MIAI 504

NEURAL NETWORKS AND DEEP LEARNING

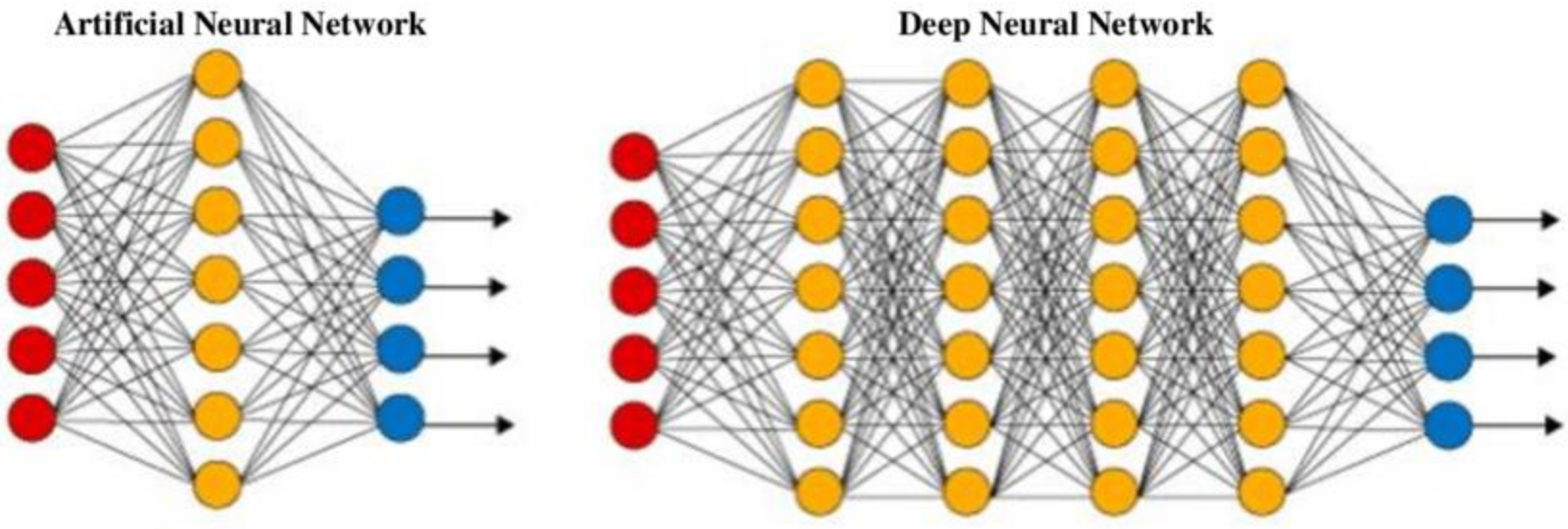
Dr. Zein Al Abidin IBRAHIM

zein.ibrahim@ul.edu.lb

SHALLOW & DEEP NETWORKS

TYPES OF NETWORKS

- Two main type of networks
  - Shallow neural networks
  - Deep (very deep, …) neural networks



SHALLOW NETWORKS

- **Shallow neural networks**
  - consist of one to two hidden layers between the input and output layers.
  - Simpler and easier to train compared to deep neural networks but are usually less capable of capturing complex patterns in data.
- **Characteristics**
  - Number of Layers: Usually 1-2 hidden layers.
  - Complexity: Lower complexity, easier to interpret.
  - Training Time: Generally faster to train due to fewer parameters.
  - Applications: Suitable for simpler tasks where the relationships in the data are relatively straightforward. Commonly used in situations where the amount of data is limited, or interpretability is important.
- **Examples**
  - Perceptrons: The most basic form of neural networks.
  - Multilayer Perceptrons (MLP): If it has one or two hidden layers, it's considered a shallow network.



DEEP NETWORKS

- **Deep neural networks**
  - have multiple hidden layers between the input and output. The term "deep" refers to the depth of these layers.
  - can model more complex patterns but require more data and computational resources to train.
- **Characteristics**
  - Number of Layers: Usually more than 2 hidden layers. modern architectures can have **dozens or even hundreds** of layers
  - Complexity: Higher complexity, capable of capturing intricate patterns and representations.
  - Training Time: Longer training times due to increased number of parameters and complexity, often requiring specialized hardware like GPUs.
  - Applications: Suitable for complex tasks such as image and speech recognition, natural language processing, and any application requiring the modelling of high-dimensional data.
- **Examples**
  - CNN, RNN (LSTM, BLSTM, GRU), GAN, Transformers, ResNet, DenseNet …



CONVOLUTIONAL NEURAL NETWORKS

DEEP LEARNING

DEEP NEURAL NETWORKS PROBLEMATICS



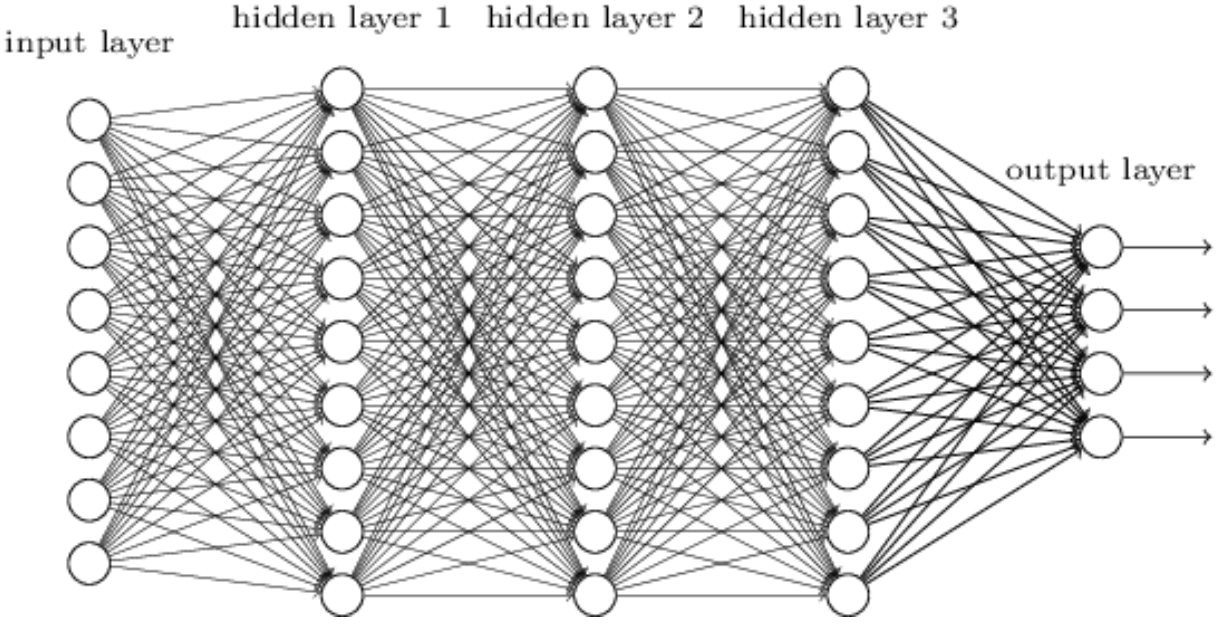
- Image size = 500x500x3
- Input dimension = 500*500*3=750000
- Number of neurons in the first hidden layer = 100

->Number of weights for the first hidden layer = 750000*100 = 75 million !!!

7

UTILITY OF FULLY CONNECTED LAYERS

- We know it is good to learn a small model.
- From this fully connected model, do we really need all the edges?
- Can some of these be shared?



CONSIDER LEARNING AN IMAGE

- Some patterns are much smaller than the whole image

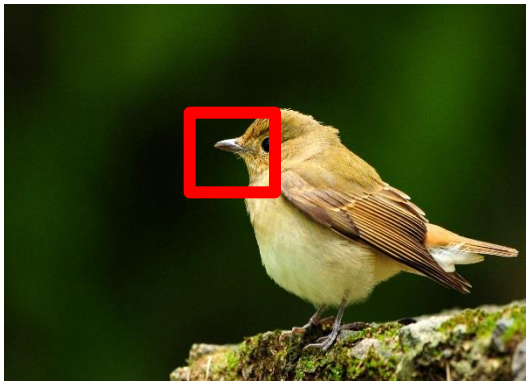
Can represent a small region with fewer parameters

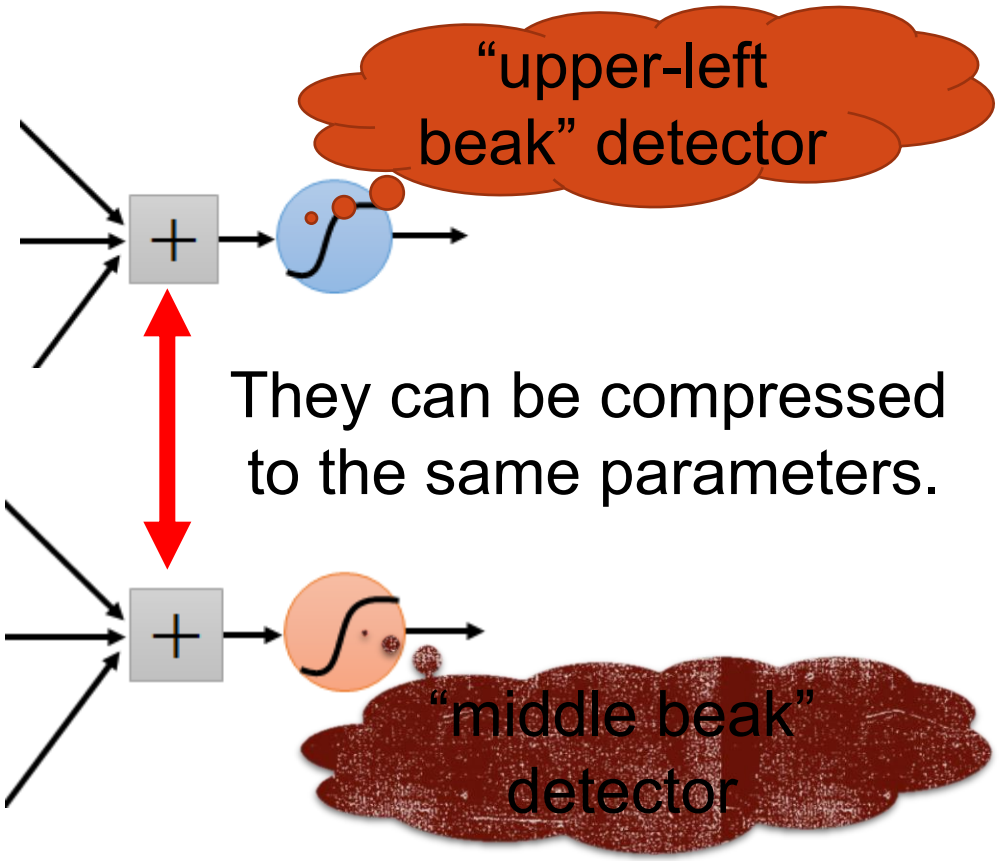


SAME PATTERN APPEARS IN DIFFERENT PLACES: THEY CAN BE COMPRESSED!

- What about training a lot of such "small" detectors and each detector must "move around".

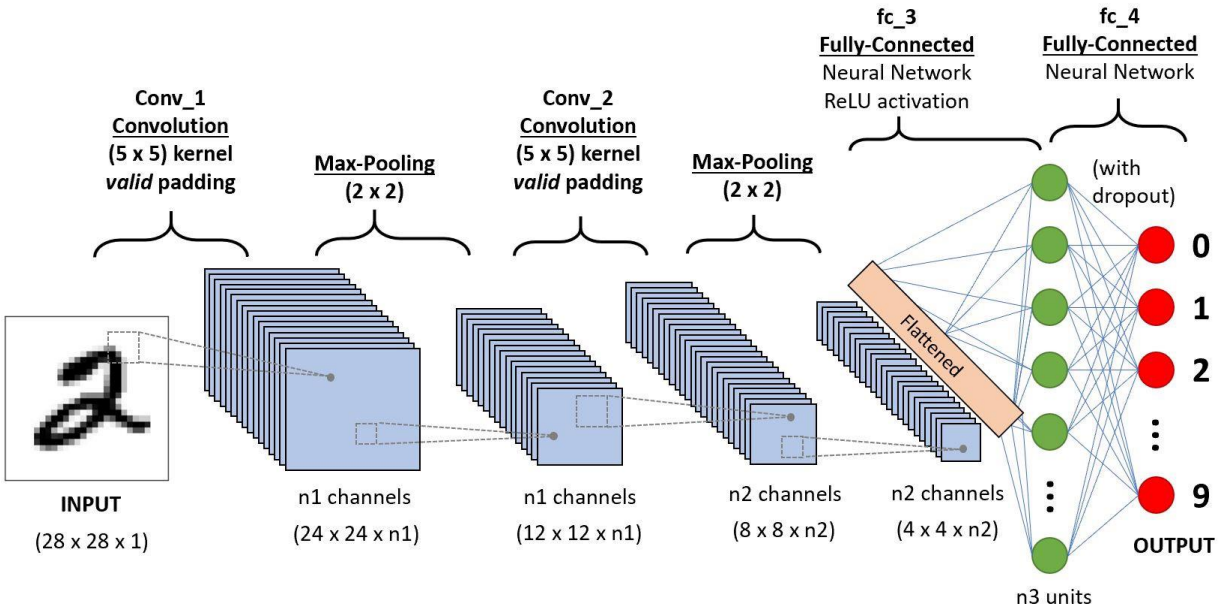






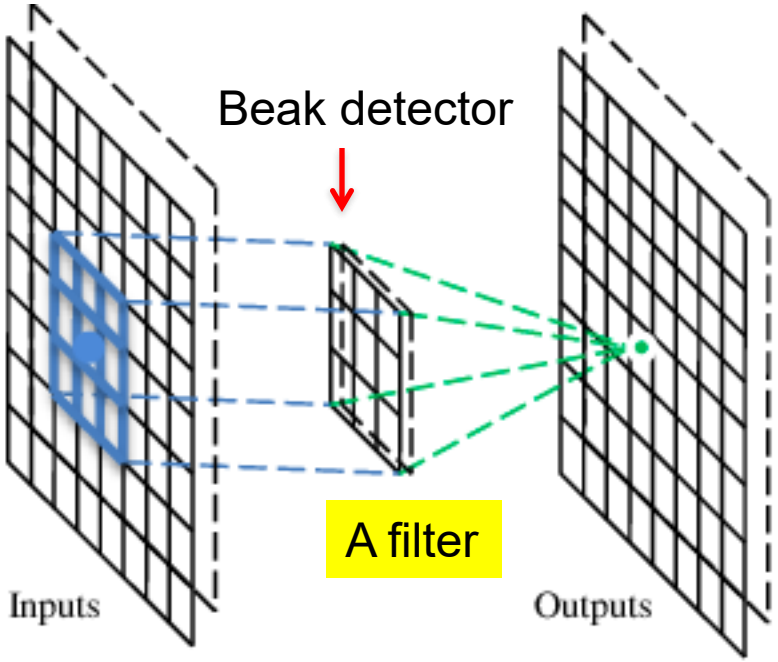
CONVOLUTION NEURAL NETWORKS

- A convolution neural network (CNN) is an MLP designed to recognize two-dimensional shapes with a high degree of invariance to translation, scaling and skewing and other sorts of distortion.
- They are mostly used in computer vision applications.



A CONVOLUTIONAL LAYER

- A CNN is a neural network with some convolutional layers (and some other layers).
- A convolutional layer has a number of filters that does convolutional operation.



CONVOLUTION

These are the network parameters to be learned.

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

⋮ ⋮

Each filter detects a small pattern (3 x 3).

CONVOLUTION

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

stride=1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

Dot product

3 -1

6 x 6 image

CONVOLUTION

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

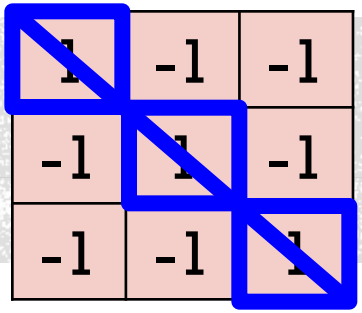
If stride=2

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

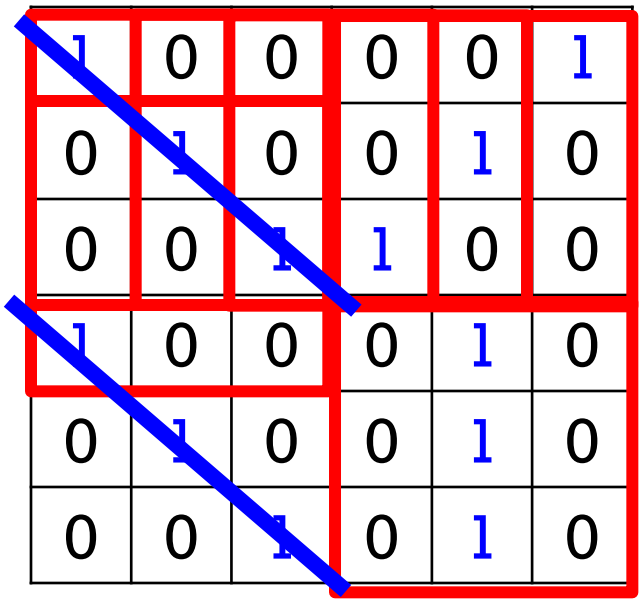
3 -3

CONVOLUTION

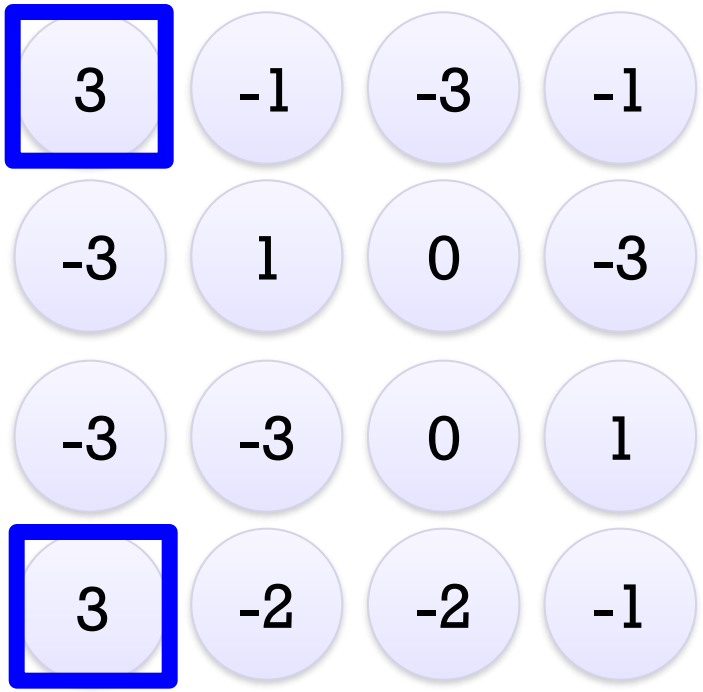


Filter 1

stride=1



6 x 6 image



CONVOLUTION

-1	1	-1
-1	1	-1
-1	1	-1

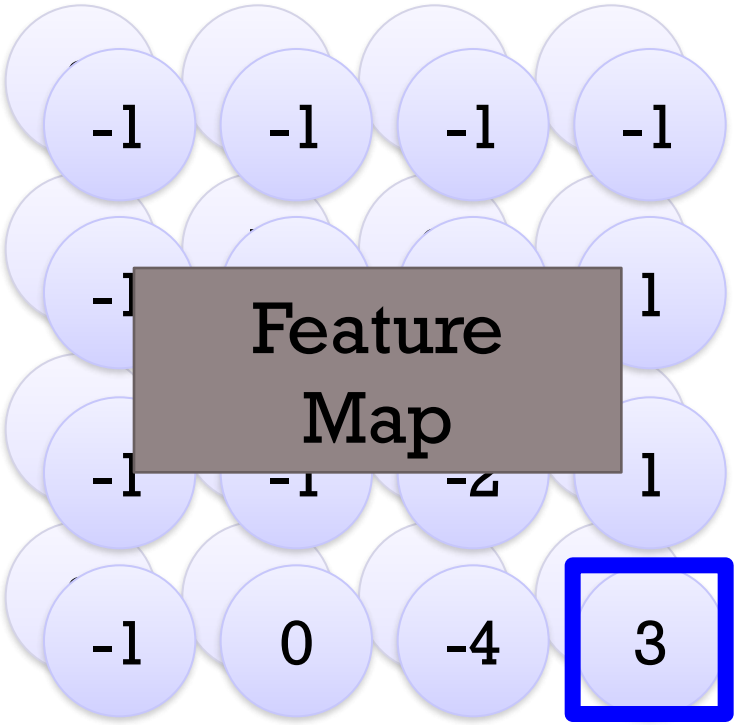
Filter 2

stride=1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

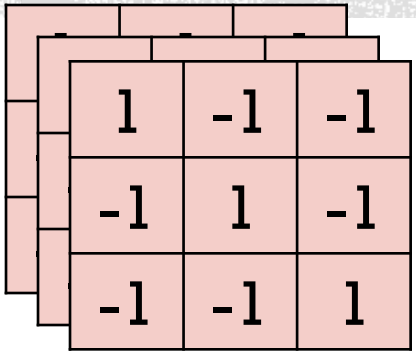
6 x 6 image

Repeat this for each filter

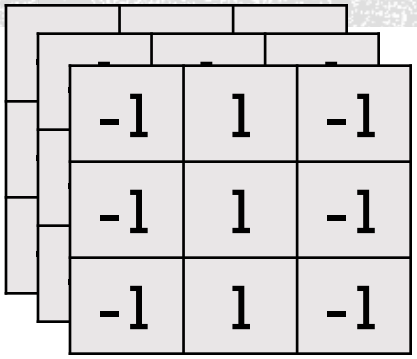


Two 4 x 4 images
Forming 2 x 4 x 4 matrix

COLOR IMAGE: RGB 3 CHANNELS

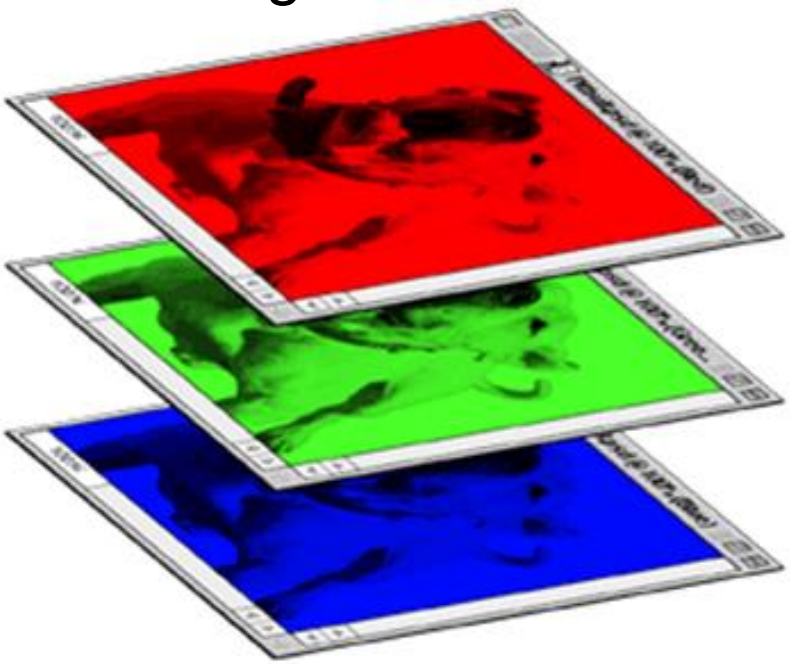


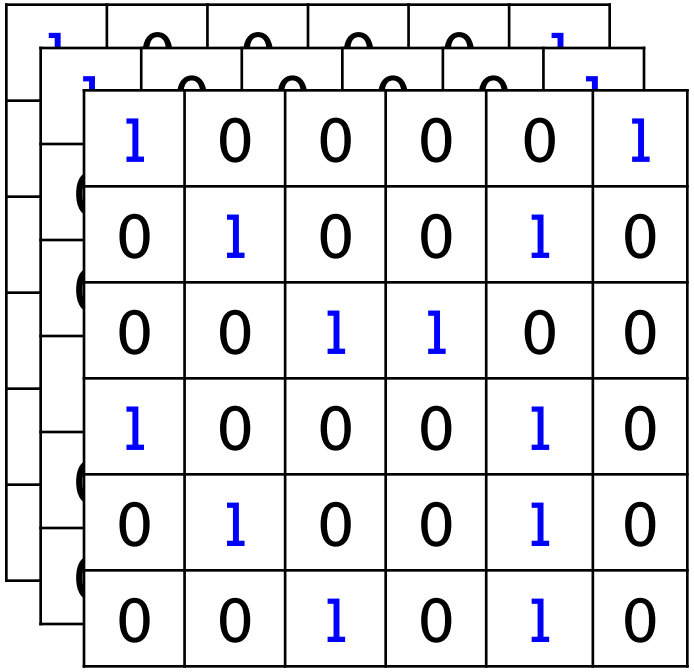
Filter 1



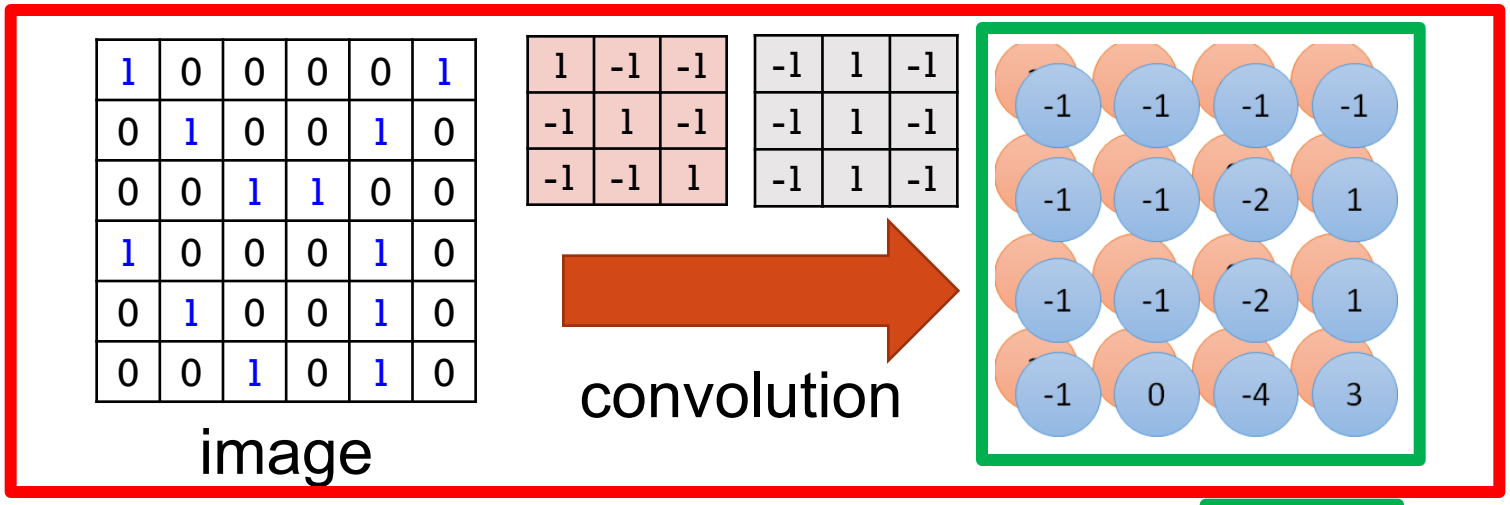
Filter 2

Color image

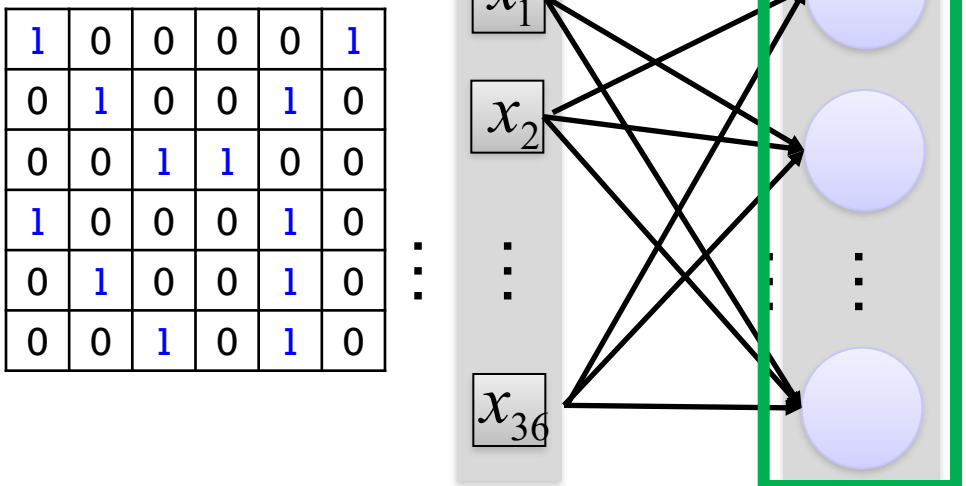


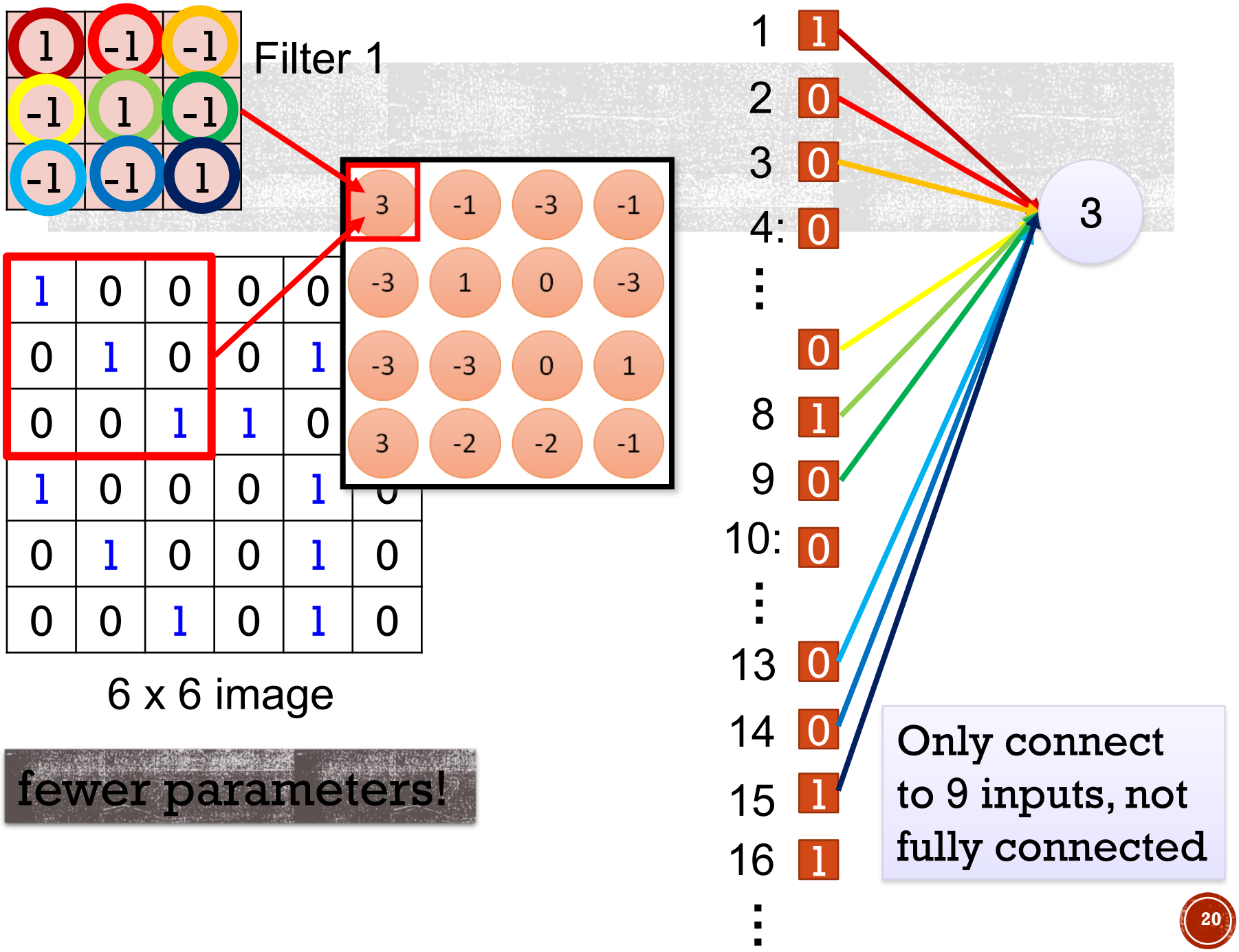


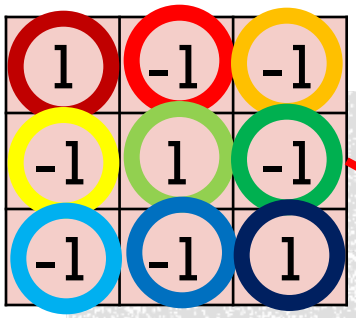
CONVOLUTION VS FULLY CONNECTED



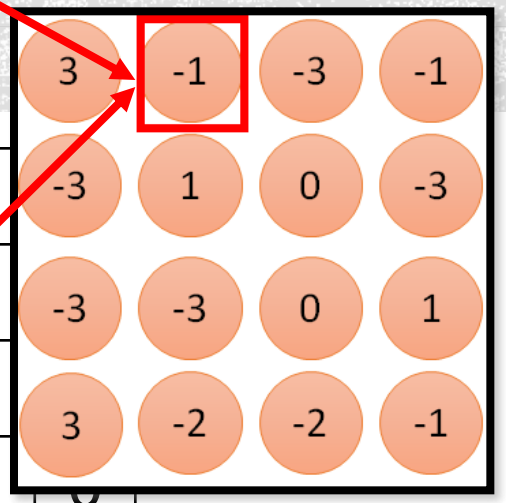
Fully-connected




1 -1 -1
-1 1 -1
-1 -1 1
Filter 1
3 -1 -3 -1
-3 1 0 -3
-3 -3 0 1
3 -2 -2 -1
1 0 0 0 0
0 1 0 0 1
0 0 1 1 0
1 0 0 0 1 0
0 1 0 0 1 0
0 0 1 0 1 0
6 x 6 image
1
2
3
4:
8
9
10:
13
14
15
16
1
0
0
0
0
1
0
0
0
1
1
3
Only connect to 9 inputs, not fully connected
fewer parameters!



Filter 1

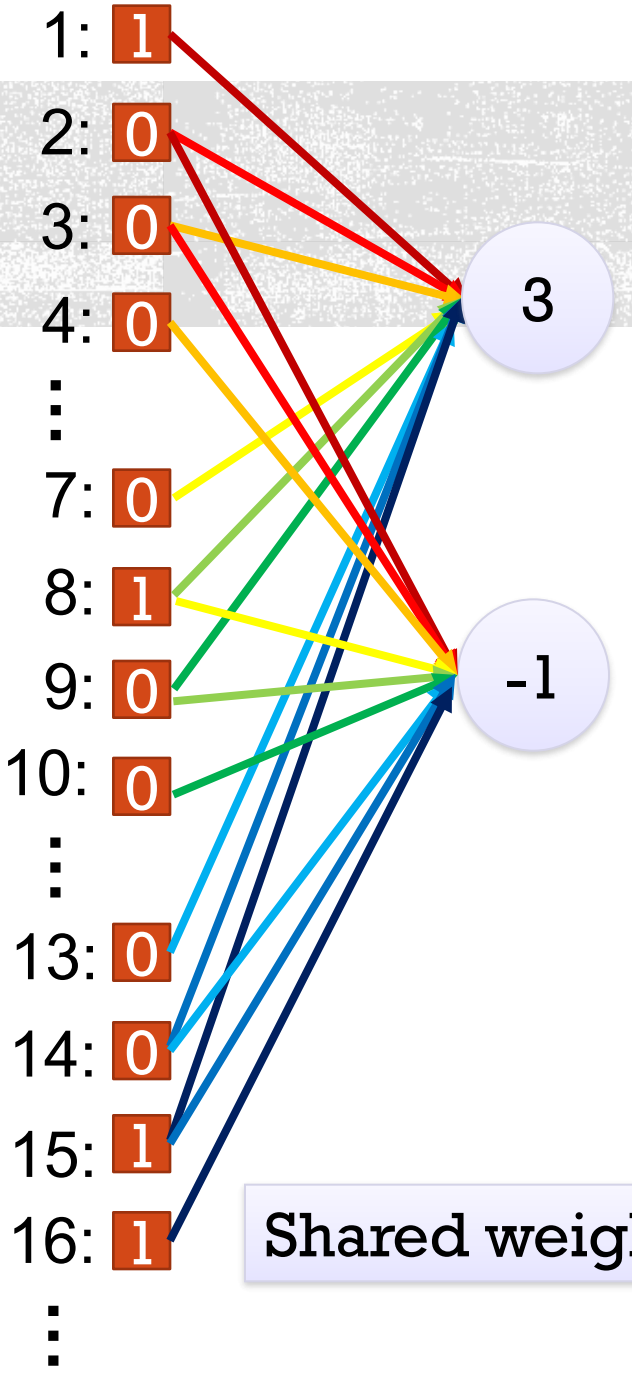


1	0	0	0	0	
0	1	0	0	1	
0	0	1	1	0	
1	0	0	0	1	
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

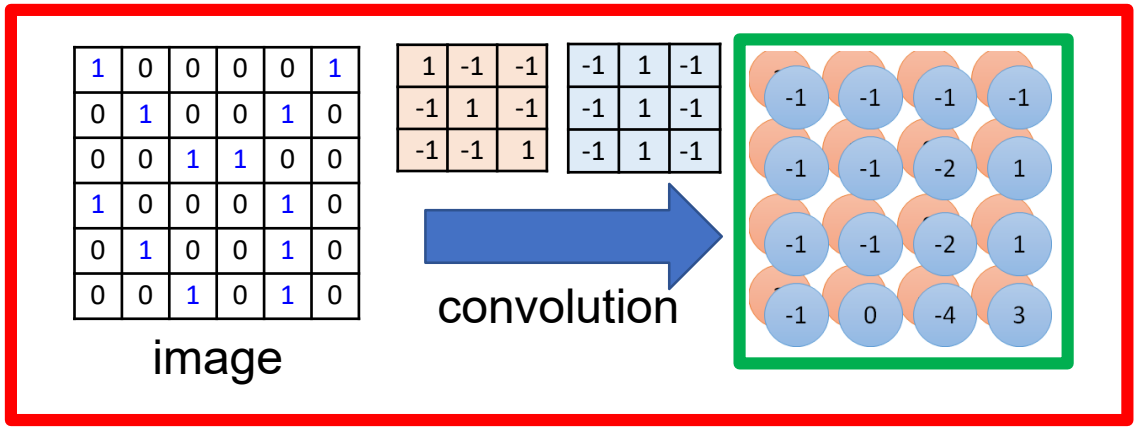
Fewer parameters

Even fewer parameters

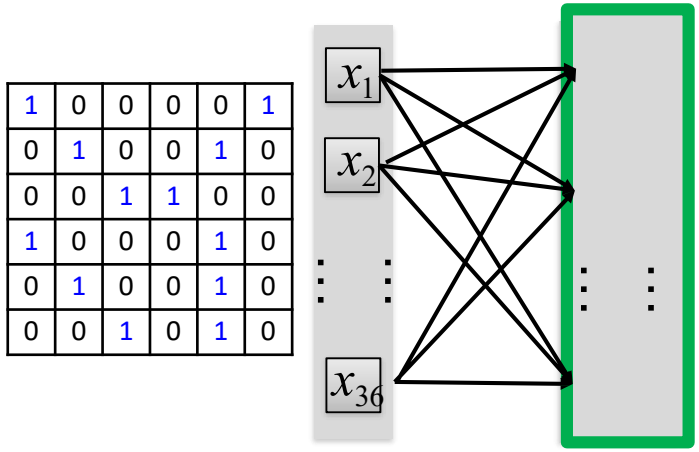


Shared weights

CONVOLUTION V.S. FULLY CONNECTED

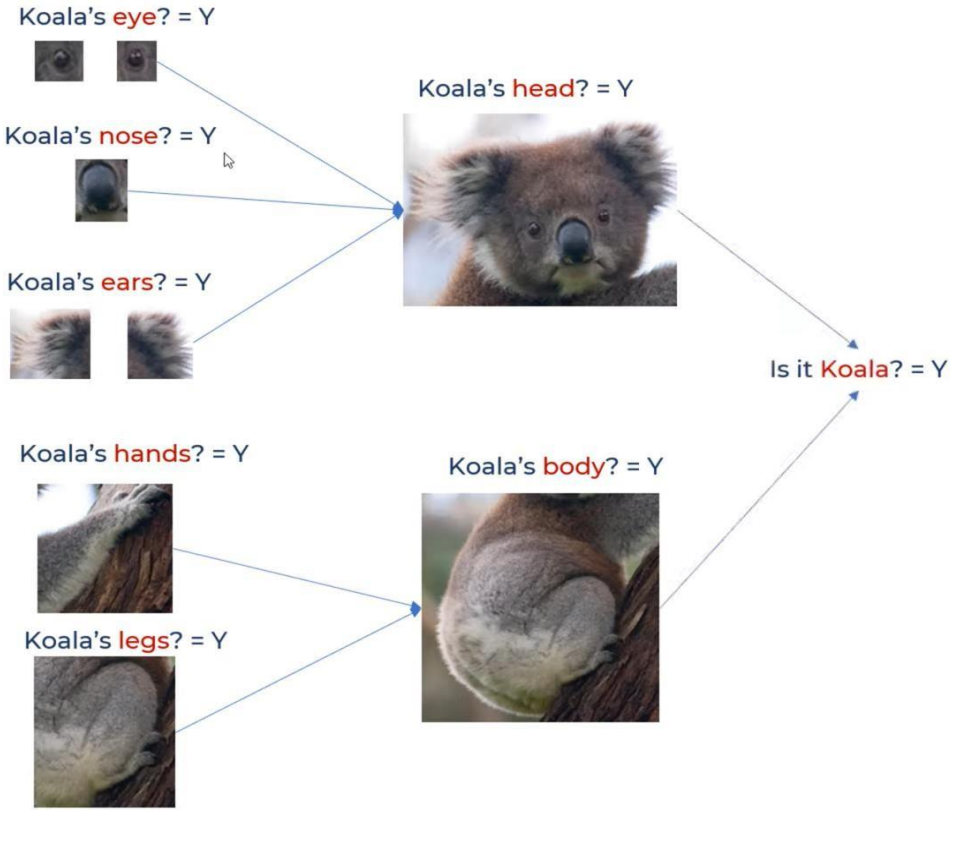


Fully-connected

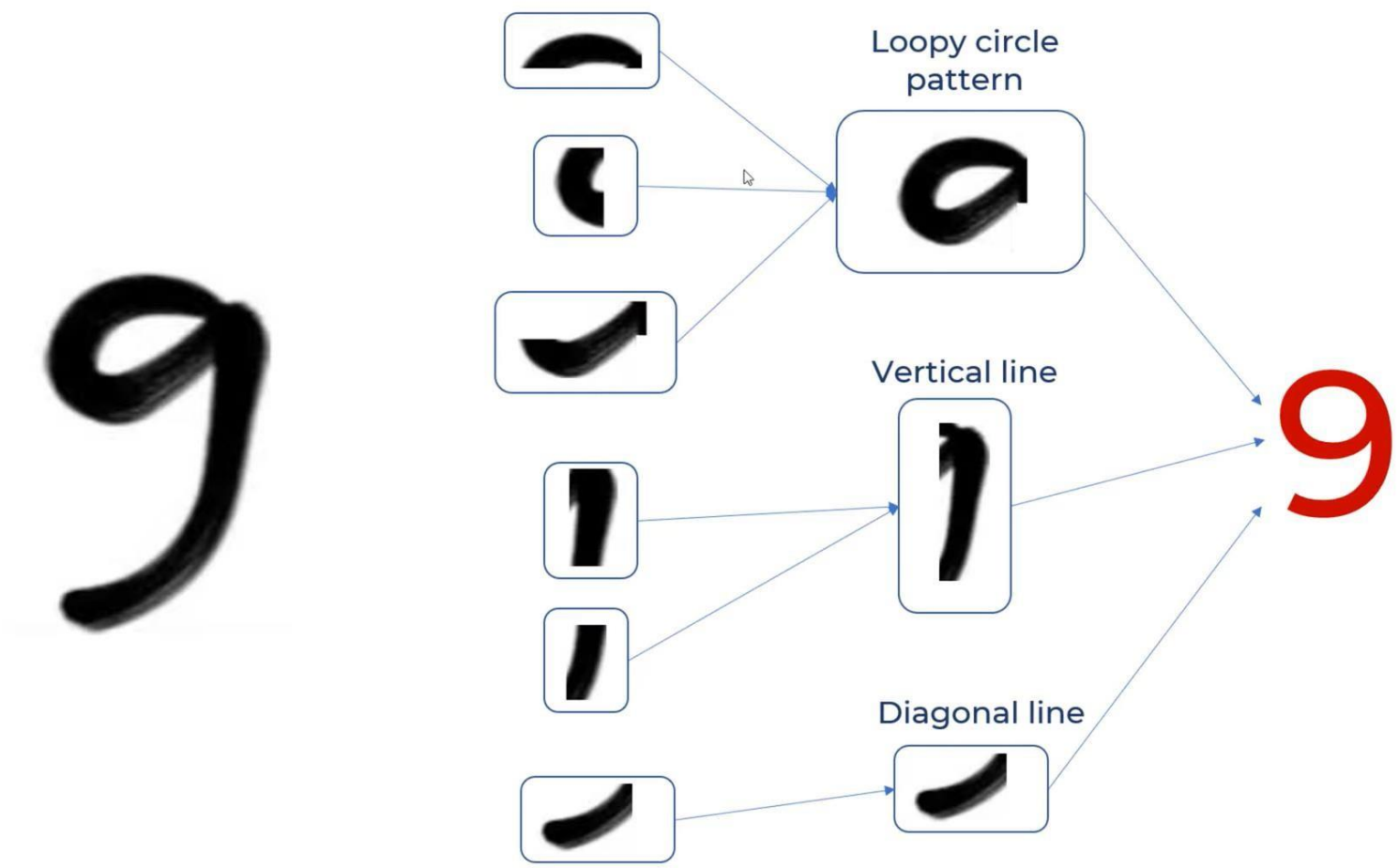


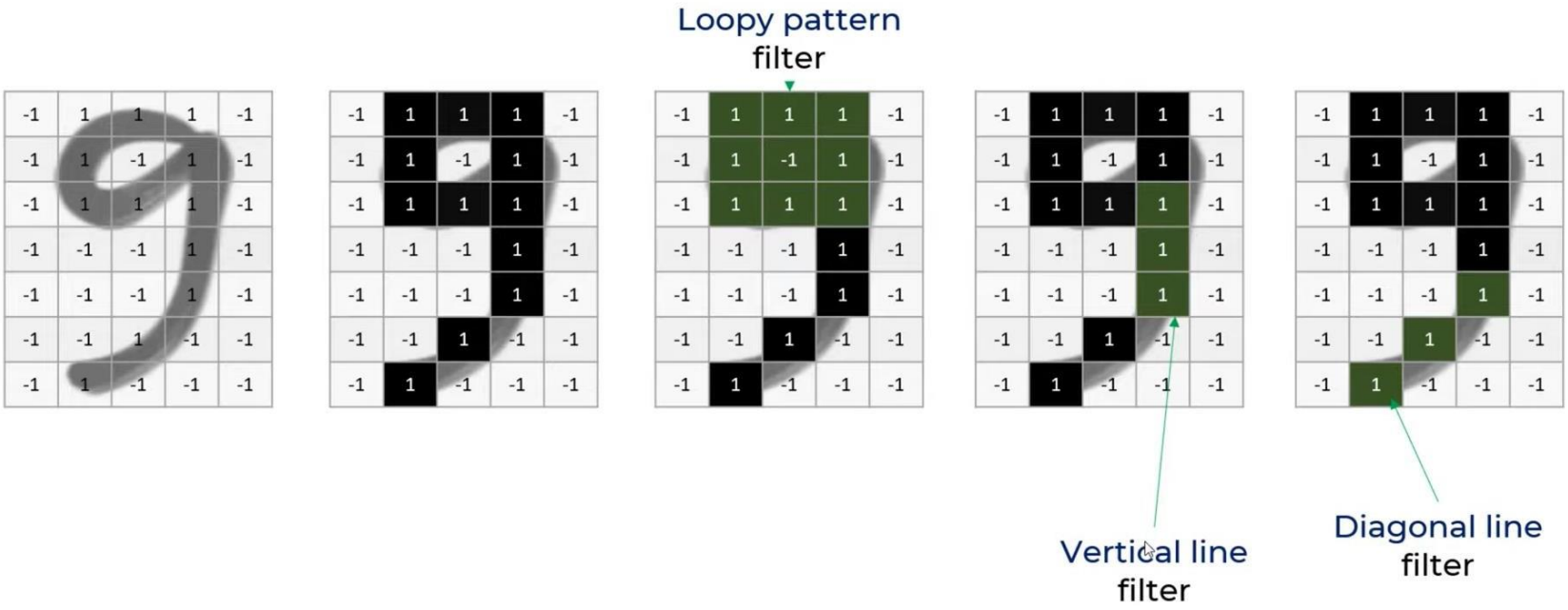
EXAMPLE

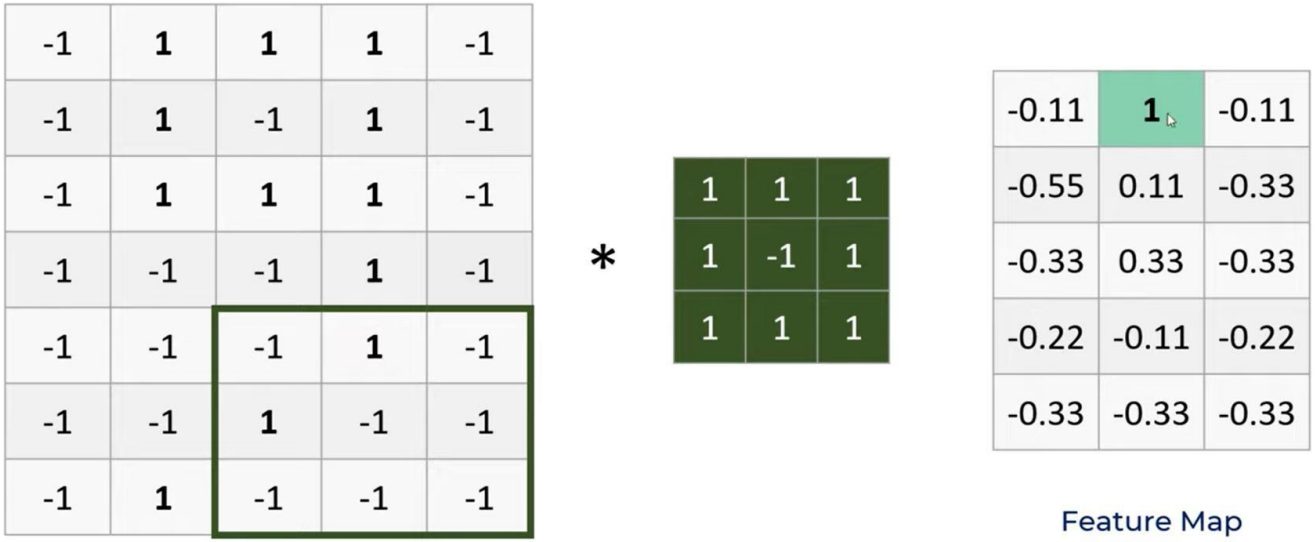


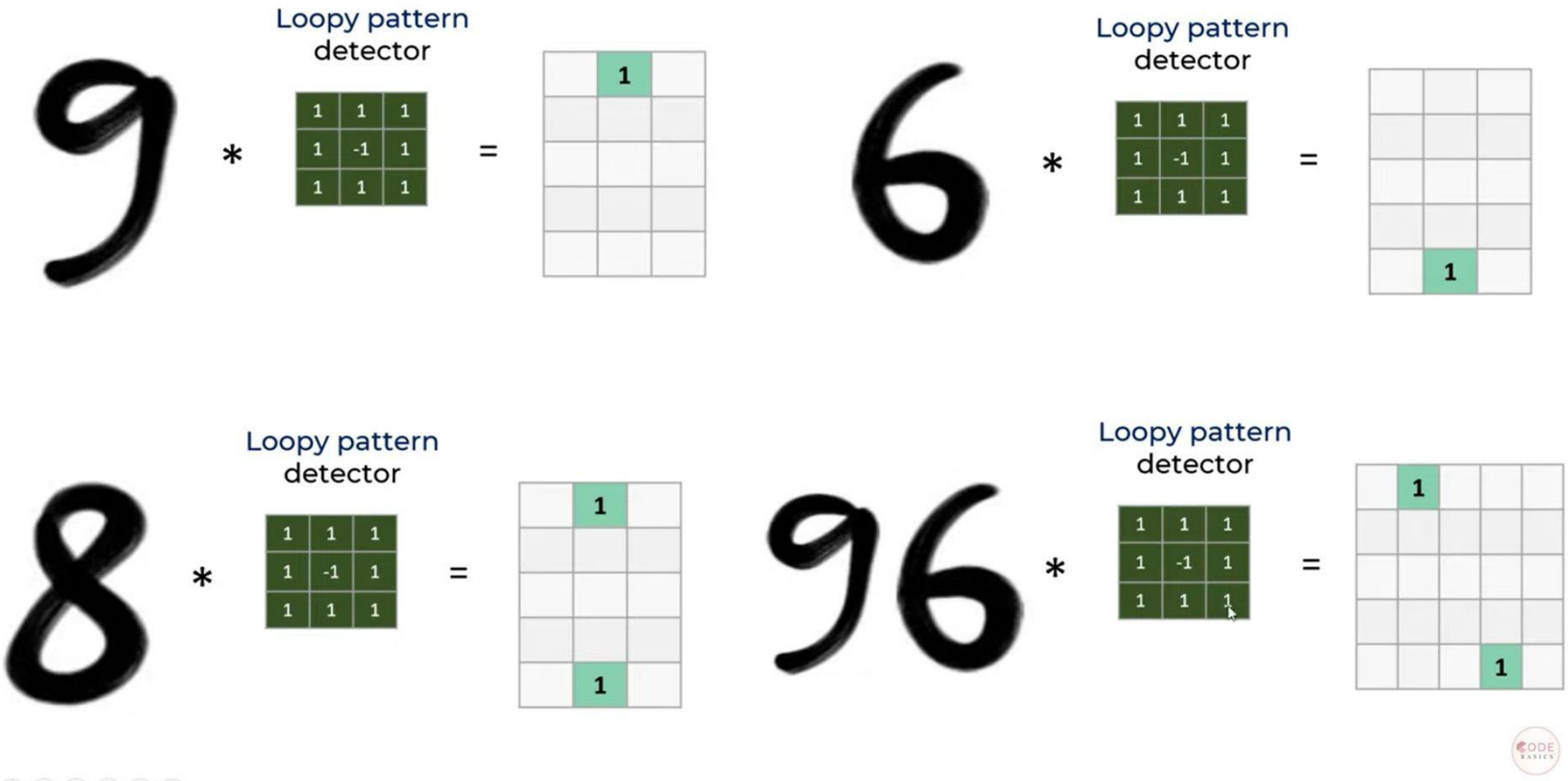

Koala's eye? = Y
Koala's nose? = Y
Koala's ears? = Y
Koala's head? = Y
Is it Koala? = Y
Koala's hands? = Y
Koala's legs? = Y
Koala's body? = Y

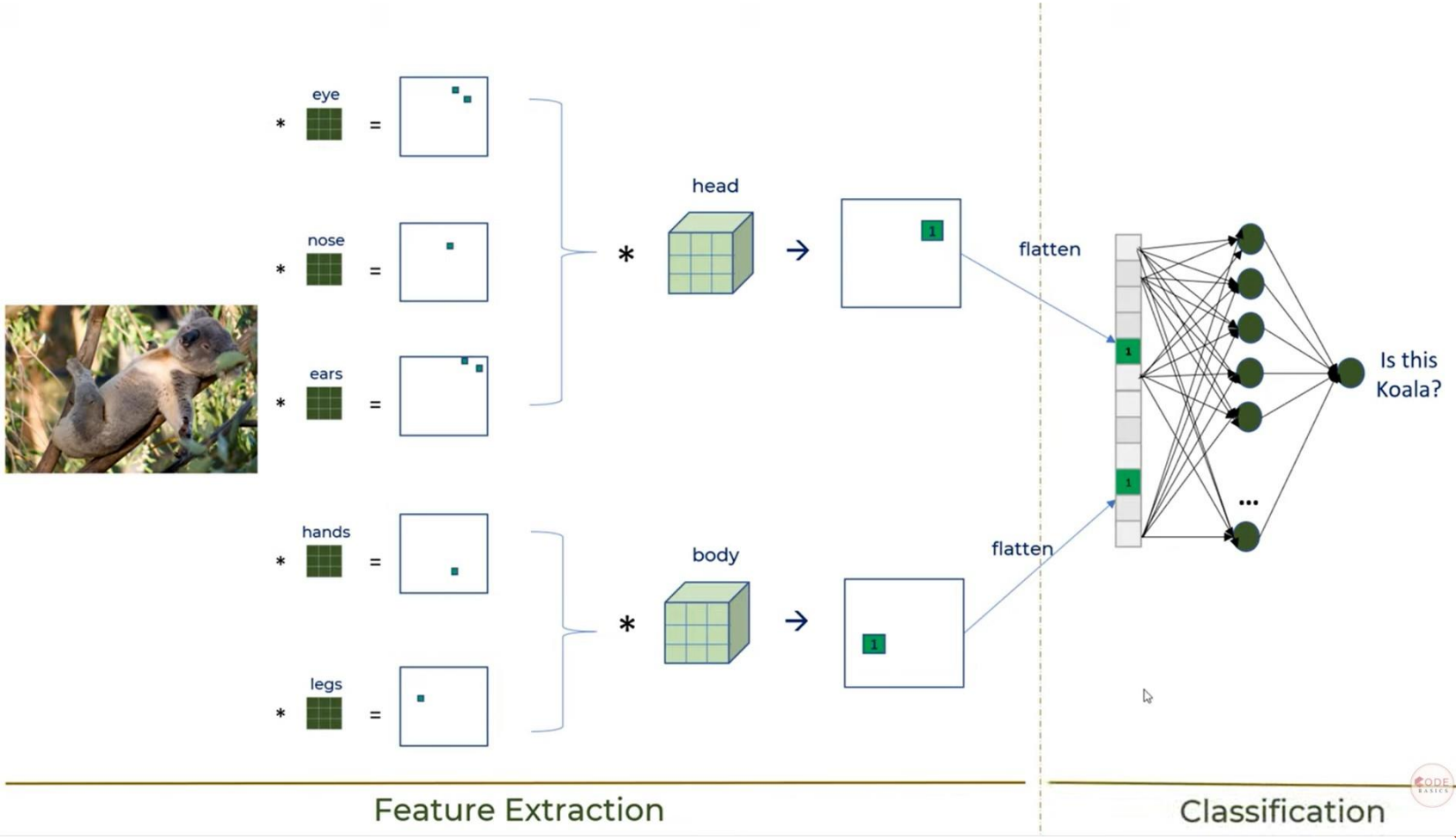
EXAMPLE




Loopy pattern filter
Vertical line filter
Diagonal line filter


-1 1 1 1 -1
-1 1 -1 1 -1
-1 1 1 1 -1
-1 -1 -1 1 -1
-1 -1 -1 1 -1
-1 -1 1 -1 -1
-1 1 -1 -1 -1
*
1 1 1
1 -1 1
1 1 1
-0.11 1 -0.11
-0.55 0.11 -0.33
-0.33 0.33 -0.33
-0.22 -0.11 -0.22
-0.33 -0.33 -0.33
Feature Map


Loopy pattern detector
1 1 1
1 -1 1
1 1 1
Loopy pattern detector
1 1 1
1 -1 1
1 1 1
Loopy pattern detector
1 1 1
1 -1 1
1 1 1
Loopy pattern detector
1 1 1
1 -1 1
1 1 1


eye
nose
ears
hands
legs
head
body
flatten
flatten
Is this Koala?
Feature Extraction
Classification



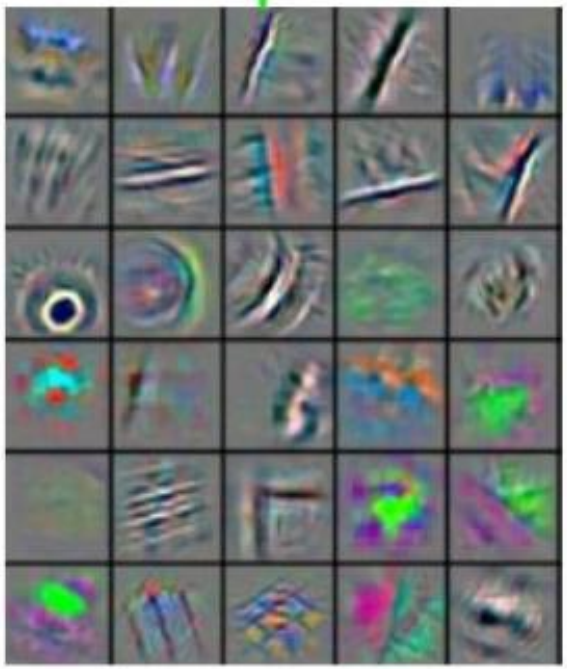
Low-Level Feature

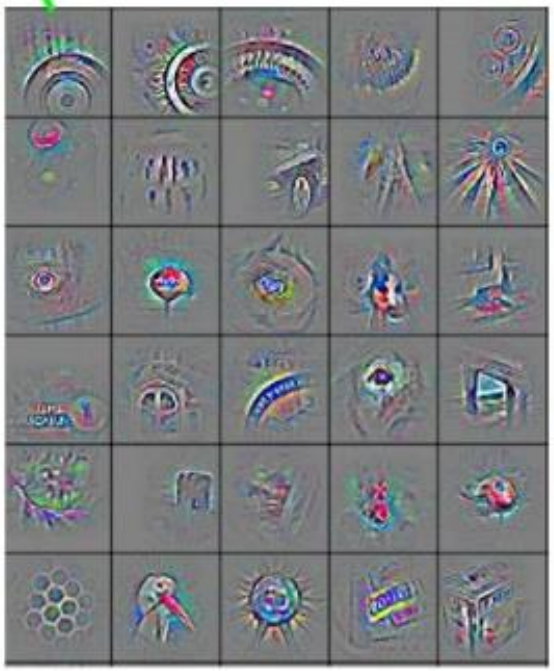
Mid-Level Feature

High-Level Feature

Trainable Classifier

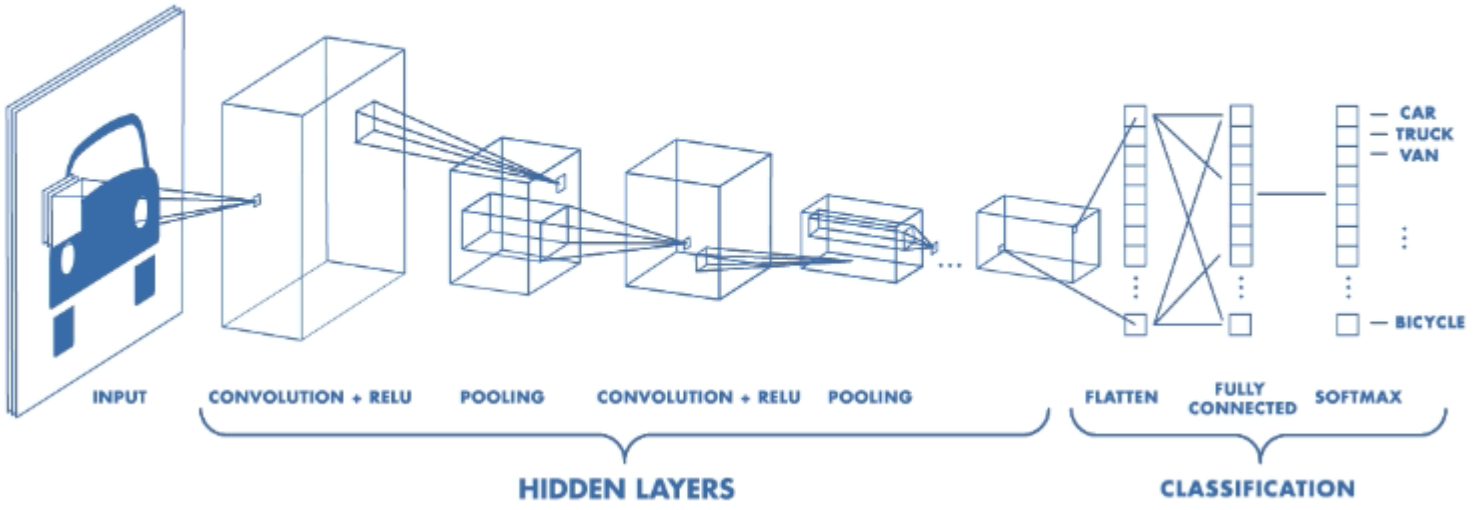






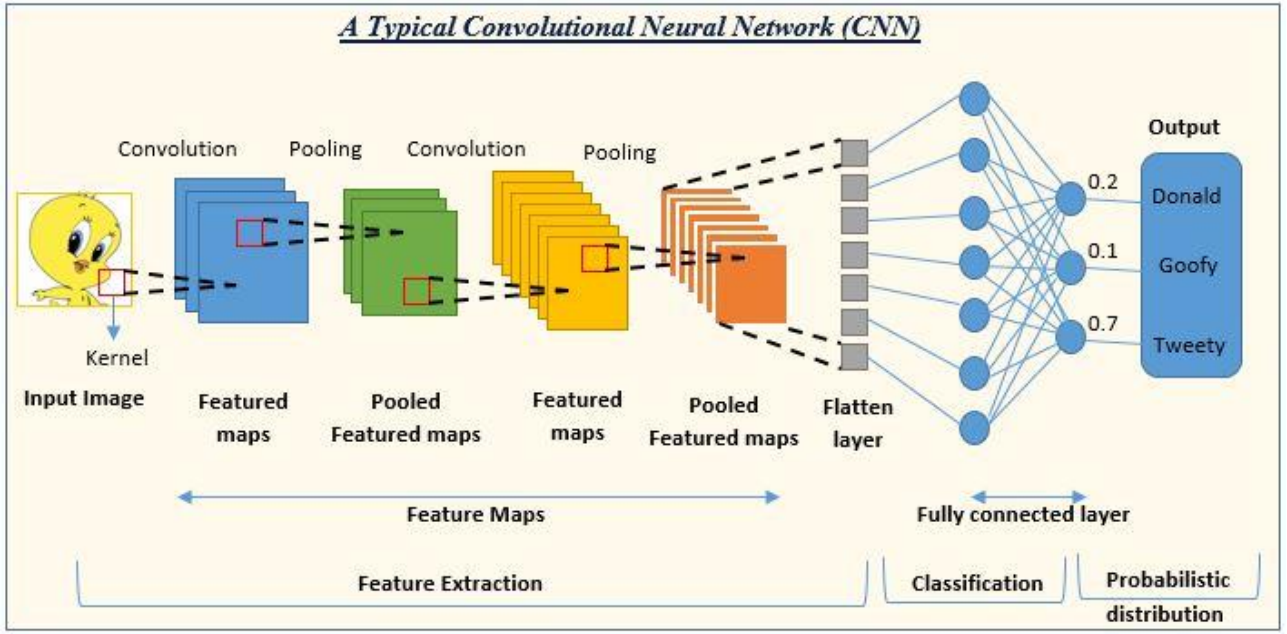
CONVOLUTIONAL NEURAL NETWORK ELABORATION

- CNN is mainly used for image recognition and classification.
- There are four operations in CNN
  - Convolution
  - Non-Linearity (ReLU) activation function
  - Subsampling or Pooling
  - Classification (fully connected layer)



CONVOLUTIONAL NEURAL NETWORK ELABORATION

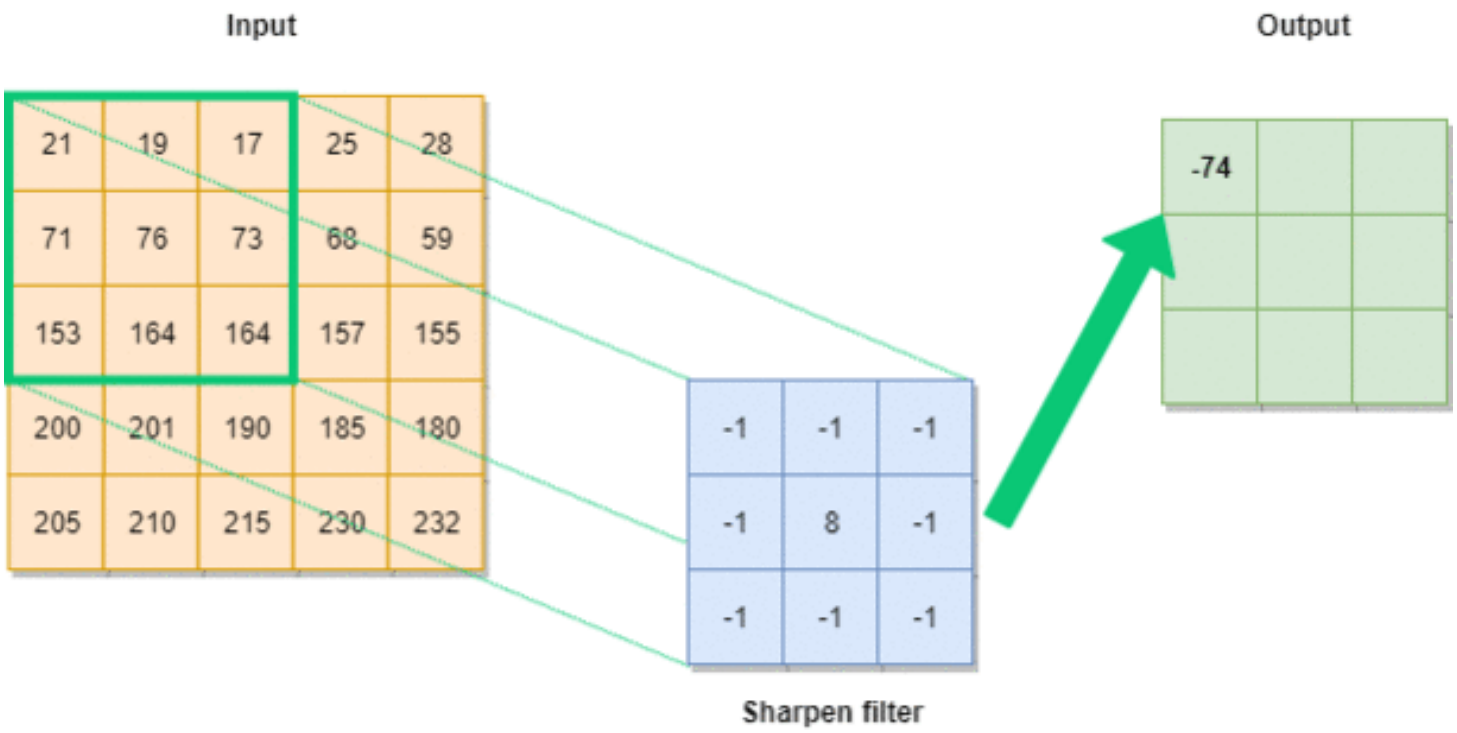
- CNN is mainly used for image recognition and classification.
- There are four operations in CNN
  - Convolution
  - Non-Linearity (ReLU) activation function
  - Subsampling or Pooling
  - Classification (fully connected layer)



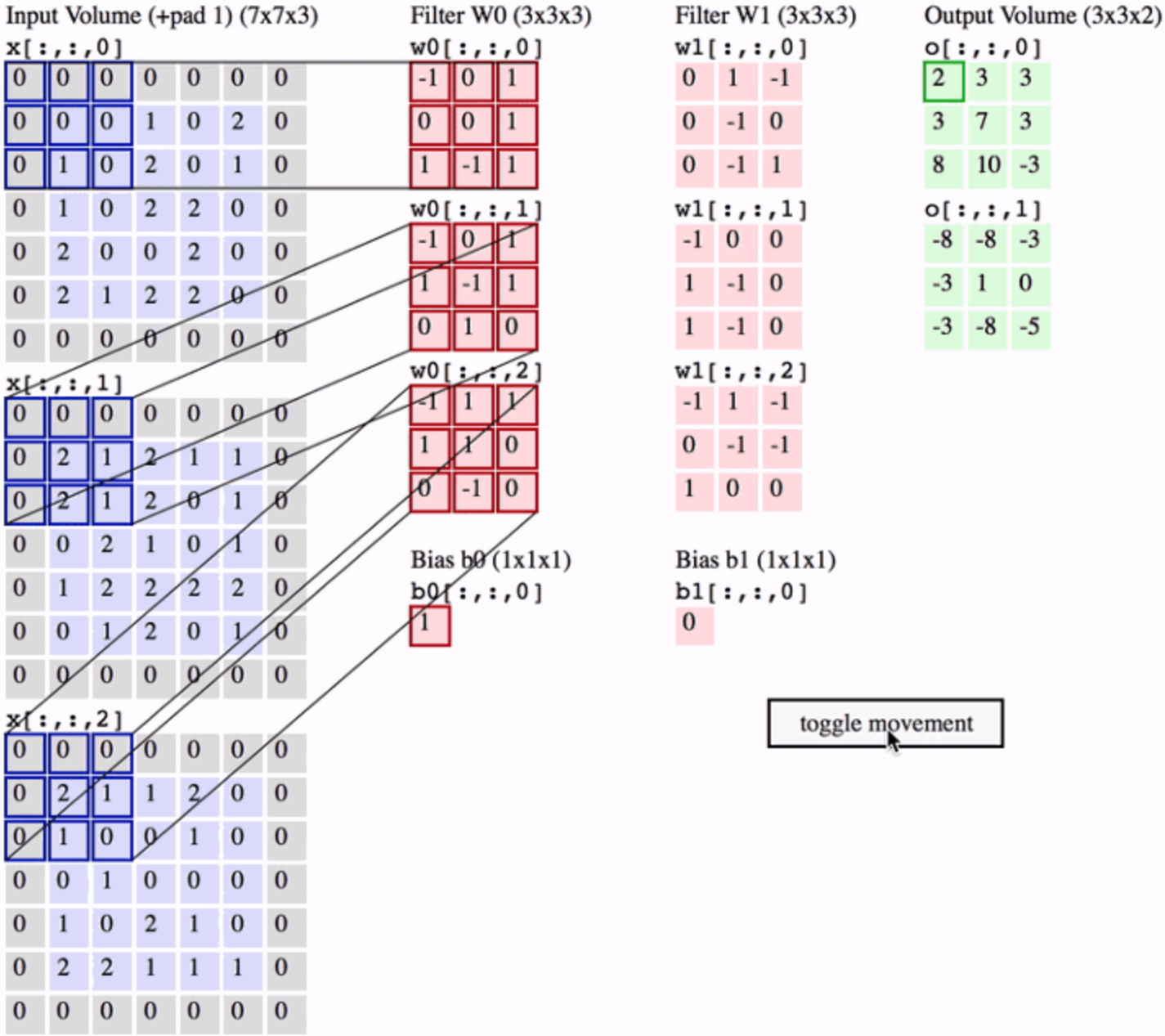
CONVOLUTIONAL NEURAL NETWORK ELABORATION

- CNN mainly used for image recognition and classification.
- There are four operations in CNN
  - Convolution
  - Non-Linearity (ReLU) activation function
  - Subsampling or Pooling
  - Classification (fully connected layer)

CONVOLUTION OPERATION



AIGeekProgrammer.com © 2019


Input Volume (+pad 1) (7x7x3)
x[:,:,0]
0 0 0 0 0 0 0
0 0 0 1 0 2 0
0 1 0 2 0 1 0
0 1 0 2 2 0 0
0 2 0 0 2 0 0
0 2 1 2 2 0 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 2 1 2 1 1 0
0 2 1 2 0 1 0
0 0 2 1 0 1 0
0 1 2 2 2 2 0
0 0 1 2 0 1 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 2 1 1 2 0 0
0 1 0 0 1 0 0
0 0 1 0 0 0 0
0 1 0 2 1 0 0
0 2 2 1 1 1 0
0 0 0 0 0 0 0
Filter W0 (3x3x3)
w0[:,:,0]
-1 0 1
0 0 1
1 -1 1
w0[:,:,1]
-1 0 1
1 -1 1
0 1 0
w0[:,:,2]
-1 1 1
1 1 0
0 -1 0
Bias b0 (1x1x1)
b0[:,:,0]
1
Filter W1 (3x3x3)
w1[:,:,0]
0 1 -1
0 -1 0
0 -1 1
w1[:,:,1]
-1 0 0
1 -1 0
1 -1 0
w1[:,:,2]
-1 1 -1
0 -1 -1
1 0 0
Bias b1 (1x1x1)
b1[:,:,0]
0
Output Volume (3x3x2)
o[:,:,0]
2 3 3
3 7 3
8 10 -3
o[:,:,1]
-8 -8 -3
-3 1 0
-3 -8 -5
toggle movement

THREE-CHANNELS IMAGE → SUMMING

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

$308 + (-498) + 164 + 1 = -25$

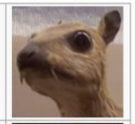
Bias = 1

Output

-25				...
				...
				...
				...
...	...	...	...	...

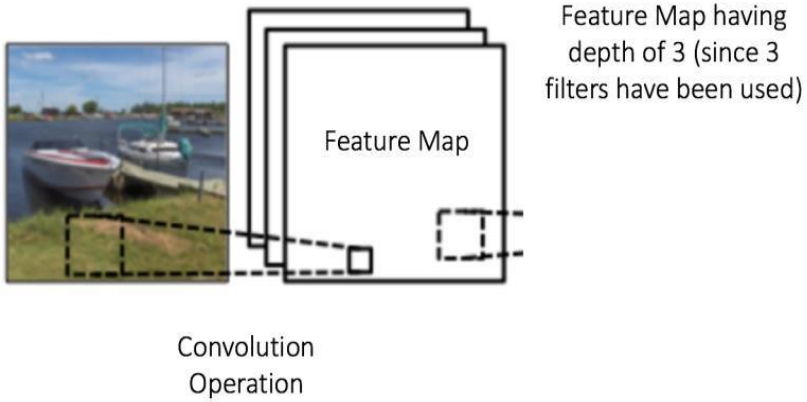
CONVOLUTIONAL NEURAL NETWORK ELABORATION

- Convolution
  - The 3x3 matrix is called a filter or kernel or feature detector which is applied by sliding (striding).
    - The matrix formed is called the feature map.
  - Many filters exits such as
    - Edge detection
    - Blur
    - Sharpen
  - In a CNN, each filter detect different feature of the input image.
  - In practice, a CNN *learns* the values of these filters on its own during the training process
    - (although we still need to specify parameters such as number of filters, filter size, architecture of the network etc. before the training process).
    - The more number of filters we have, the more image features get extracted and the better our network becomes at recognizing patterns in unseen images.

Operation	Filter	Convolved Image
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur (approximation)	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

CONVOLUTIONAL NEURAL NETWORK ELABORATION

- The size of the Feature Map (Convolved Feature) is controlled by three parameters that we need to specify before the convolution step is performed:
  - **Depth:** Depth corresponds to the number of filters we use for the convolution operation in the network.



In the above figure, we are performing convolution of the original boat image using three distinct filters, thus producing three different feature maps as shown.

- You can think of these three feature maps as stacked 2d matrices, so, the 'depth' of the feature map would be three.

CONVOLUTIONAL NEURAL NETWORK ELABORATION

- **Stride** is the number of pixels by which we slide our filter matrix over the input matrix.
  - When the stride is 1 then we move the filters one pixel at a time.
  - When the stride is 2, then the filters jump 2 pixels at a time as we slide them around.
    - Having a larger stride will produce smaller feature maps.
- **Zero-padding:** Sometimes, it is convenient to pad the input matrix with zeros around the border, so that we can apply the filter to bordering elements of our input image matrix.

CONVOLUTIONAL NEURAL NETWORK ELABORATION

Padding
(of 1)

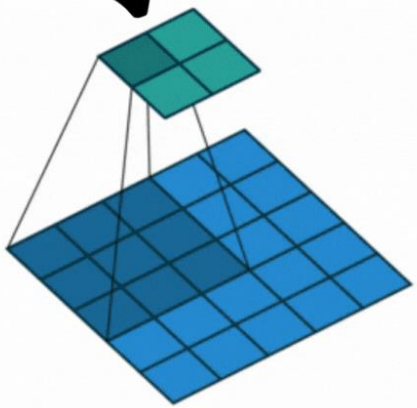
0	0	0	0	0	0	0
0	60	113	56	139	85	0
0	73	121	54	84	128	0
0	131	99	70	129	127	0
0	80	57	115	69	134	0
0	104	126	123	95	130	0
0	0	0	0	0	0	0

Kernel

0	-1	0
-1	5	-1
0	-1	0

114				

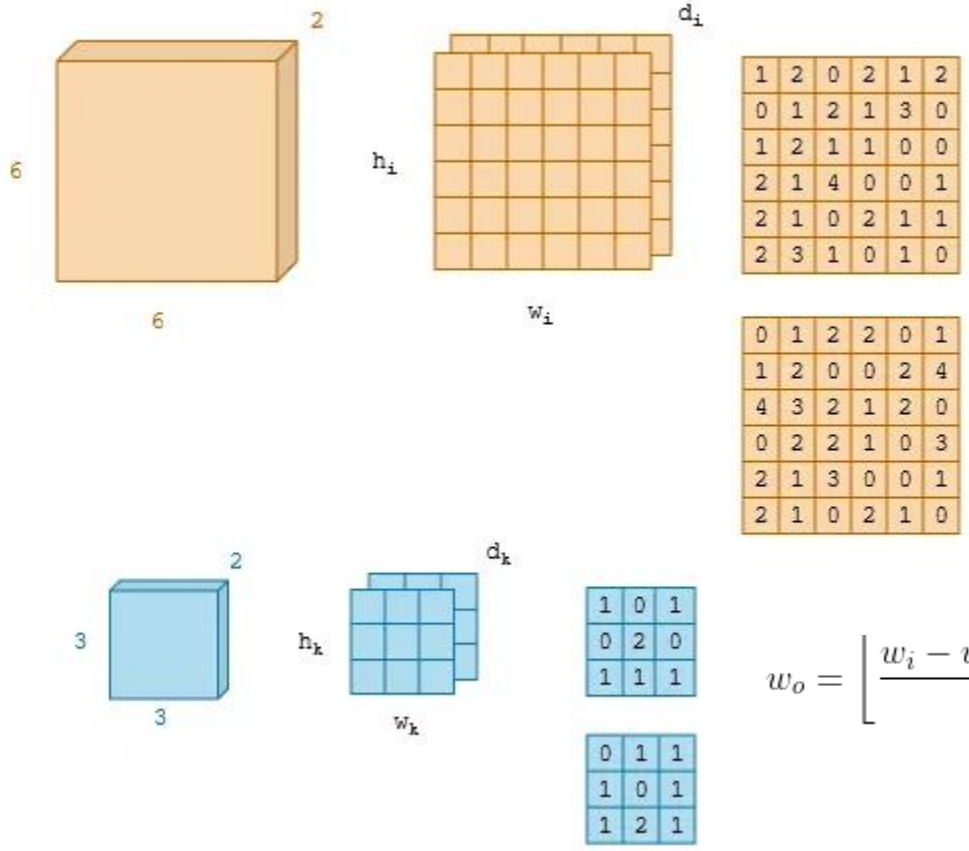
Stride
(of 2)



0_2	0_0	0_1	0	0	0	0
0_1	2_0	2_0	3	3	3	0
0_0	0_1	1_1	3	0	3	0
0	2	3	0	1	3	0
0	3	3	2	1	2	0
0	3	3	0	2	3	0
0	0	0	0	0	0	0

1	6	5
7	10	9
7	10	8

SIZE OF OUTPUT IN TERMS OF INPUTS



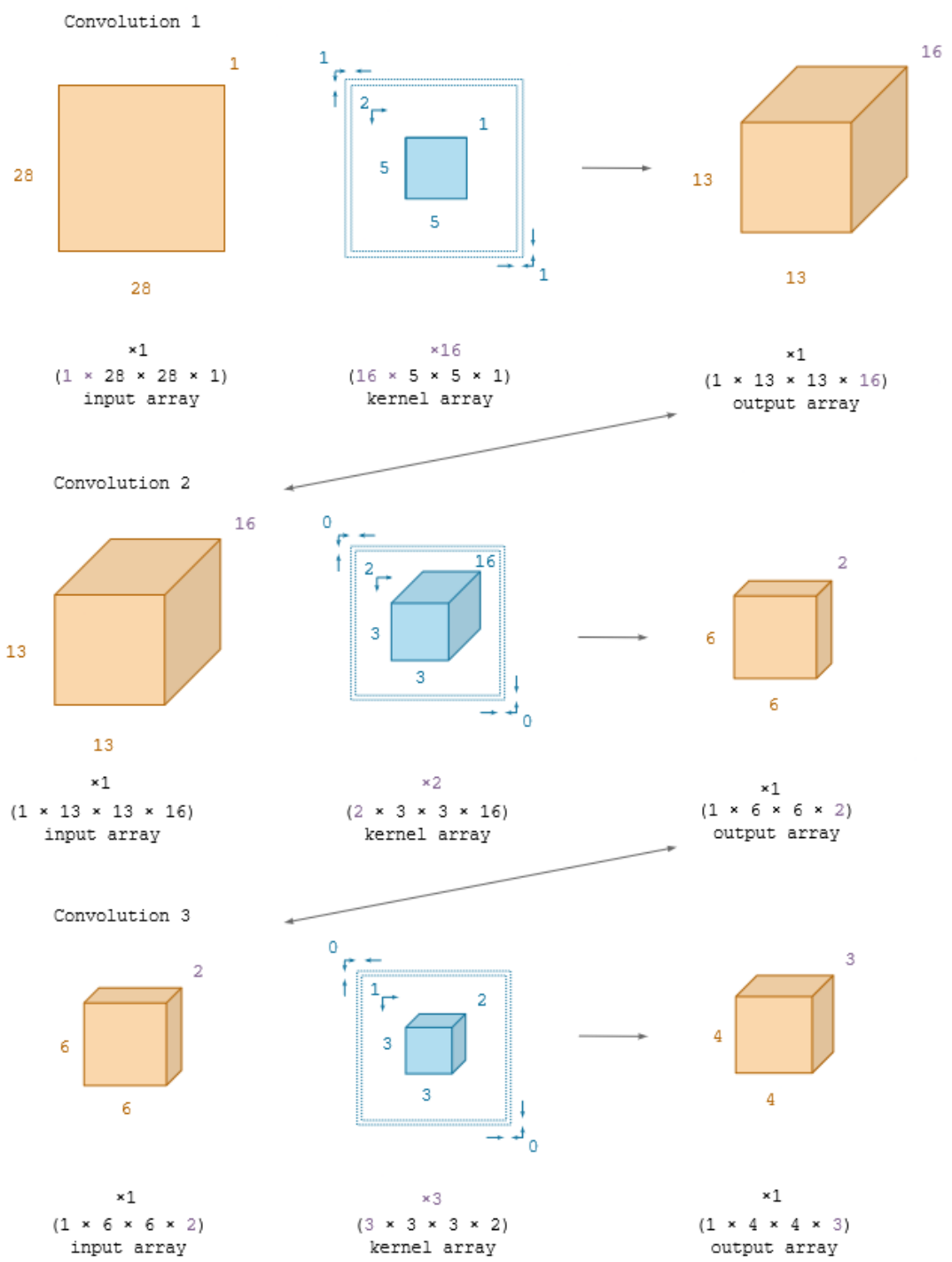
$$w_o = \left\lfloor \frac{w_i - w_k + 2 \cdot p_w}{s_w} \right\rfloor + 1 \; ; \; h_o = \left\lfloor \frac{h_i - h_k + 2 \cdot p_h}{s_h} \right\rfloor + 1$$

$$o = \left\lfloor \frac{6 - 3 + 2 \cdot 0}{2} \right\rfloor + 1 = \left\lfloor \frac{3}{2} \right\rfloor + 1 = 2$$

EXAMPLE

$$w_o = \left\lfloor \frac{w_i - w_k + 2 \cdot p_w}{s_w} \right\rfloor + 1 \; ;$$

$$h_o = \left\lfloor \frac{h_i - h_k + 2 \cdot p_h}{s_h} \right\rfloor + 1$$

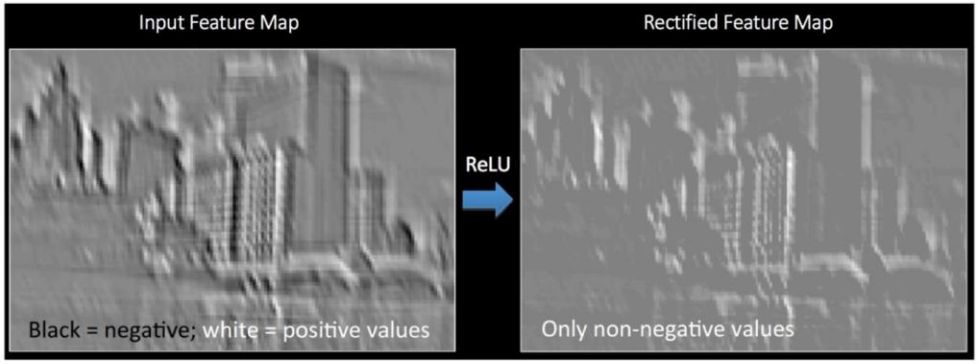


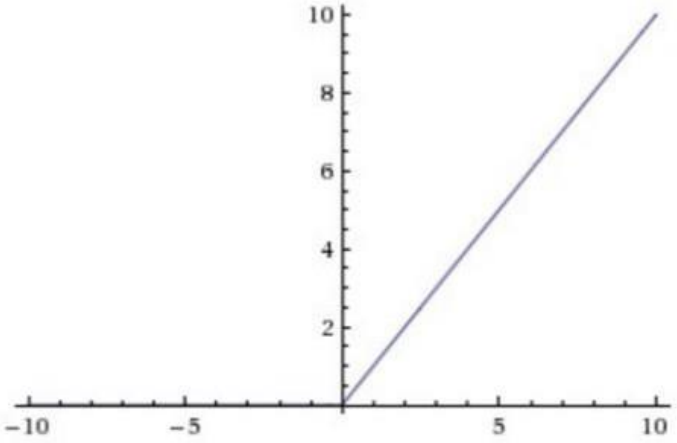
CONVOLUTIONAL NEURAL NETWORK ELABORATION

- CNN mainly used for image recognition and classification.
- There are four operations in CNN
  - Convolution
  - Non-Linearity (ReLU) activation function
  - Subsampling or Pooling
  - Classification (fully connected layer)

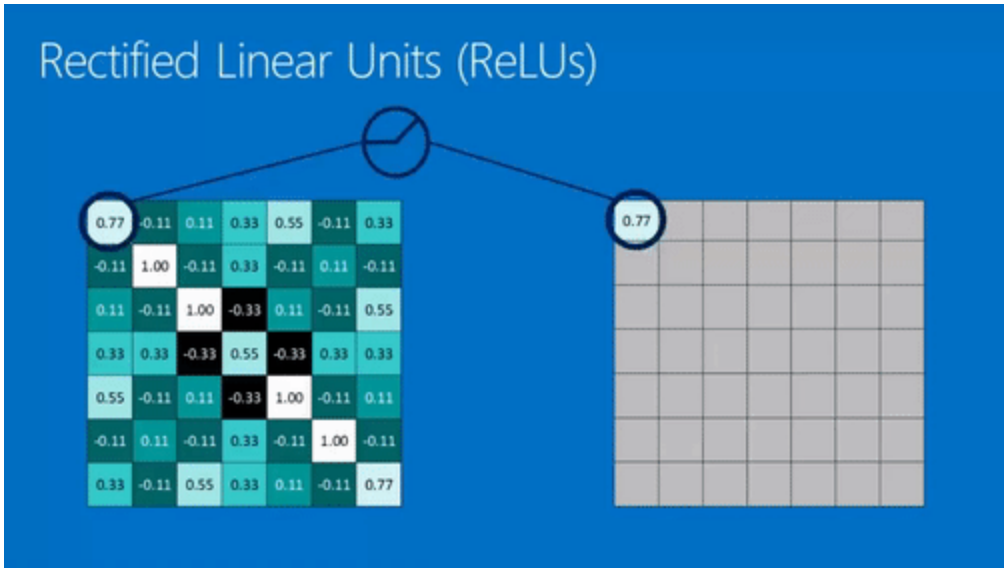
CONVOLUTIONAL NEURAL NETWORK ELABORATION

- Non-Linearity (ReLU)
  - An additional operation called ReLU has been used after every Convolution operation
  - ReLU is an element wise operation (applied per pixel) and replaces all negative pixel values in the feature map by zero. The purpose of ReLU is to introduce non-linearity
  - There are non-linear functions such as **tanh** or **sigmoid** can also be used instead of ReLU, but ReLU has been found to perform better in most situations.





RELU

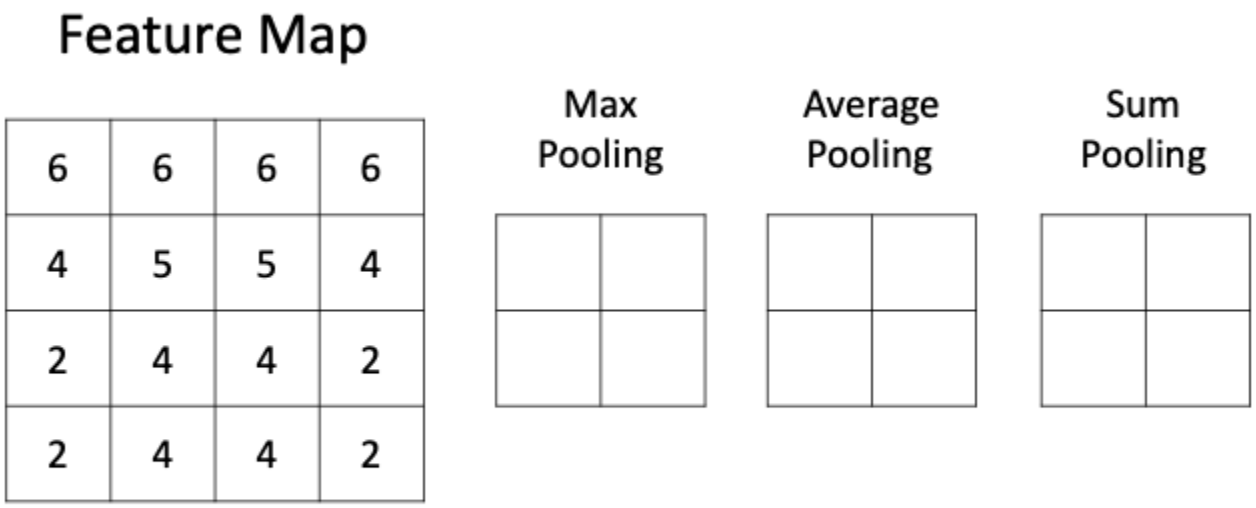

Rectified Linear Units (ReLUs)
0.77 -0.11 0.11 0.33 0.55 -0.11 0.33
-0.11 1.00 -0.11 0.33 -0.11 0.11 -0.11
0.11 -0.11 1.00 -0.33 0.11 -0.11 0.55
0.33 0.33 -0.33 0.55 -0.33 0.33 0.33
0.55 -0.11 0.11 -0.33 1.00 -0.11 0.11
-0.11 0.11 -0.11 0.33 -0.11 1.00 -0.11
0.33 -0.11 0.55 0.33 0.11 -0.11 0.77
0.77

CONVOLUTIONAL NEURAL NETWORK ELABORATION

- CNN mainly used for image recognition and classification.
- There are four operations in CNN
  - Convolution
  - Non-Linearity (ReLU) activation function
  - Subsampling or Pooling
  - Classification (fully connected layer)

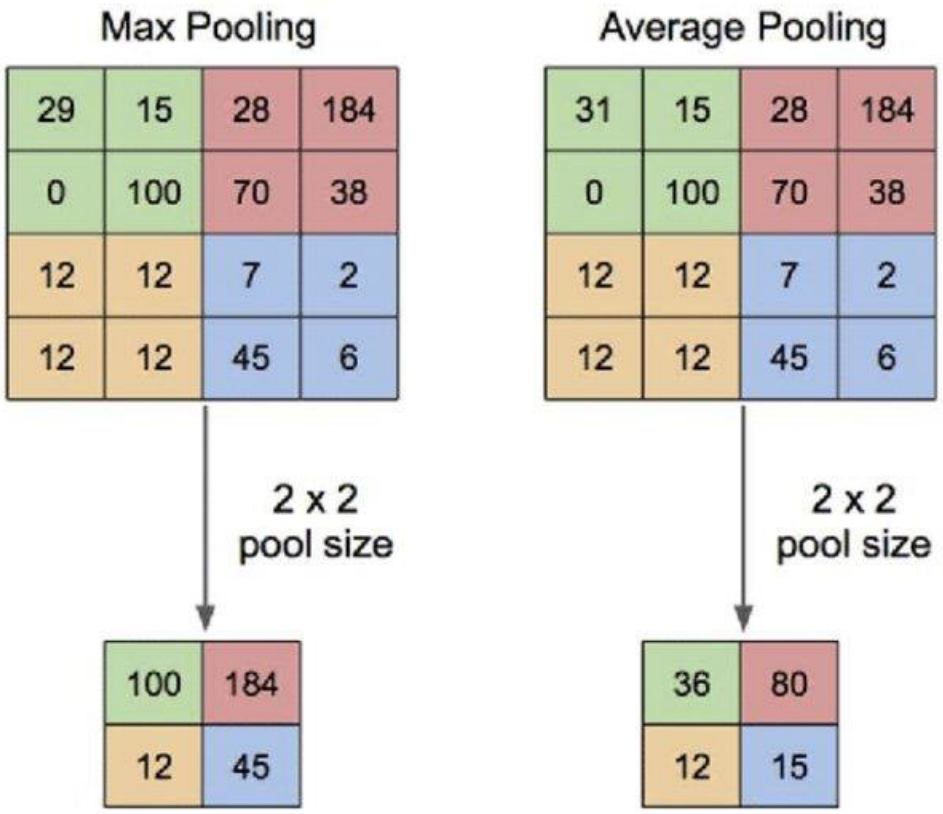
CONVOLUTIONAL NEURAL NETWORK ELABORATION

- Subsampling
  - Spatial Pooling (also called subsampling or down-sampling) reduces the dimensionality of each feature map but retains the most important information.
  - Spatial Pooling can be of different types: Max, Average, Sum etc.



CONVOLUTIONAL NEURAL NETWORK ELABORATION

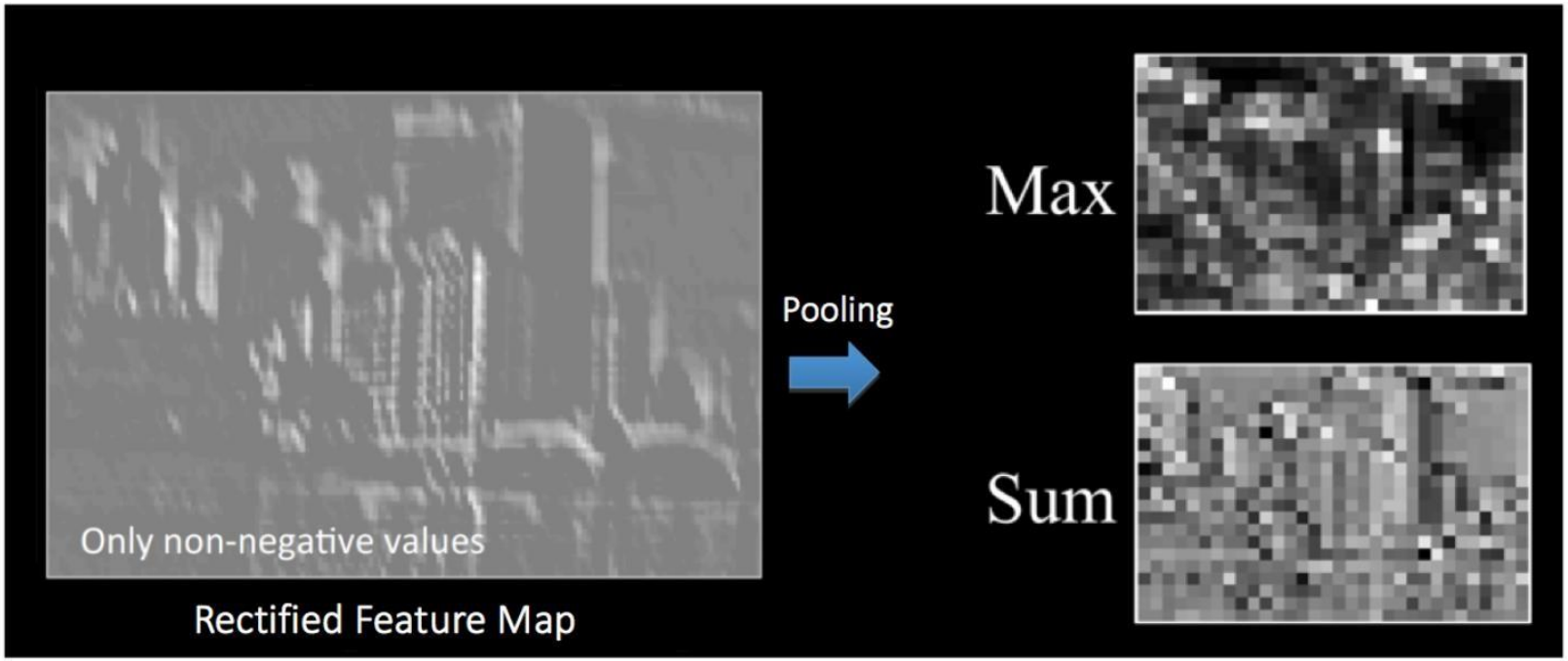
- The below figure shows an example of Max Pool and Average Pool on a rectified image (obtained after convolution + ReLU operation) by using a 2×2 window and stride = 2.



We slide our 2 x 2 window by 2 cells (also called 'stride') and take the maximum/average value in each region. As shown in **the figure** , this reduces the dimensionality of our feature map.

CONVOLUTIONAL NEURAL NETWORK ELABORATION

- The below figure shows the effect of Subsampling on the Rectified Feature Map we received after the ReLU operation



CONVOLUTIONAL NEURAL NETWORK ELABORATION

- The function of Subsampling is to progressively reduce the spatial size of the input representation. In particular, subsampling
  - makes the input representations (feature dimension) smaller and more manageable and reduces the number of parameters and computations in the network, therefore, controlling overfitting.
  - makes the network invariant to small transformations, distortions and translations in the input image (a small distortion in input will not change the output of Pooling – since we take the maximum / average value in a local neighbourhood).
  - helps us arrive at an almost scale invariant representation of our image (the exact term is “equivariant”). This is very powerful since we can detect objects in an image no matter where they are located.

Convolution

- Connections sparsity reduces overfitting
- Conv + Pooling gives location invariant feature detection
- Parameter sharing

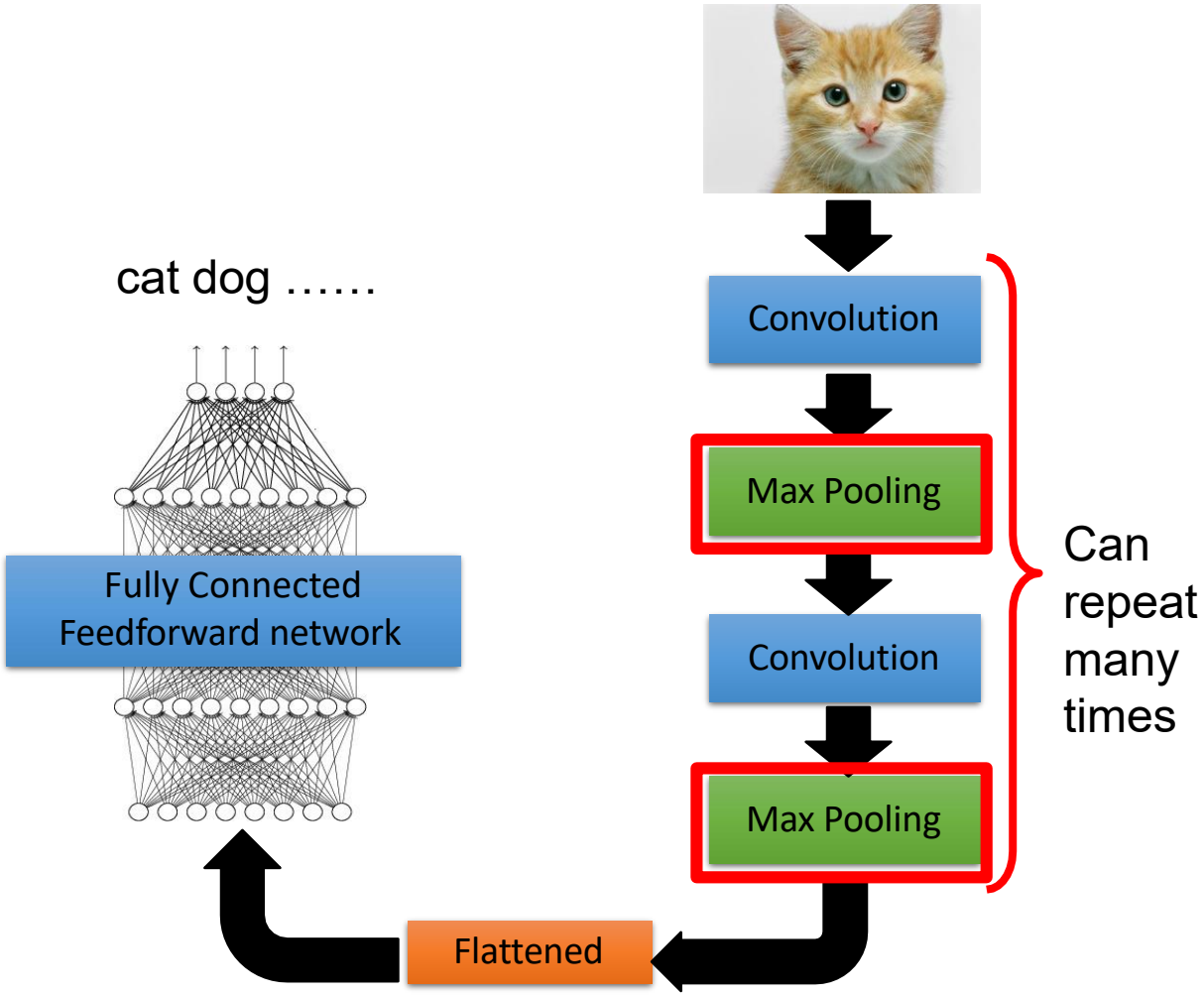
ReLU

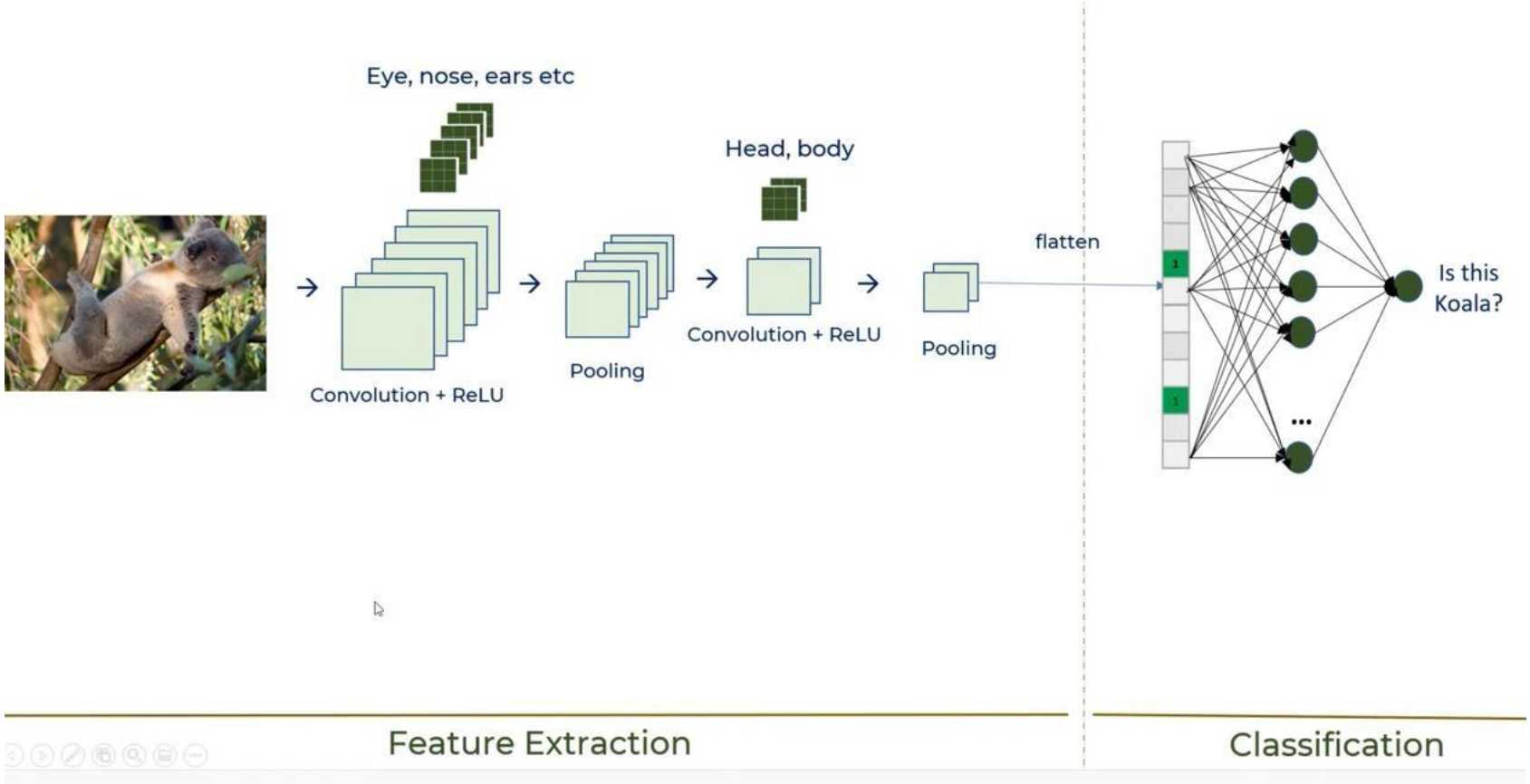
- Introduces nonlinearity
- Speeds up training, faster to compute

Pooling

- Reduces dimensions and computation
- Reduces overfitting
- Makes the model tolerant towards small distortion and variations

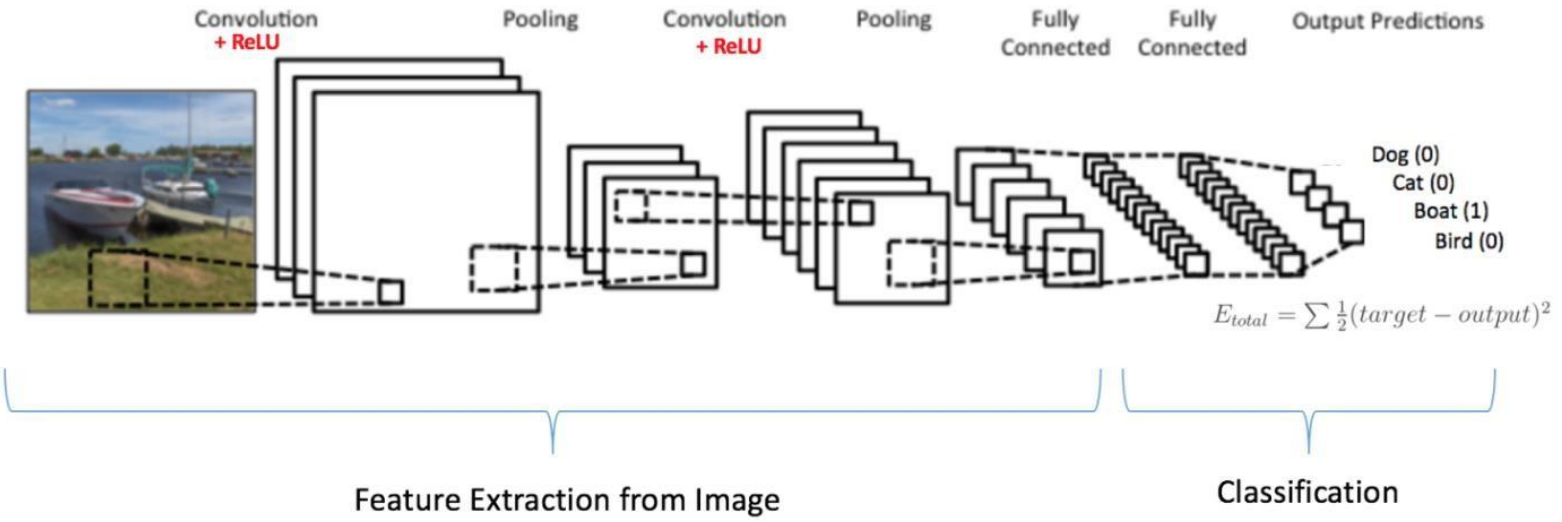
THE WHOLE CNN




Eye, nose, ears etc
Head, body
Convolution + ReLU
Pooling
Convolution + ReLU
Pooling
flatten
Is this Koala?
Feature Extraction
Classification

CONVOLUTIONAL NEURAL NETWORK ELABORATION

- Putting it all together

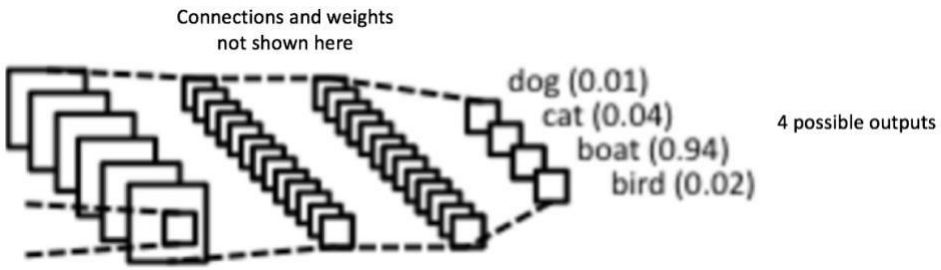


CONVOLUTIONAL NEURAL NETWORK ELABORATION

- CNN is mainly used for image recognition and classification.
- There are four operations in CNN
  - Convolution
  - Non-Linearity (ReLU) activation function
  - Subsampling or Pooling
  - Classification (fully connected layer)

CONVOLUTIONAL NEURAL NETWORK ELABORATION

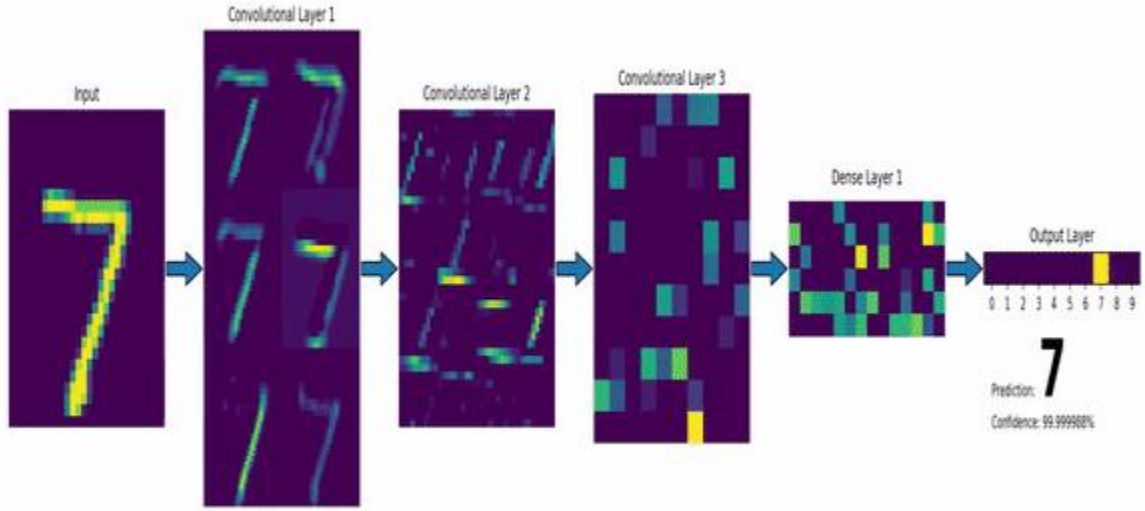
- The Fully Connected layer is a traditional MLP using SoftMax activation function
  - The SoftMax function takes a vector of arbitrary real-valued scores and squashes it to a vector of values between zero and one.
- The output from the convolutional and pooling layers represent high-level features of the input image.
- The purpose of the Fully Connected layer is to use these features for classifying the input image into various classes based on the training dataset.



CONVOLUTION NETWORKS

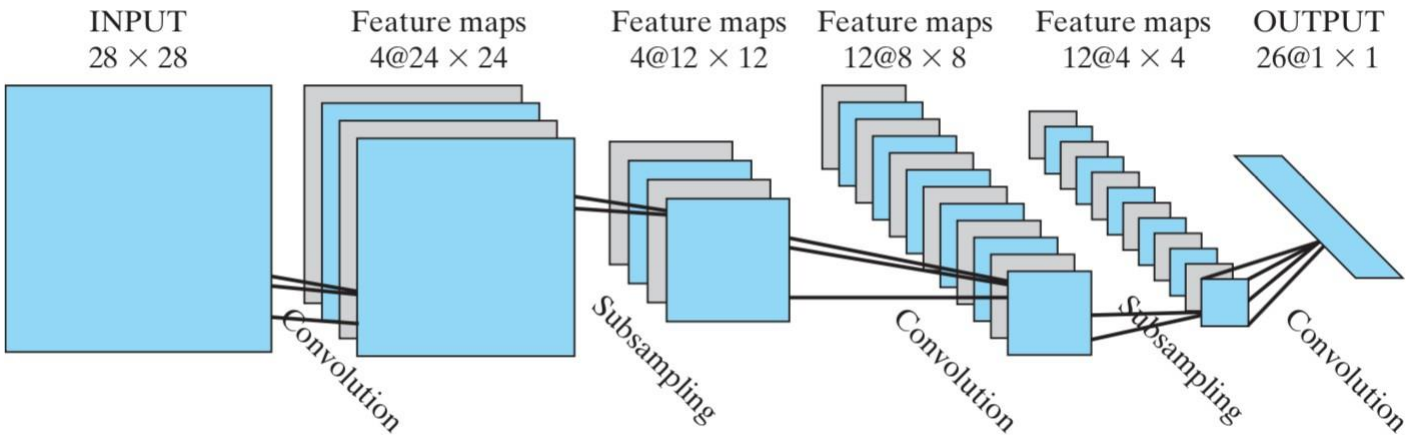
- **Feature extraction:**
  - Each neuron takes its synaptic inputs from a local *receptive field* in the previous layer, thereby forcing it to extract local features.
  - Once a feature has been extracted, its exact location becomes less important, as long as its position relative to other features is approximately preserved.

- **Feature mapping**
  - Each computational layer of the network is composed of multiple *feature maps*
    - *Shift invariance*, forced into the operation of a feature map through the use of *convolution* with a kernel of small size, followed by a nonlinear function (sigmoid, RELU, …);
    - *Reduction in the number of parameters*, accomplished through the use of *weight sharing.*


Input
Convolutional Layer 1
Convolutional Layer 2
Convolutional Layer 3
Dense Layer 1
Output Layer
0 1 2 3 4 5 6 7 8 9
Prediction:
7
Confidence: 99.999988%

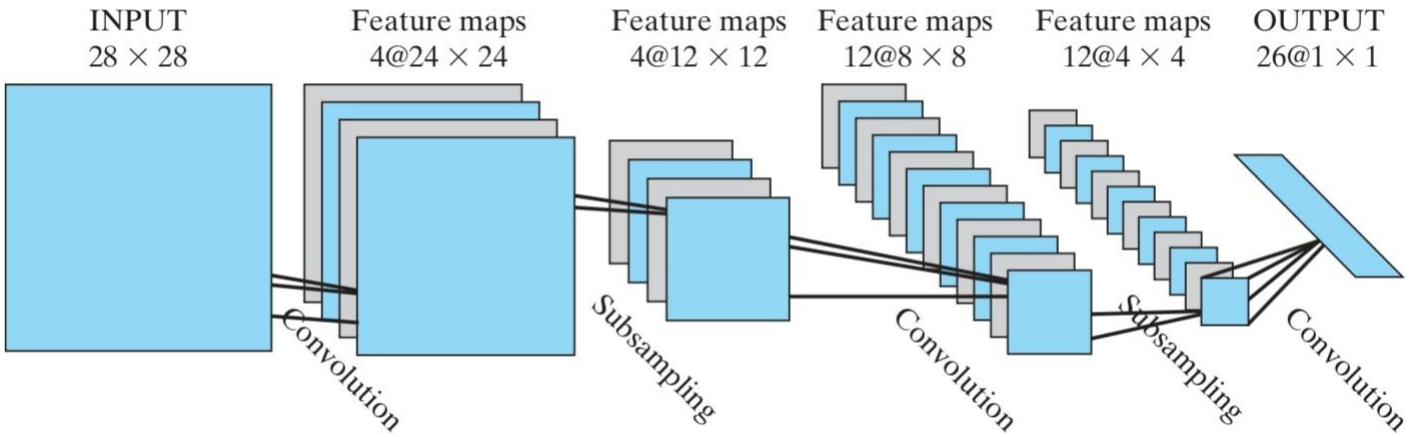
CNN EXAMPLE

- The below convolution network is made up of four hidden layers and an output layer.
- The input layer is 28 x 28 sensory nodes receiving images of different characters that have been approximately centred and normalized



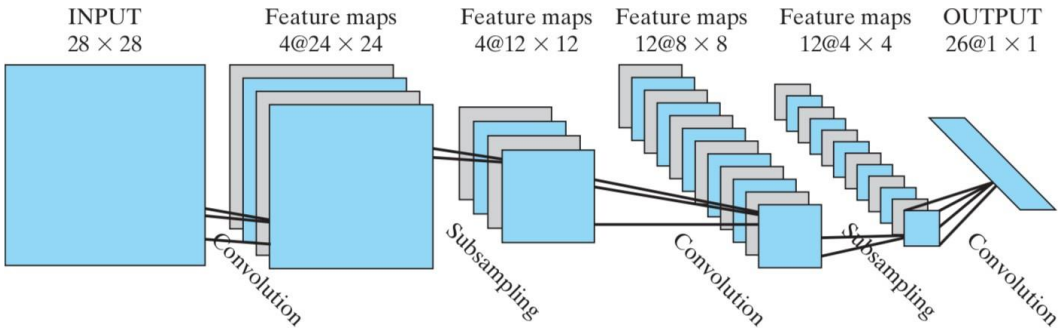
CNN EXAMPLE

- The first hidden layer performs convolution. It consists of four feature maps, with each feature map consisting of 24 x24 neurons.
  - Each neuron is assigned a receptive field of size 5 x 5.
- The second hidden layer performs subsampling and local averaging. It also consists of four feature maps, but each feature map is now made up of 12 x 12 neurons.



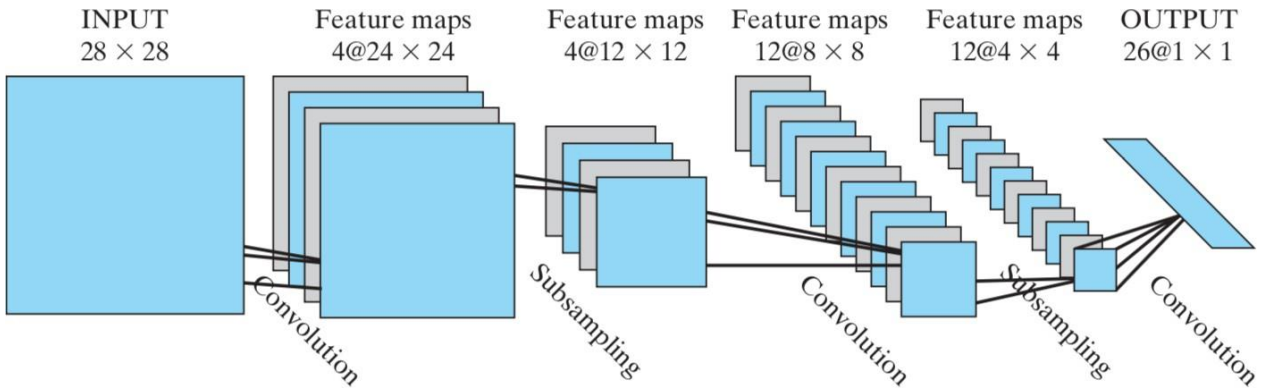
CNN EXAMPLE

- The third hidden layer performs a second convolution.
  - It consists of 12 feature maps, with each feature map consisting of 8 x 8 neurons.
  - Each neuron in this hidden layer may have synaptic connections from several feature maps in the previous hidden layer.
  - Otherwise, it operates in a manner similar to the first convolutional layer.
- The fourth hidden layer performs a second subsampling and local averaging.
  - It consists of 12 feature maps, but with each feature map consisting of 4 x 4 neurons.
  - Otherwise, it operates in a manner similar to the first subsampling layer.
- The output layer performs one final stage of convolution.
  - It consists of 26 neurons, with each neuron assigned to one of 26 possible characters.
  - As before, each neuron is assigned a receptive field of size 4 x 4.



CNN EXAMPLE

- The MLP shown below contains approximately 100,000 synaptic connection, but only 2600 synaptic weight
  - This is done by weight sharing
  - The capacity of the learning machine is thereby reduced, which in turn improves the machine's ability to generalize.
  - The adjustment of the parameters are done using stochastic mode of the BP algorithm.

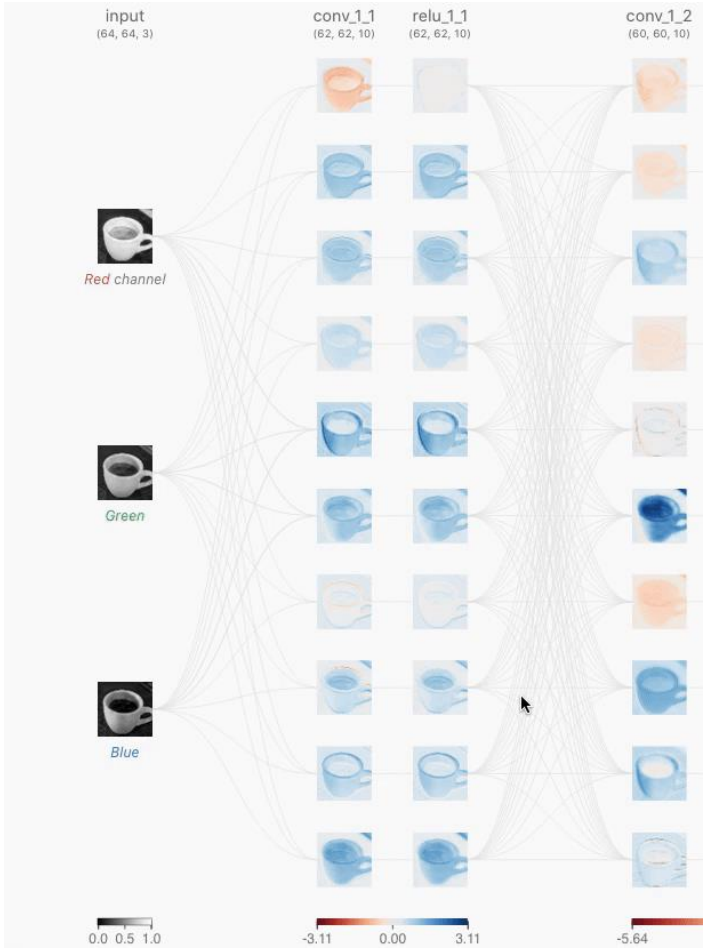


CONVOLUTIONAL NEURAL NETWORK TRAINING

- **Step1:** We initialize all filters and parameters / weights with random values
- **Step2:** The network takes a training image as input, goes through the forward propagation step (convolution, ReLU and subsampling operations along with forward propagation in the Fully Connected layer) and finds the output probabilities for each class.
  - Let's say the output probabilities are [0.2, 0.4, 0.1, 0.3]
  - Since weights are randomly assigned for the first training example, output probabilities are also random.
- **Step3:** Calculate the total error at the output layer (summation over all 4 classes)
  - **Total Error = $\sum$ ½ (target probability – output probability) ^{2}**

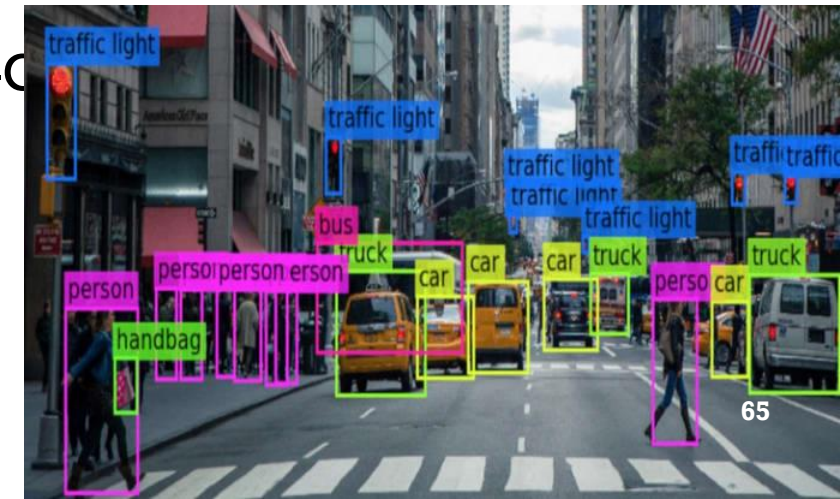
CONVOLUTIONAL NEURAL NETWORK TRAINING

- **Step4:** Use Backpropagation to calculate the *gradients* of the error with respect to all weights in the network and use *gradient descent* to update all filter values / weights and parameter values to minimize the output error.
  - The weights are adjusted in proportion to their contribution to the total error.
  - When the same image is input again, output probabilities might now be [0.1, 0.1, 0.7, 0.1], which is closer to the target vector [0, 0, 1, 0].
  - This means that the network has learnt to classify this particular image correctly by adjusting its weights / filters such that the output error is reduced.
  - Parameters like number of filters, filter sizes, architecture of the network etc. have all been fixed before Step 1 and do not change during training process – only the values of the filter matrix and connection weights get updated.
- **Step5:** Repeat steps 2-4 with all images in the training set.


input
(64, 64, 3)
conv_1_1
(62, 62, 10)
relu_1_1
(62, 62, 10)
conv_1_2
(60, 60, 10)
Red channel
Green
Blue
0.0 0.5 1.0
-3.11 0.00 3.11
-5.64

ARCHITECTURES VS APPLICATIONS

- **Image classification**
  - ✓AlexNet
  - ✓VGG (e.g., VGG16, VGG19)
  - ✓GoogLeNet (also known as Inception)
  - ✓ResNet (e.g., ResNet50, ResNet101)
  - ✓DenseNet
  - ✓MobileNet
- **Object detection**
  - ✓R-CNN (e.g., Fast R-CNN, Faster R-C
  - ✓YOLO (You Only Look Once)
  - ✓SSD (Single Shot Detector)
  - ✓RetinaNet

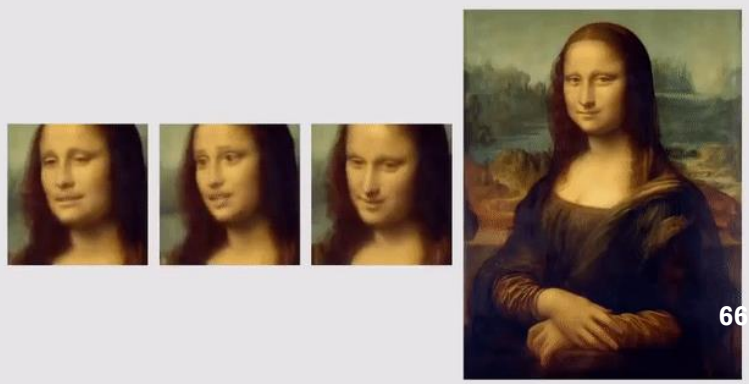


ARCHITECTURES VS APPLICATIONS

- **Semantic segmentation**
  - ✓FCN (Fully Convolutional Network)
  - ✓U-Net
  - ✓SegNet
  - ✓DeepLab



- **Generative modeling**
  - ✓GAN (Generative Adversarial Network)
  - ✓VAE (Variational Autoencoder)
  - ✓PixelCNN



SOME GAN-BASED APPLICATIONS

- ✓ **Image generation**: GANs can generate high-quality images of various types, such as human faces, landscapes, and animals. This has applications in art, design, and advertising.
- ✓ **Image-to-image translation**: GANs can be used to translate an image from one domain to another, such as converting a daytime image to a nighttime image, or converting a sketch to a photorealistic image.
- ✓ **Super-resolution**: GANs can be used to generate high-resolution images from low-resolution images, which has applications in medical imaging, satellite imaging, and surveillance.
- ✓ **Video synthesis**: GANs can generate realistic videos by combining generated frames with real frames, which has applications in entertainment and virtual reality.
- ✓ **3D object generation**: GANs can generate 3D objects from 2D images, which has applications in manufacturing, prototyping, and product design.
- ✓ **Text-to-image synthesis**: GANs can generate images from textual descriptions, which has applications in e-commerce, advertising, and visual storytelling.

SOME GAN-BASED APPLICATIONS

- ✓ **Style transfer**: GANs can be used to transfer the style of one image to another, which has applications in art and design.
- ✓ **Data augmentation**: GANs can be used to generate synthetic data to augment real data for training machine learning models, which can improve model accuracy.
- ✓ **Image inpainting**: GANs can fill in missing parts of an image based on the surrounding context, which has applications in photo editing and restoration.
- ✓ **Image manipulation**: GANs can be used to manipulate images in various ways, such as changing the pose or expression of a person in an image, or changing the background. This has applications in entertainment, advertising, and social media.
- ✓ **Image colorization**: GANs can be used to colorize grayscale images, which has applications in photography and art.
- ✓ **Face swapping**: GANs can be used to swap faces in images or videos, which has applications in entertainment and social media.
- ✓ ……