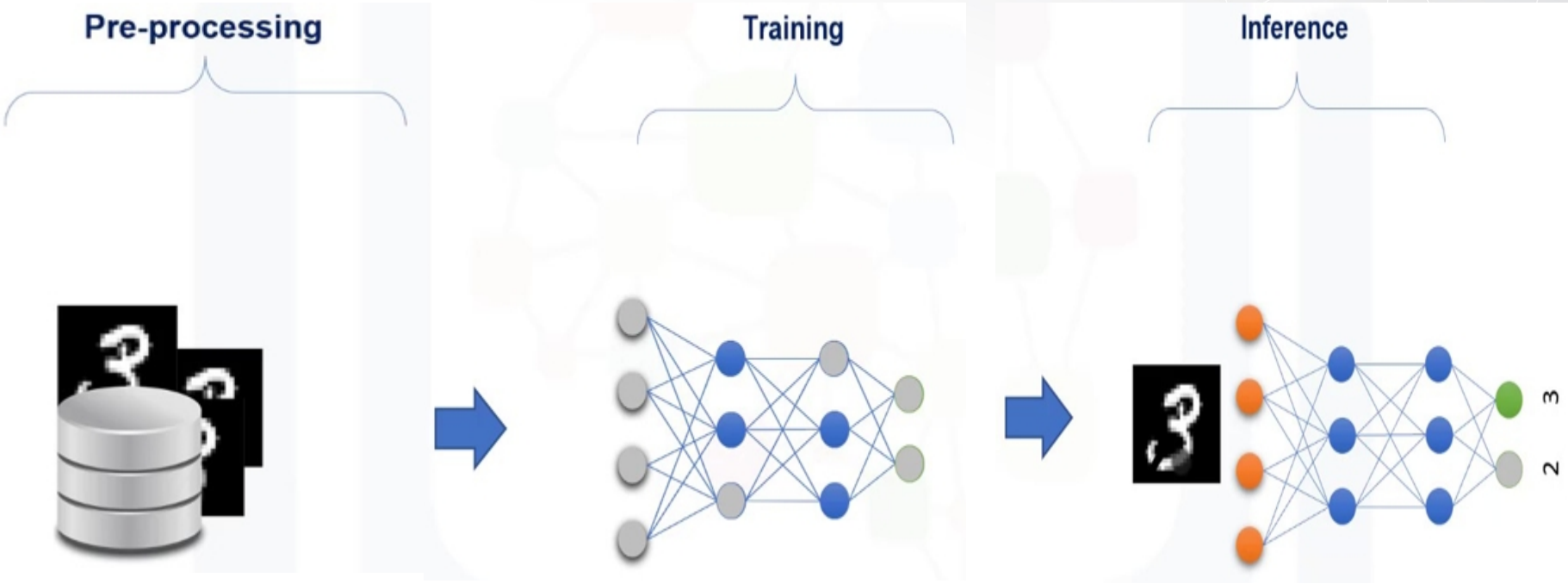
Training and Inference



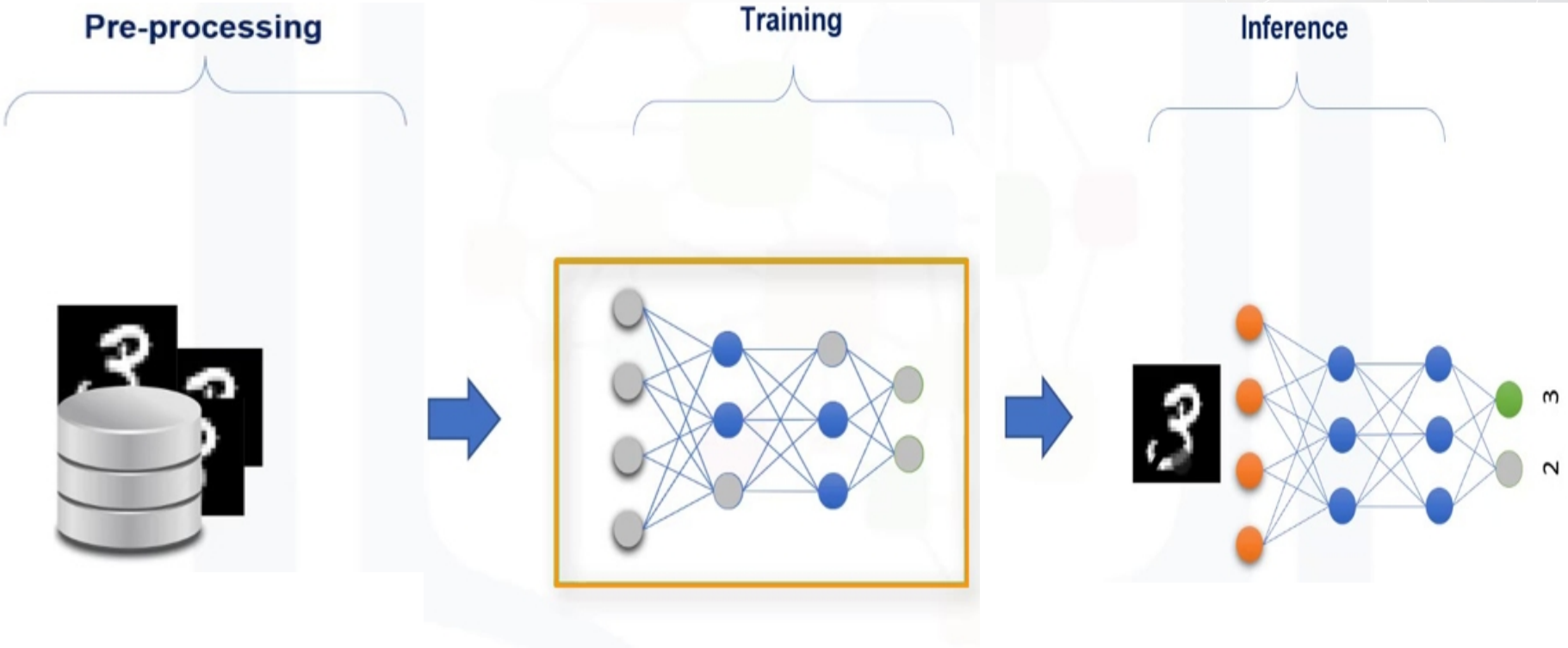
Training and Inference



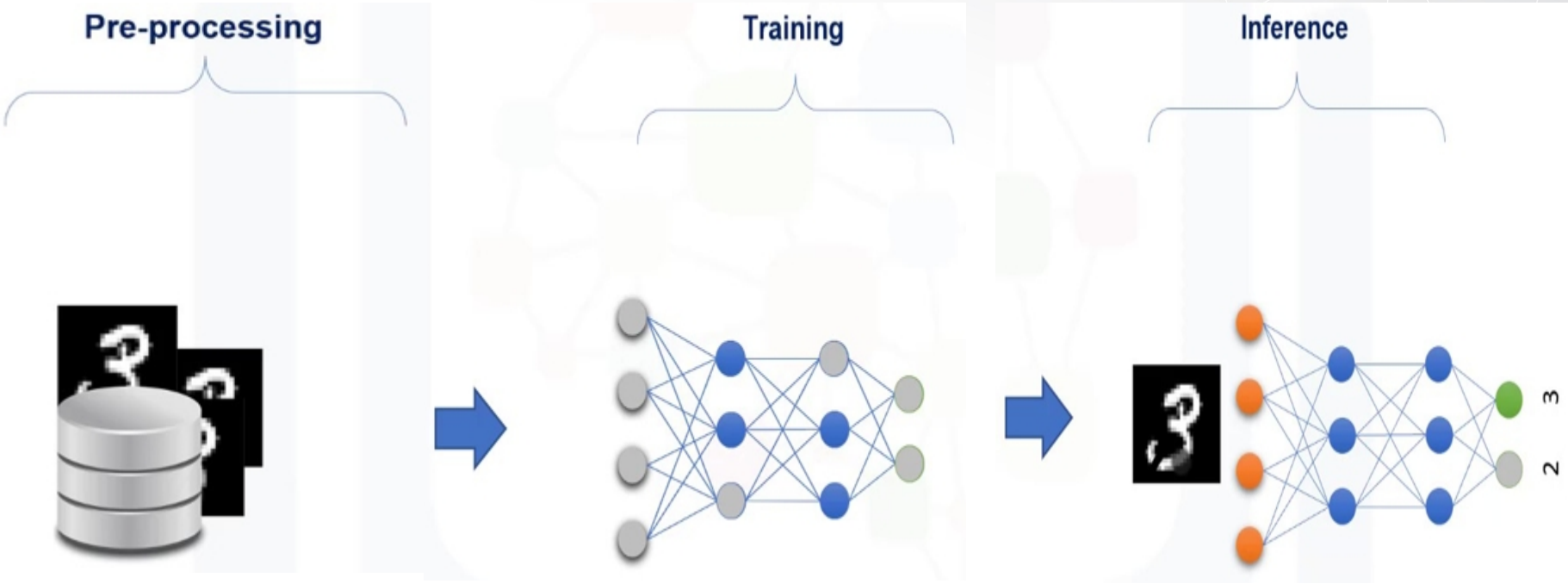
Training and Inference



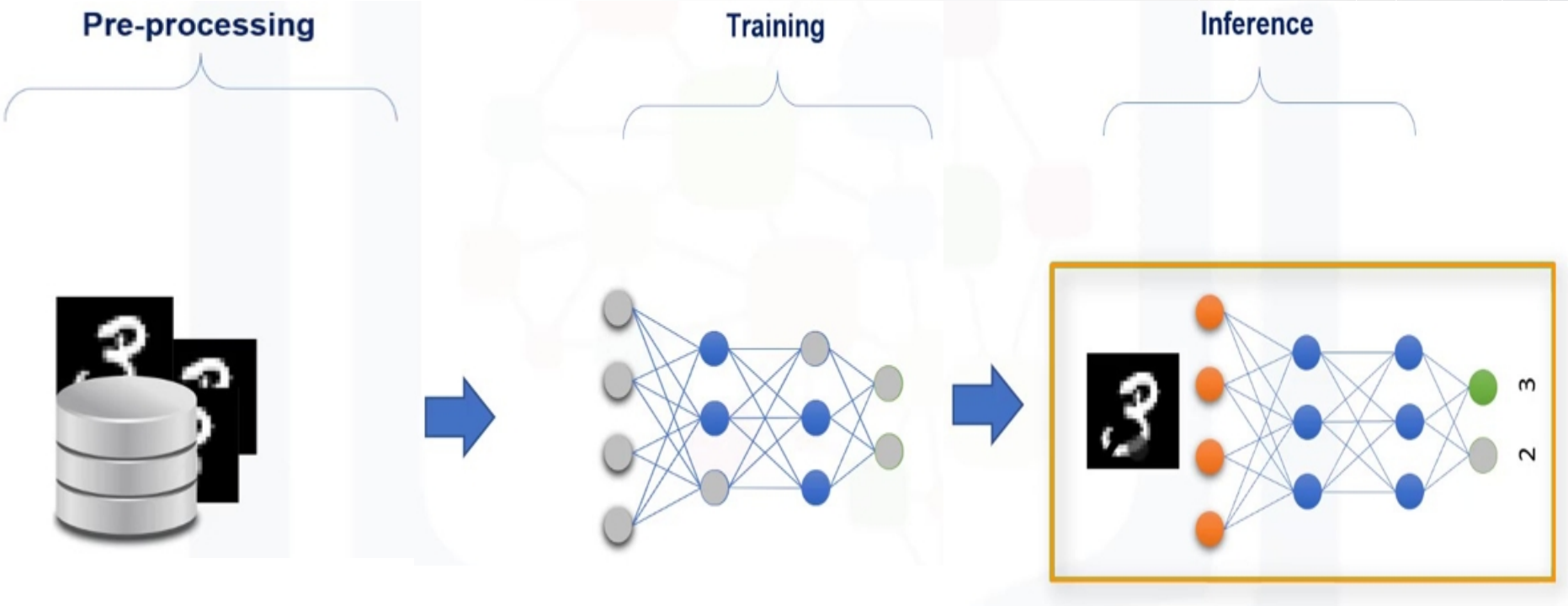
Training and Inference



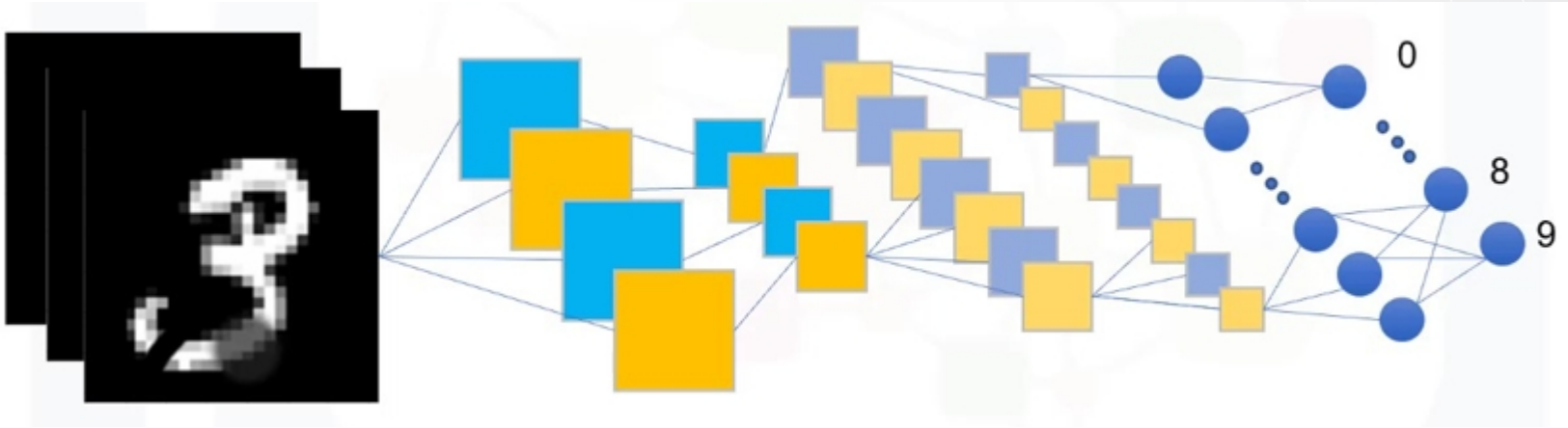
Training and Inference



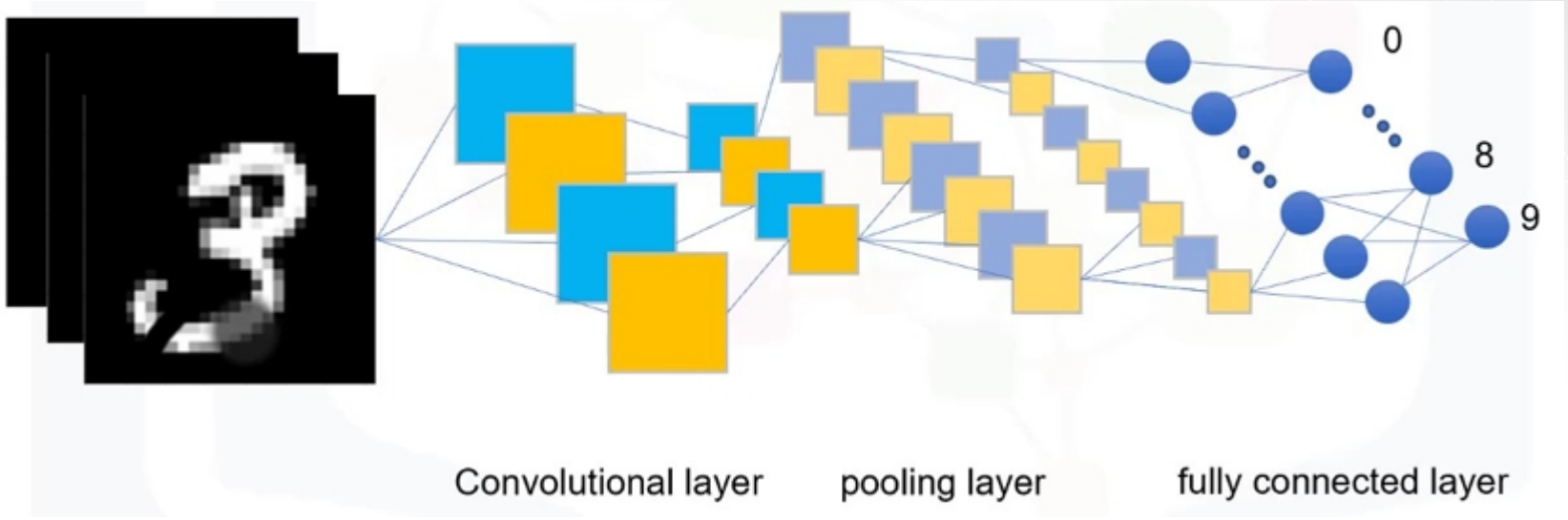
Training and Inference



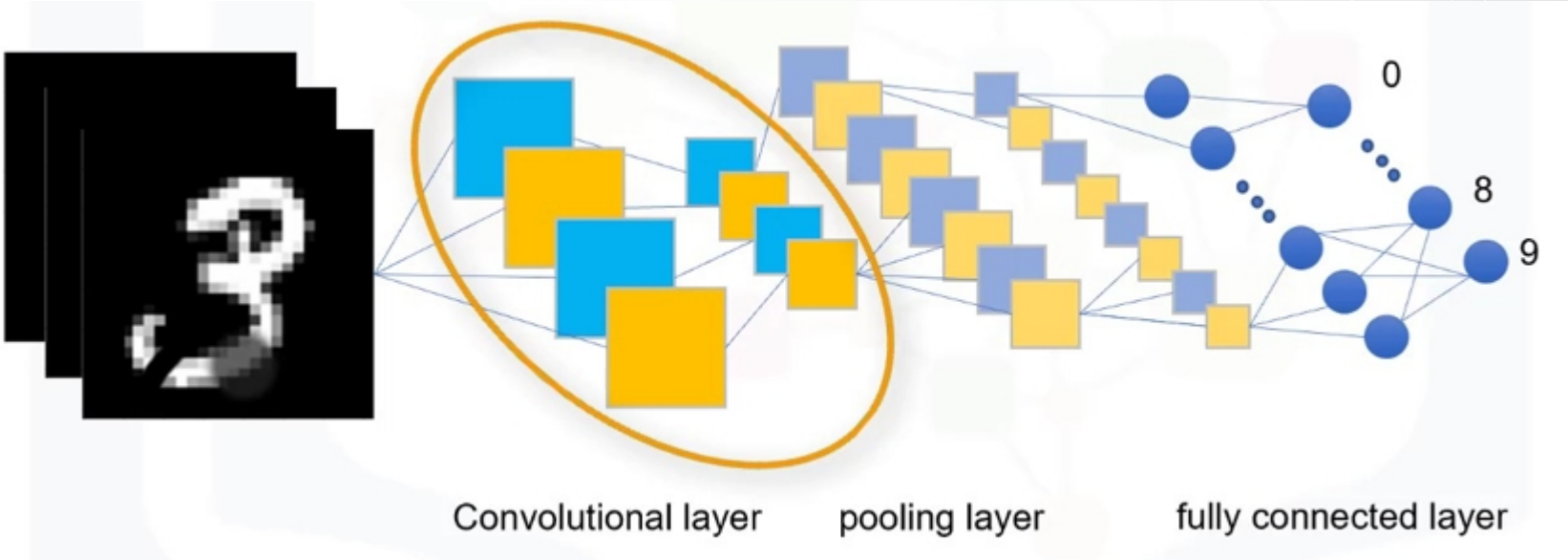
Digit Recognition



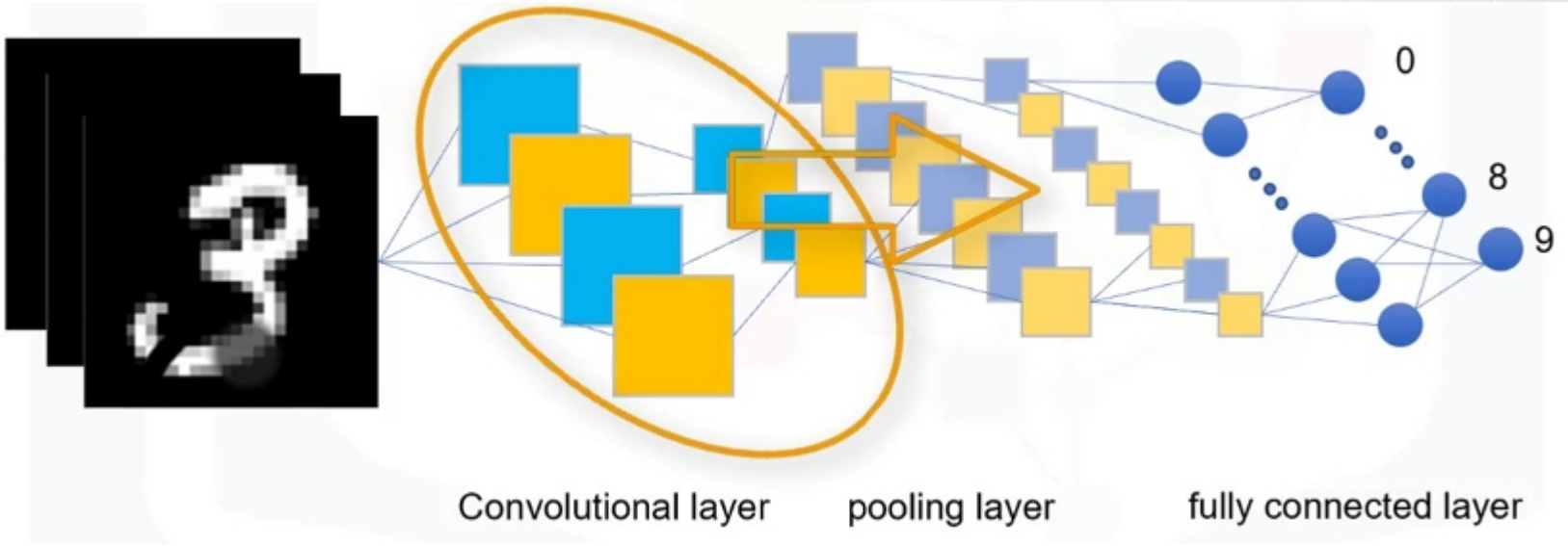
Digit Recognition



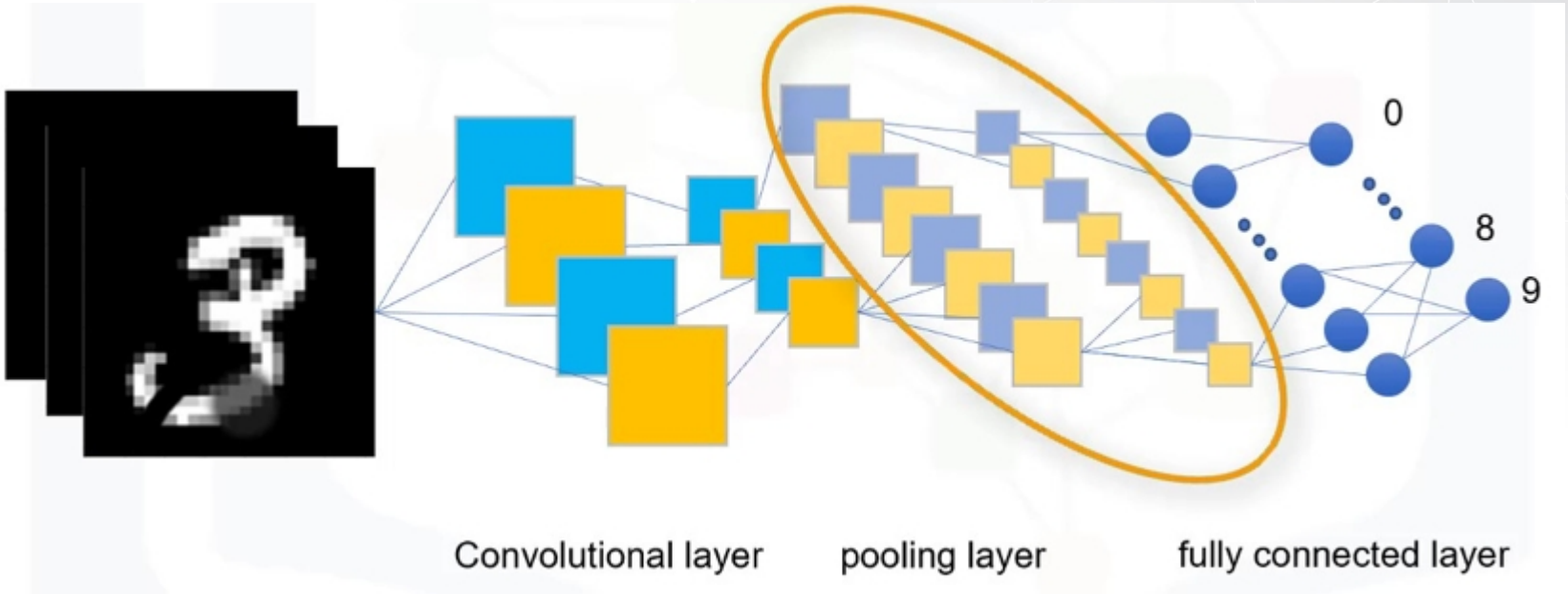
Digit Recognition



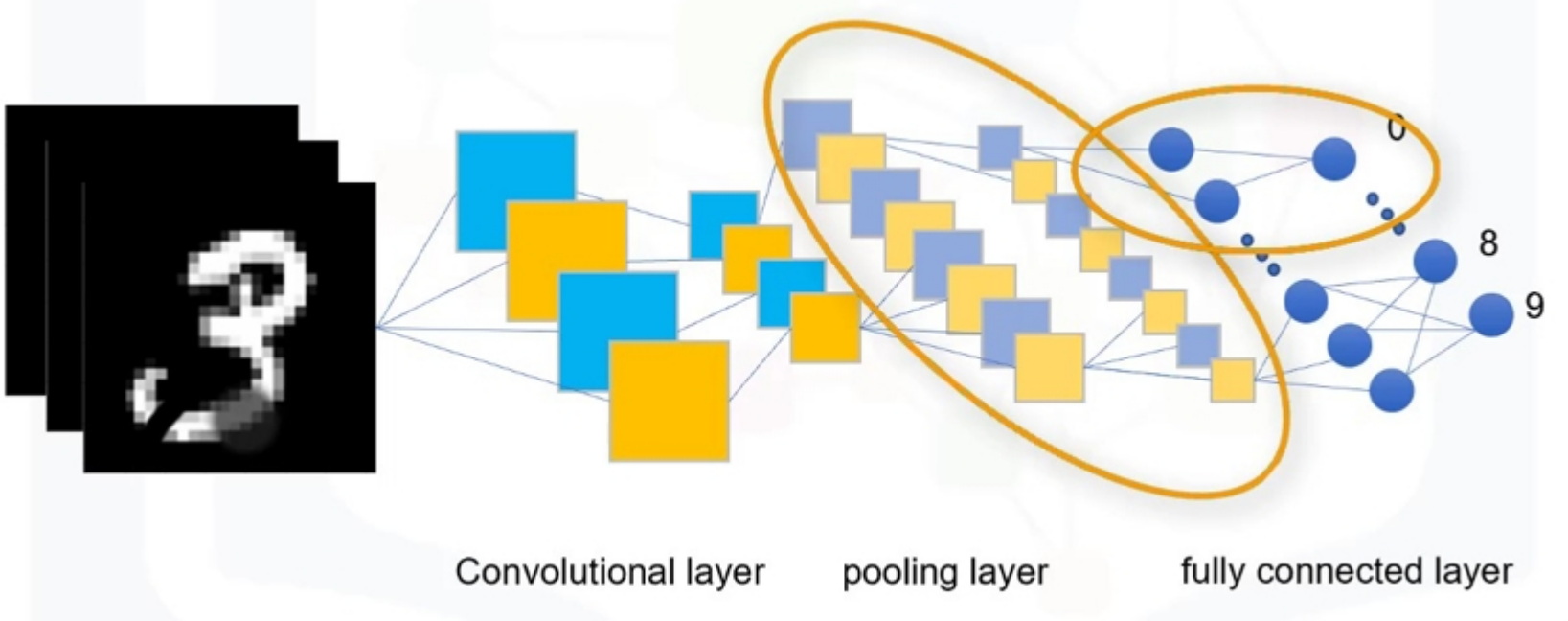
Digit Recognition



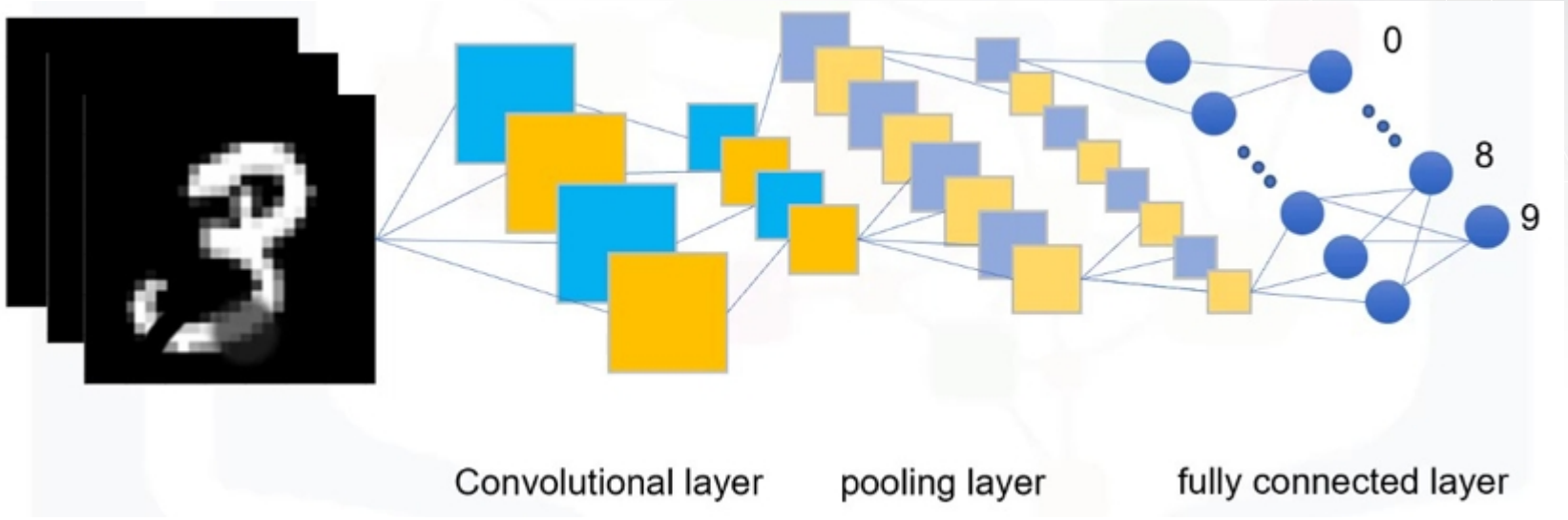
Digit Recognition



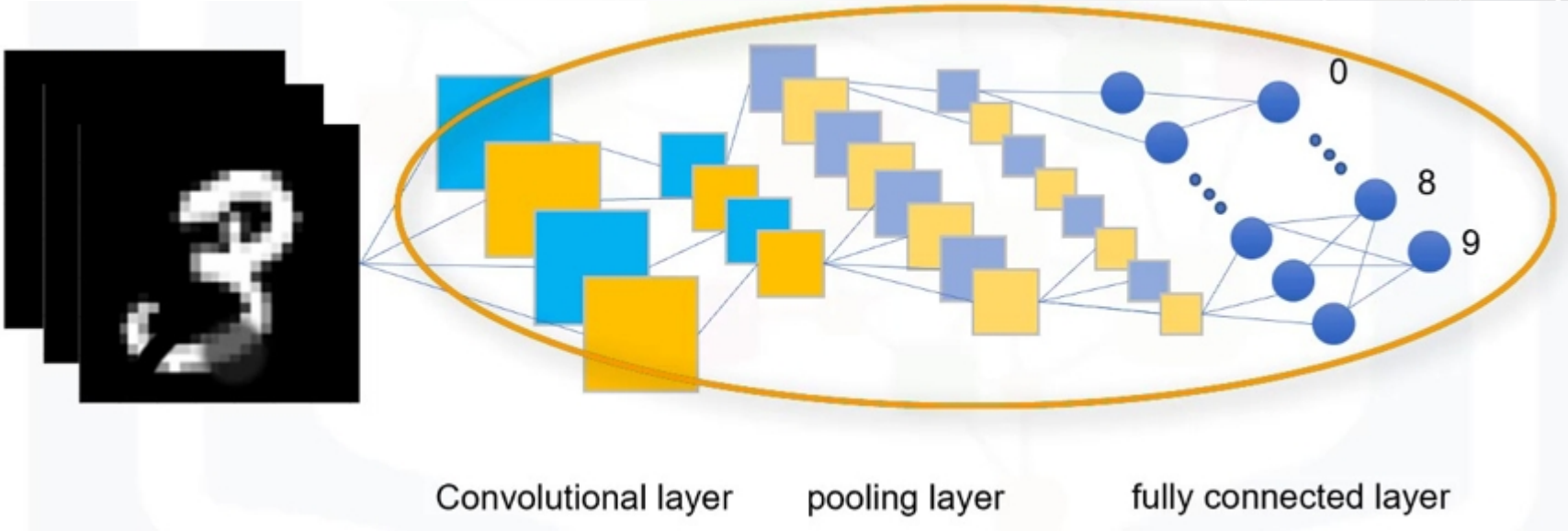
Digit Recognition



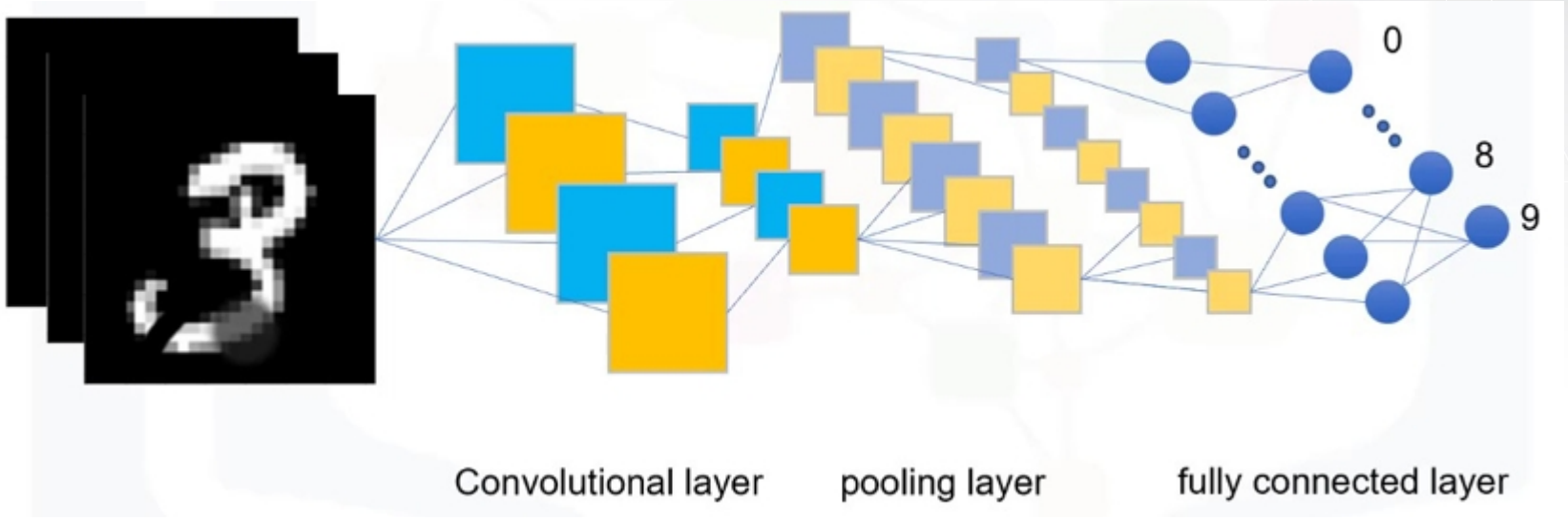
Digit Recognition



Digit Recognition



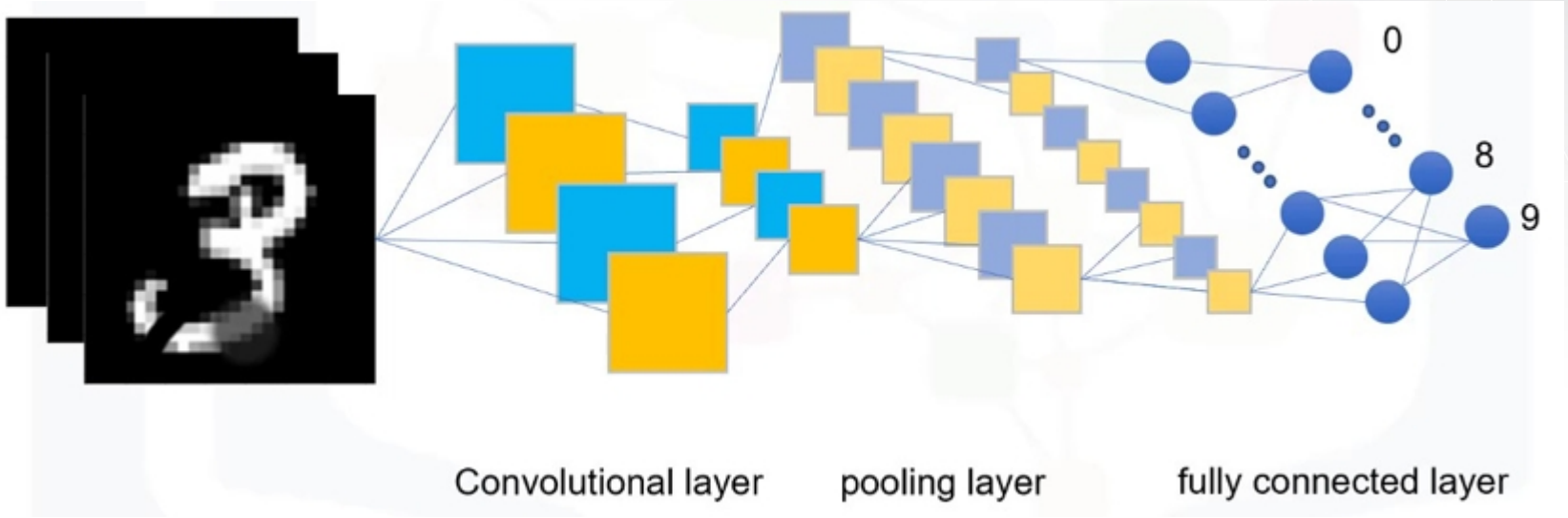
Digit Recognition



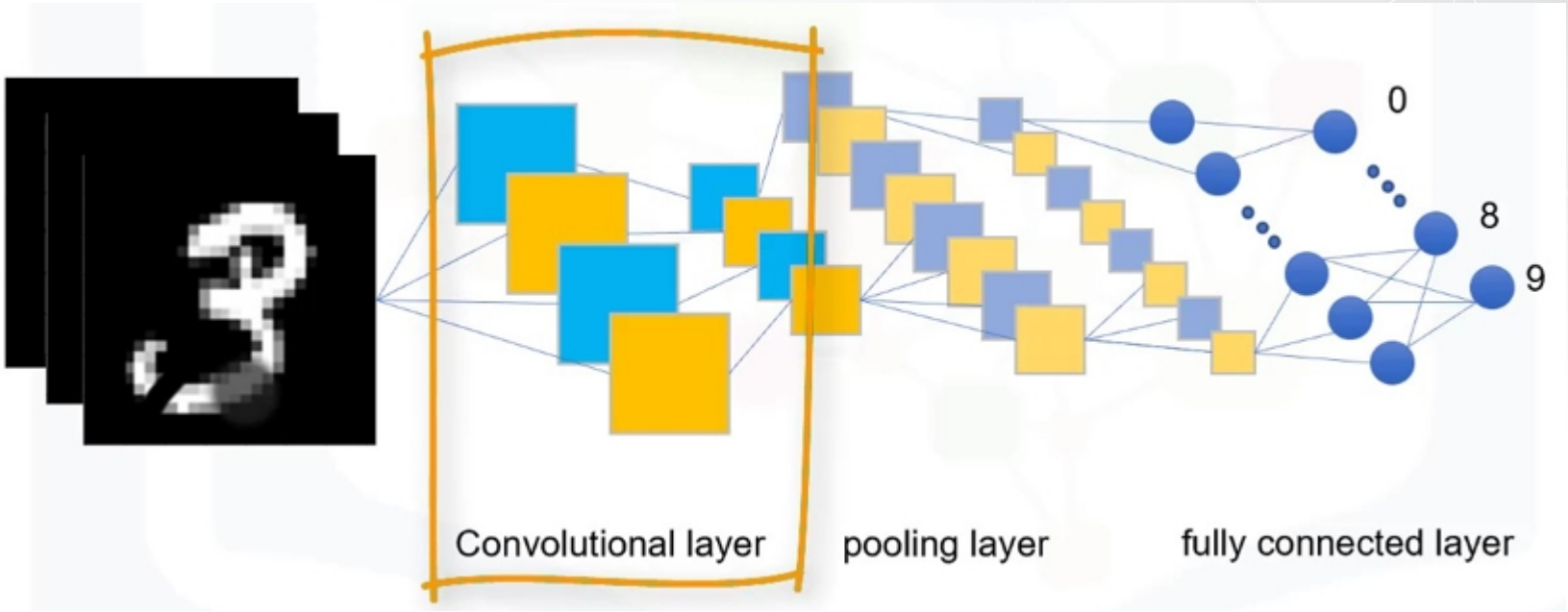
Bagian 3

Convolutional Neural Networks

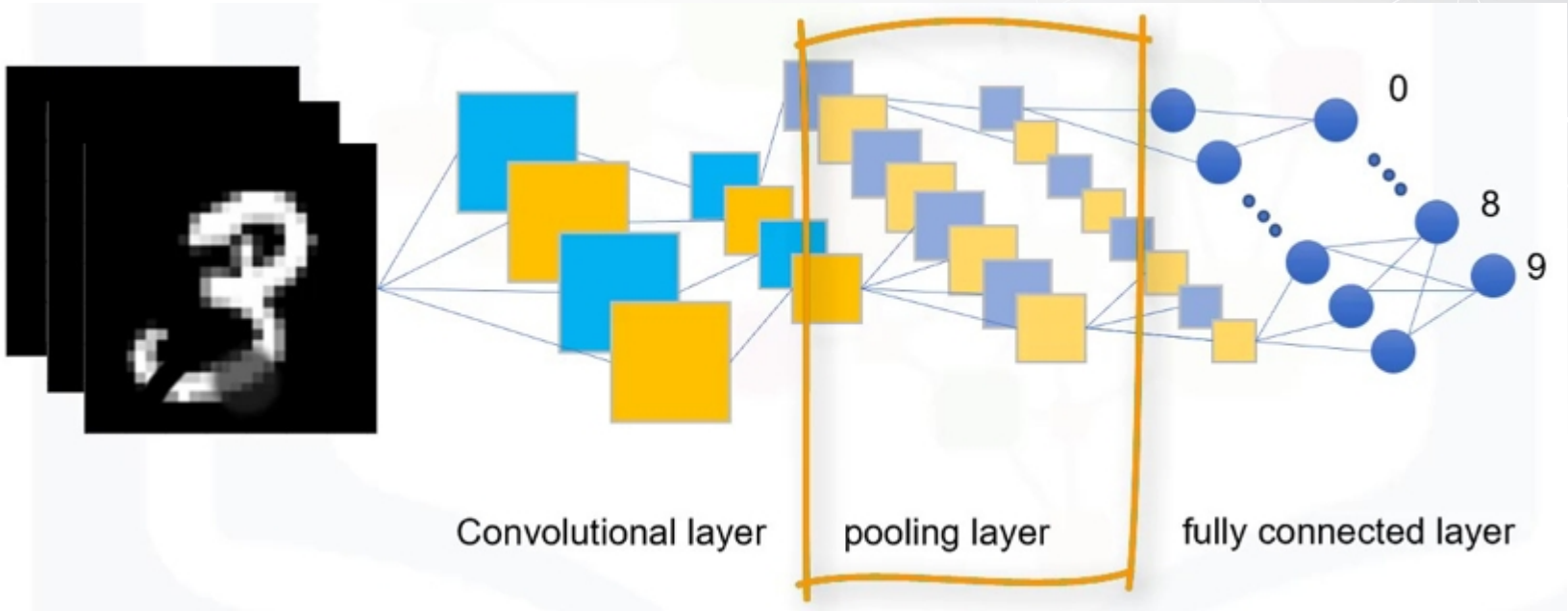
Digit Recognition



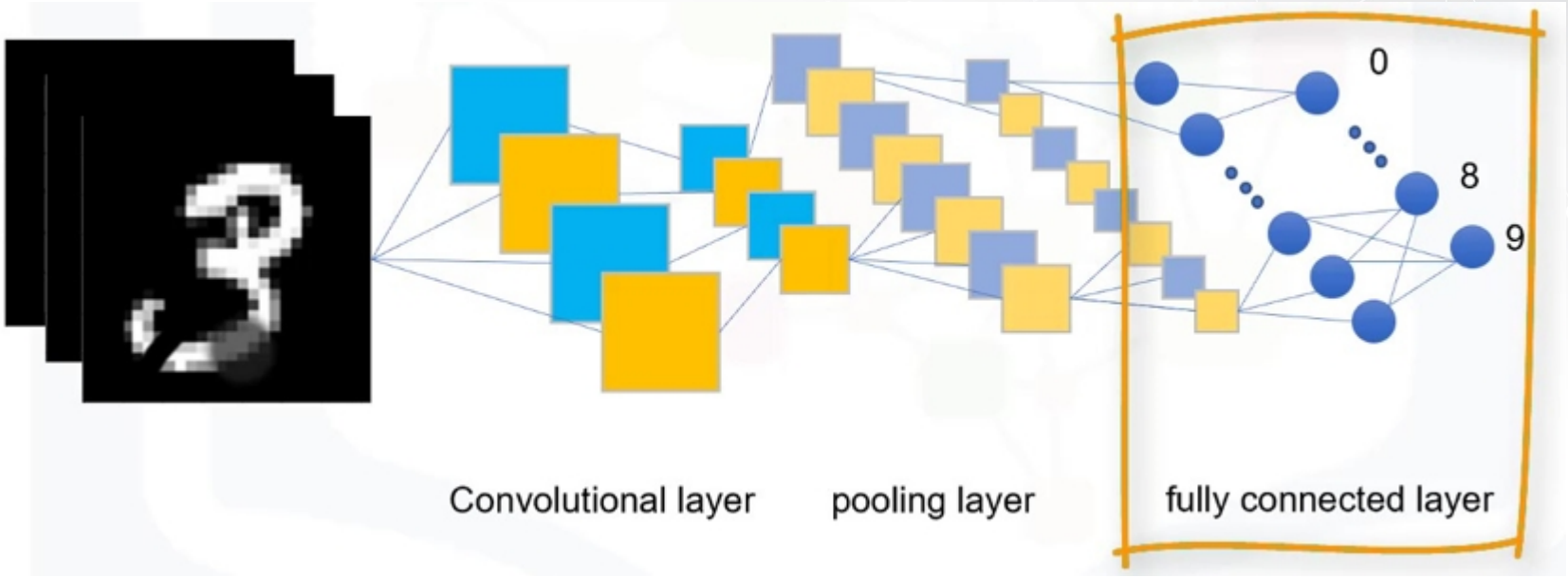
Digit Recognition



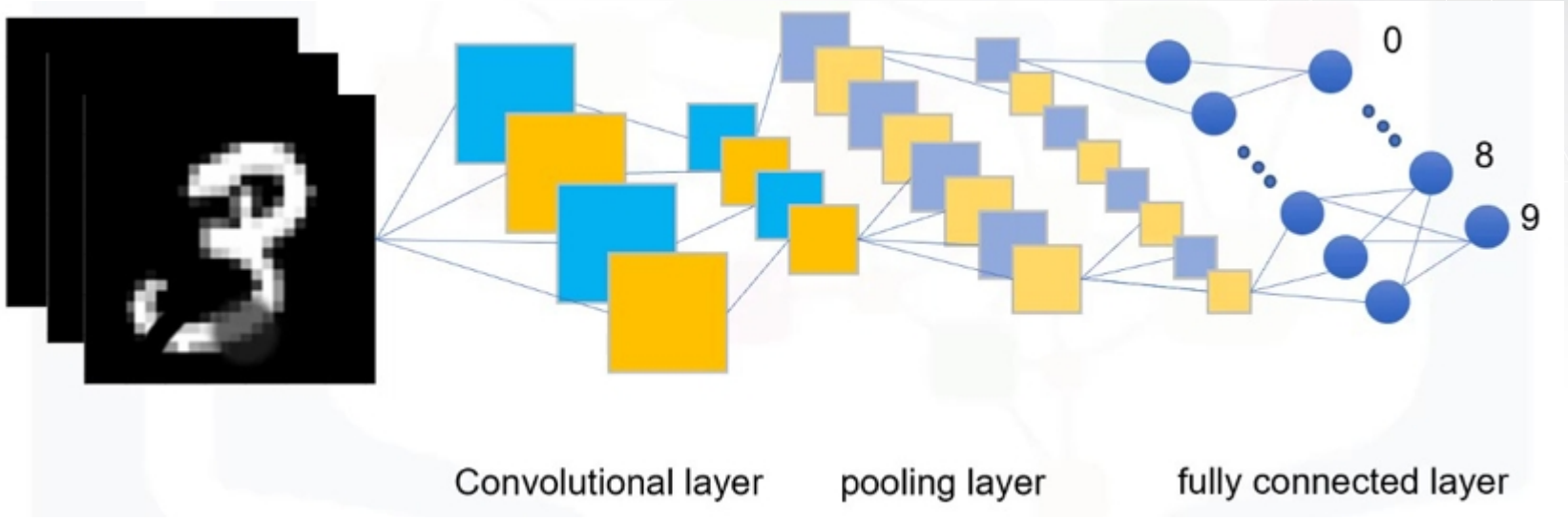
Digit Recognition



Digit Recognition



Digit Recognition

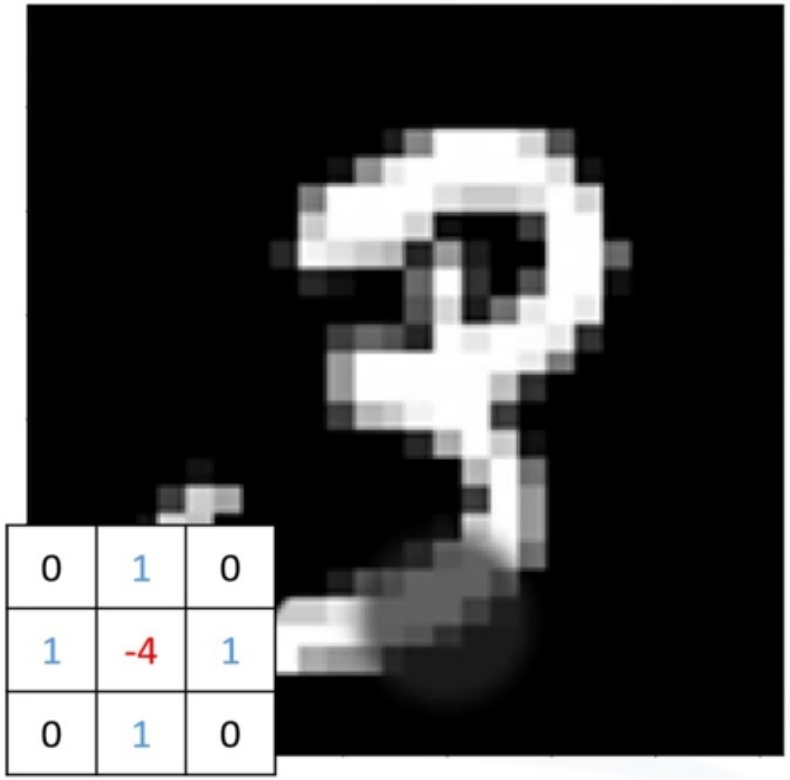


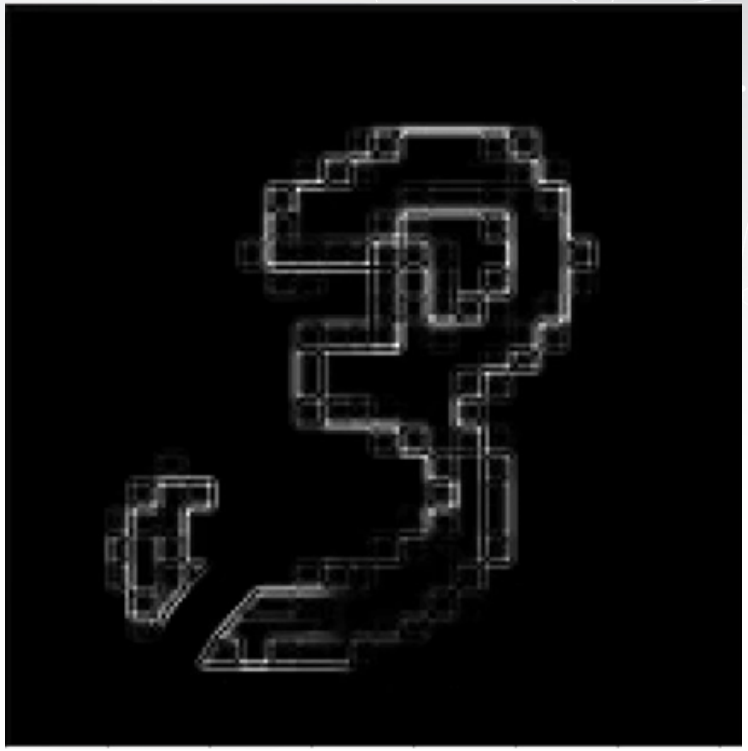
Understanding Convolution

Understanding Convolution

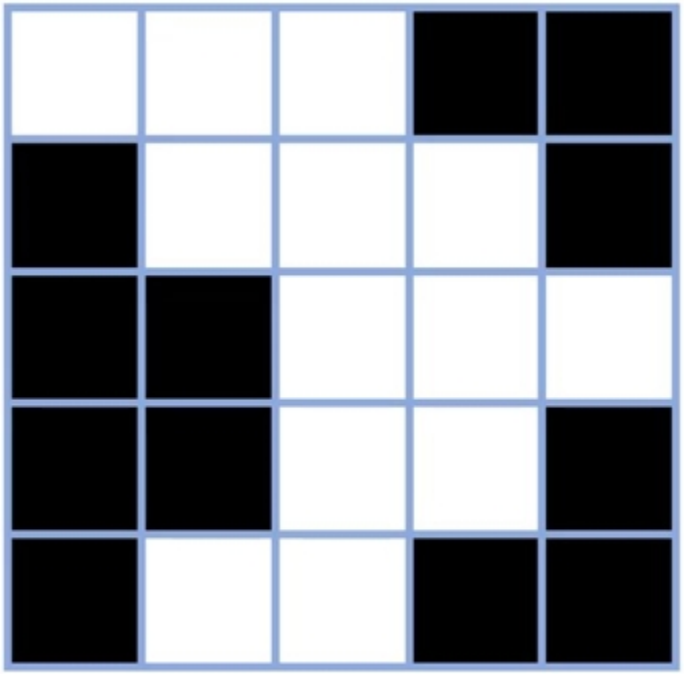
0	1	0
1	-4	1
0	1	0

Understanding Convolution





Understanding Convolution



Image

Understanding Convolution

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

Understanding Convolution

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

Understanding Convolution

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

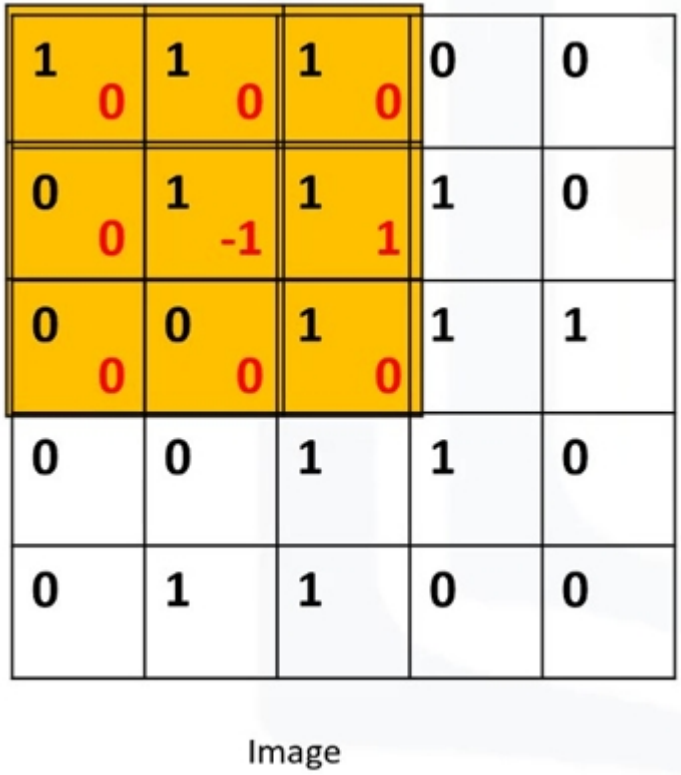
Understanding Convolution

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

0	0	0
0	-1	1
0	0	0

Understanding Convolution

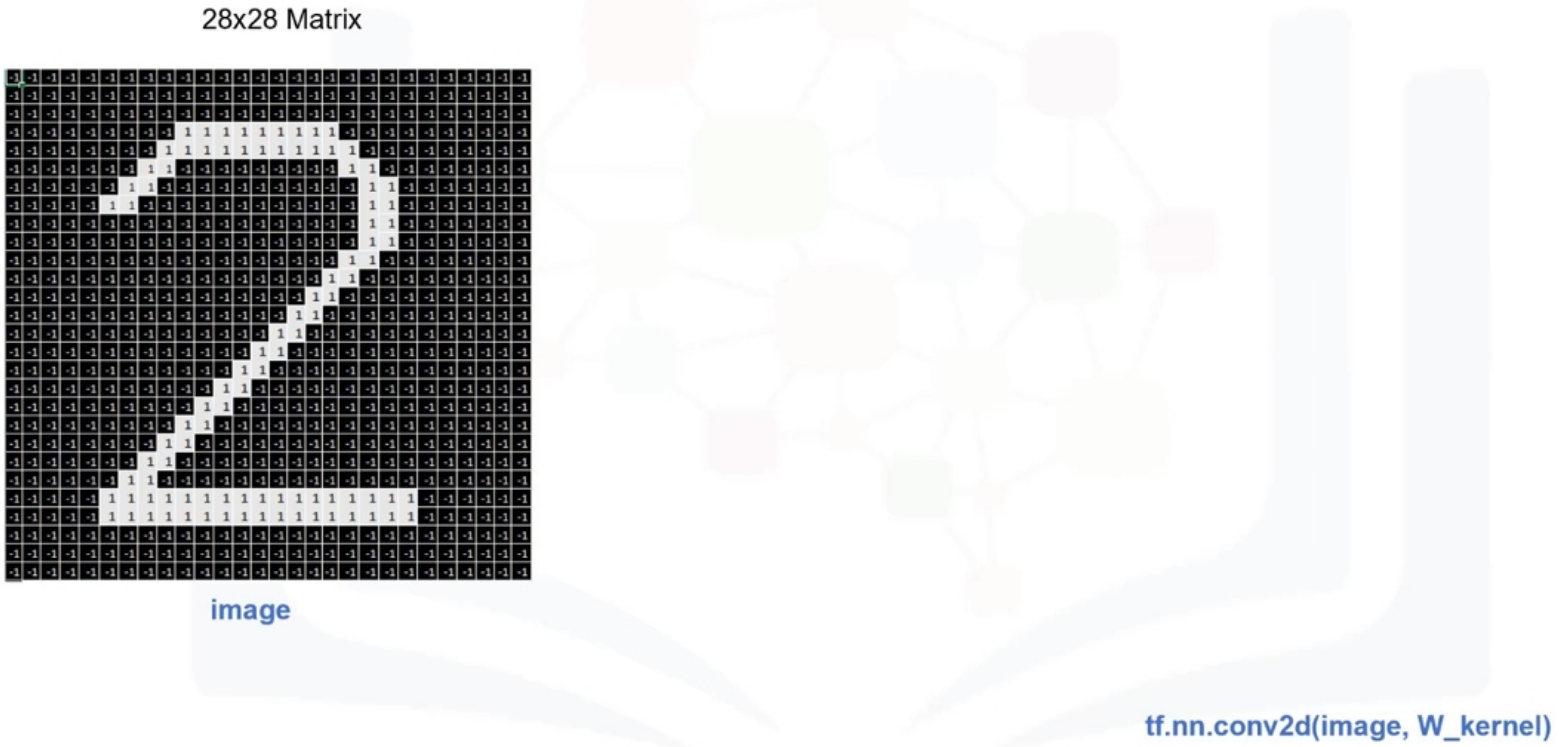


Image

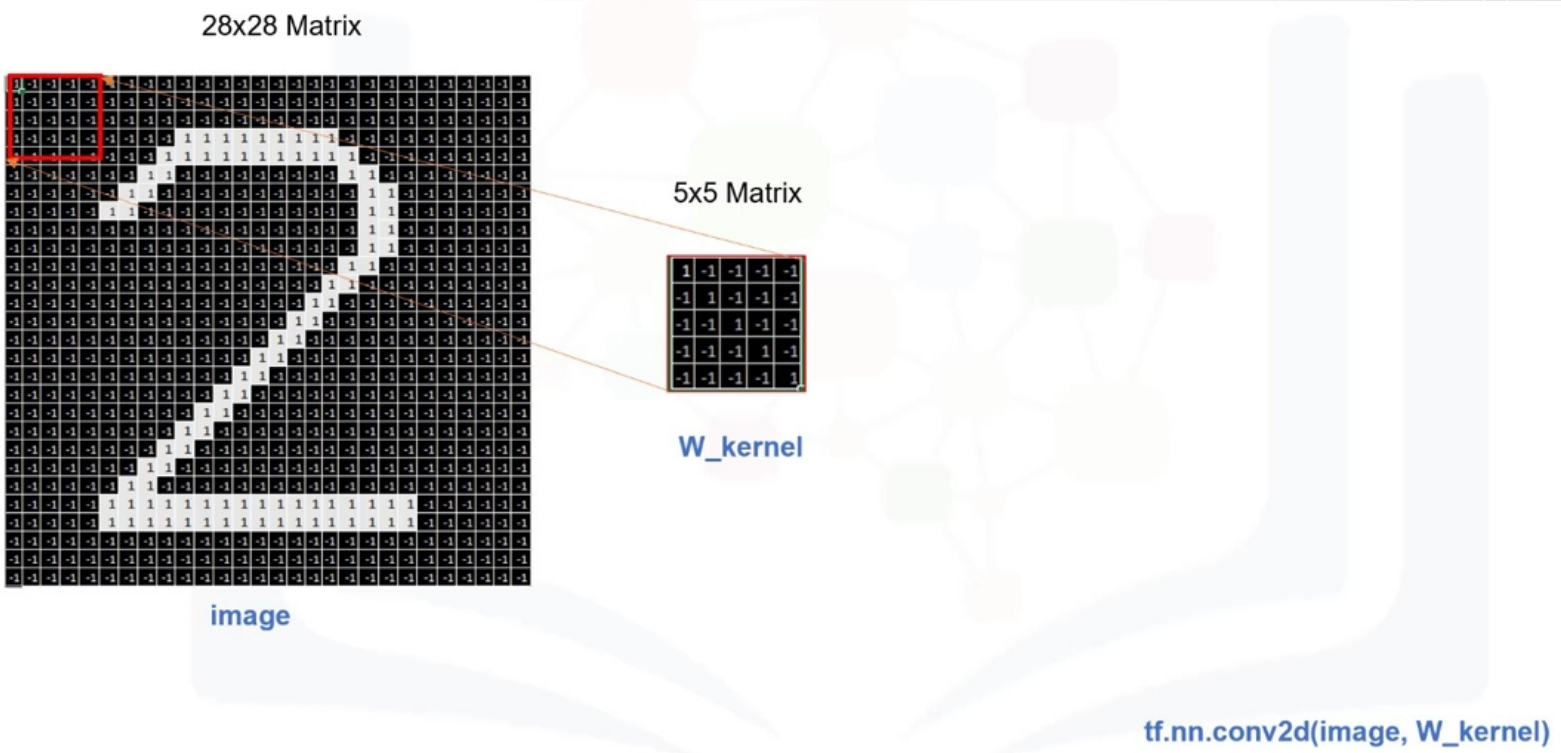
Understanding Convolution

Understanding Convolution

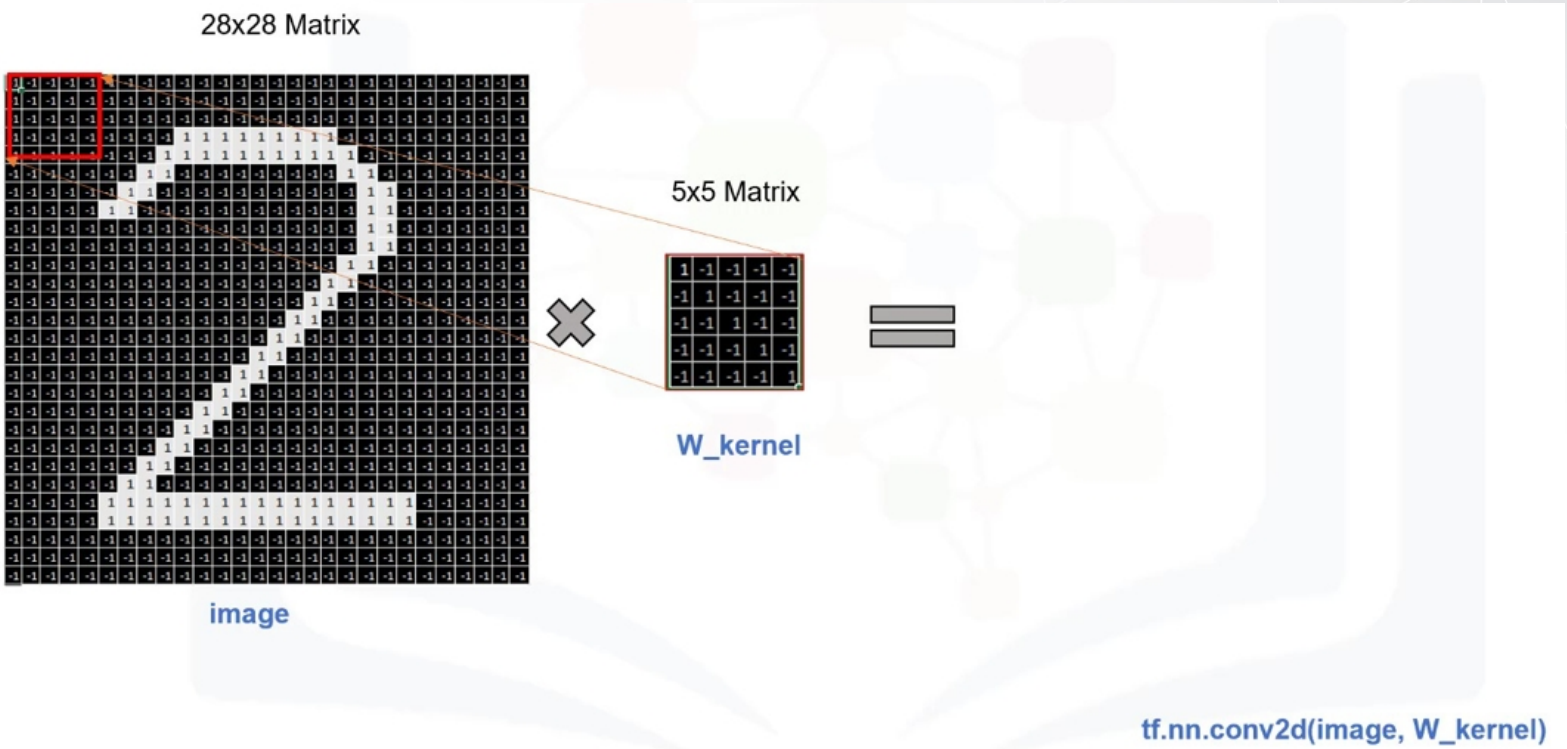
Understanding Convolution



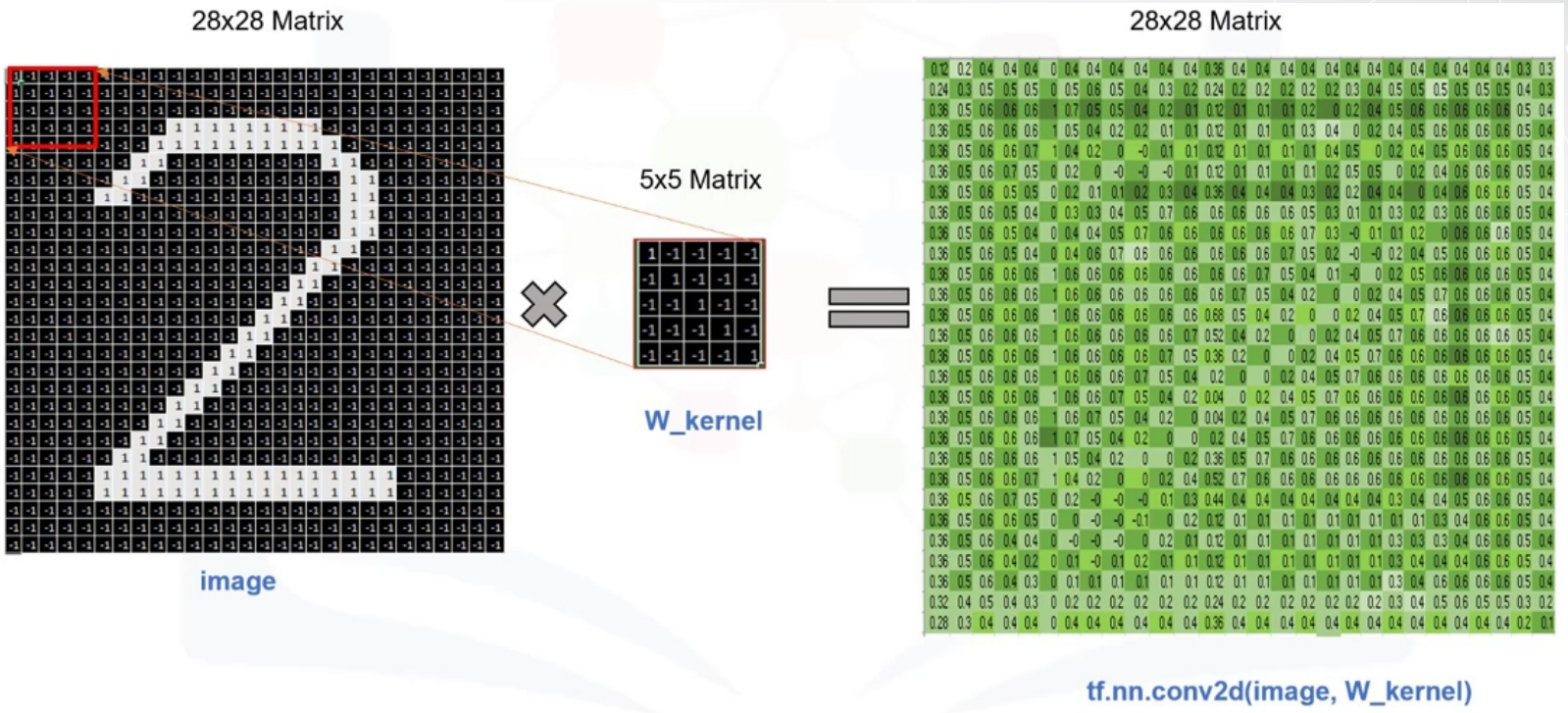
Understanding Convolution



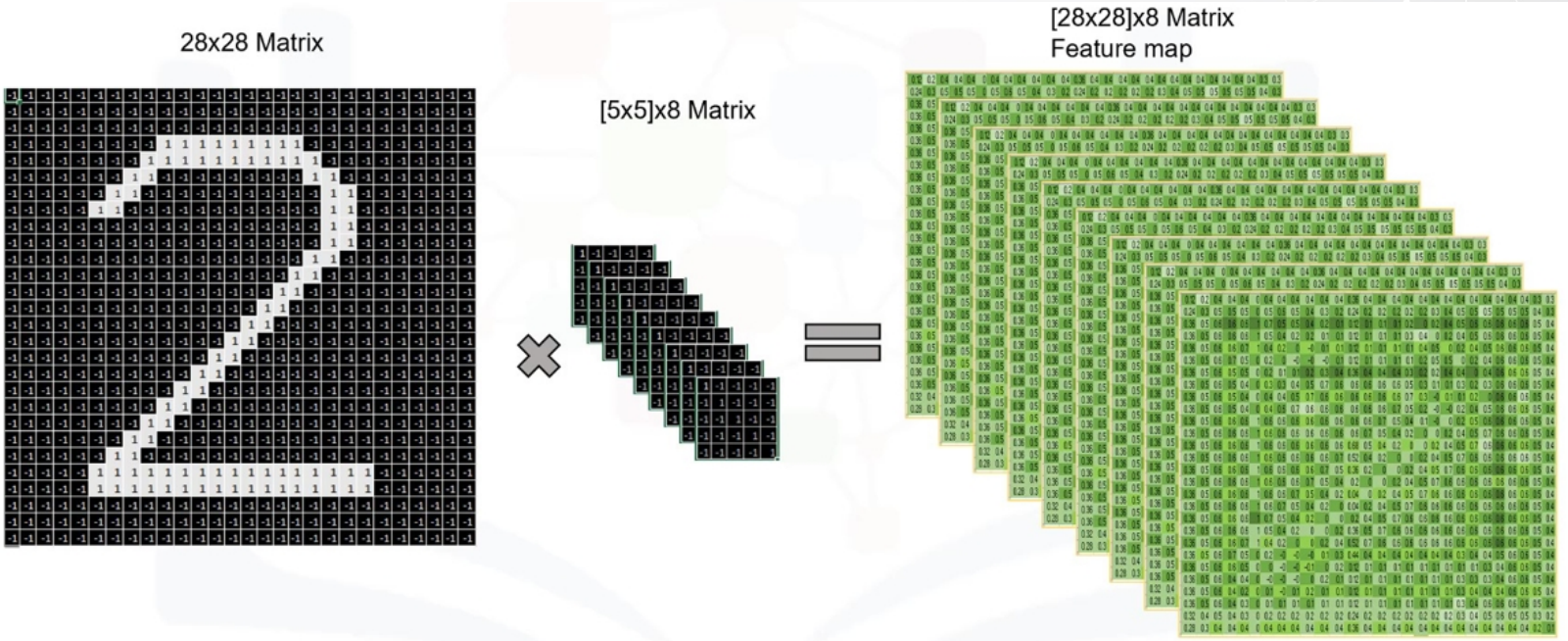
Understanding Convolution



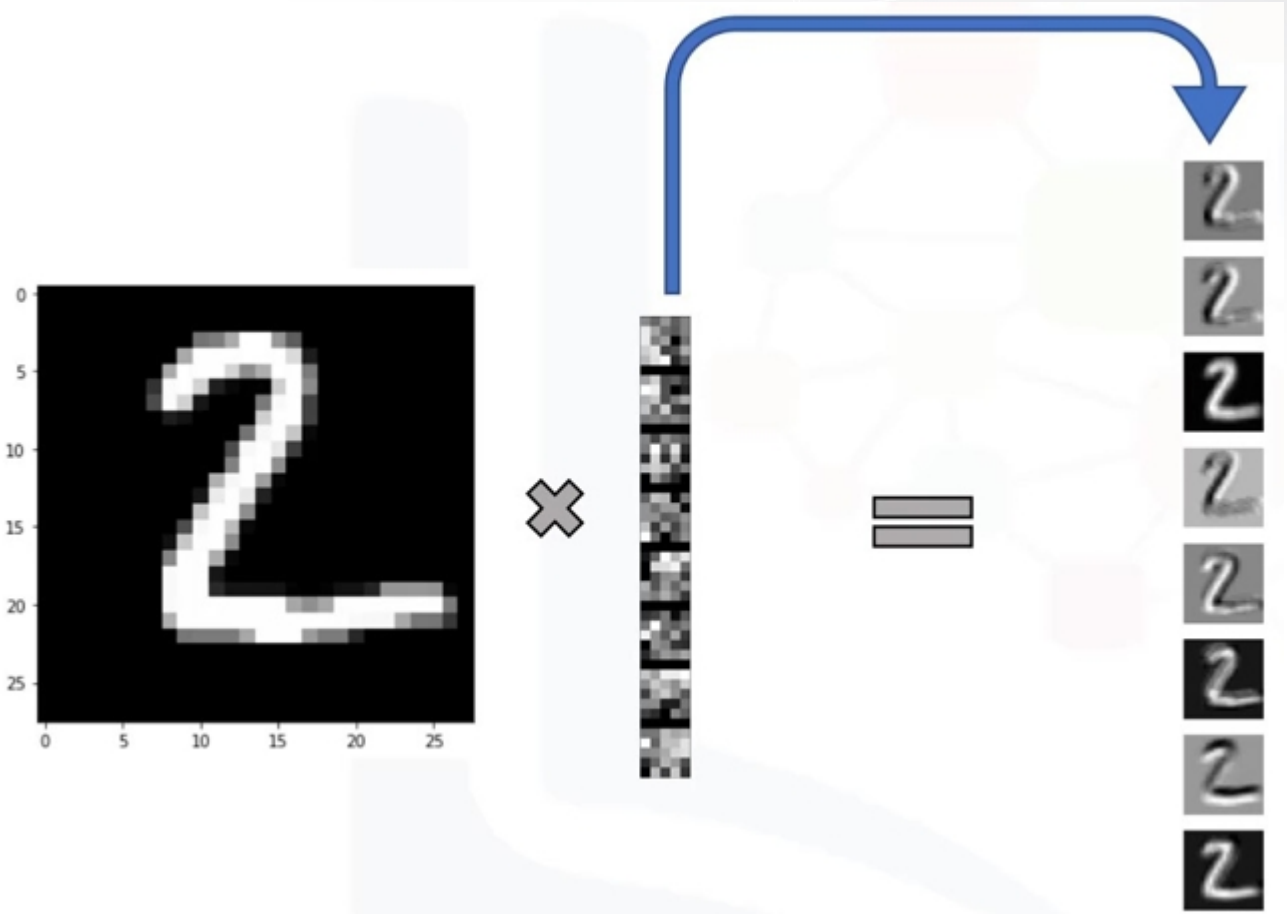
Understanding Convolution



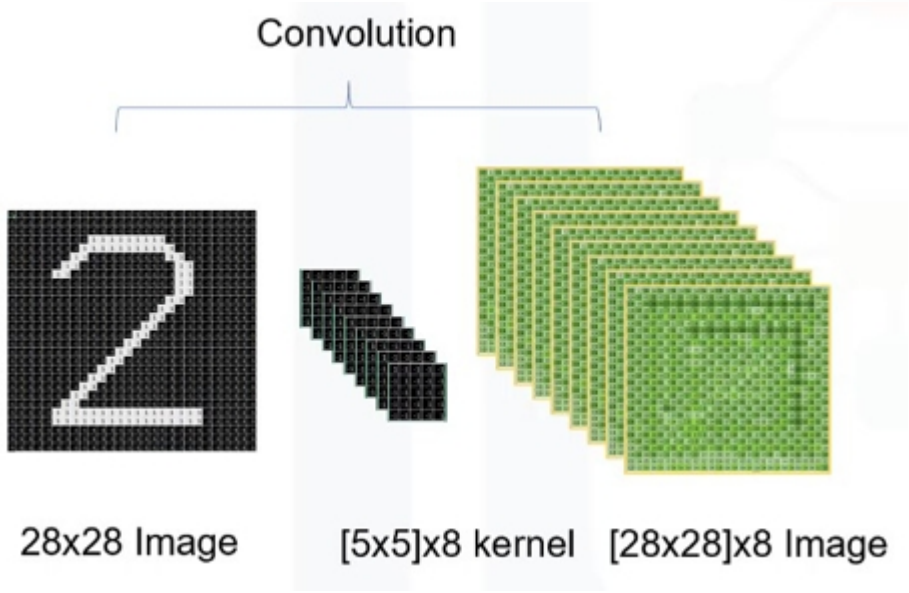
Understanding Convolution



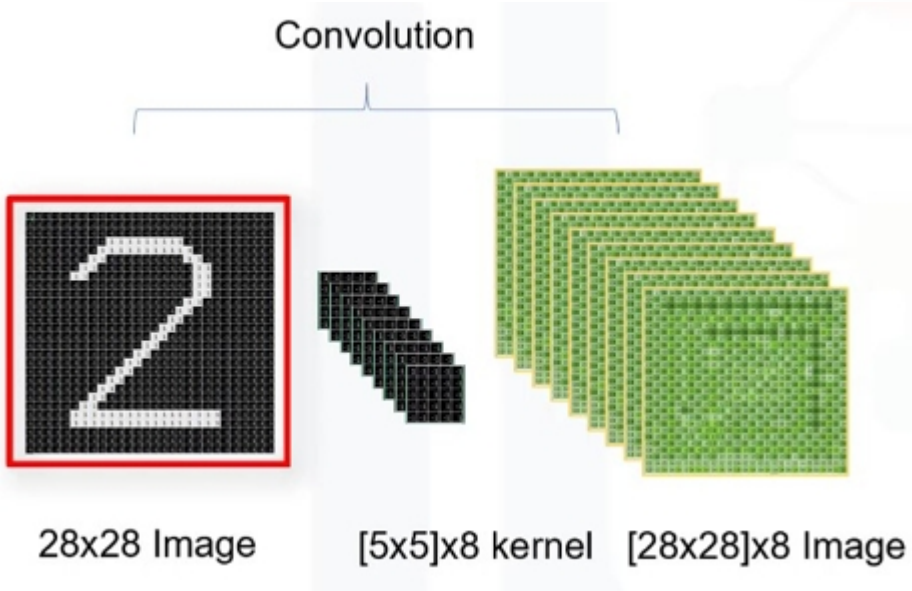
Convolution Layer



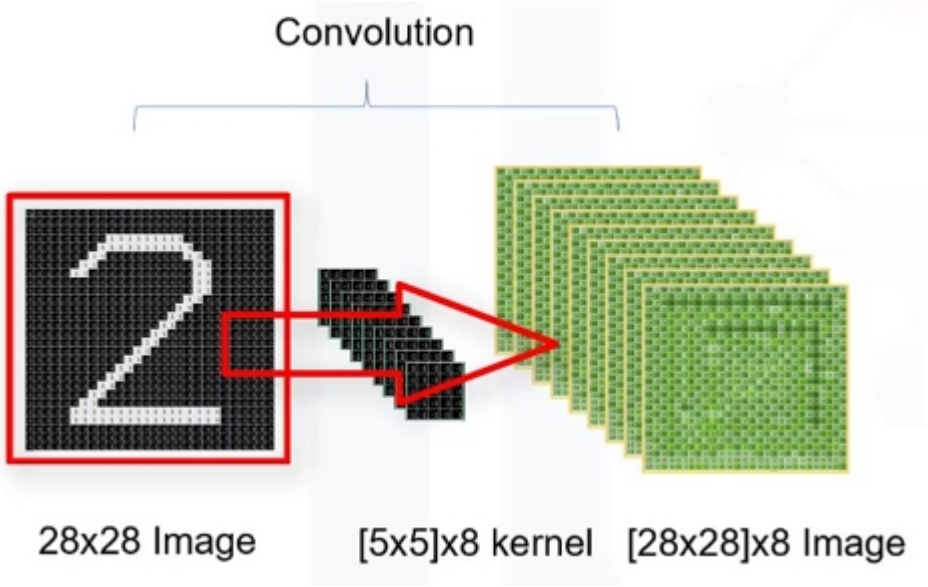
CNN Architecture



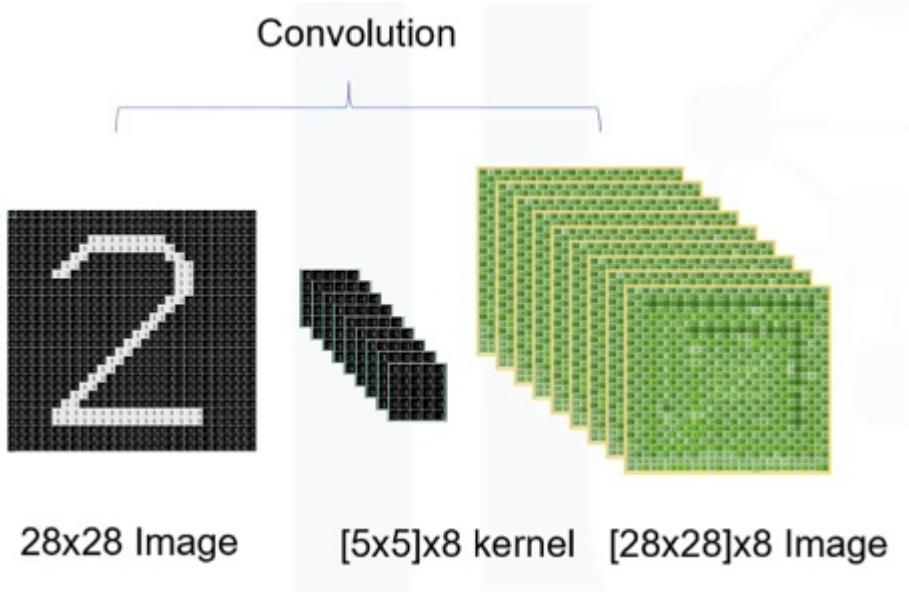
CNN Architecture



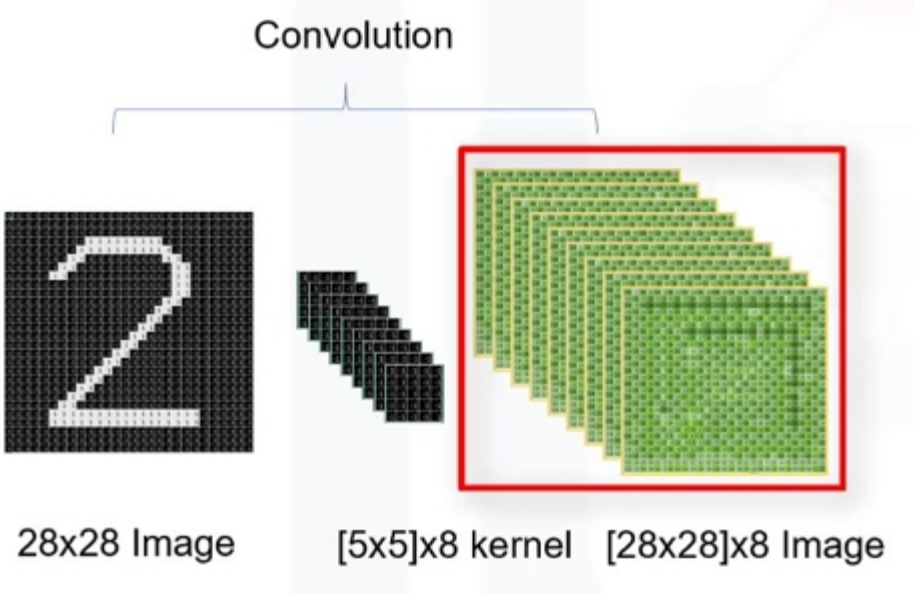
CNN Architecture



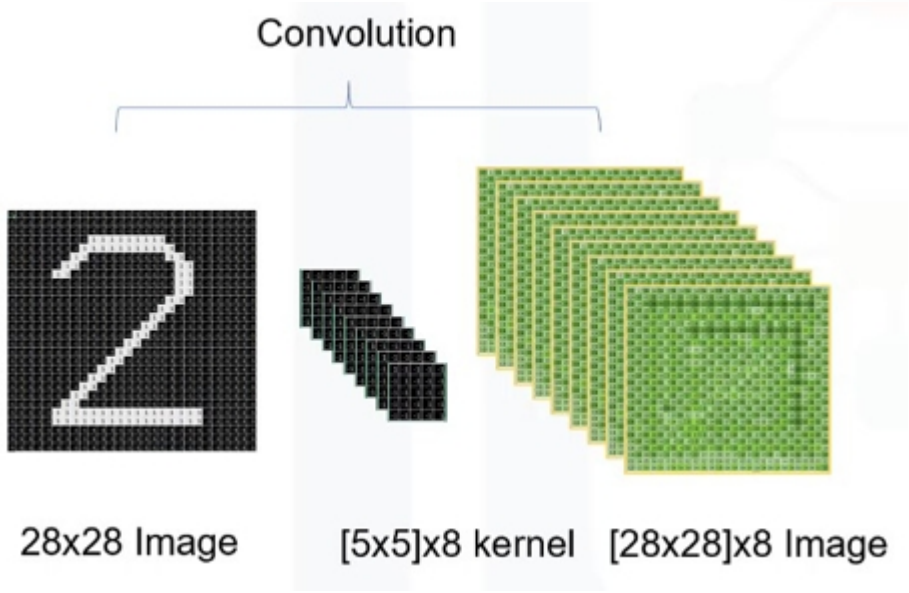
CNN Architecture



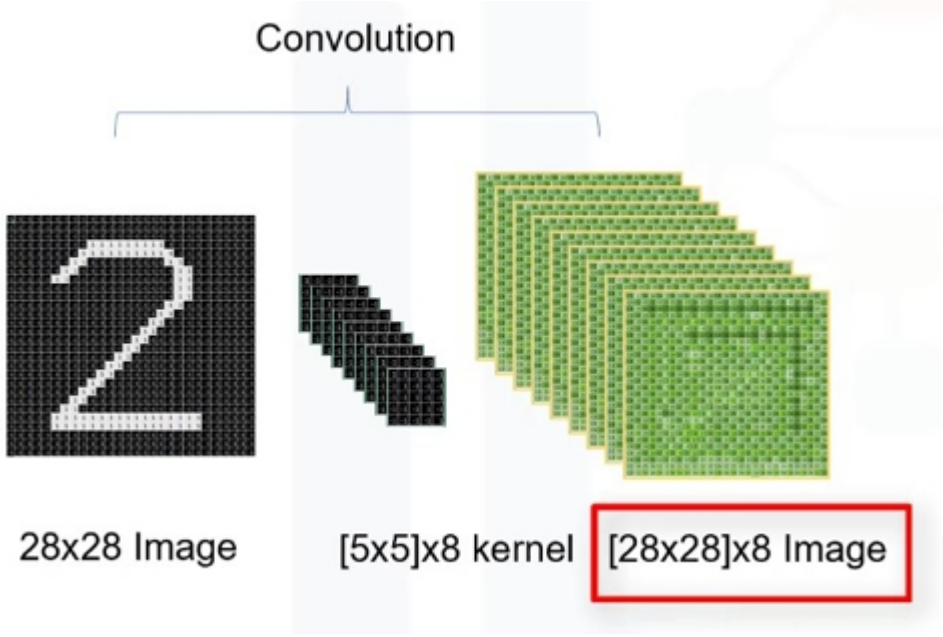
CNN Architecture



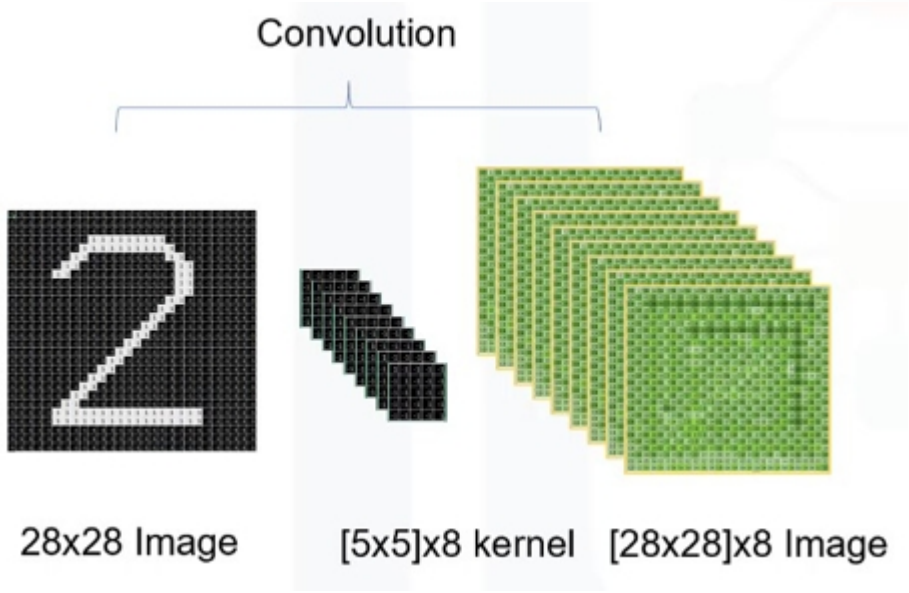
CNN Architecture



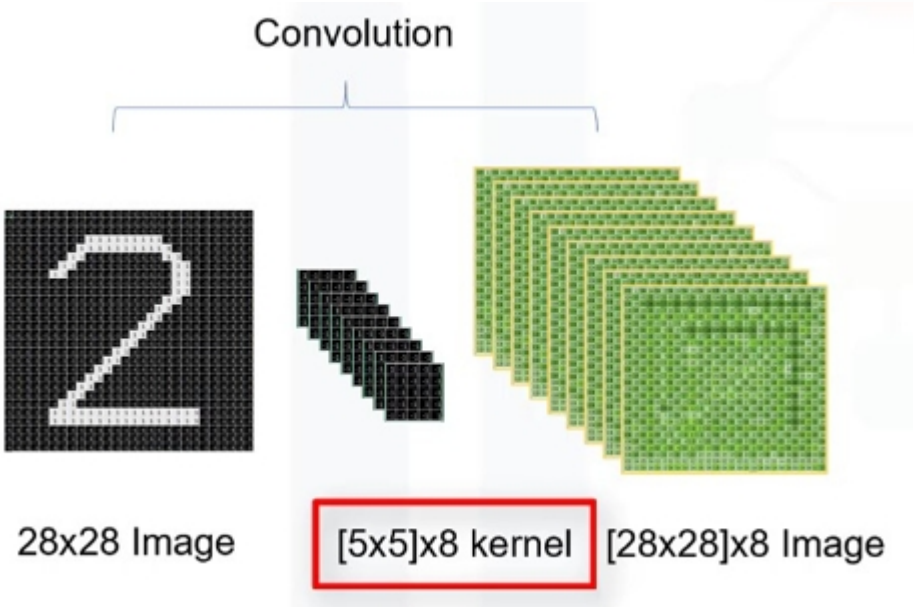
CNN Architecture



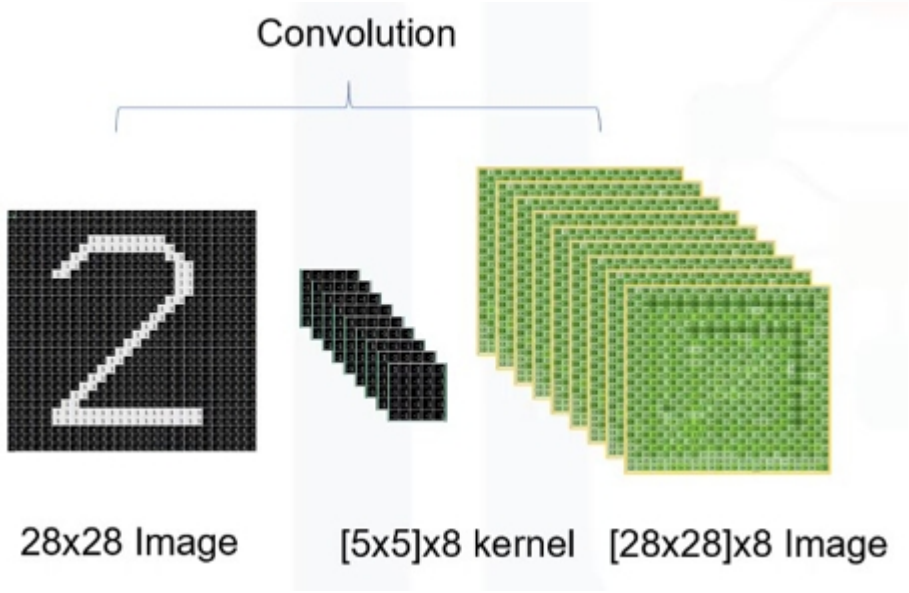
CNN Architecture



CNN Architecture



CNN Architecture



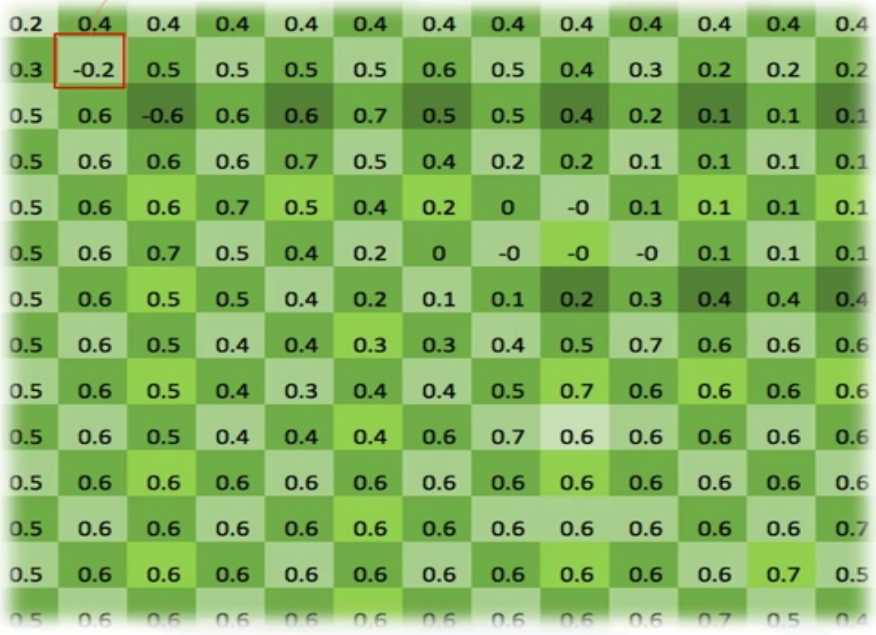
ReLu

0.2	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4
0.3	-0.2	0.5	0.5	0.5	0.5	0.6	0.5	0.4	0.3	0.2	0.2	0.2
0.5	0.6	-0.6	0.6	0.6	0.7	0.5	0.5	0.4	0.2	0.1	0.1	0.1
0.5	0.6	0.6	0.6	0.7	0.5	0.4	0.2	0.2	0.1	0.1	0.1	0.1
0.5	0.6	0.6	0.7	0.5	0.4	0.2	0	-0	0.1	0.1	0.1	0.1
0.5	0.6	0.7	0.5	0.4	0.2	0	-0	-0	-0	0.1	0.1	0.1
0.5	0.6	0.5	0.5	0.4	0.2	0.1	0.1	0.2	0.3	0.4	0.4	0.4
0.5	0.6	0.5	0.4	0.4	0.3	0.3	0.4	0.5	0.7	0.6	0.6	0.6
0.5	0.6	0.5	0.4	0.3	0.4	0.4	0.5	0.7	0.6	0.6	0.6	0.6
0.5	0.6	0.5	0.4	0.4	0.4	0.6	0.7	0.6	0.6	0.6	0.6	0.6
0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6
0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7
0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5
0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5	0.4

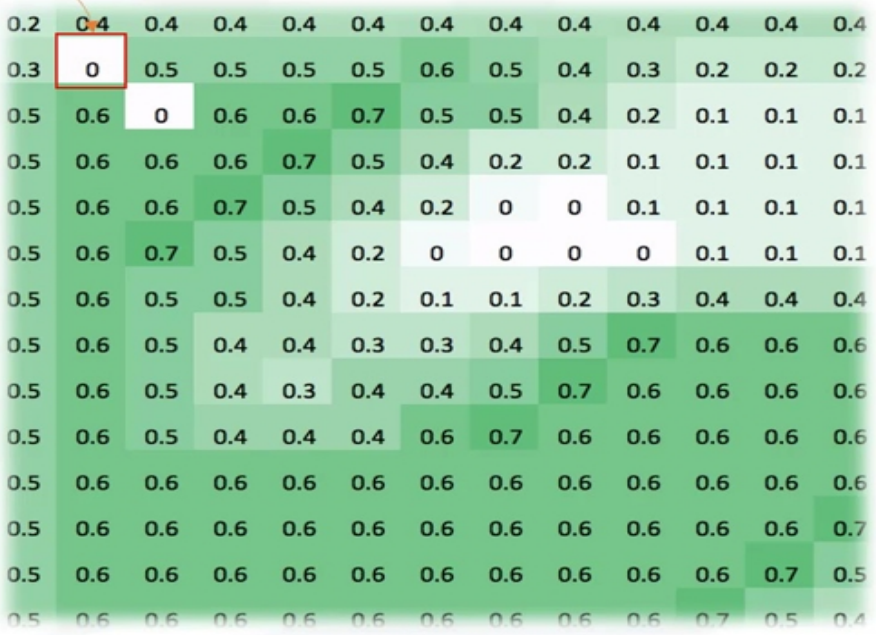
convolved1

0.2	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4
0.3	0	0.5	0.5	0.5	0.5	0.6	0.5	0.4	0.3	0.2	0.2	0.2
0.5	0.6	0	0.6	0.6	0.7	0.5	0.5	0.4	0.2	0.1	0.1	0.1
0.5	0.6	0.6	0.6	0.7	0.5	0.4	0.2	0.2	0.1	0.1	0.1	0.1
0.5	0.6	0.6	0.7	0.5	0.4	0.2	0	0	0.1	0.1	0.1	0.1
0.5	0.6	0.7	0.5	0.4	0.2	0	0	0	0	0.1	0.1	0.1
0.5	0.6	0.5	0.5	0.4	0.2	0.1	0.1	0.2	0.3	0.4	0.4	0.4
0.5	0.6	0.5	0.4	0.4	0.3	0.3	0.4	0.5	0.7	0.6	0.6	0.6
0.5	0.6	0.5	0.4	0.3	0.4	0.4	0.5	0.7	0.6	0.6	0.6	0.6
0.5	0.6	0.5	0.4	0.4	0.4	0.6	0.7	0.6	0.6	0.6	0.6	0.6
0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6
0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7
0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5
0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5

ReLu

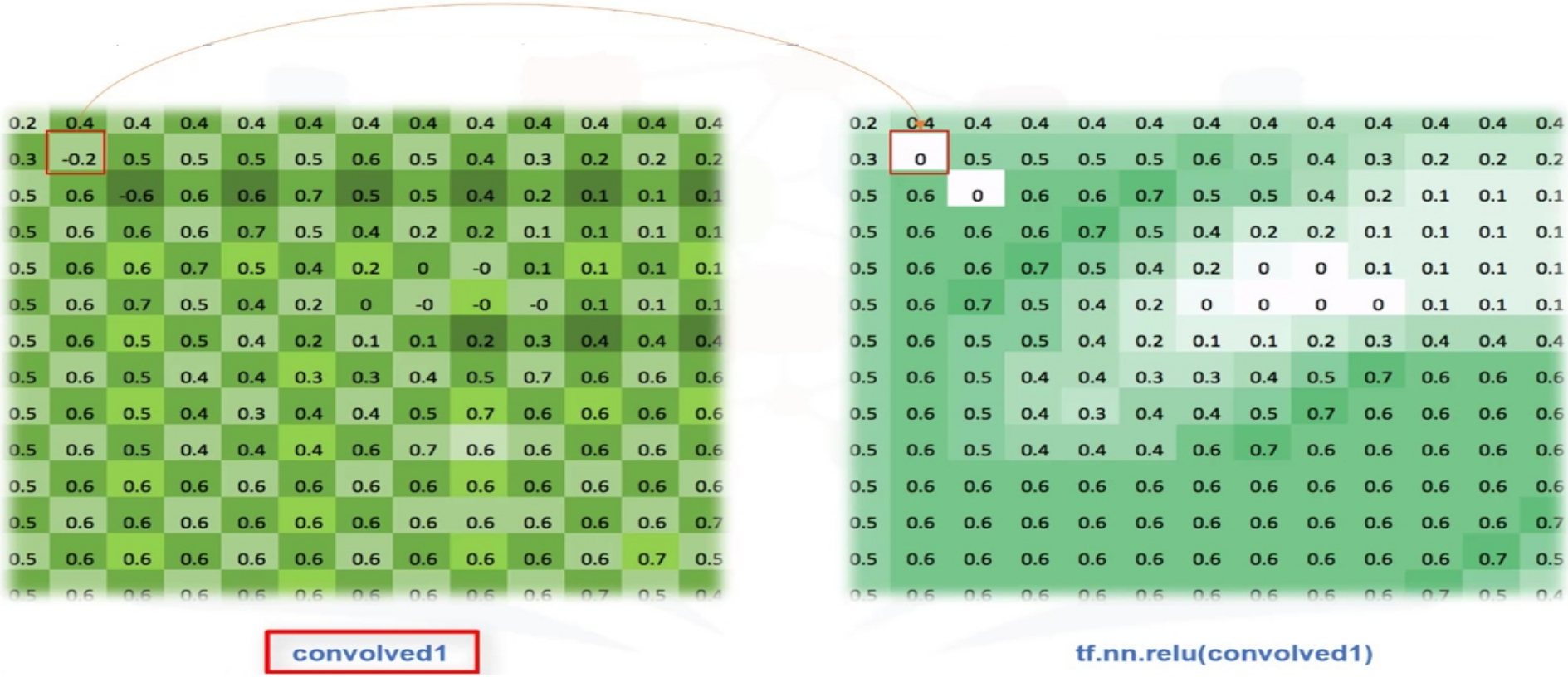


convolved1

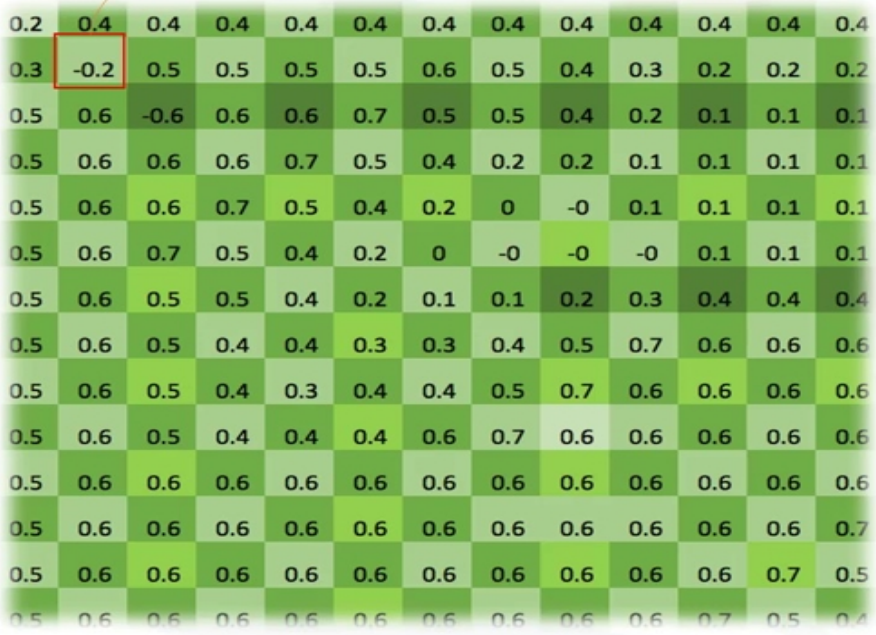


tf.nn.relu(convolved1)

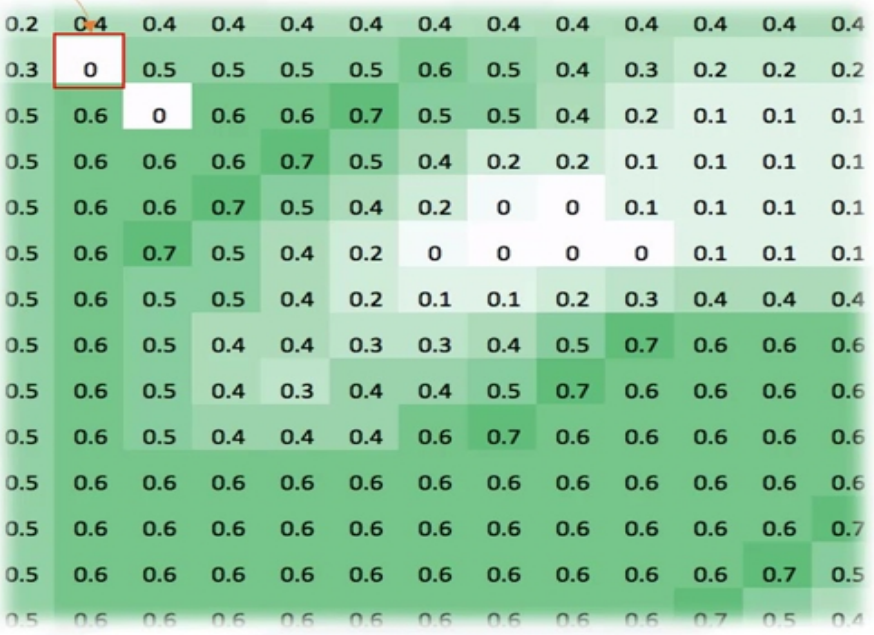
ReLu



ReLu

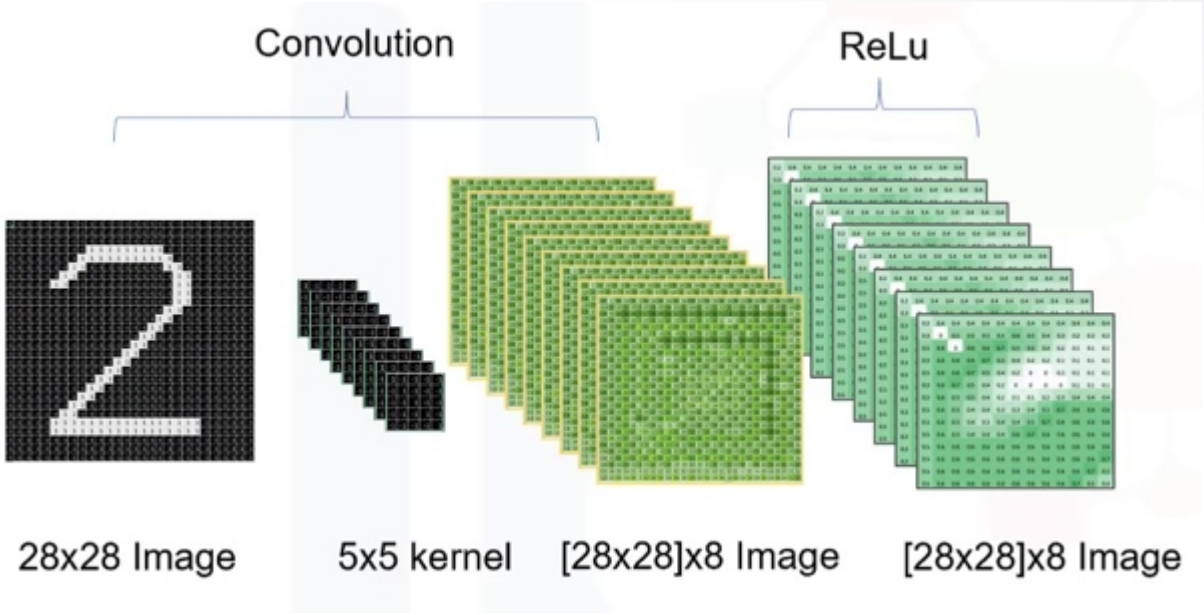


convolved1



tf.nn.relu(convolved1)

CNN Architecture - ReLu



Max Pooling

0.1	0.2	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4
0.2	0.3	0.5	0.5	0.5	0.5	0.5	0.6	0.5	0.4	0.3	0.2	0.2	0.2	0.2
0.4	0.5	0.6	0.6	0.6	0.6	0.7	0.5	0.5	0.4	0.2	0.1	0.1	0.1	0.1
0.4	0.5	0.6	0.6	0.6	0.7	0.5	0.4	0.2	0.2	0.1	0.1	0.1	0.1	0.1
0.4	0.5	0.6	0.6	0.7	0.5	0.4	0.2	0	0	0.1	0.1	0.1	0.1	0.1
0.4	0.5	0.6	0.7	0.5	0.4	0.2	0	0	0	0	0.1	0.1	0.1	0.1
0.4	0.5	0.6	0.5	0.5	0.4	0.2	0.1	0.1	0.2	0.3	0.4	0.4	0.4	0.4
0.4	0.5	0.6	0.5	0.4	0.4	0.3	0.3	0.4	0.5	0.7	0.6	0.6	0.6	0.6
0.4	0.5	0.6	0.5	0.4	0.3	0.4	0.4	0.5	0.7	0.6	0.6	0.6	0.6	0.6
0.4	0.5	0.6	0.5	0.4	0.4	0.4	0.6	0.7	0.6	0.6	0.6	0.6	0.6	0.6
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5	0.4
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5	0.4	0.2
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5	0.4	0.2	0
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5	0.4	0.2	0	0

conv1_ReLu

0.3	0.5	0.5	0.6	0.5	0.4	0.4	0.4
0.5	0.6	0.7	0.7	0.5	0.2	0.1	0.1
0.5	0.7	0.7	0.4	0	0.1	0.1	0.1
0.5	0.6	0.5	0.3	0.5	0.7	0.6	0.6
0.5	0.6	0.4	0.6	0.7	0.6	0.6	0.7
0.5	0.6	0.6	0.6	0.6	0.6	0.7	0.7
0.5	0.6	0.6	0.6	0.6	0.7	0.7	0.4
0.5	0.6	0.6	0.6	0.7	0.7	0.4	0.2

tf.nn.max_pool(conv1_ReLu, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1])

Max Pooling

0.1	0.2	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4
0.2	0.3	0.5	0.5	0.5	0.5	0.5	0.6	0.5	0.4	0.3	0.2	0.2	0.2	0.2
0.4	0.5	0.6	0.6	0.6	0.6	0.7	0.5	0.5	0.4	0.2	0.1	0.1	0.1	0.1
0.4	0.5	0.6	0.6	0.6	0.7	0.5	0.4	0.2	0.2	0.1	0.1	0.1	0.1	0.1
0.4	0.5	0.6	0.6	0.7	0.5	0.4	0.2	0	0	0.1	0.1	0.1	0.1	0.1
0.4	0.5	0.6	0.7	0.5	0.4	0.2	0	0	0	0	0.1	0.1	0.1	0.1
0.4	0.5	0.6	0.5	0.5	0.4	0.2	0.1	0.1	0.2	0.3	0.4	0.4	0.4	0.4
0.4	0.5	0.6	0.5	0.4	0.4	0.3	0.3	0.4	0.5	0.7	0.6	0.6	0.6	0.6
0.4	0.5	0.6	0.5	0.4	0.3	0.4	0.4	0.5	0.7	0.6	0.6	0.6	0.6	0.6
0.4	0.5	0.6	0.5	0.4	0.4	0.4	0.6	0.7	0.6	0.6	0.6	0.6	0.6	0.6
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5	0.4
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5	0.4	0.2
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5	0.4	0.2	0
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5	0.4	0.2	0	0

conv1_ReLu

0.3	0.5	0.5	0.6	0.5	0.4	0.4	0.4
0.5	0.6	0.7	0.7	0.5	0.2	0.1	0.1
0.5	0.7	0.7	0.4	0	0.1	0.1	0.1
0.5	0.6	0.5	0.3	0.5	0.7	0.6	0.6
0.5	0.6	0.4	0.6	0.7	0.6	0.6	0.7
0.5	0.6	0.6	0.6	0.6	0.6	0.7	0.7
0.5	0.6	0.6	0.6	0.6	0.7	0.7	0.4
0.5	0.6	0.6	0.6	0.7	0.7	0.4	0.2

```
tf.nn.max_pool(conv1_ReLu, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1])
```

Max Pooling

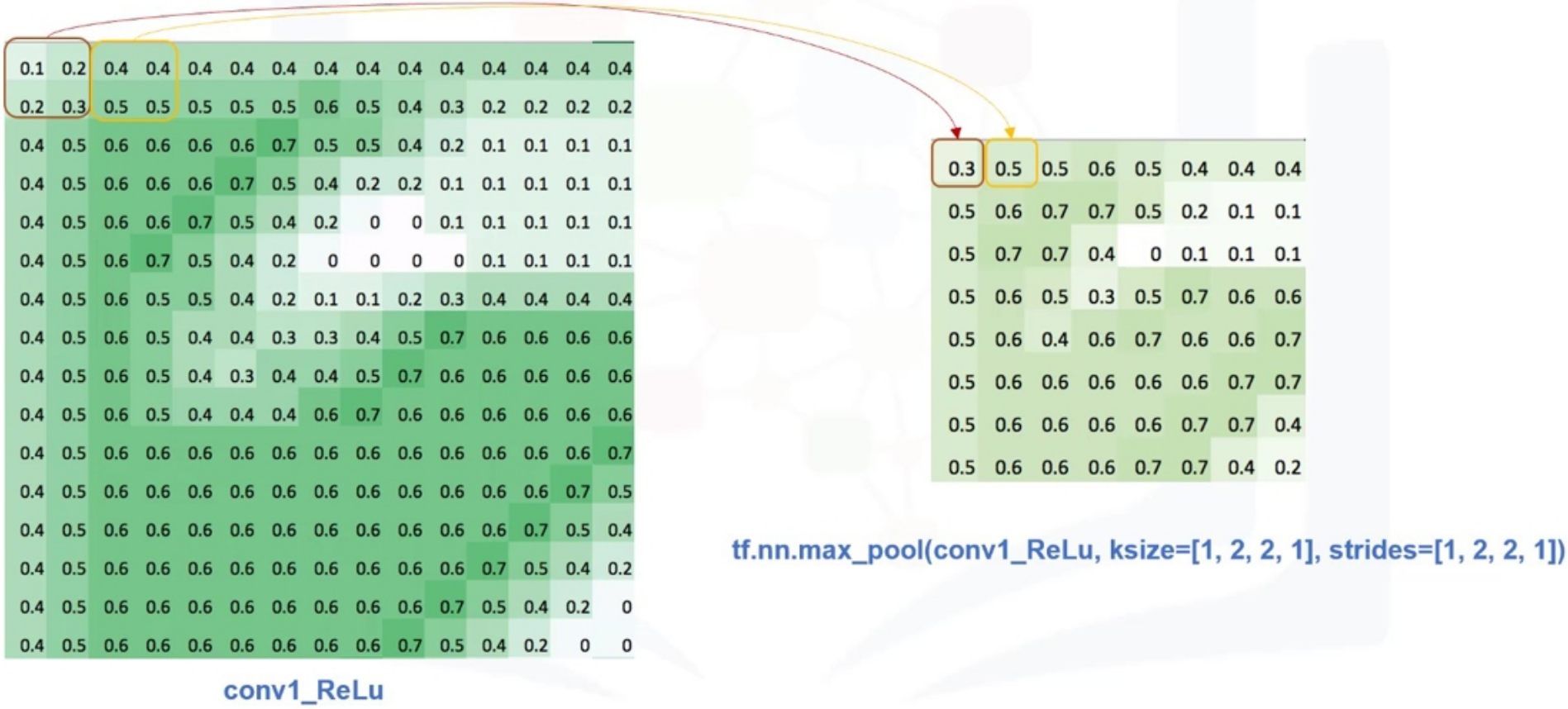
0.1	0.2	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4
0.2	0.3	0.5	0.5	0.5	0.5	0.5	0.6	0.5	0.4	0.3	0.2	0.2	0.2	0.2
0.4	0.5	0.6	0.6	0.6	0.6	0.7	0.5	0.5	0.4	0.2	0.1	0.1	0.1	0.1
0.4	0.5	0.6	0.6	0.6	0.7	0.5	0.4	0.2	0.2	0.1	0.1	0.1	0.1	0.1
0.4	0.5	0.6	0.6	0.7	0.5	0.4	0.2	0	0	0.1	0.1	0.1	0.1	0.1
0.4	0.5	0.6	0.7	0.5	0.4	0.2	0	0	0	0	0.1	0.1	0.1	0.1
0.4	0.5	0.6	0.5	0.5	0.4	0.2	0.1	0.1	0.2	0.3	0.4	0.4	0.4	0.4
0.4	0.5	0.6	0.5	0.4	0.4	0.3	0.3	0.4	0.5	0.7	0.6	0.6	0.6	0.6
0.4	0.5	0.6	0.5	0.4	0.3	0.4	0.4	0.5	0.7	0.6	0.6	0.6	0.6	0.6
0.4	0.5	0.6	0.5	0.4	0.4	0.4	0.6	0.7	0.6	0.6	0.6	0.6	0.6	0.6
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5	0.4
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5	0.4	0.2
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5	0.4	0.2	0
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5	0.4	0.2	0	0

conv1_ReLu

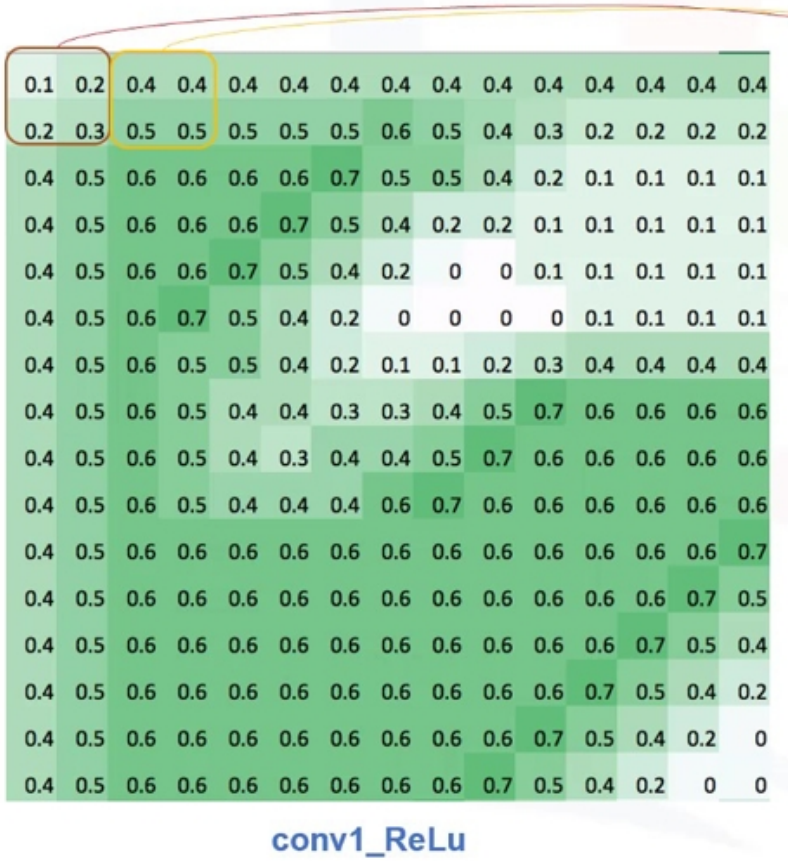
0.3	0.5	0.5	0.6	0.5	0.4	0.4	0.4
0.5	0.6	0.7	0.7	0.5	0.2	0.1	0.1
0.5	0.7	0.7	0.4	0	0.1	0.1	0.1
0.5	0.6	0.5	0.3	0.5	0.7	0.6	0.6
0.5	0.6	0.4	0.6	0.7	0.6	0.6	0.7
0.5	0.6	0.6	0.6	0.6	0.6	0.7	0.7
0.5	0.6	0.6	0.6	0.6	0.7	0.7	0.4
0.5	0.6	0.6	0.6	0.7	0.7	0.4	0.2

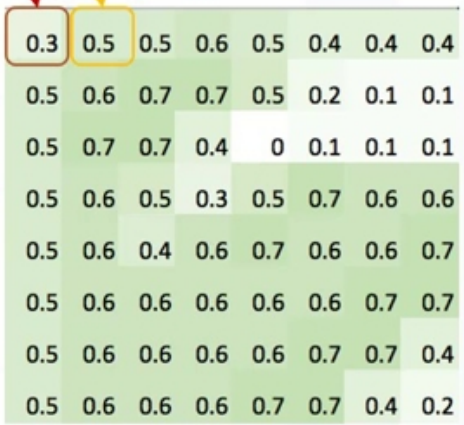
tf.nn.max_pool(conv1_ReLu, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1])

Max Pooling


0.1 0.2 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4
0.2 0.3 0.5 0.5 0.5 0.5 0.5 0.6 0.5 0.4 0.3 0.2 0.2 0.2 0.2
0.4 0.5 0.6 0.6 0.6 0.6 0.7 0.5 0.5 0.4 0.2 0.1 0.1 0.1 0.1
0.4 0.5 0.6 0.6 0.6 0.7 0.5 0.4 0.2 0.2 0.1 0.1 0.1 0.1 0.1
0.4 0.5 0.6 0.6 0.7 0.5 0.4 0.2 0 0 0.1 0.1 0.1 0.1 0.1
0.4 0.5 0.6 0.7 0.5 0.4 0.2 0 0 0 0 0.1 0.1 0.1 0.1
0.4 0.5 0.6 0.5 0.5 0.4 0.2 0.1 0.1 0.2 0.3 0.4 0.4 0.4 0.4
0.4 0.5 0.6 0.5 0.4 0.4 0.3 0.3 0.4 0.5 0.7 0.6 0.6 0.6 0.6
0.4 0.5 0.6 0.5 0.4 0.3 0.4 0.4 0.5 0.7 0.6 0.6 0.6 0.6 0.6
0.4 0.5 0.6 0.5 0.4 0.4 0.4 0.6 0.7 0.6 0.6 0.6 0.6 0.6 0.6
0.4 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7
0.4 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7 0.5
0.4 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7 0.5 0.4
0.4 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7 0.5 0.4 0.2
0.4 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7 0.5 0.4 0.2 0
0.4 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7 0.5 0.4 0.2 0 0
conv1_ReLu
0.3 0.5 0.5 0.6 0.5 0.4 0.4 0.4
0.5 0.6 0.7 0.7 0.5 0.2 0.1 0.1
0.5 0.7 0.7 0.4 0 0.1 0.1 0.1
0.5 0.6 0.5 0.3 0.5 0.7 0.6 0.6
0.5 0.6 0.4 0.6 0.7 0.6 0.6 0.7
0.5 0.6 0.6 0.6 0.6 0.6 0.7 0.7
0.5 0.6 0.6 0.6 0.6 0.7 0.7 0.4
0.5 0.6 0.6 0.6 0.7 0.7 0.4 0.2
tf.nn.max_pool(conv1_ReLu, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1])

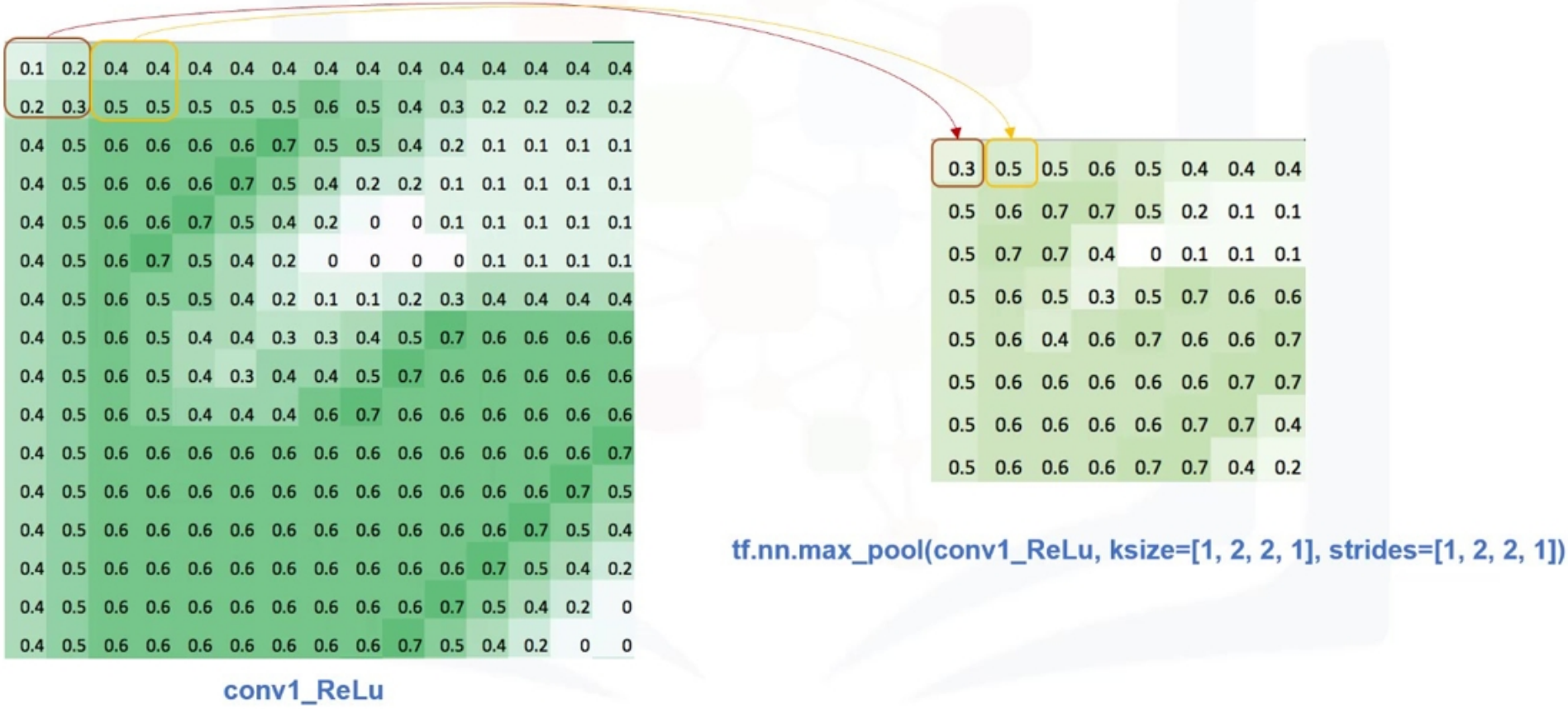
Max Pooling



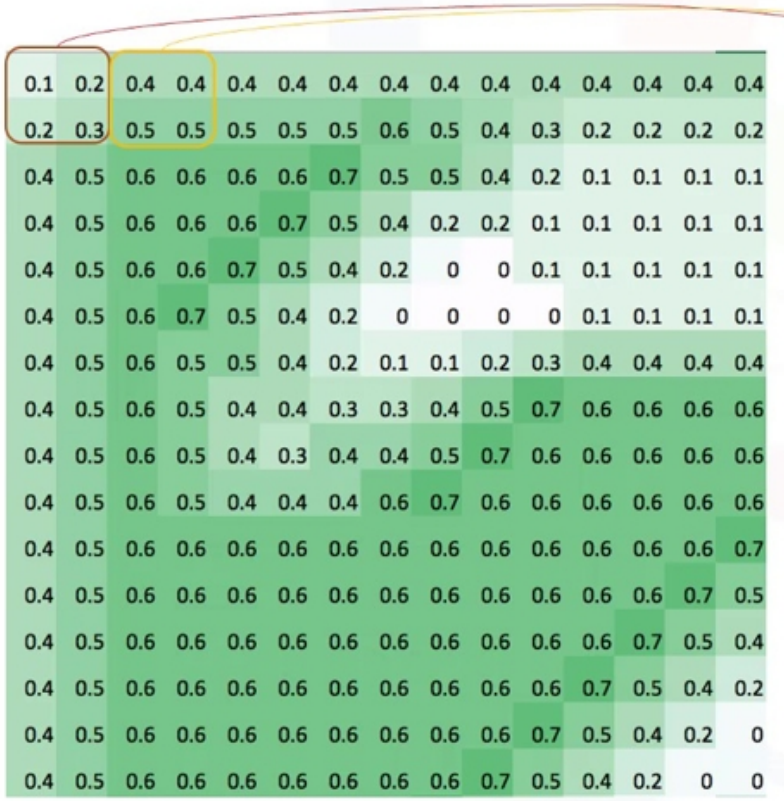



```
tf.nn.max_pool(conv1_ReLu, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1])
```

Max Pooling


0.1 0.2 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4
0.2 0.3 0.5 0.5 0.5 0.5 0.5 0.6 0.5 0.4 0.3 0.2 0.2 0.2 0.2
0.4 0.5 0.6 0.6 0.6 0.6 0.7 0.5 0.5 0.4 0.2 0.1 0.1 0.1 0.1
0.4 0.5 0.6 0.6 0.6 0.7 0.5 0.4 0.2 0.2 0.1 0.1 0.1 0.1 0.1
0.4 0.5 0.6 0.6 0.7 0.5 0.4 0.2 0 0 0.1 0.1 0.1 0.1 0.1
0.4 0.5 0.6 0.7 0.5 0.4 0.2 0 0 0 0 0.1 0.1 0.1 0.1
0.4 0.5 0.6 0.5 0.5 0.4 0.2 0.1 0.1 0.2 0.3 0.4 0.4 0.4 0.4
0.4 0.5 0.6 0.5 0.4 0.4 0.3 0.3 0.4 0.5 0.7 0.6 0.6 0.6 0.6
0.4 0.5 0.6 0.5 0.4 0.3 0.4 0.4 0.5 0.7 0.6 0.6 0.6 0.6 0.6
0.4 0.5 0.6 0.5 0.4 0.4 0.4 0.6 0.7 0.6 0.6 0.6 0.6 0.6 0.6
0.4 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7
0.4 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7 0.5
0.4 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7 0.5 0.4
0.4 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7 0.5 0.4 0.2
0.4 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7 0.5 0.4 0.2 0
0.4 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7 0.5 0.4 0.2 0 0
conv1_ReLu
0.3 0.5 0.5 0.6 0.5 0.4 0.4 0.4
0.5 0.6 0.7 0.7 0.5 0.2 0.1 0.1
0.5 0.7 0.7 0.4 0 0.1 0.1 0.1
0.5 0.6 0.5 0.3 0.5 0.7 0.6 0.6
0.5 0.6 0.4 0.6 0.7 0.6 0.6 0.7
0.5 0.6 0.6 0.6 0.6 0.6 0.7 0.7
0.5 0.6 0.6 0.6 0.6 0.7 0.7 0.4
0.5 0.6 0.6 0.6 0.7 0.7 0.4 0.2
tf.nn.max_pool(conv1_ReLu, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1])

Max Pooling



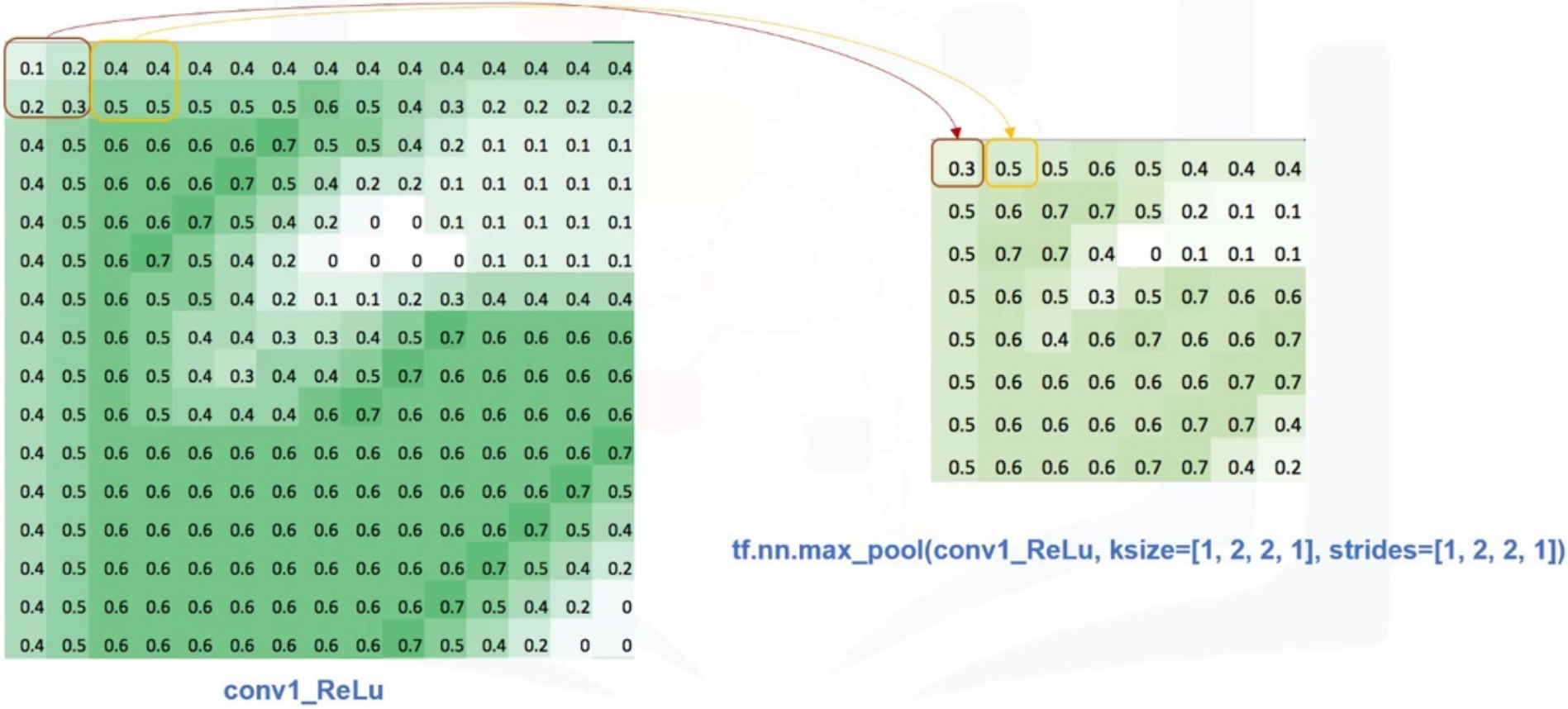
0.1	0.2	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4
0.2	0.3	0.5	0.5	0.5	0.5	0.5	0.6	0.5	0.4	0.3	0.2	0.2	0.2	0.2
0.4	0.5	0.6	0.6	0.6	0.6	0.7	0.5	0.5	0.4	0.2	0.1	0.1	0.1	0.1
0.4	0.5	0.6	0.6	0.6	0.7	0.5	0.4	0.2	0.2	0.1	0.1	0.1	0.1	0.1
0.4	0.5	0.6	0.6	0.7	0.5	0.4	0.2	0	0	0.1	0.1	0.1	0.1	0.1
0.4	0.5	0.6	0.7	0.5	0.4	0.2	0	0	0	0	0.1	0.1	0.1	0.1
0.4	0.5	0.6	0.5	0.5	0.4	0.2	0.1	0.1	0.2	0.3	0.4	0.4	0.4	0.4
0.4	0.5	0.6	0.5	0.4	0.4	0.3	0.3	0.4	0.5	0.7	0.6	0.6	0.6	0.6
0.4	0.5	0.6	0.5	0.4	0.3	0.4	0.4	0.5	0.7	0.6	0.6	0.6	0.6	0.6
0.4	0.5	0.6	0.5	0.4	0.4	0.4	0.6	0.7	0.6	0.6	0.6	0.6	0.6	0.6
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5	0.4
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5	0.4	0.2
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5	0.4	0.2	0
0.4	0.5	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.5	0.4	0.2	0	0

conv1_ReLu

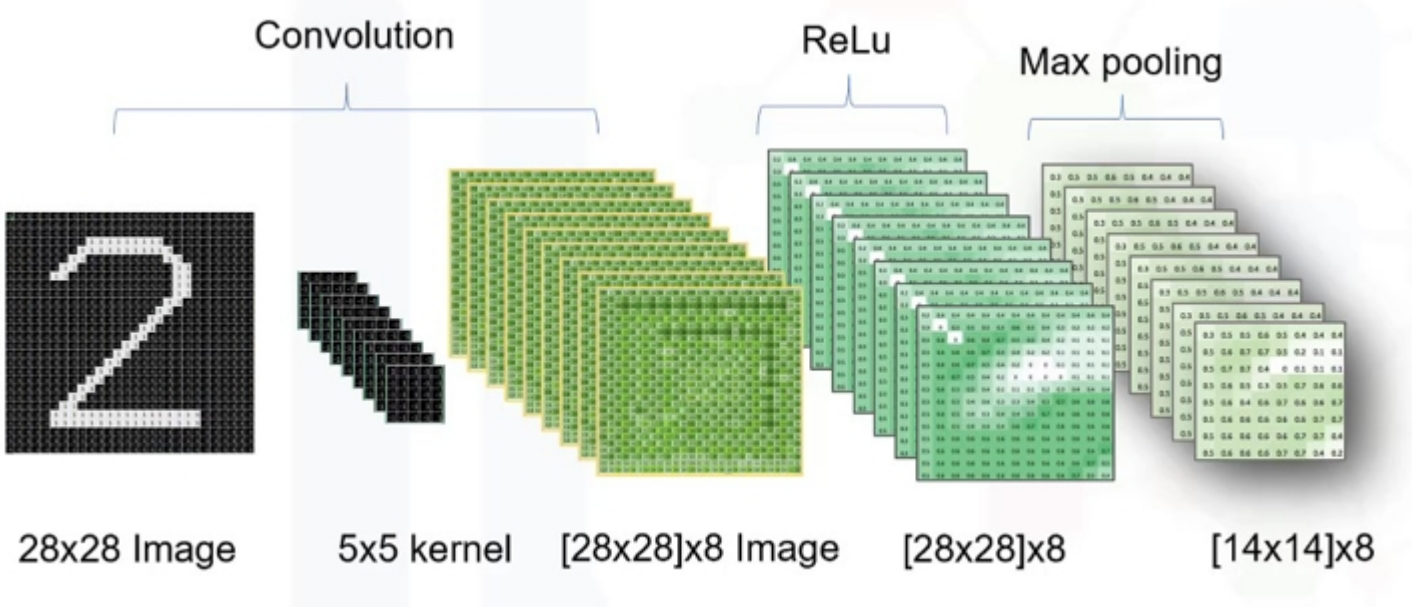
0.3	0.5	0.5	0.6	0.5	0.4	0.4	0.4
0.5	0.6	0.7	0.7	0.5	0.2	0.1	0.1
0.5	0.7	0.7	0.4	0	0.1	0.1	0.1
0.5	0.6	0.5	0.3	0.5	0.7	0.6	0.6
0.5	0.6	0.4	0.6	0.7	0.6	0.6	0.7
0.5	0.6	0.6	0.6	0.6	0.6	0.7	0.7
0.5	0.6	0.6	0.6	0.6	0.7	0.7	0.4
0.5	0.6	0.6	0.6	0.7	0.7	0.4	0.2

```
tf.nn.max_pool(conv1_ReLu, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1])
```

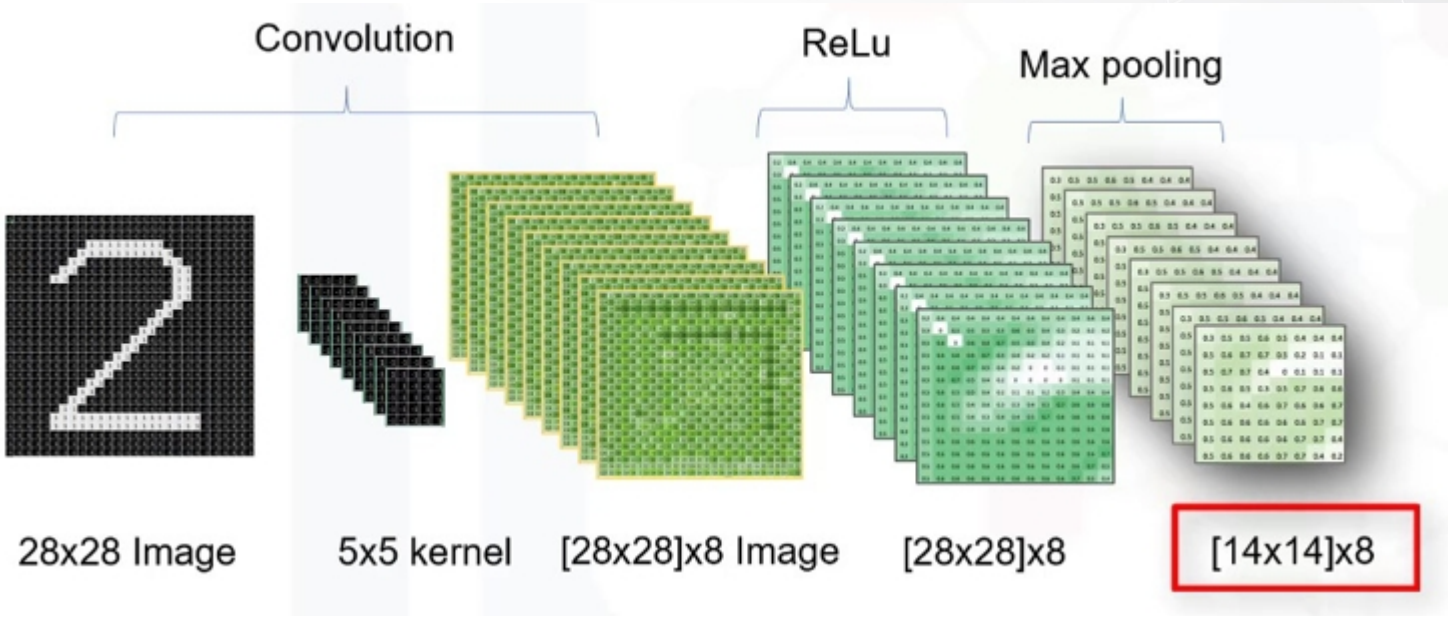
Max Pooling


0.1 0.2 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4
0.2 0.3 0.5 0.5 0.5 0.5 0.5 0.6 0.5 0.4 0.3 0.2 0.2 0.2 0.2
0.4 0.5 0.6 0.6 0.6 0.6 0.7 0.5 0.5 0.4 0.2 0.1 0.1 0.1 0.1
0.4 0.5 0.6 0.6 0.6 0.7 0.5 0.4 0.2 0.2 0.1 0.1 0.1 0.1 0.1
0.4 0.5 0.6 0.6 0.7 0.5 0.4 0.2 0 0 0.1 0.1 0.1 0.1 0.1
0.4 0.5 0.6 0.7 0.5 0.4 0.2 0 0 0 0 0.1 0.1 0.1 0.1
0.4 0.5 0.6 0.5 0.5 0.4 0.2 0.1 0.1 0.2 0.3 0.4 0.4 0.4 0.4
0.4 0.5 0.6 0.5 0.4 0.4 0.3 0.3 0.4 0.5 0.7 0.6 0.6 0.6 0.6
0.4 0.5 0.6 0.5 0.4 0.3 0.4 0.4 0.5 0.7 0.6 0.6 0.6 0.6 0.6
0.4 0.5 0.6 0.5 0.4 0.4 0.4 0.6 0.7 0.6 0.6 0.6 0.6 0.6 0.6
0.4 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7
0.4 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7 0.5
0.4 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7 0.5 0.4
0.4 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7 0.5 0.4 0.2
0.4 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7 0.5 0.4 0.2 0
0.4 0.5 0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.7 0.5 0.4 0.2 0 0
conv1_ReLu
0.3 0.5 0.5 0.6 0.5 0.4 0.4 0.4
0.5 0.6 0.7 0.7 0.5 0.2 0.1 0.1
0.5 0.7 0.7 0.4 0 0.1 0.1 0.1
0.5 0.6 0.5 0.3 0.5 0.7 0.6 0.6
0.5 0.6 0.4 0.6 0.7 0.6 0.6 0.7
0.5 0.6 0.6 0.6 0.6 0.6 0.7 0.7
0.5 0.6 0.6 0.6 0.6 0.7 0.7 0.4
0.5 0.6 0.6 0.6 0.7 0.7 0.4 0.2
tf.nn.max_pool(conv1_ReLu, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1])

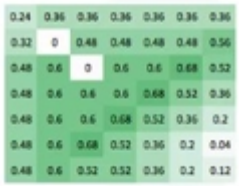
CNN Architecture ReLu

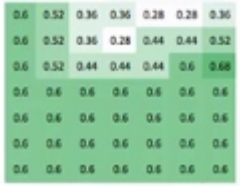


CNN Architecture ReLu

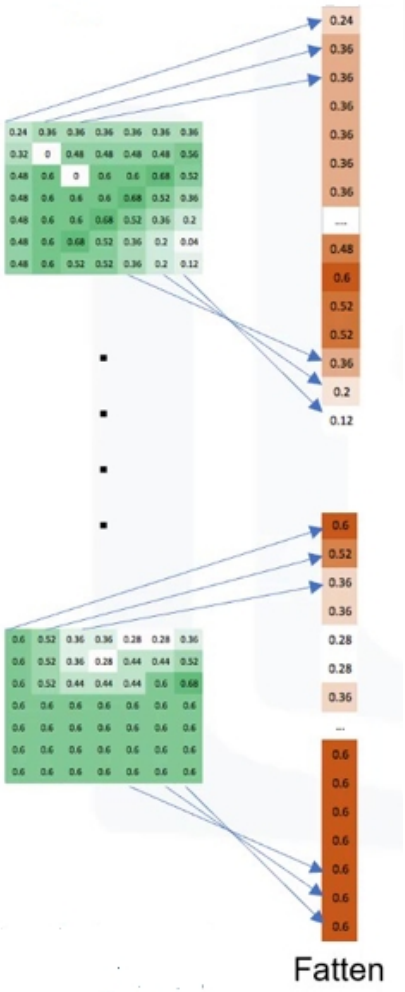


Fully Connected Layer

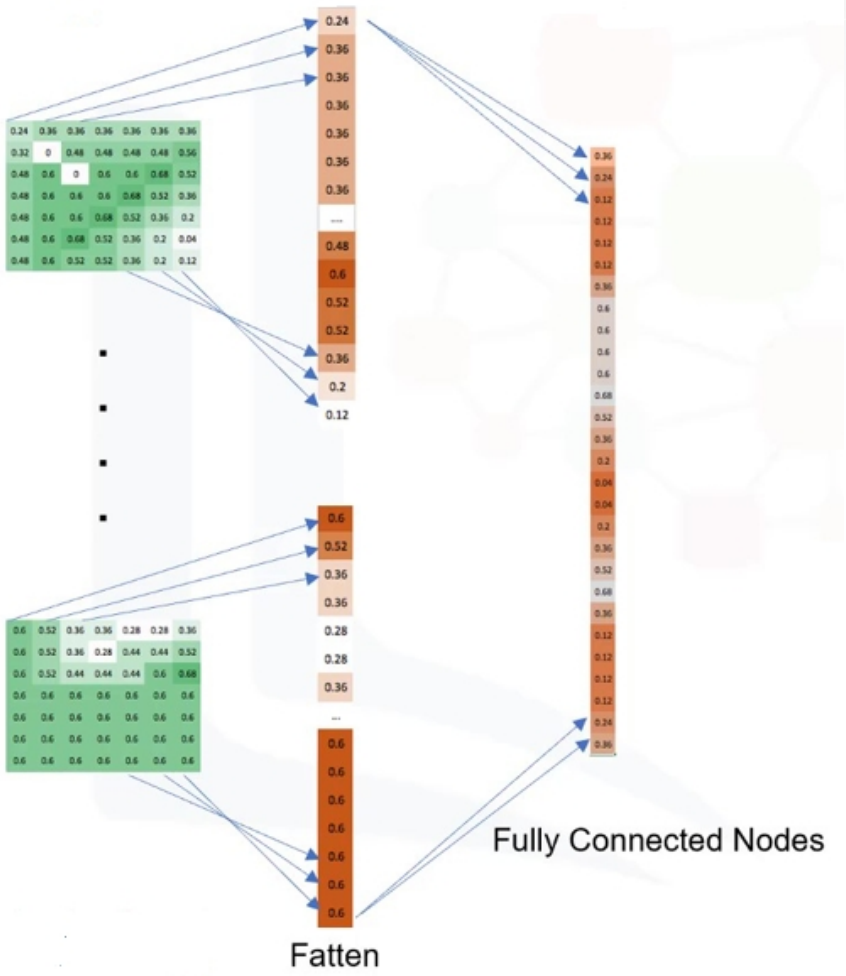




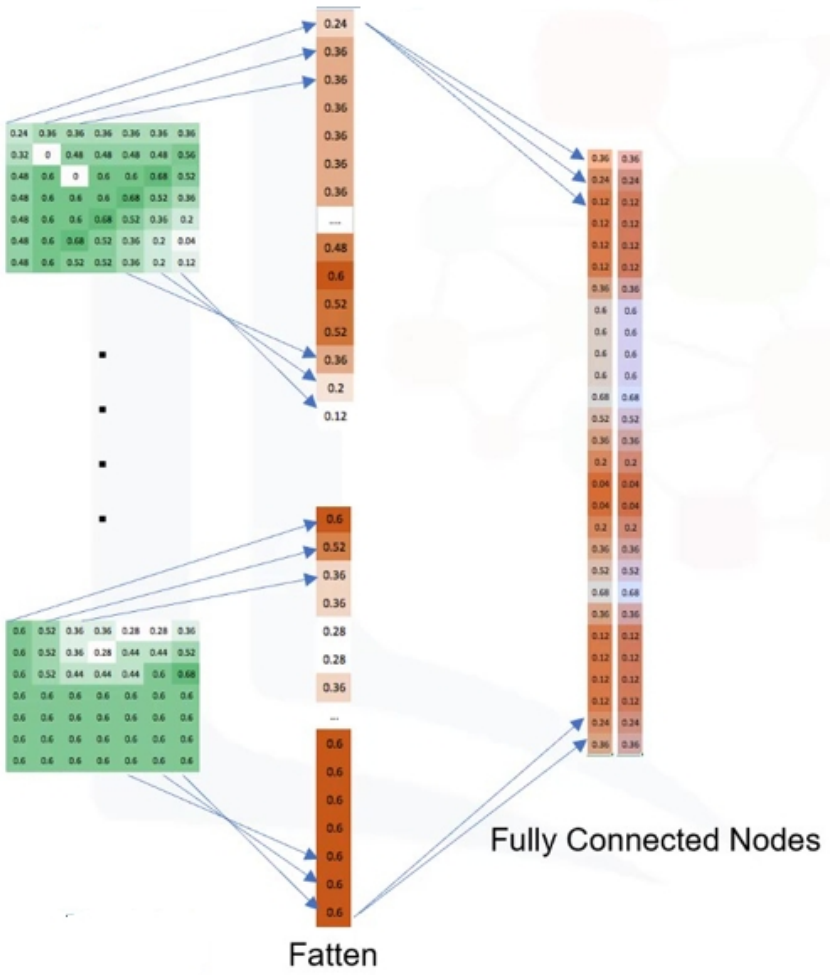
Fully Connected Layer



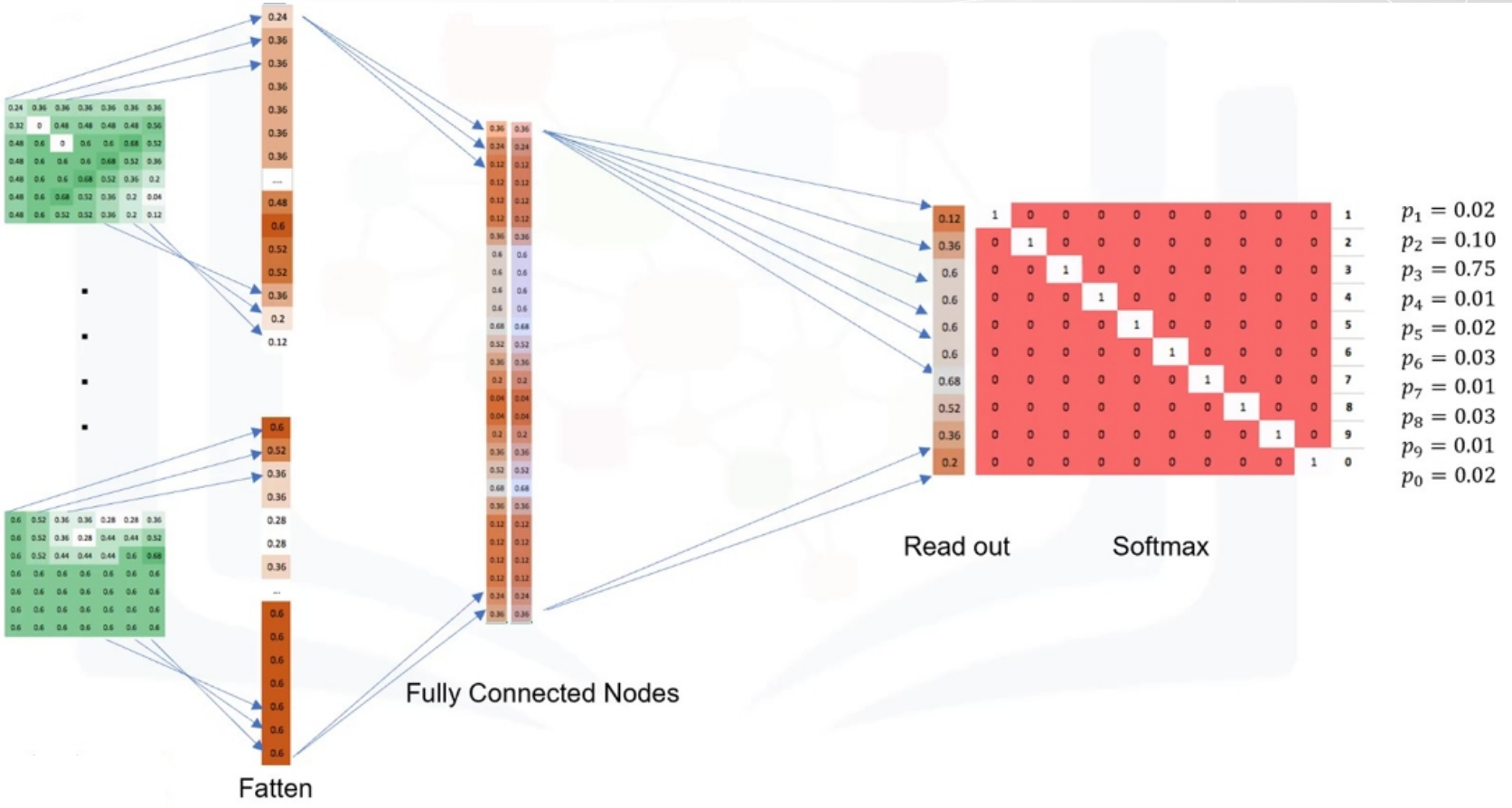
Fully Connected Layer



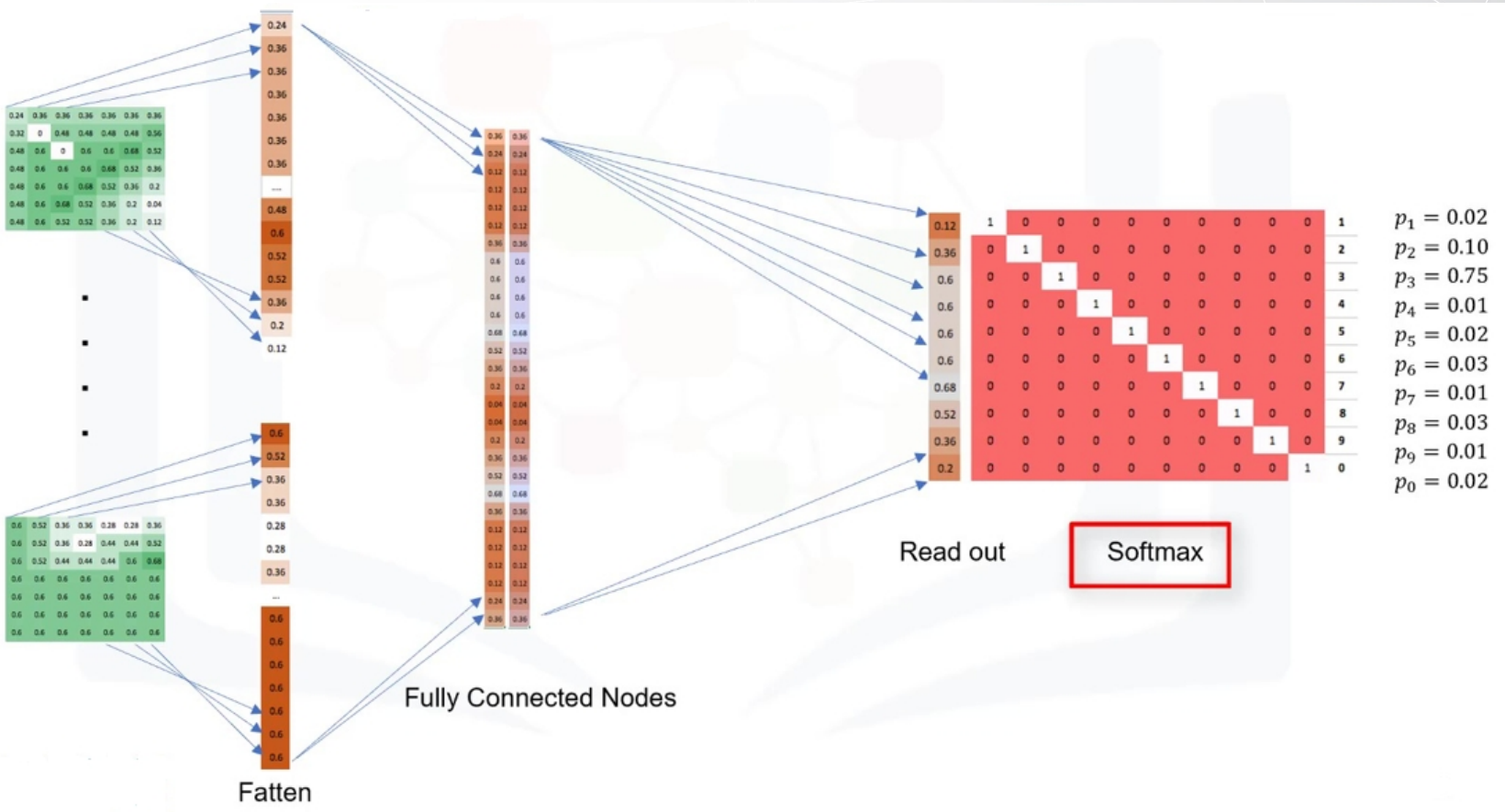
Fully Connected Layer



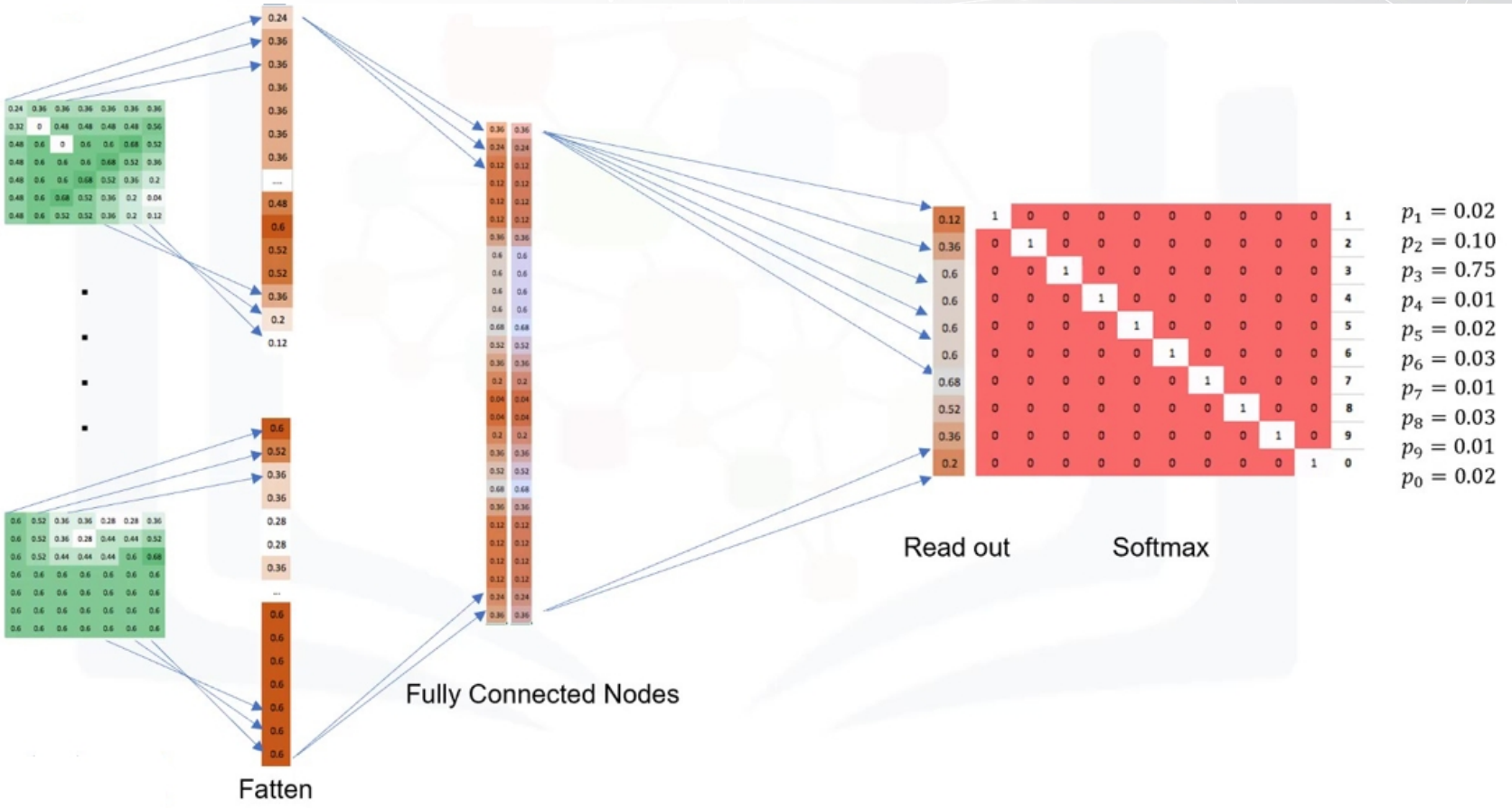
Fully Connected Layer



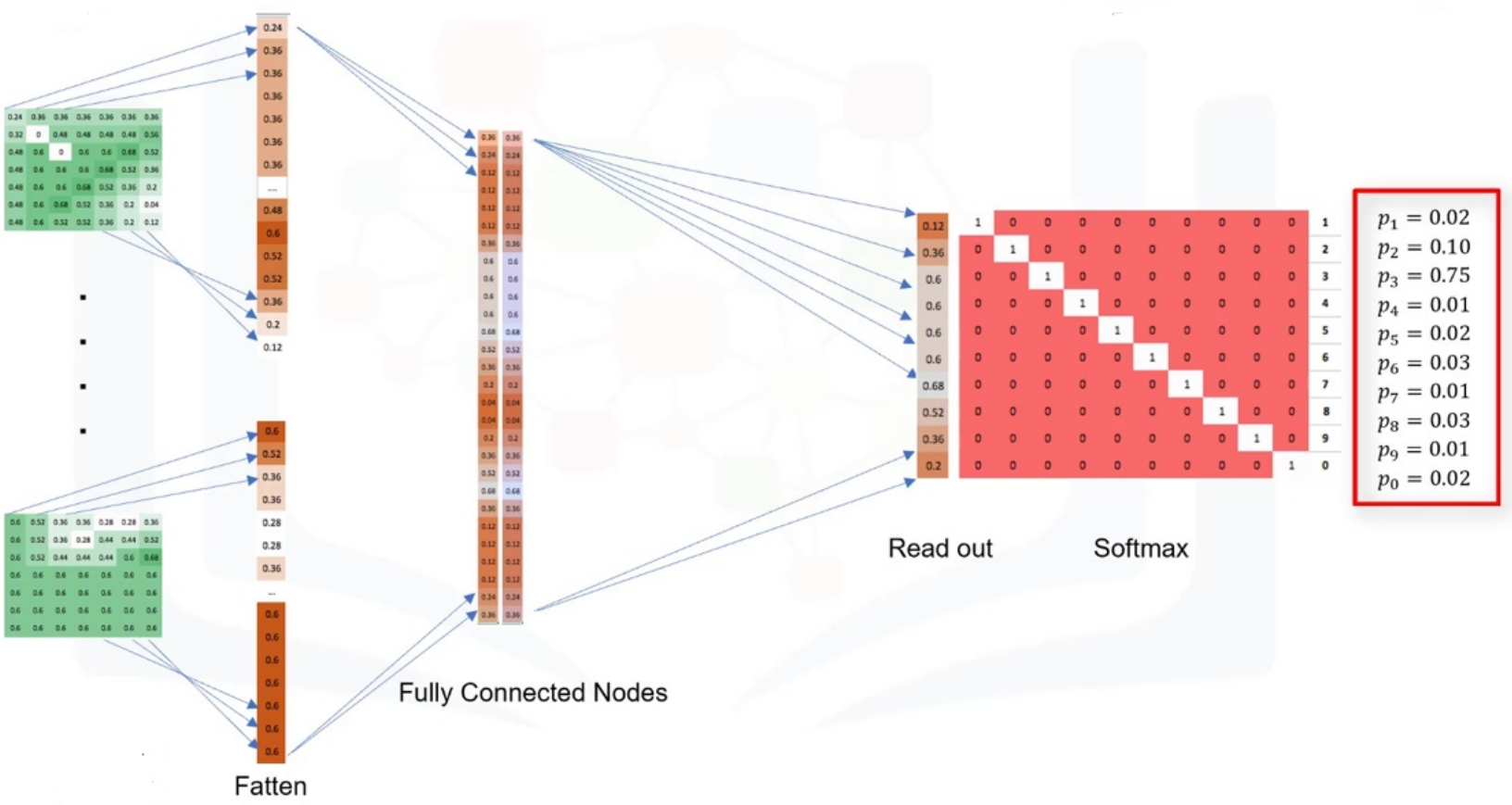
Fully Connected Layer



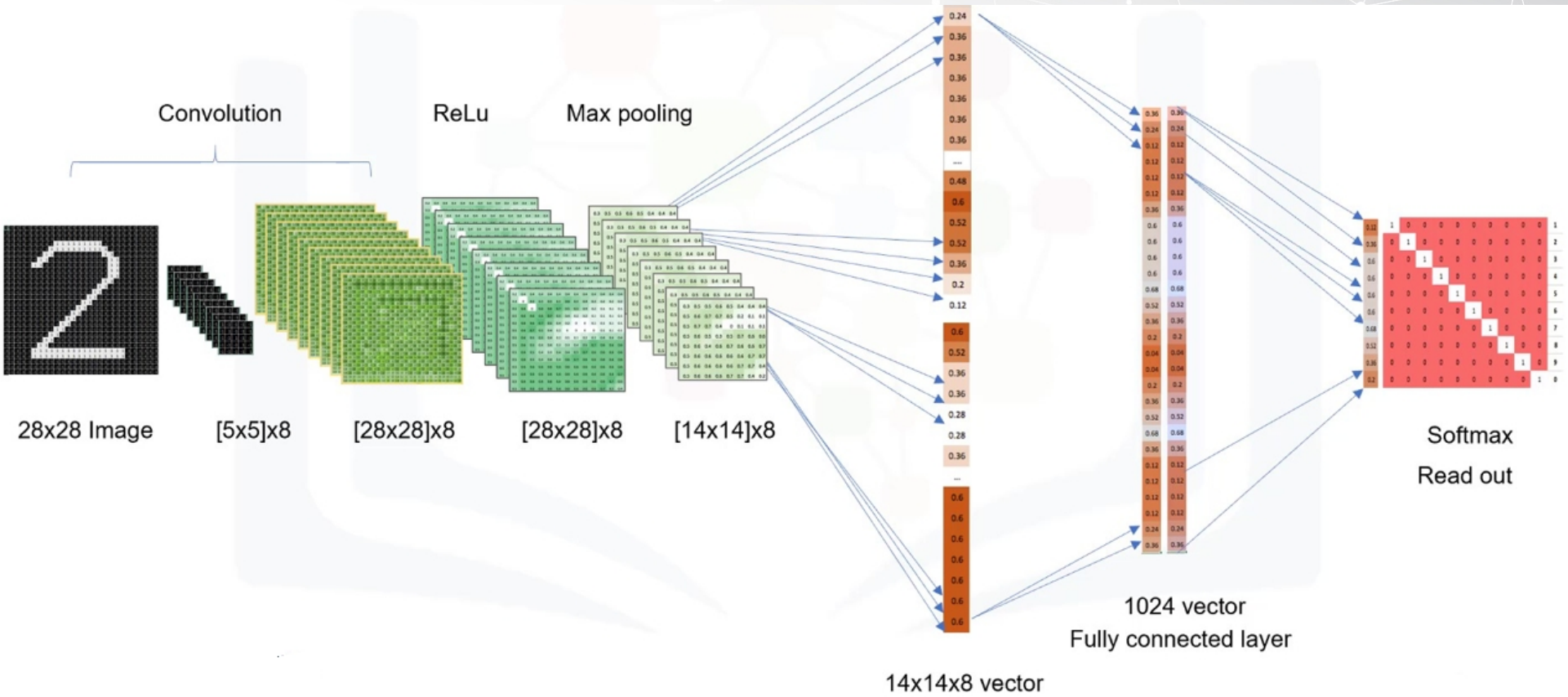
Fully Connected Layer



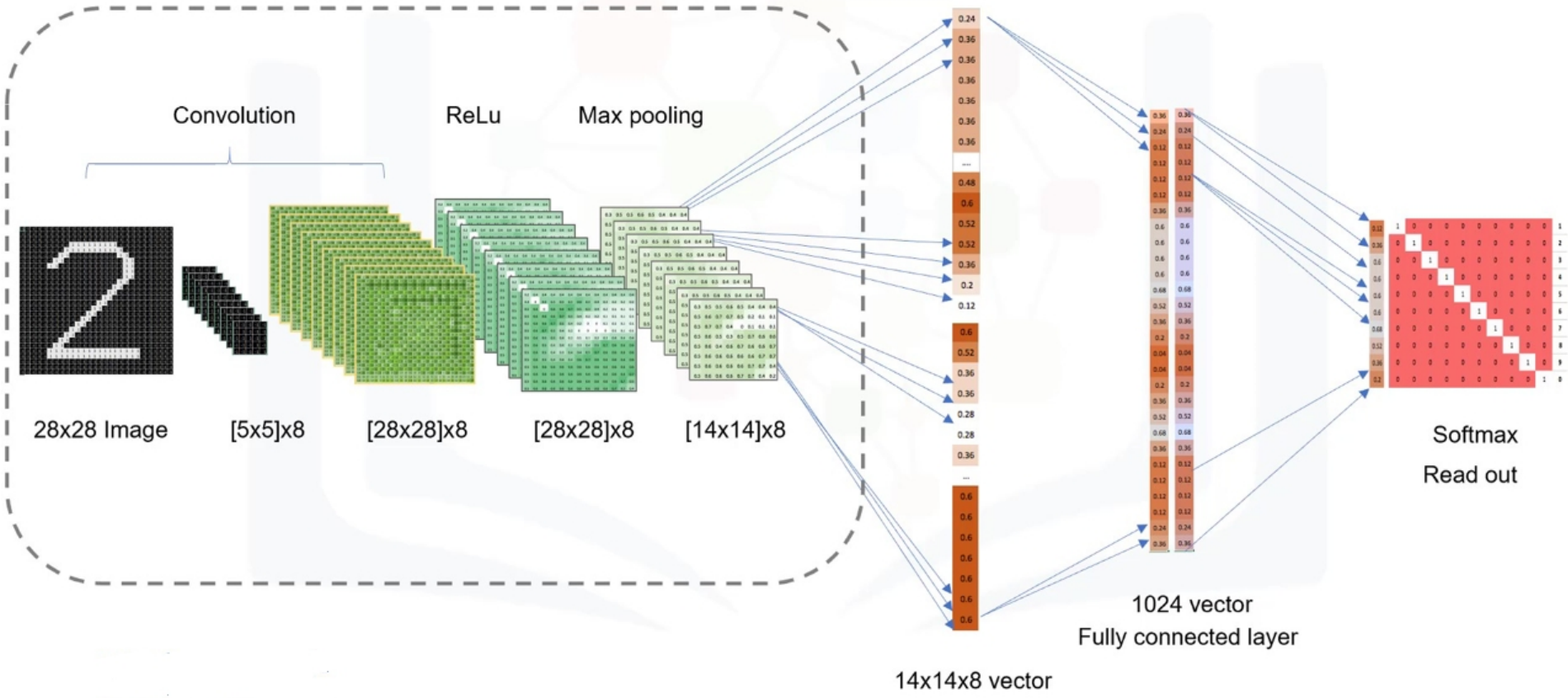
Fully Connected Layer



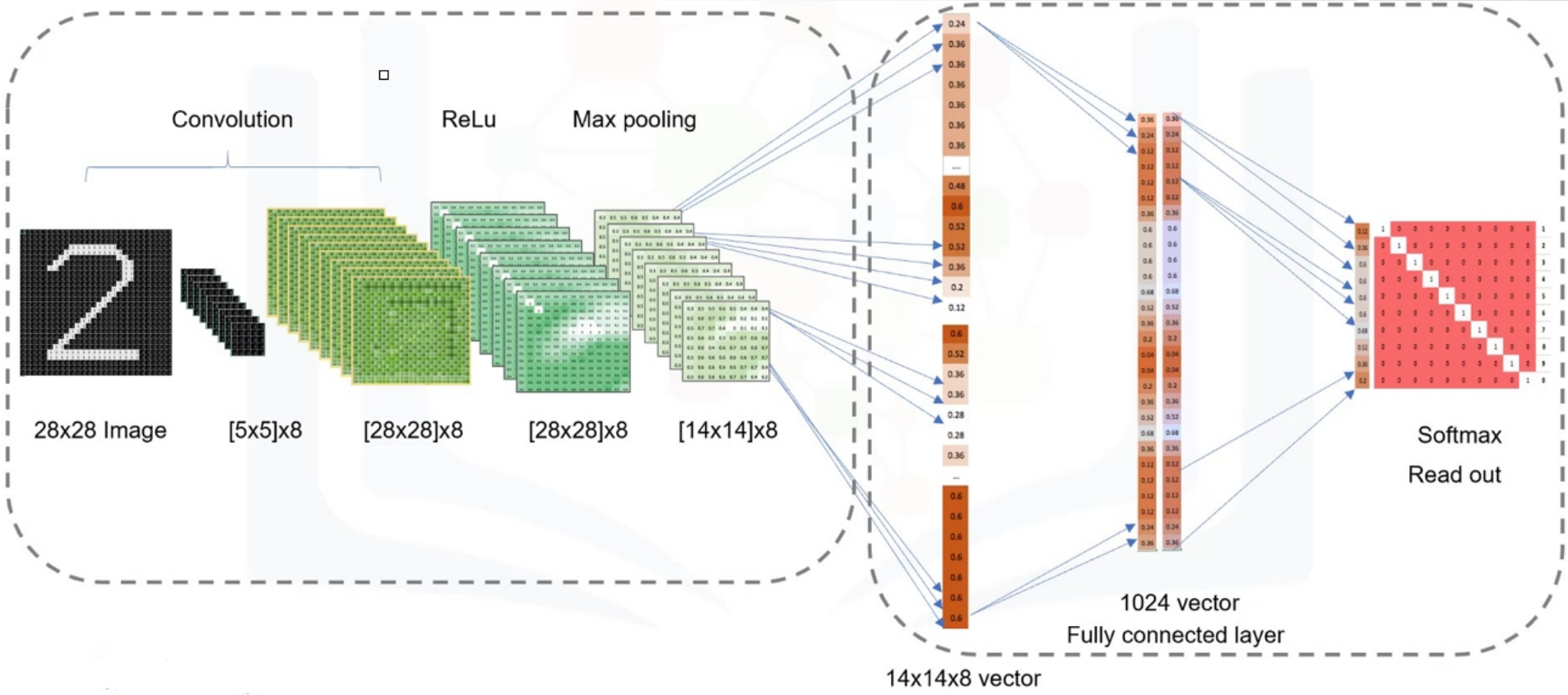
CNN Architecture



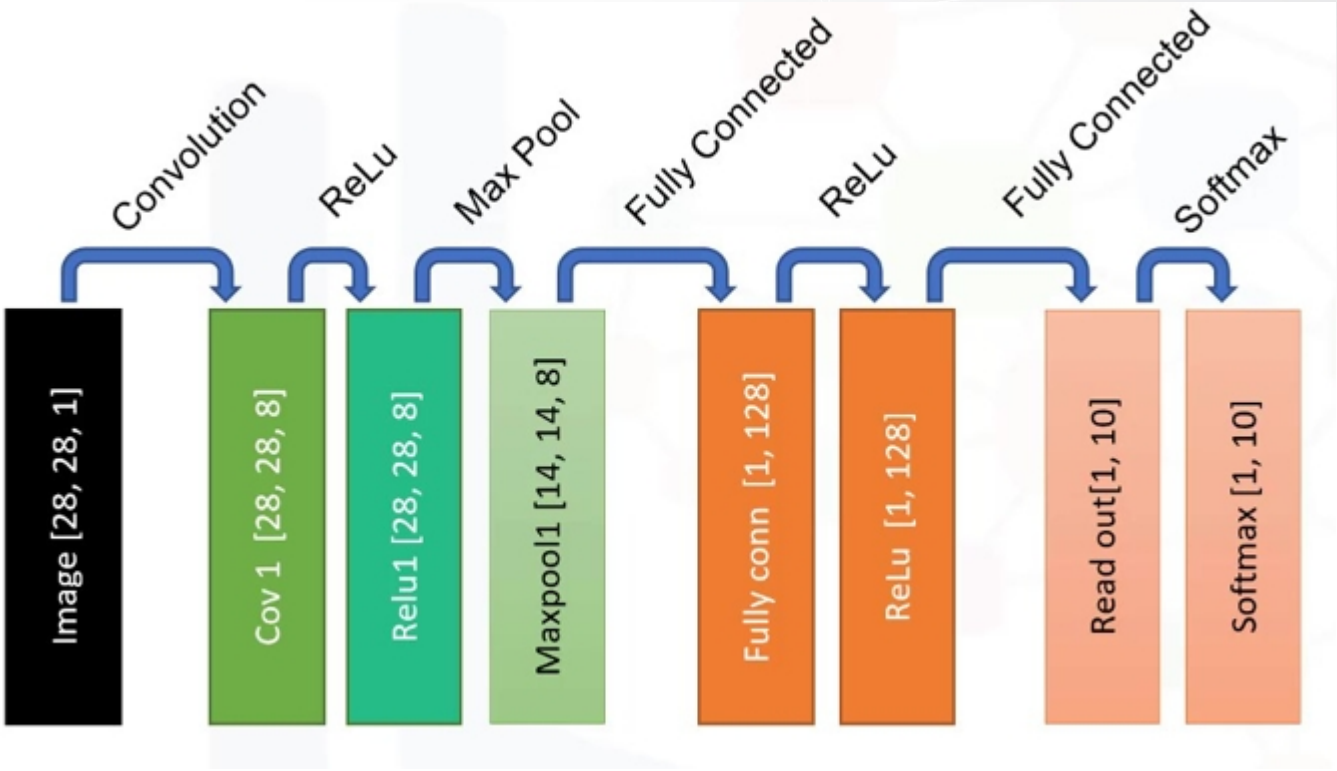
CNN Architecture



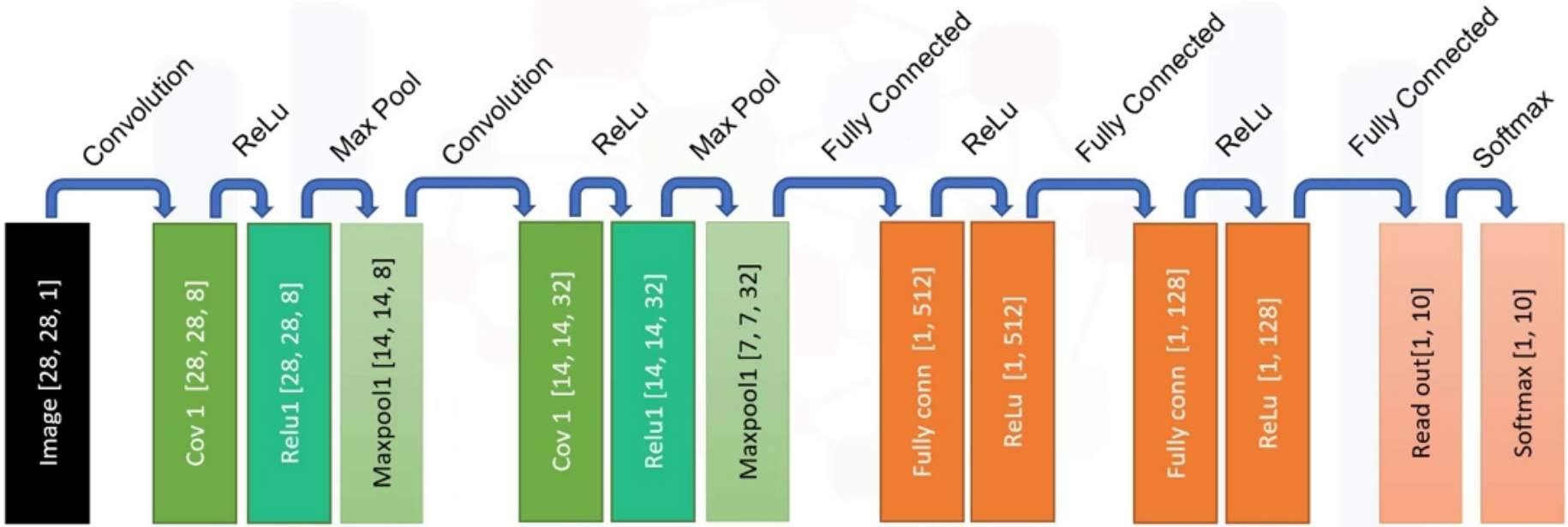
CNN Architecture



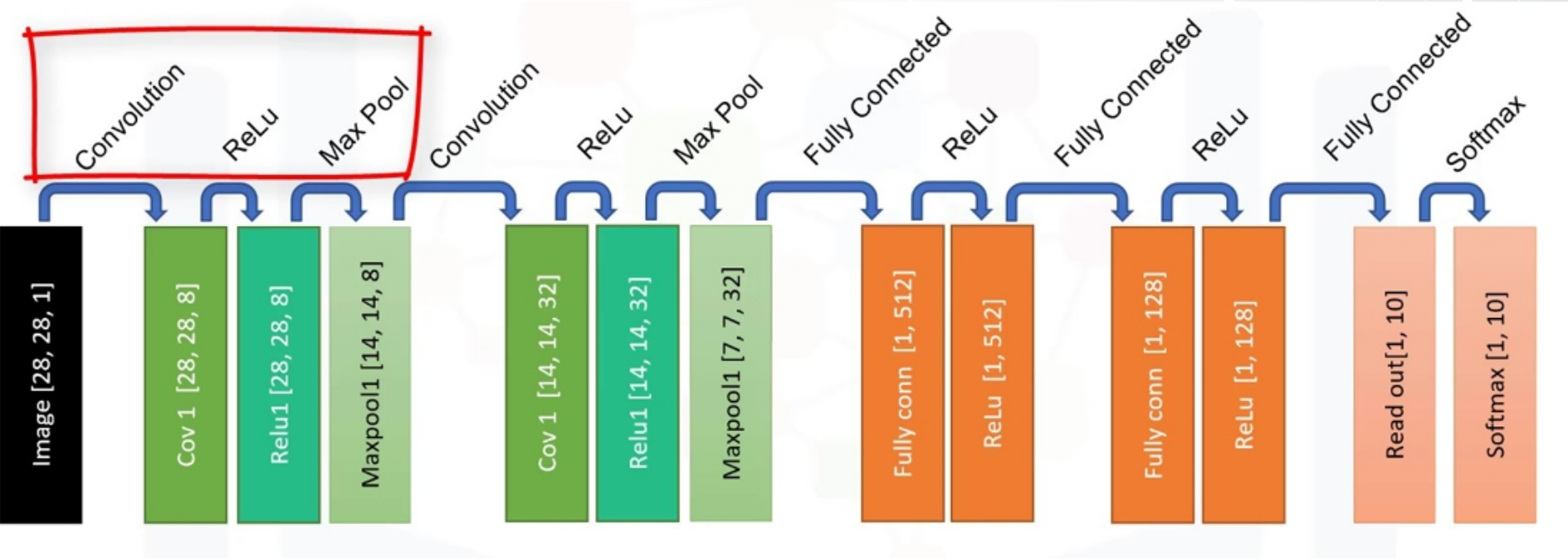
CNN Architecture



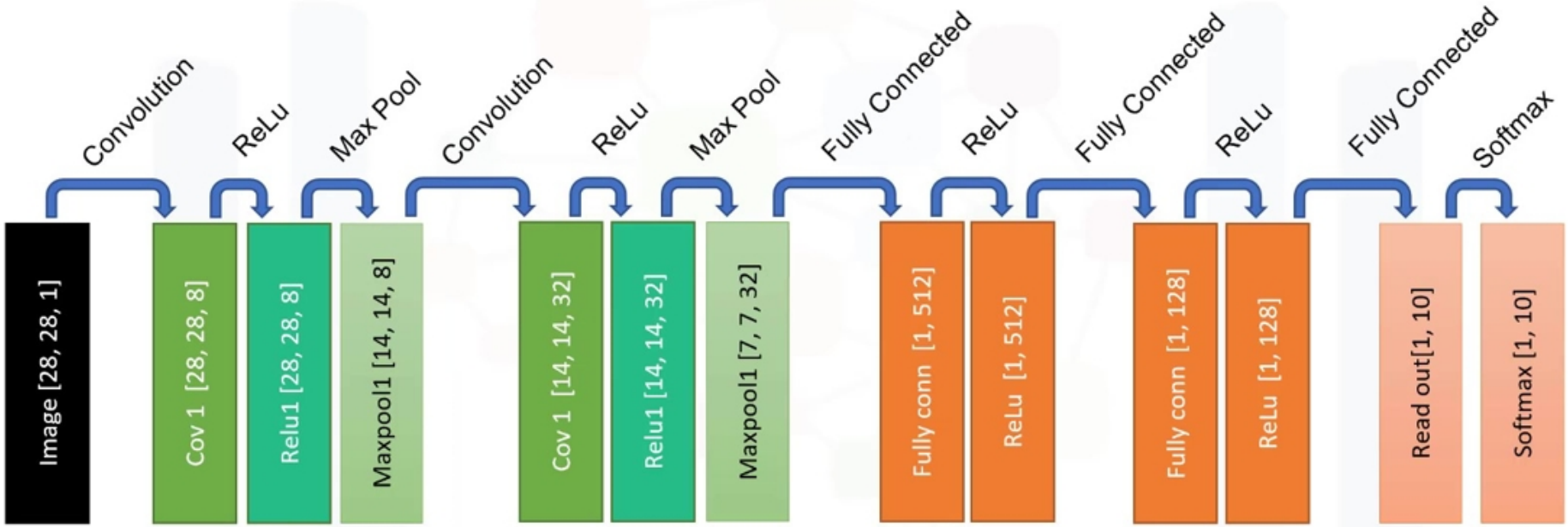
CNN Architecture



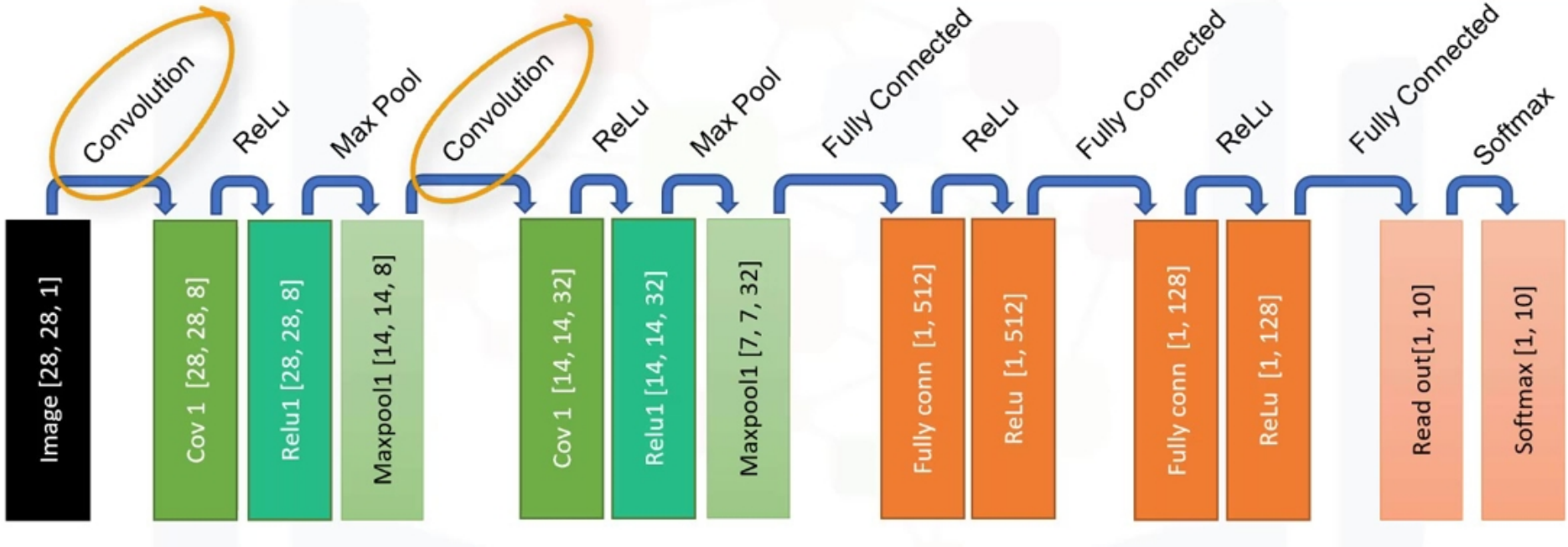
CNN Architecture



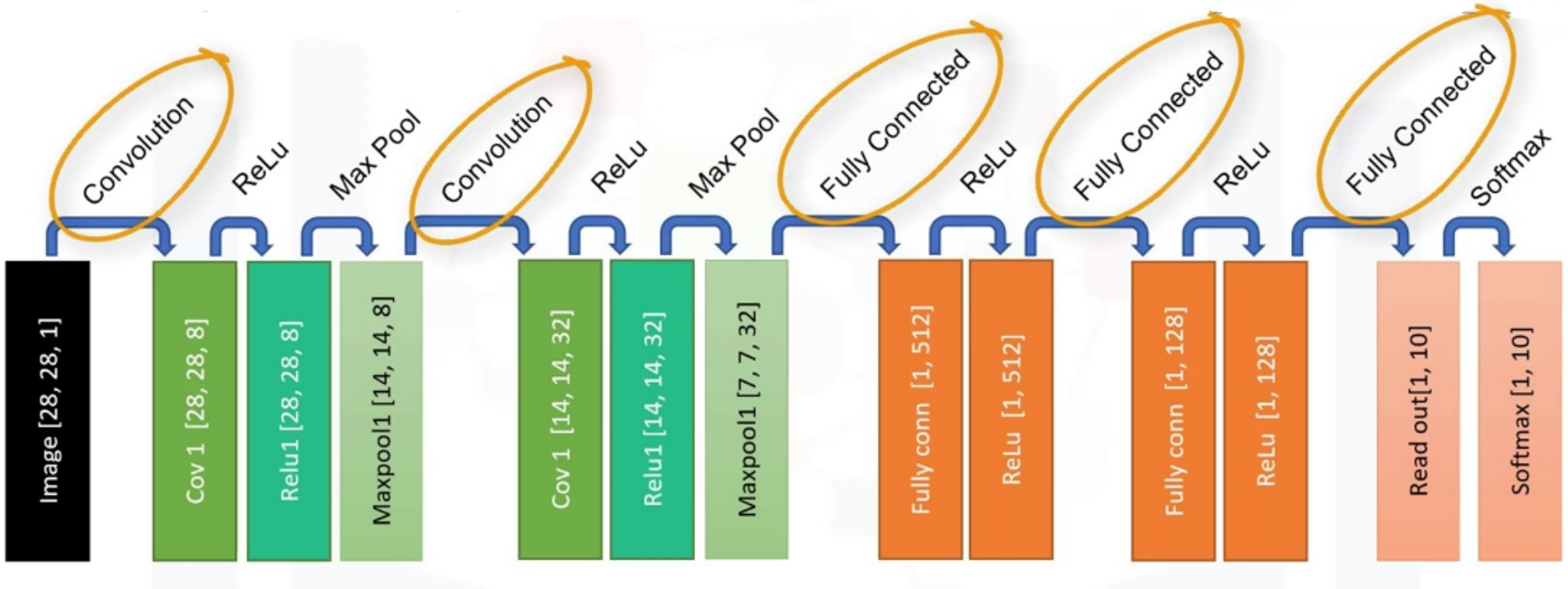
CNN Architecture



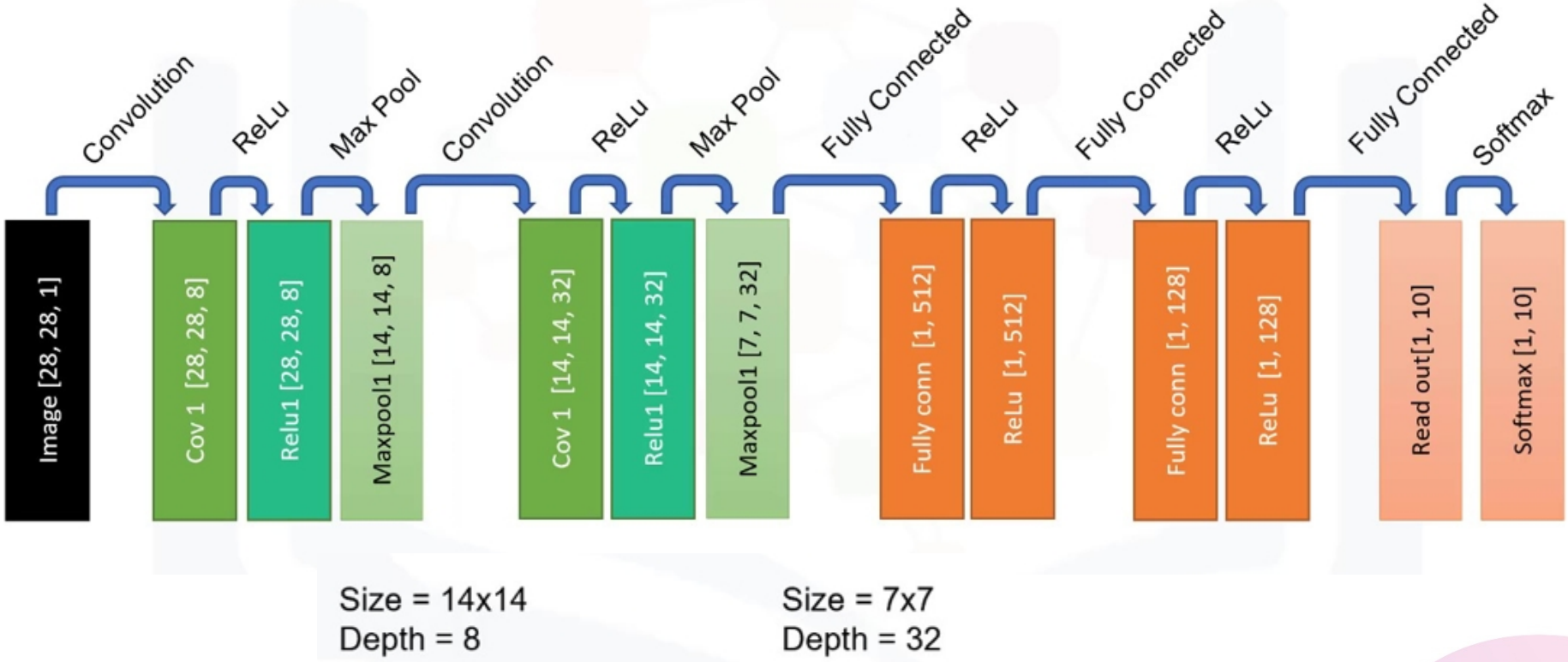
CNN Architecture



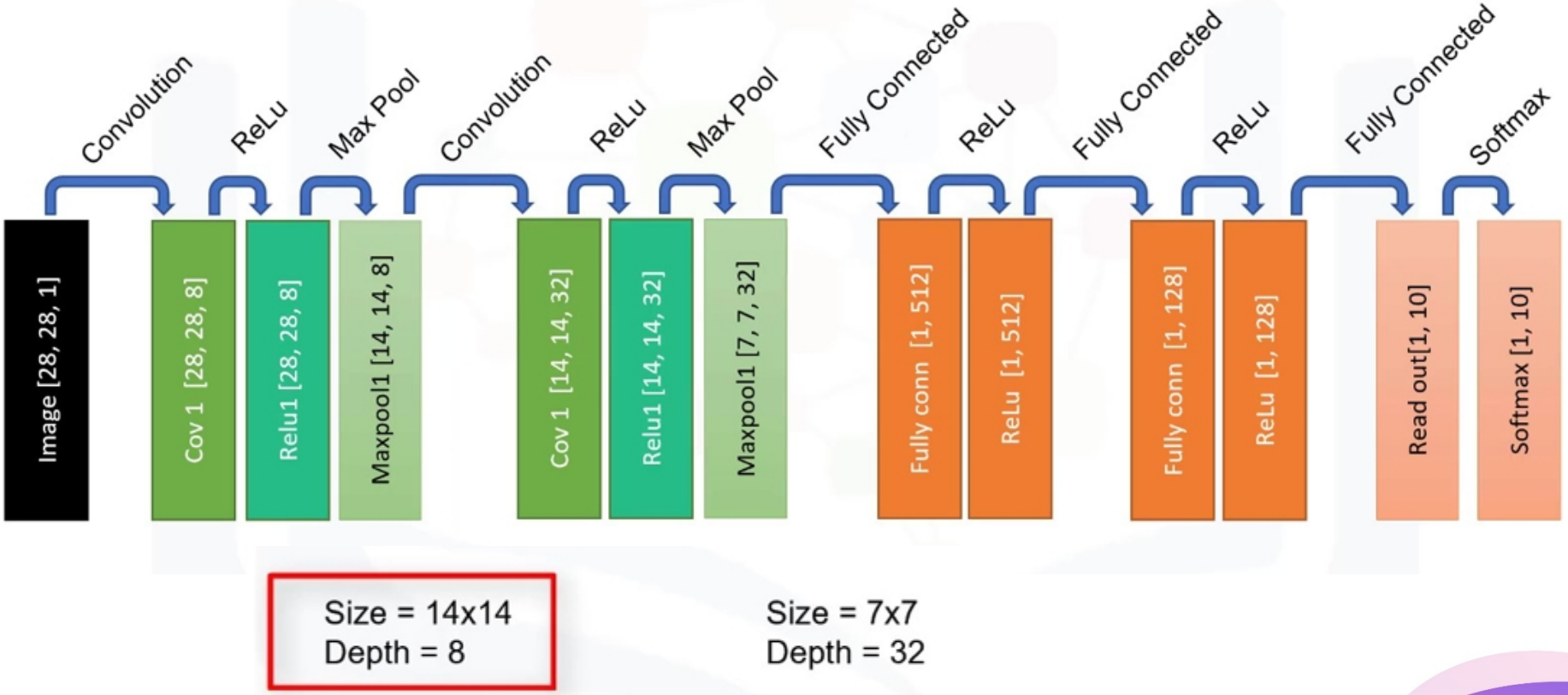
CNN Architecture



CNN Architecture

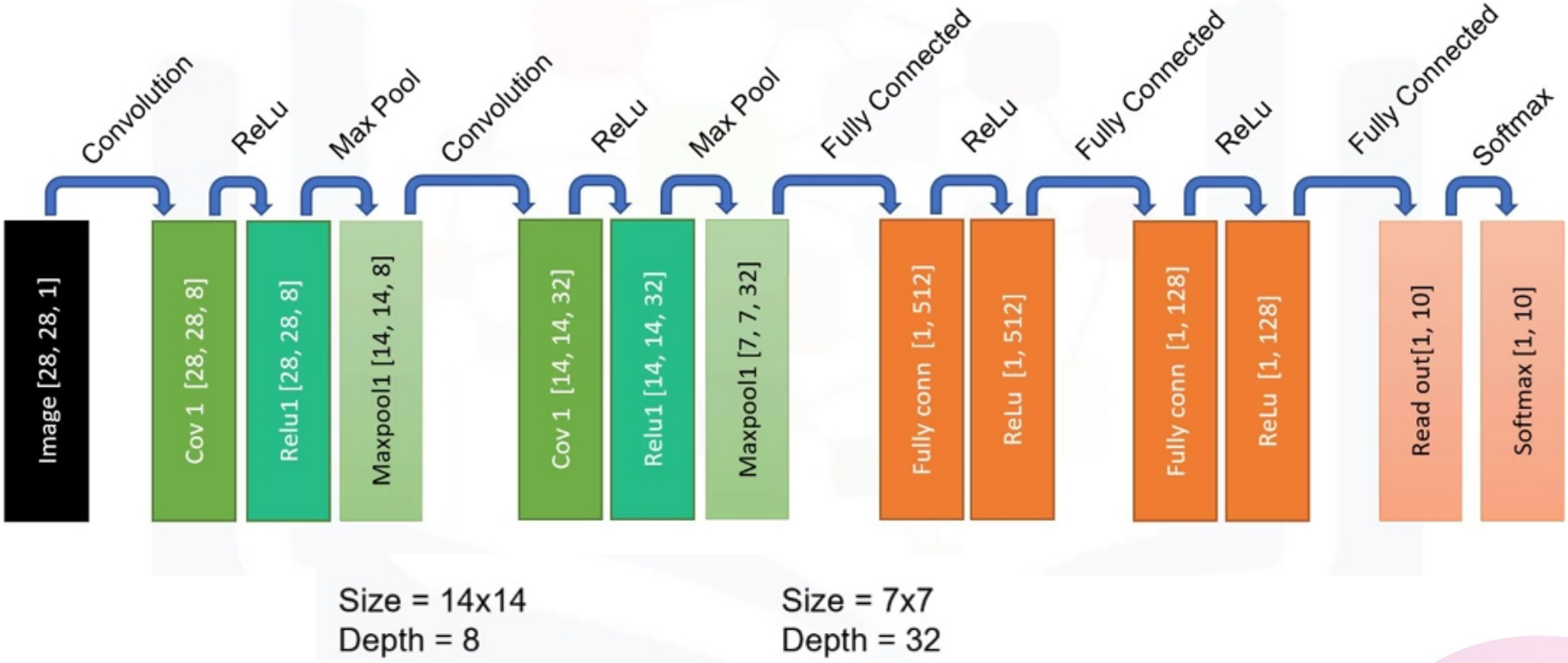


CNN Architecture

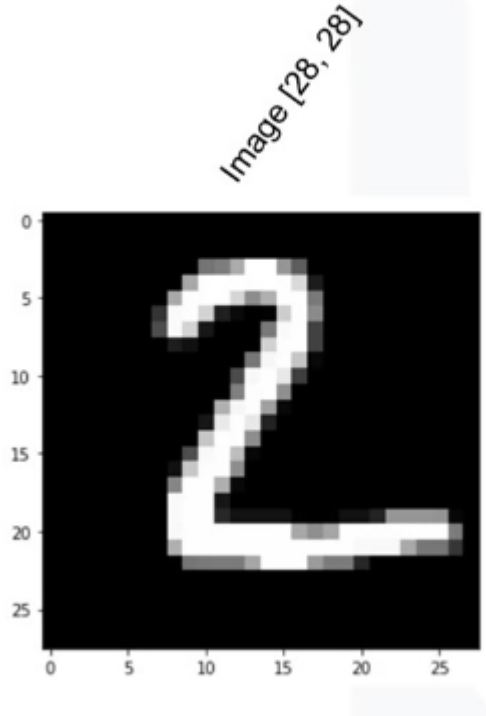


CNN Architecture

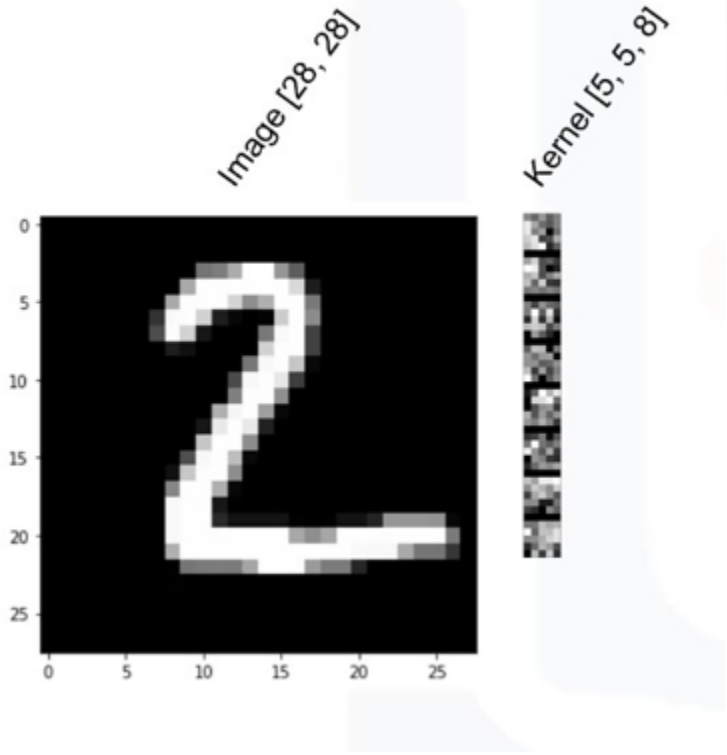
CNN Architecture



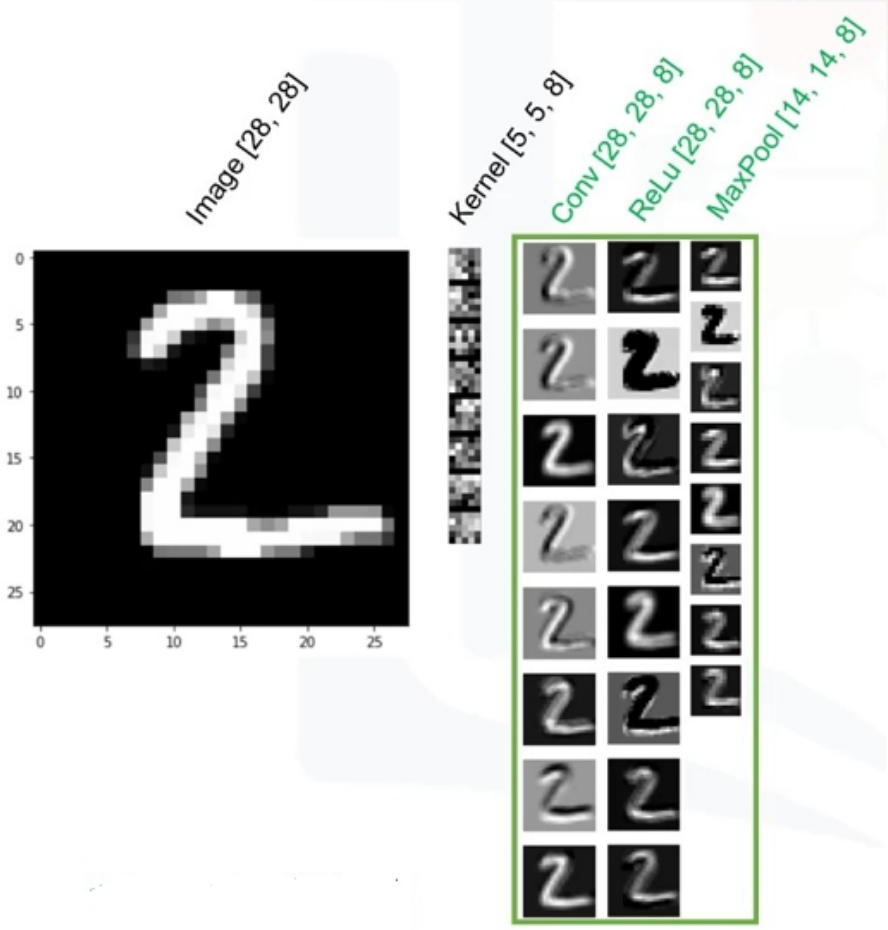
CNN Architecture



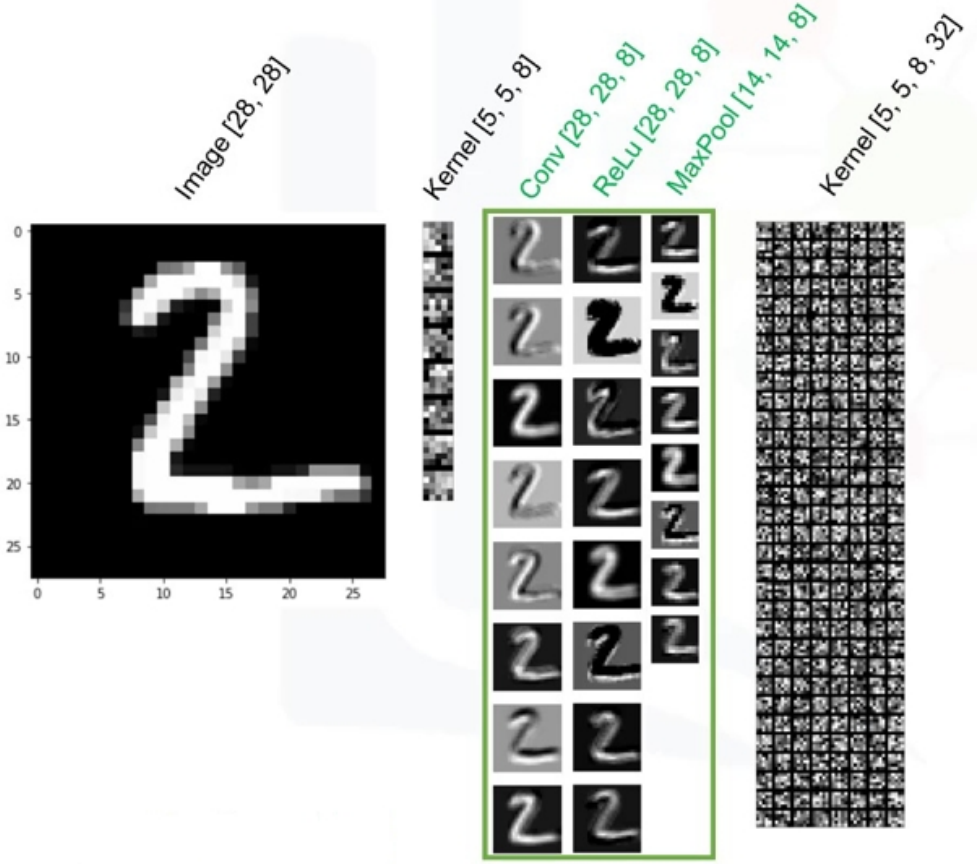
CNN Architecture



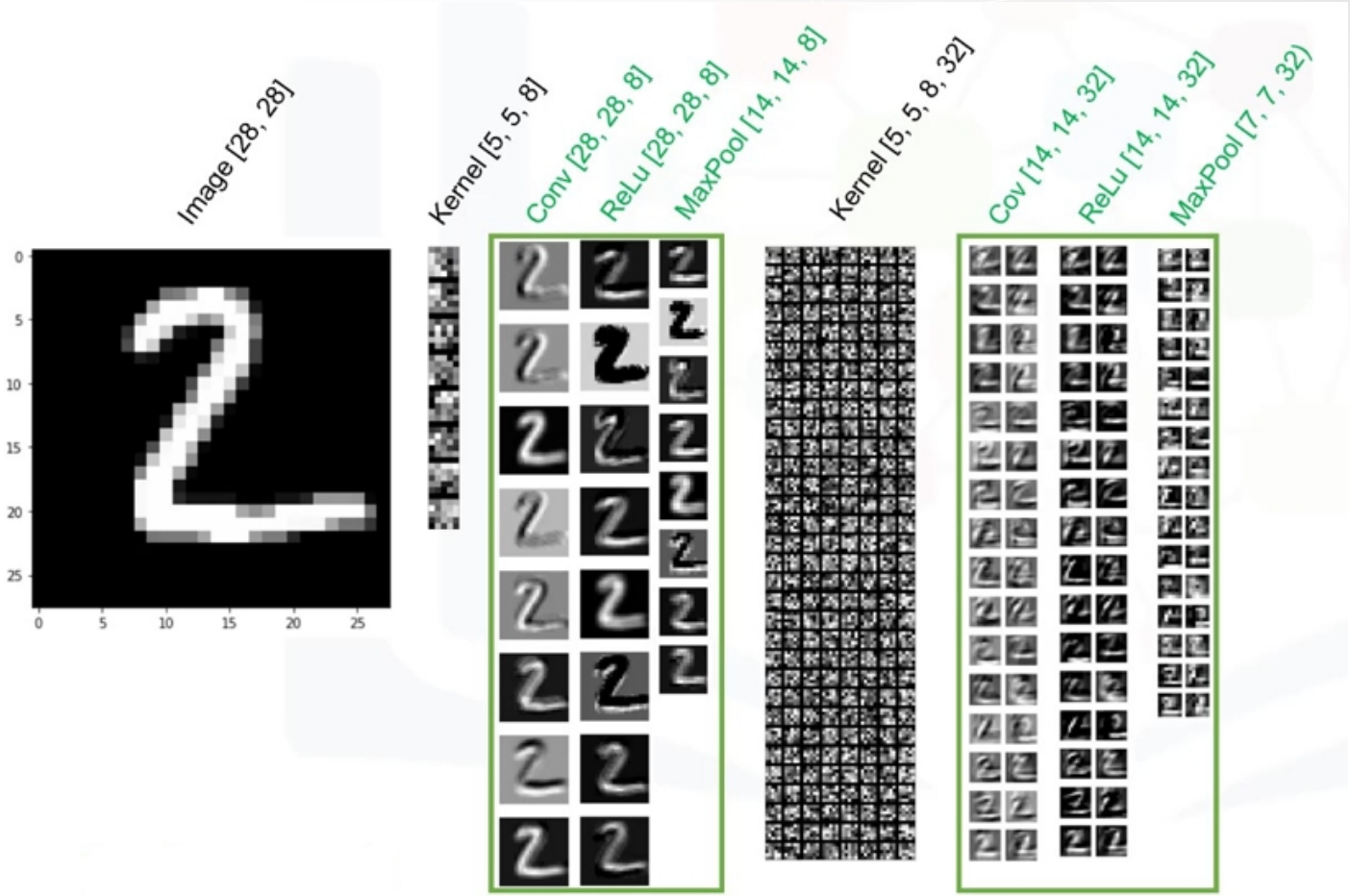
CNN Architecture



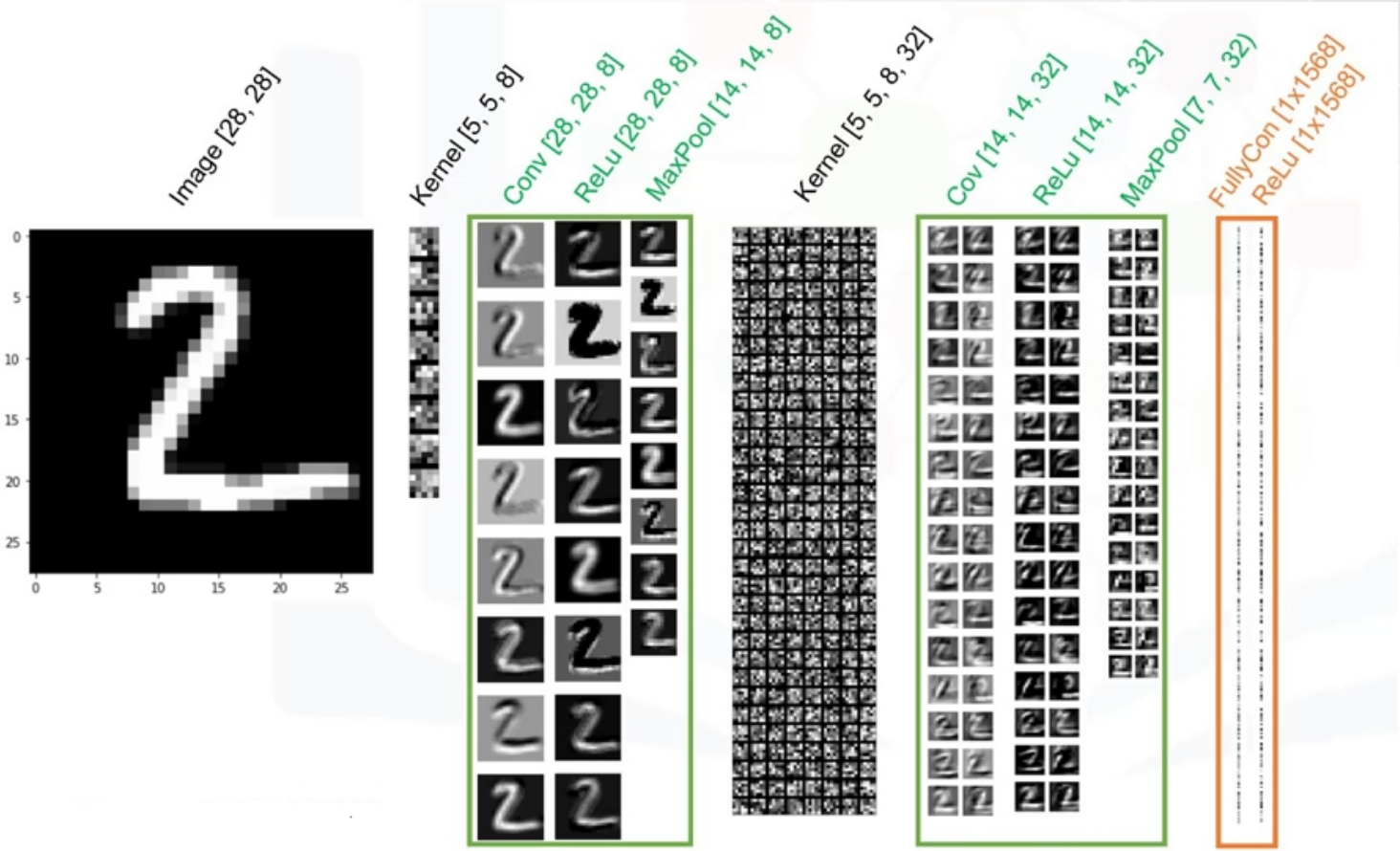
CNN Architecture



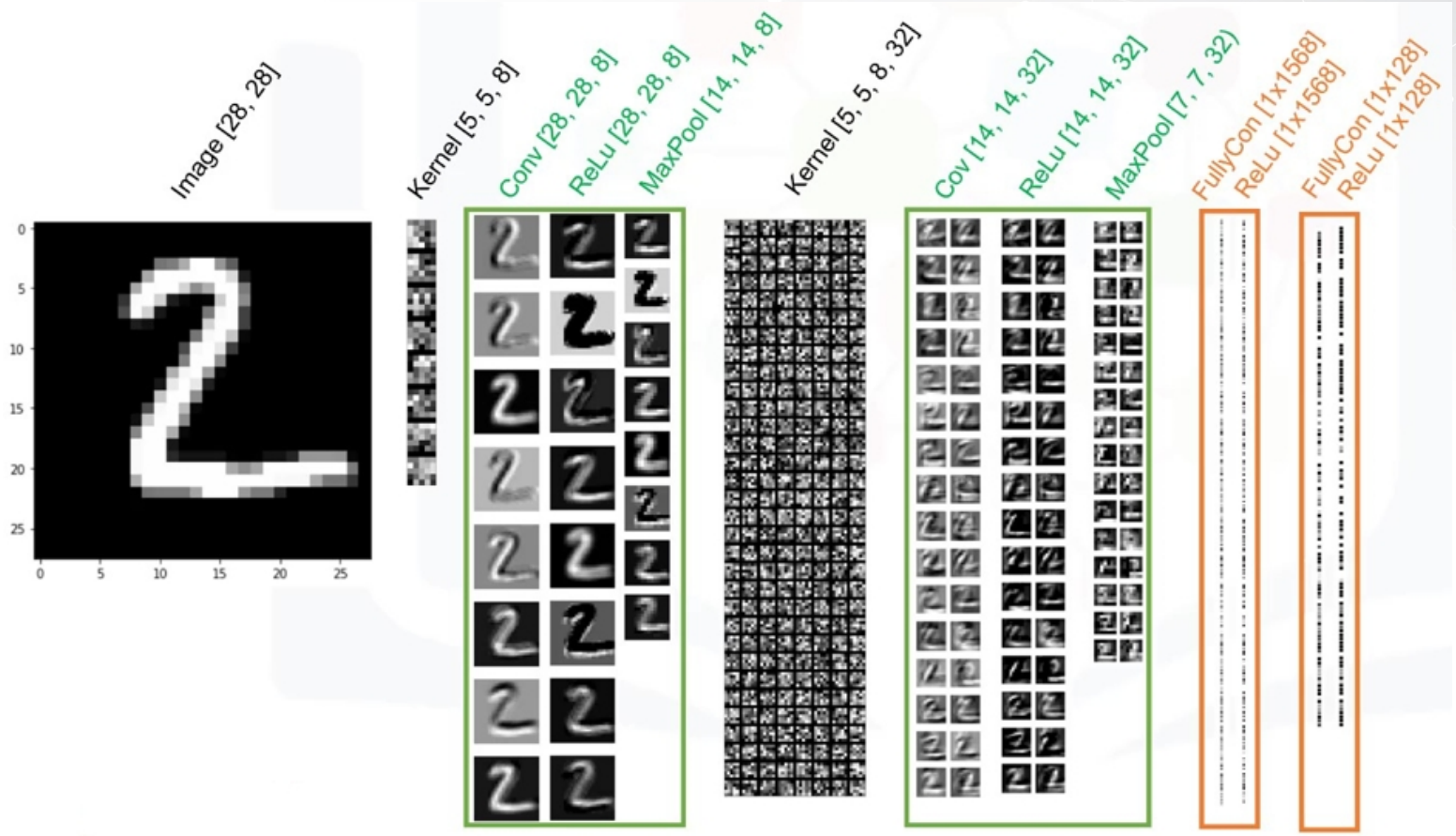
CNN Architecture



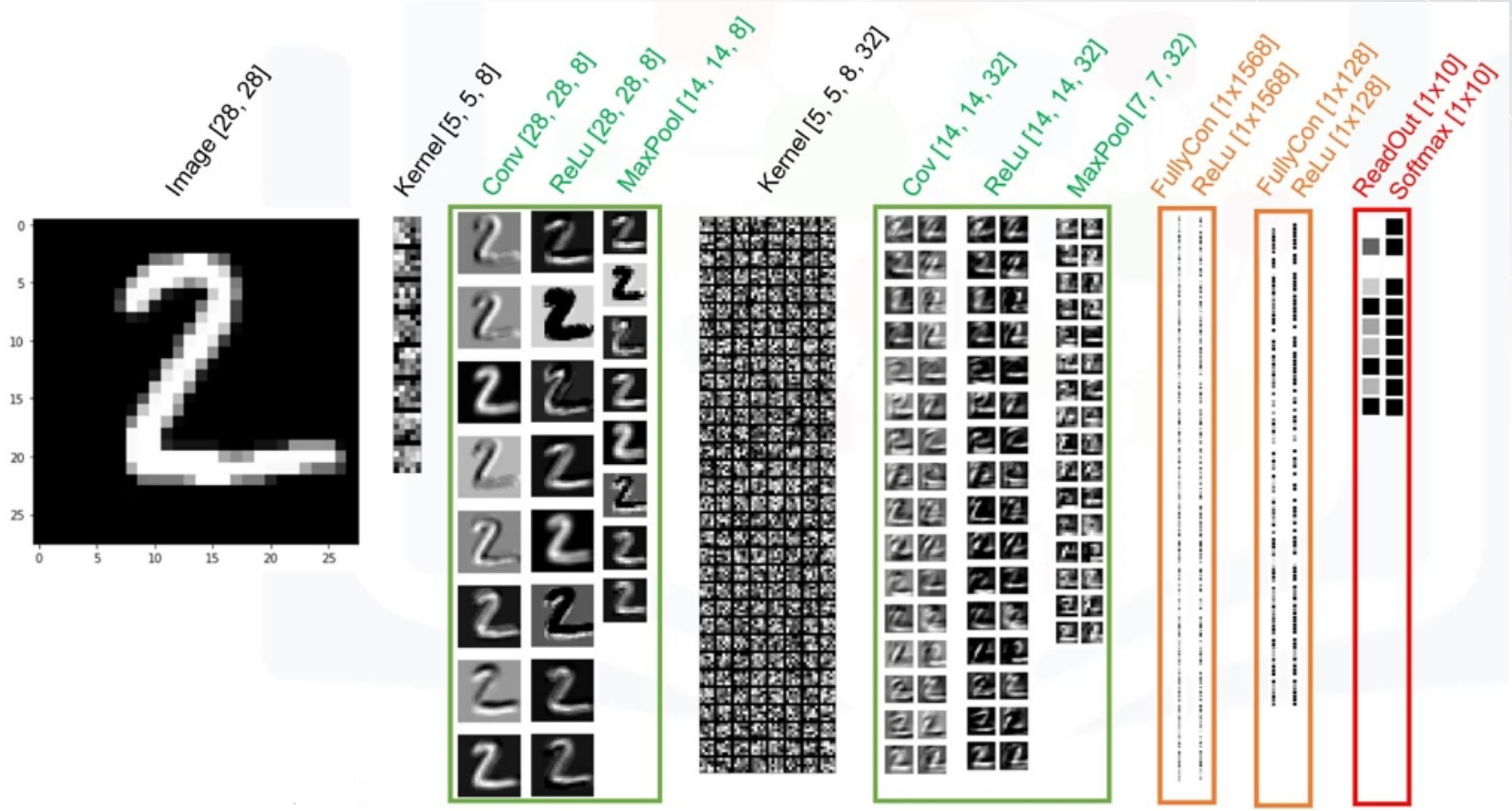
CNN Architecture



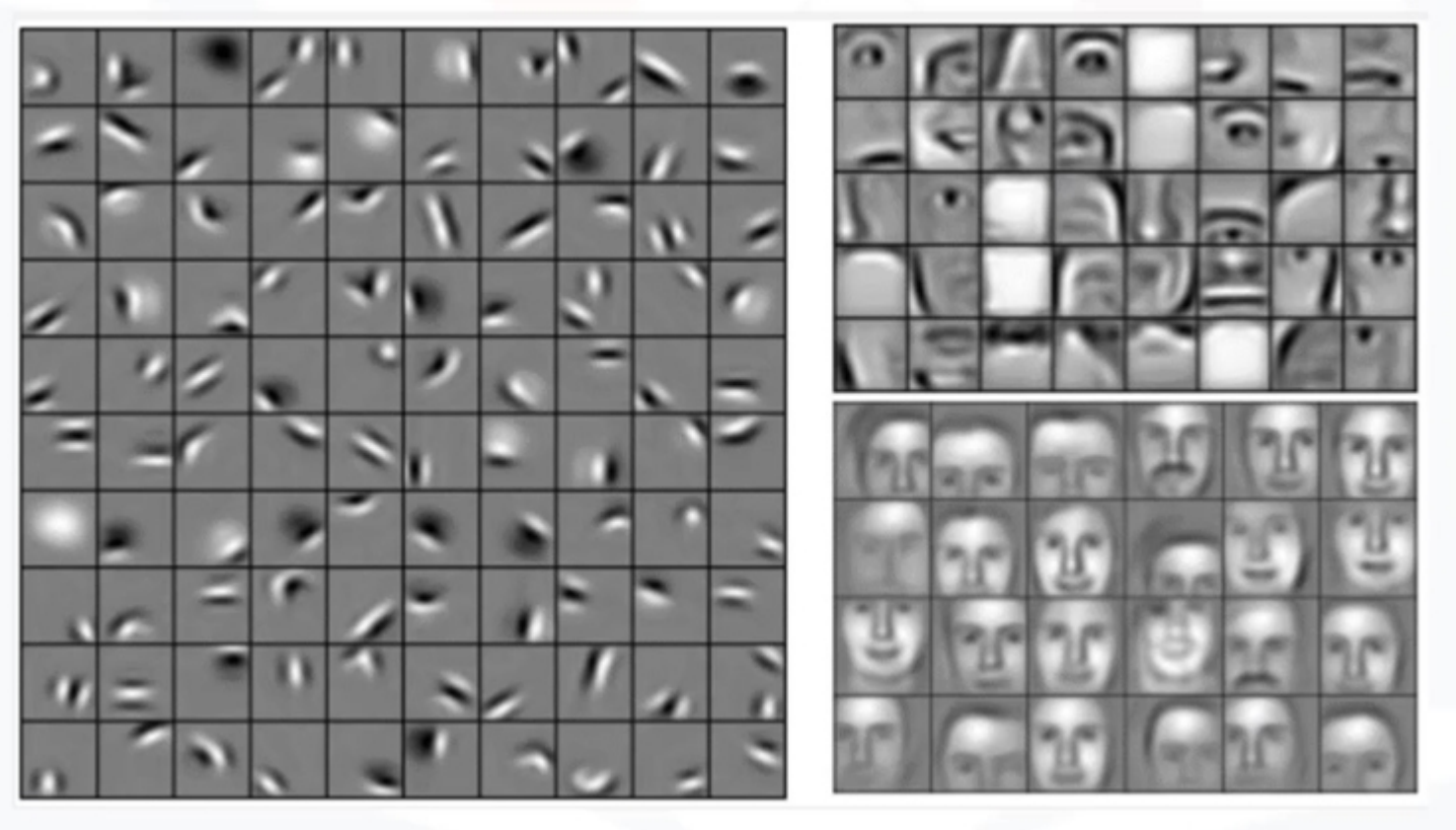
CNN Architecture



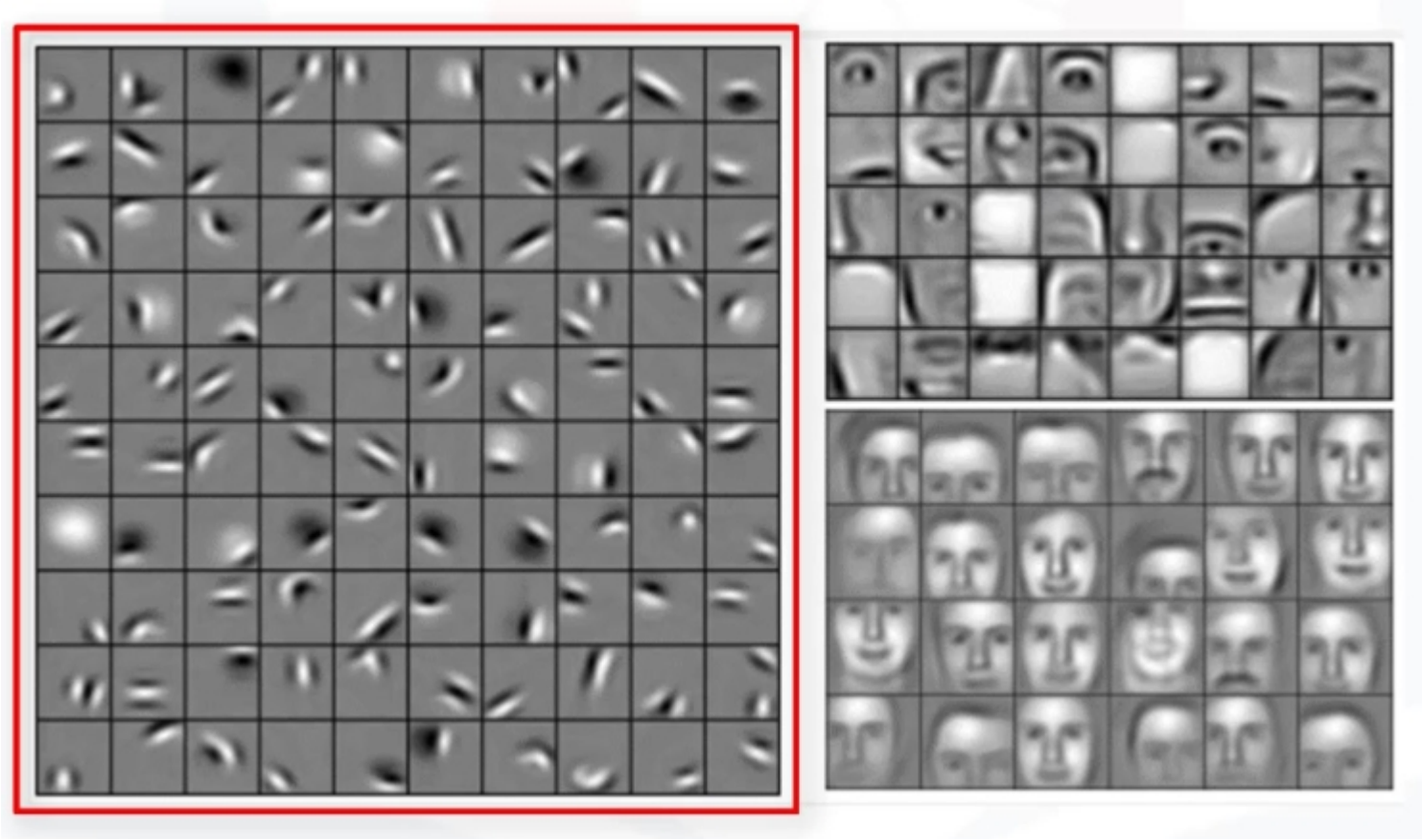
CNN Architecture



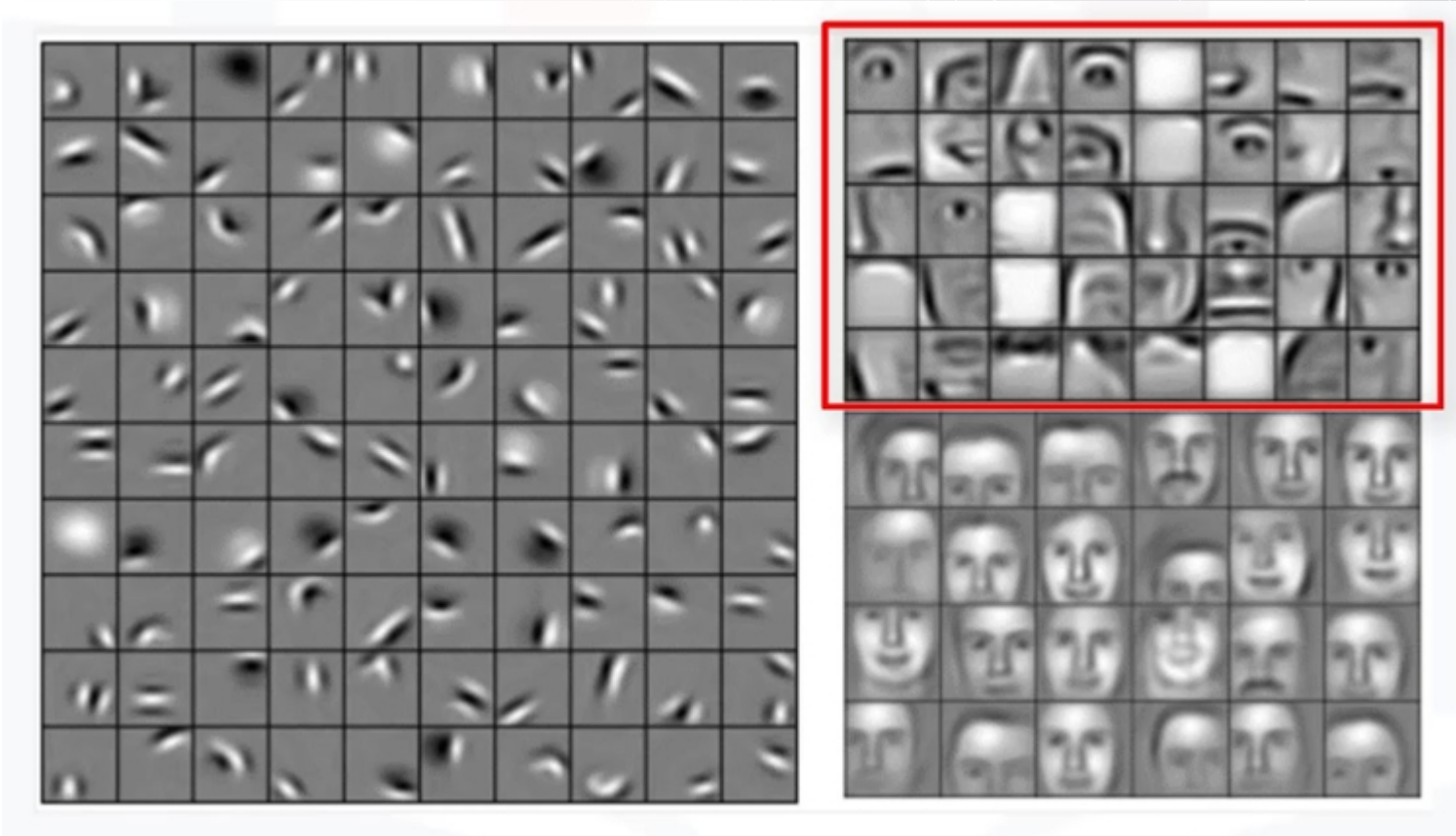
CNN on Faces



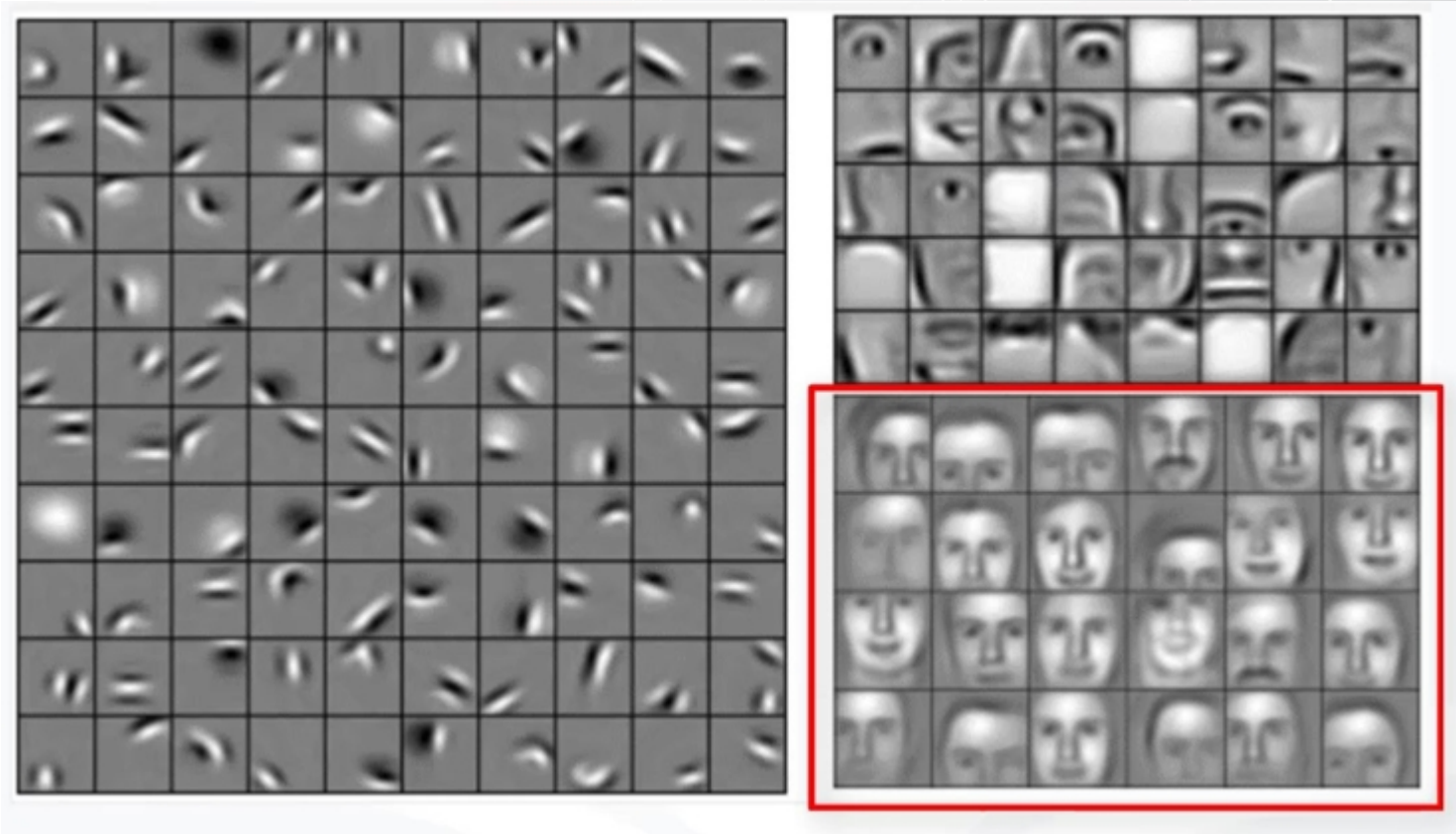
CNN on Faces



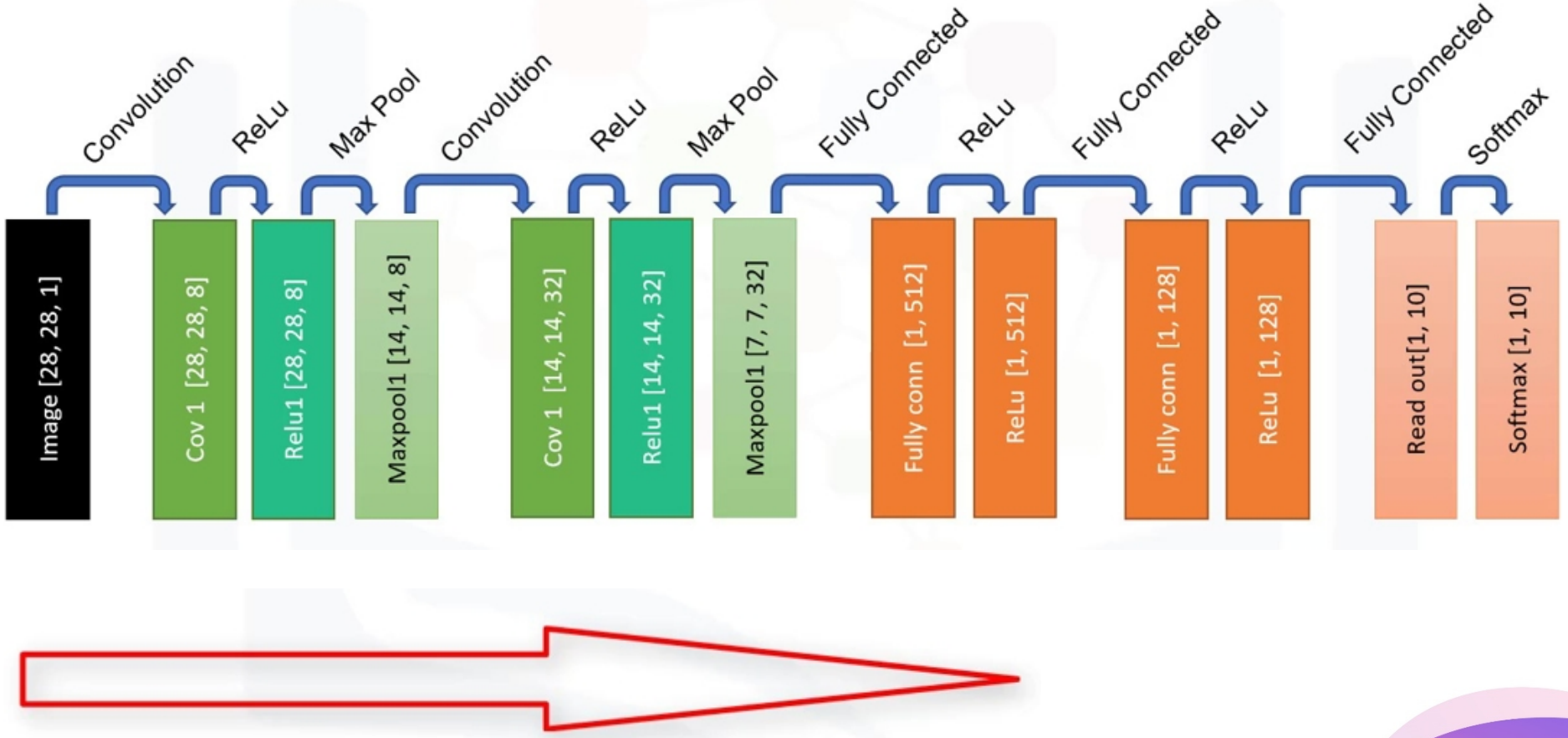
CNN on Faces



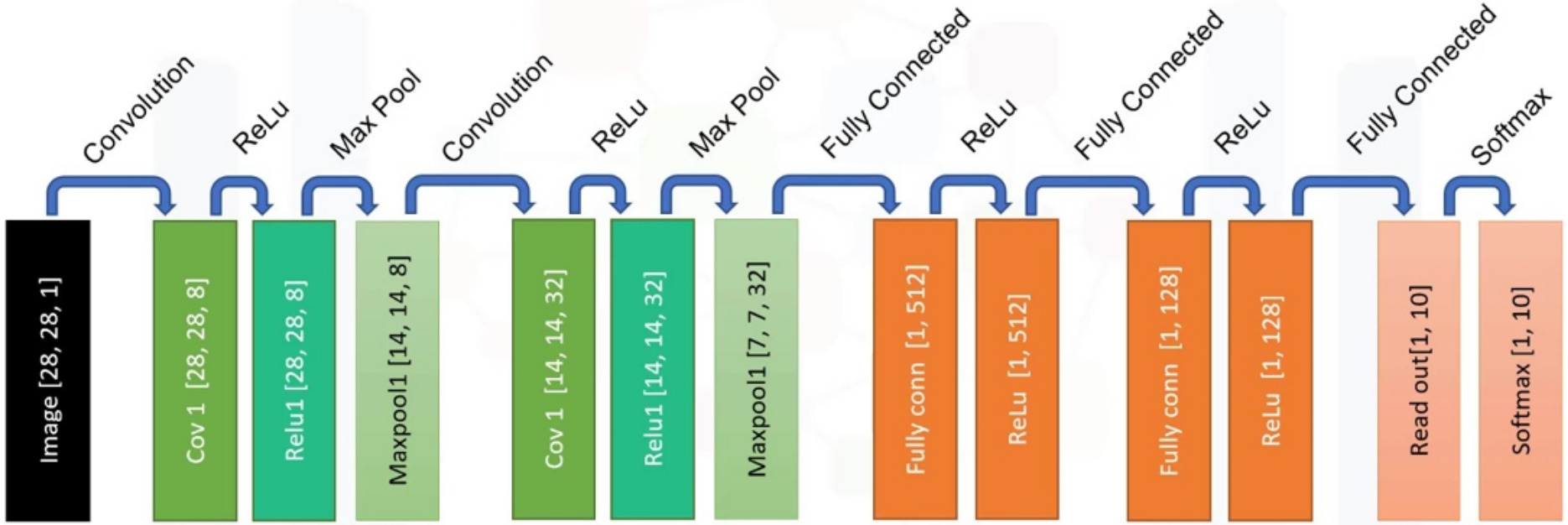
CNN on Faces



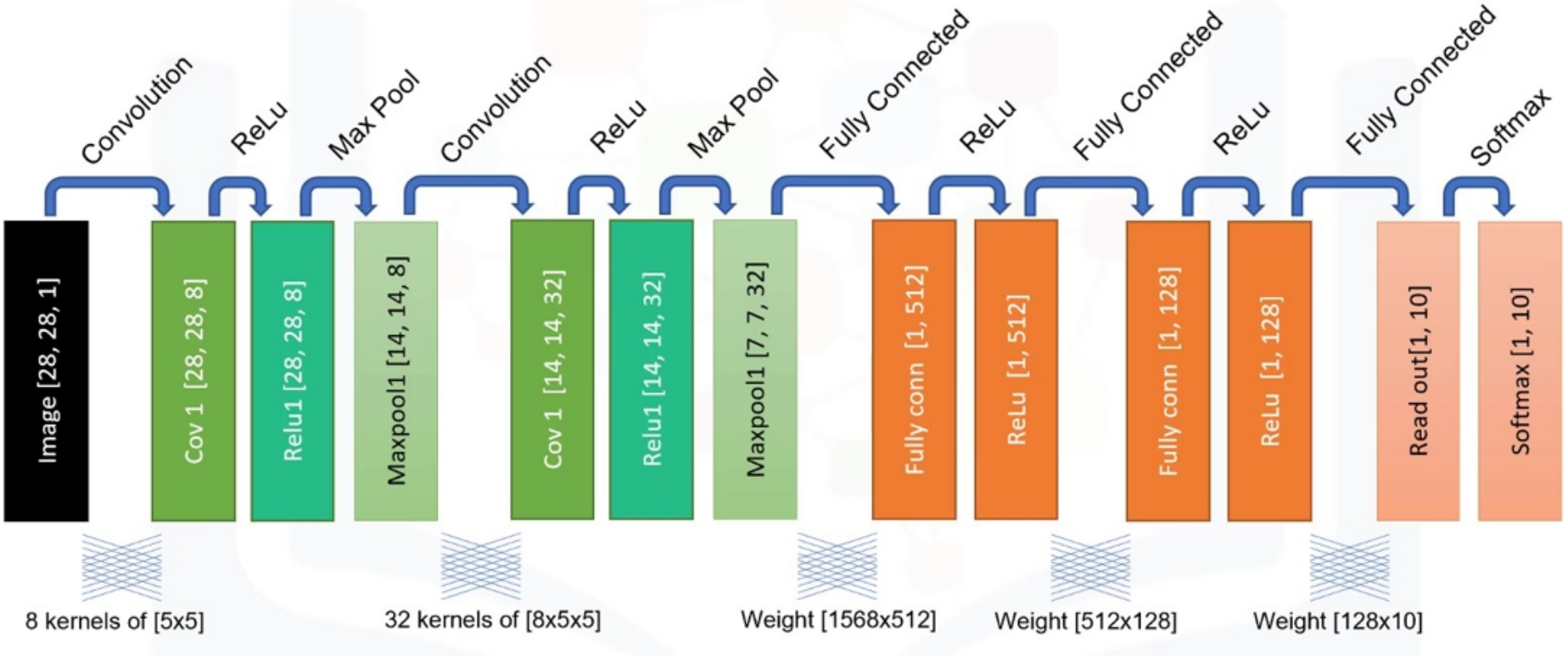
CNN Training



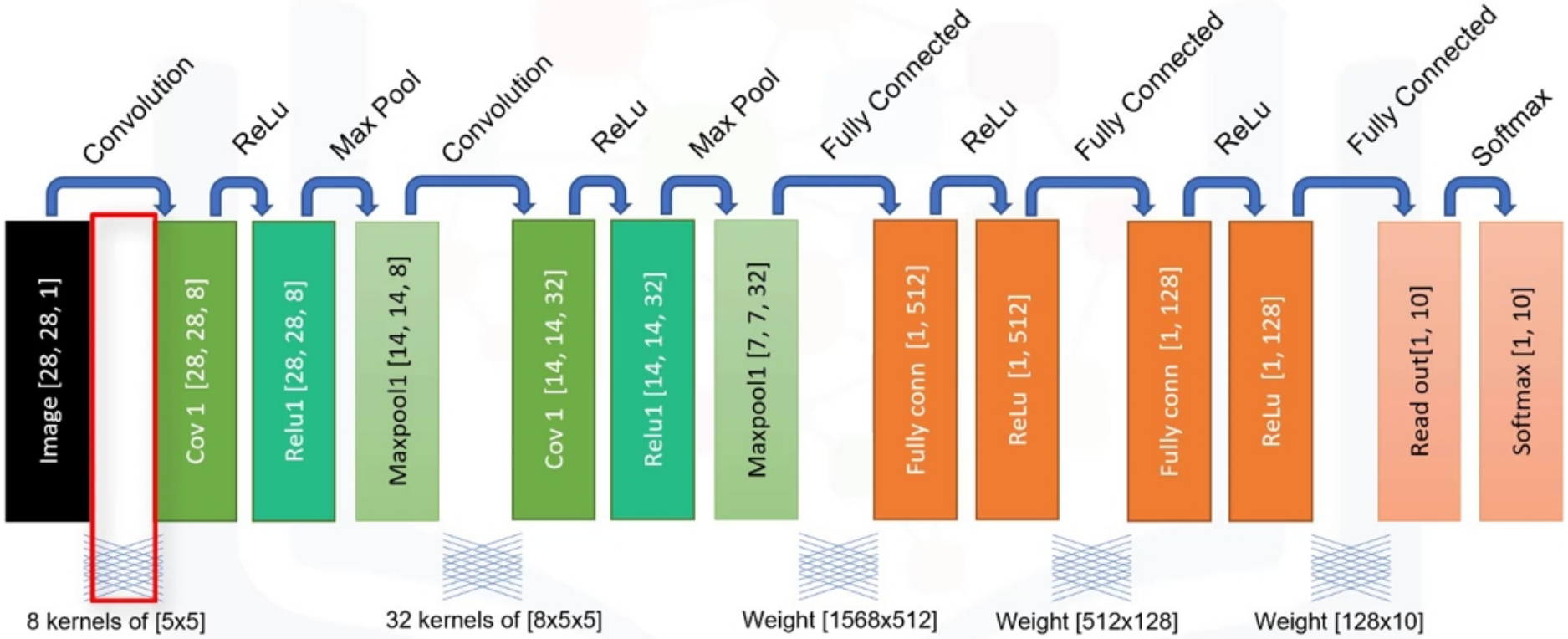
CNN Training



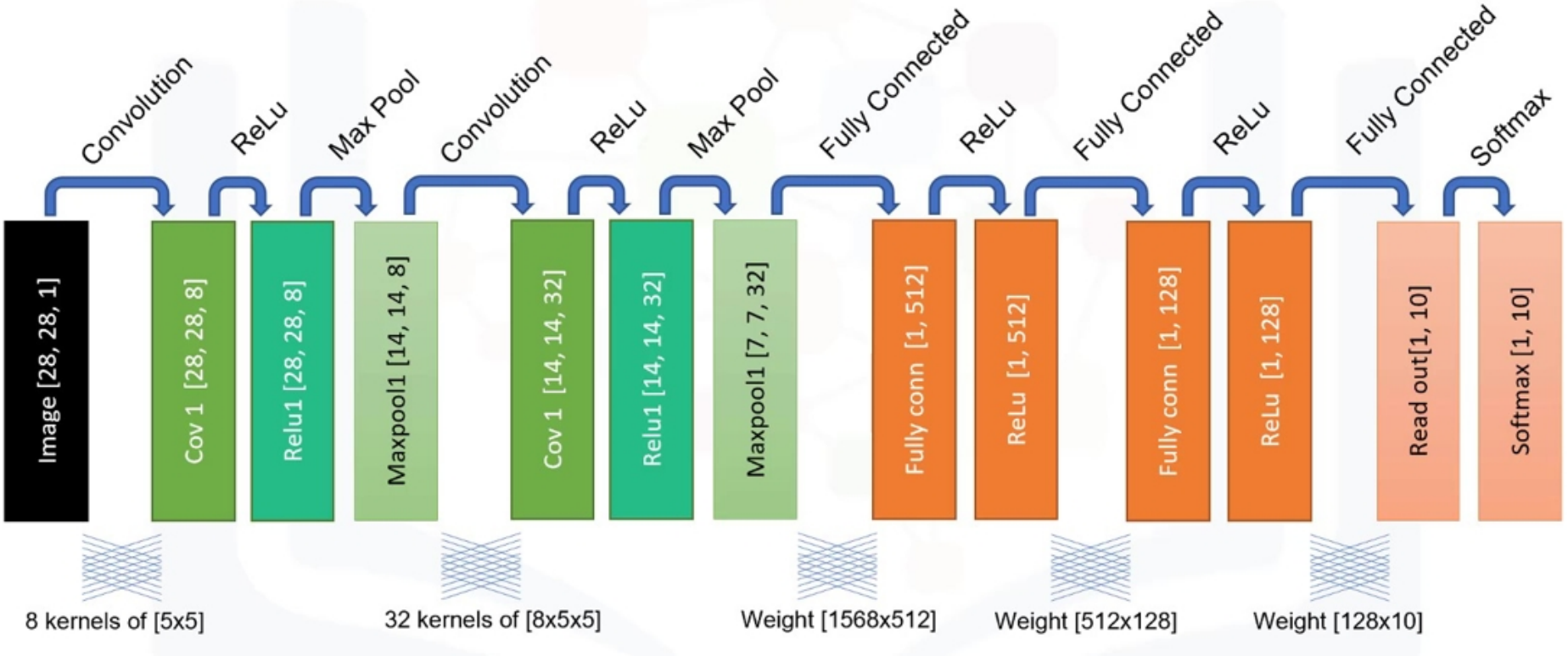
CNN Training



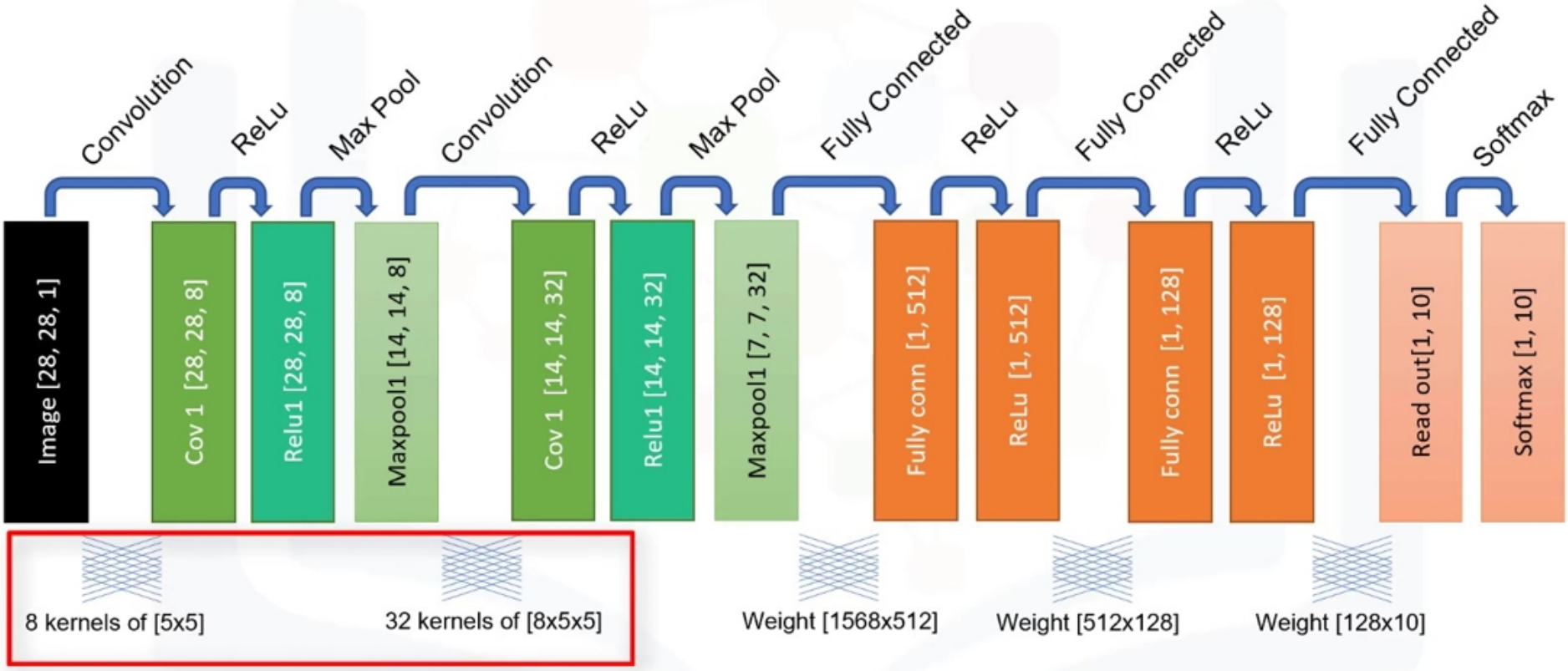
CNN Training



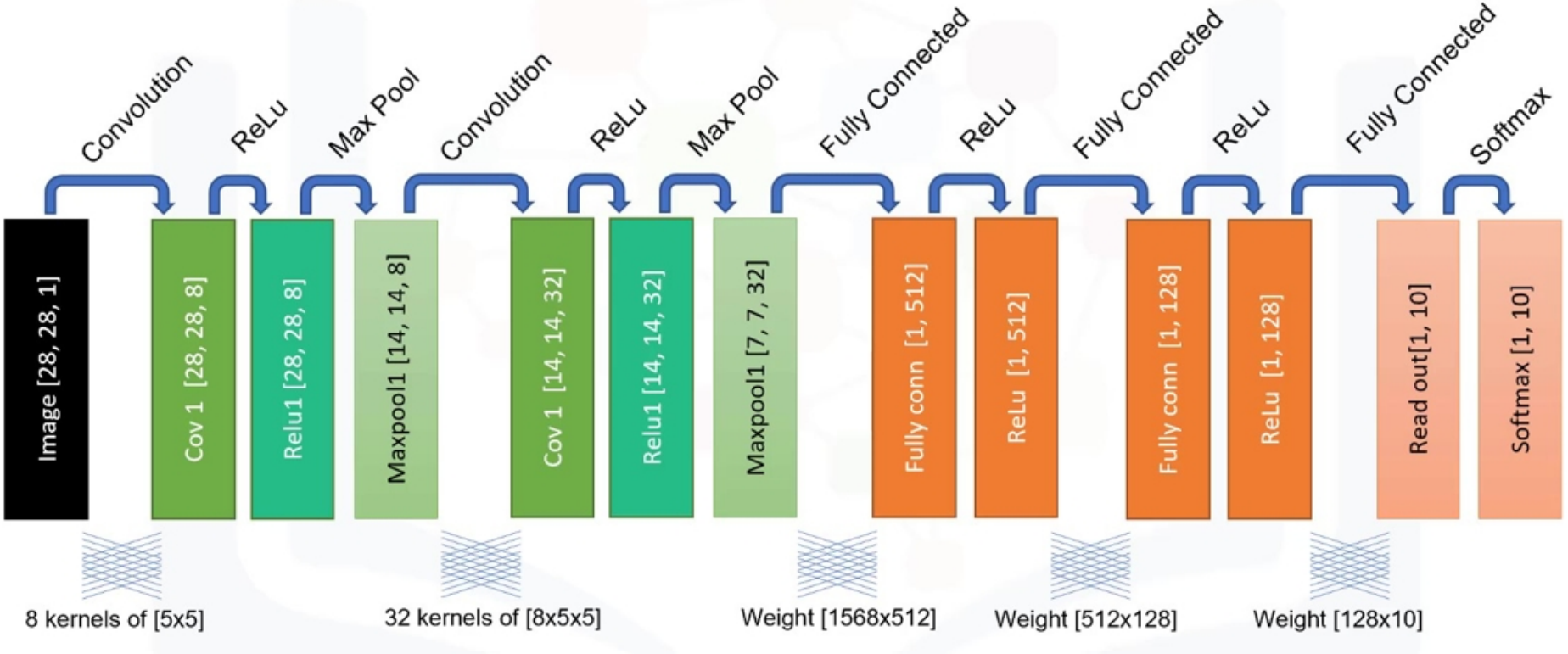
CNN Training



CNN Training

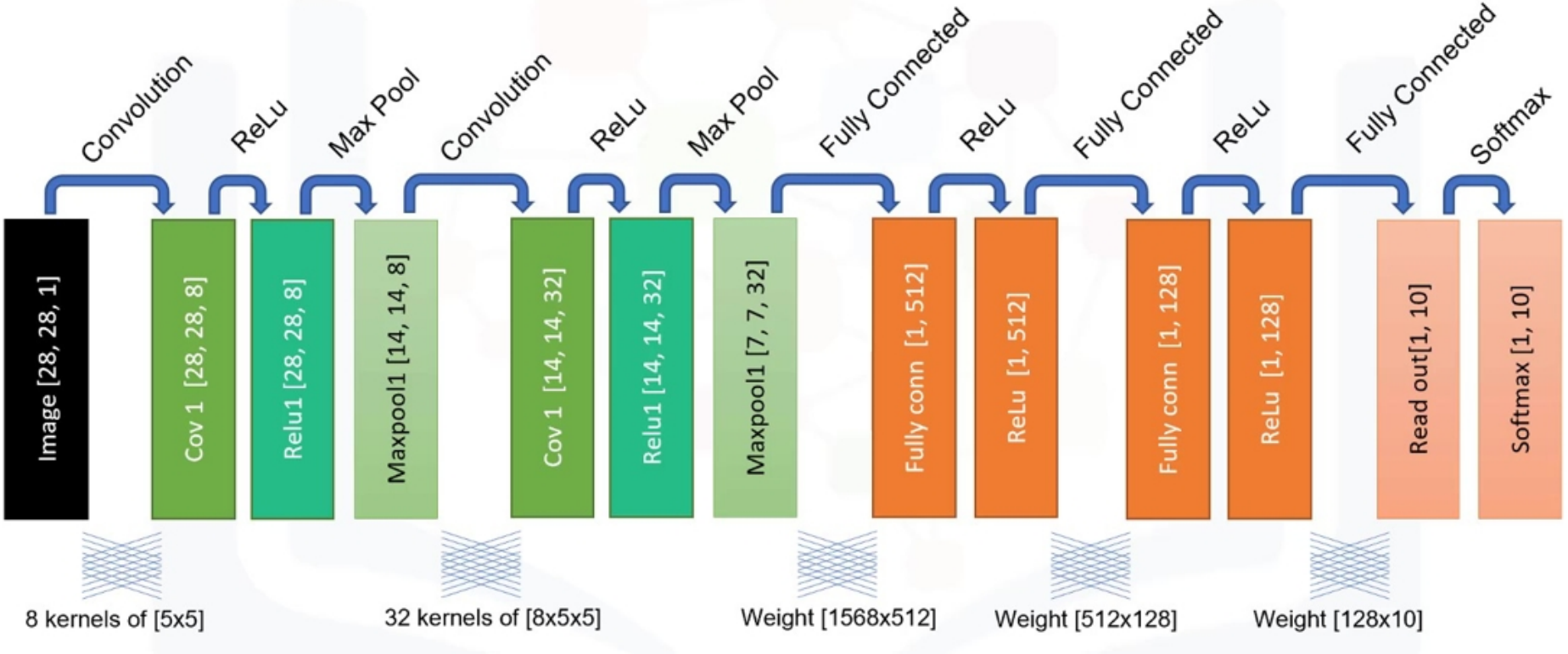


CNN Training



CNN Training

CNN Training



IKUTI KAMI

DIGITAL TALENT SCHOLARSHIP

digitalent.kominfo
digitalent.kominfo
DTS_kominfo
Digital Talent Scholarship 2019

Pusat Pengembangan Profesi dan Sertifikasi
Badan Penelitian dan Pengembangan SDM
Kementerian Komunikasi dan Informatika
Jl. Medan Merdeka Barat No. 9
(Gd. Belakang Lt. 4 - 5)
Jakarta Pusat, 10110