Mondragon Unibertsitatea
Faculty of Engineering

Advanced Machine Learning

Lecture 2: Supervised Deep Learning

Syllabus

Mondragon Unibertsitatea
Faculty of Engineering

- Deep Sequential Modeling
  - RNN
    - LSTM
    - GRU
  - Bidirectional RNNs

- Convolutional NN (CNN)
  - What are they for?
  - Step 1: Convolution
  - Step 1(b) - ReLU
  - Step 2 - Pooling
  - Step 3 - Flattening
  - Step 4 - Full Connection

Deep Sequence modelling

Why do we need sequence modelling?

I Can assure, this ball is moving:

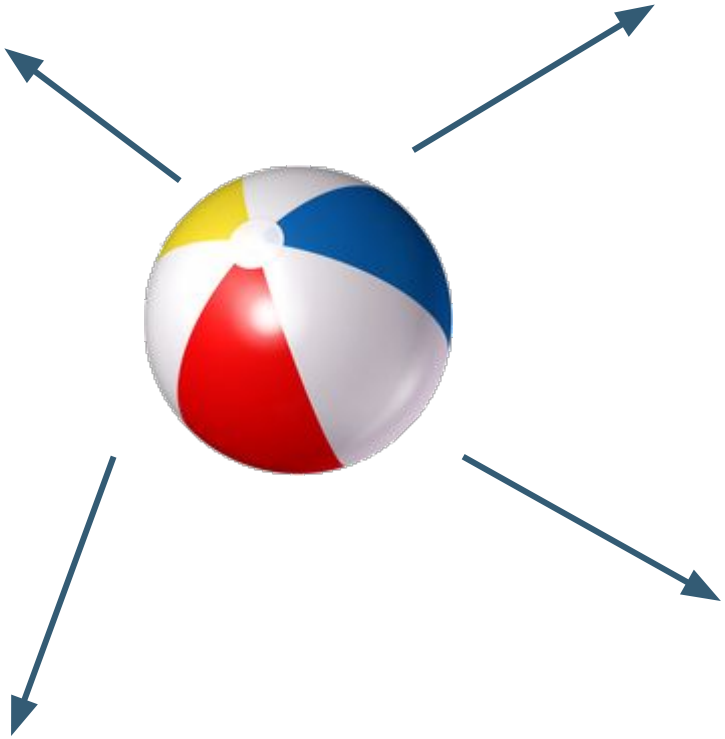
Can you predict the direction this ball is heading?

Why do we need sequence modelling?

Mondragon Unibertsitatea
Faculty of Engineering

Can you predict the direction this ball is heading?

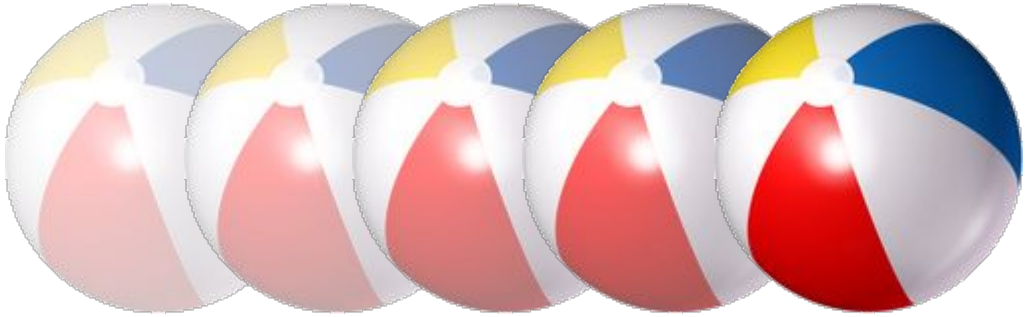
With no other information, it can be ANYWHERE



Why do we need sequence modelling?

Mondragon Unibertsitatea
Faculty of Engineering

Can you predict the direction this ball is heading, if I tell you that its last movements were these ones?

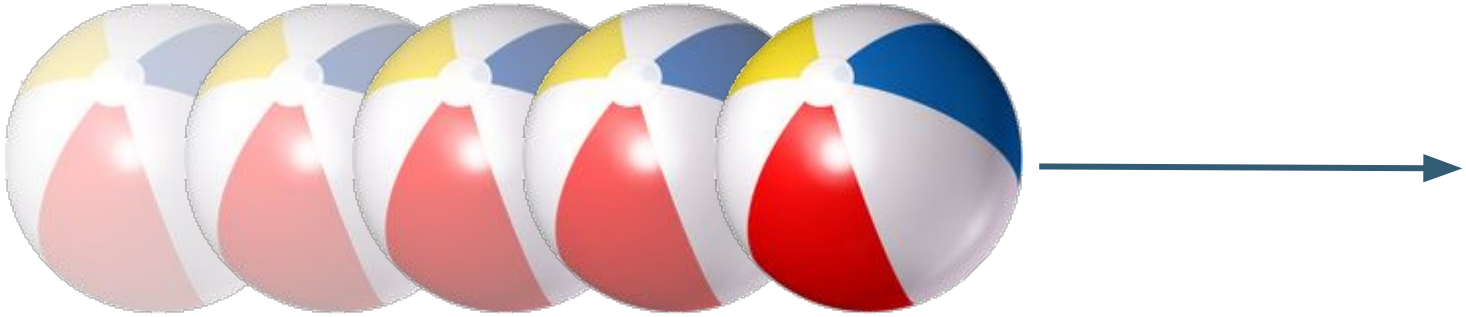


Why do we need sequence modelling?

Mondragon Unibertsitatea
Faculty of Engineering

Can you predict the direction this ball is heading, if I tell you that its last movements were these ones?

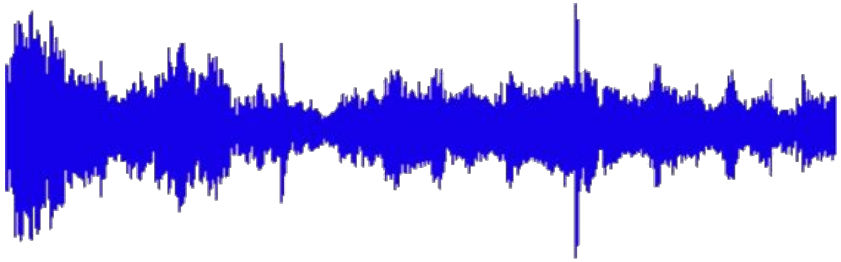
It is easy…. it is heading East



Why do we need sequence modelling?

Mondragon Unibertsitatea
Faculty of Engineering

- Basically, in order to predict, sometimes, **past knowledge is needed**.
- Examples of Sequence data (or TimeSeries):
  - Audio or sensor data, is a sequence of values
  - Text can be seen as sequences of characters or words



Audio, Sensor data, ...

character sequence:
H E L L O M Y N A M E I S E K H I

Word sequence:
Hello My Name Is Ekhi

Text data

Can we predict Sequences using FFNN?

Mondragon Unibertsitatea
Faculty of Engineering

"This morning I took my cat for a walk."

given these words — predict the next word

- Limitation:
  - Inputs in FFNN are fixed, so we would not be able to predict variable length sentences.

"This morning I took my cat for a walk."

given these two words — predict the next word

- OK, let us take only 2 words and try to predict the 3rd one
- Limitation:
  - We loose long time dependency:

I am **Basque**, right now I live in Guatemala, but I still speak fluent ____

- There are even more Limitations:
  - Order importance
  - No information Share across the sequence

The food was good, not bad at all.

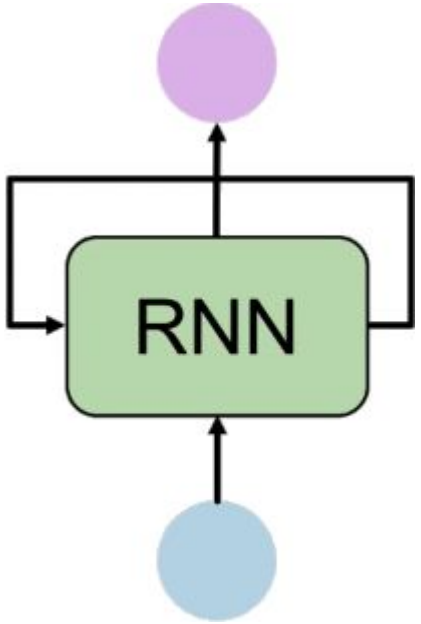
vs.

The food was bad, not good at all.

Sequence Modelling: Design Criteria

Mondragon Unibertsitatea
Faculty of Engineering

1. Handle Variable length Sequences
2. Track long-term dependencies
3. Maintain order information
4. Share parameters across the sequence



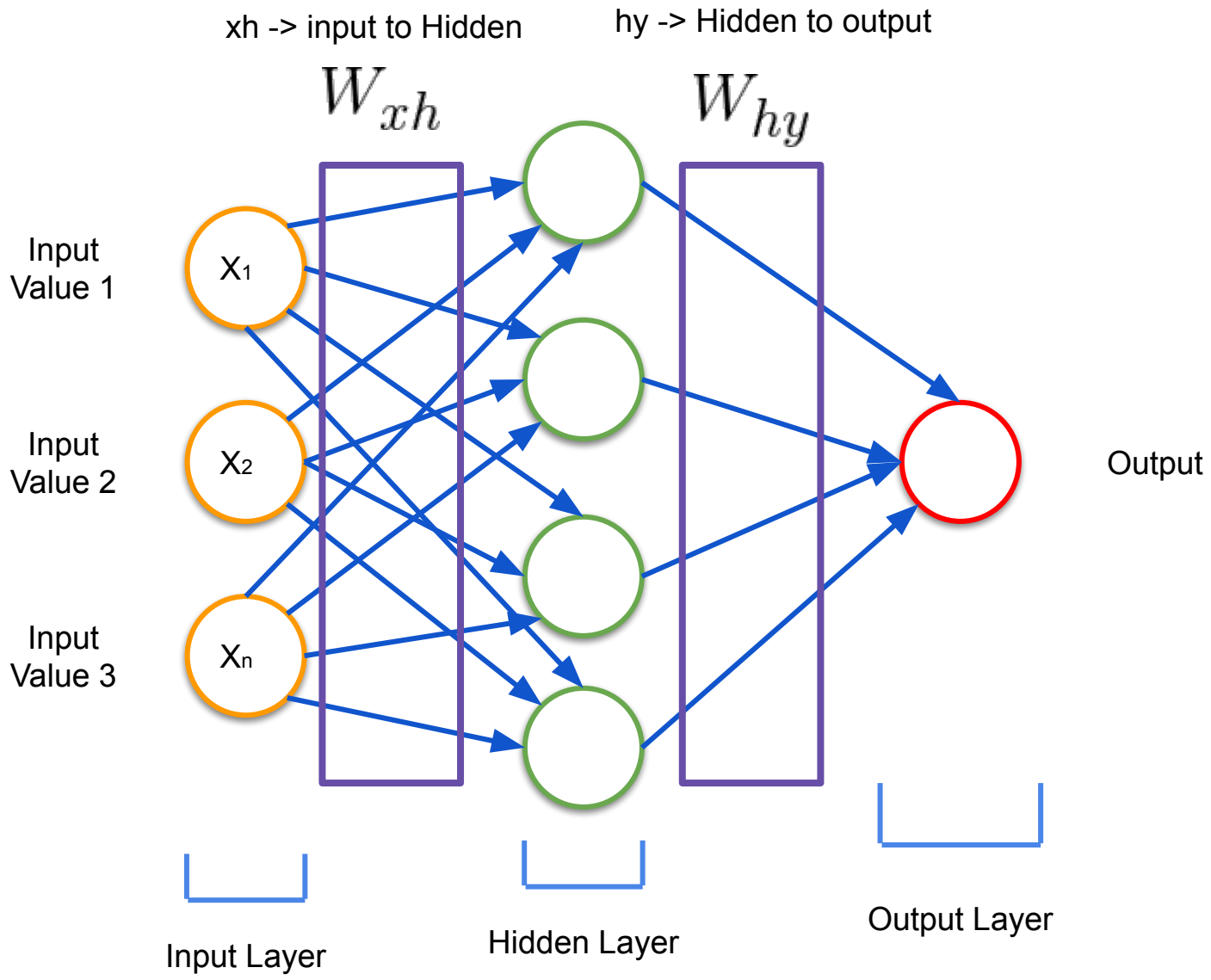
RNN in action - SunsPring

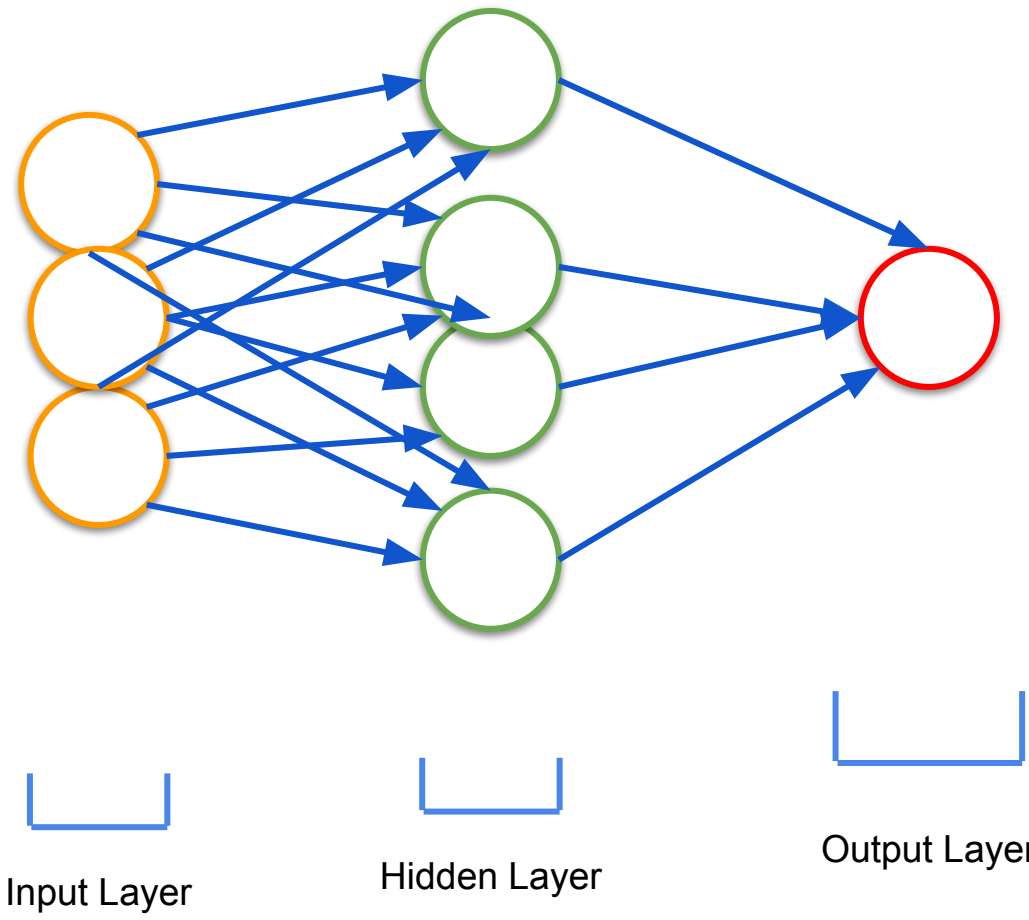
https://www.imdb.com/title/tt5794766/

Short movie written by an RNN (llamado Benjamin)

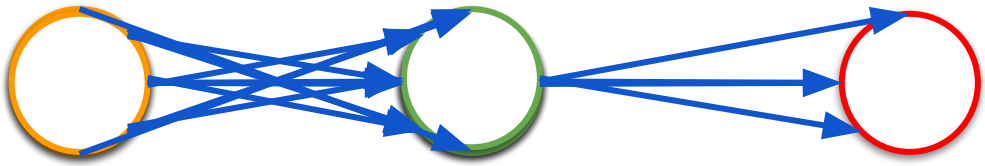
Learning: Hundreds of science fiction film plots
although each phrase has a meaning, there is a lack of a global vision

Recurrent Neural Networks (RNN)


xh -> input to Hidden
hy -> Hidden to output
W_{xh}
W_{hy}
Input Value 1
Input Value 2
Input Value 3
X_1
X_2
X_n
Output
Input Layer
Hidden Layer
Output Layer


Input Layer
Hidden Layer
Output Layer

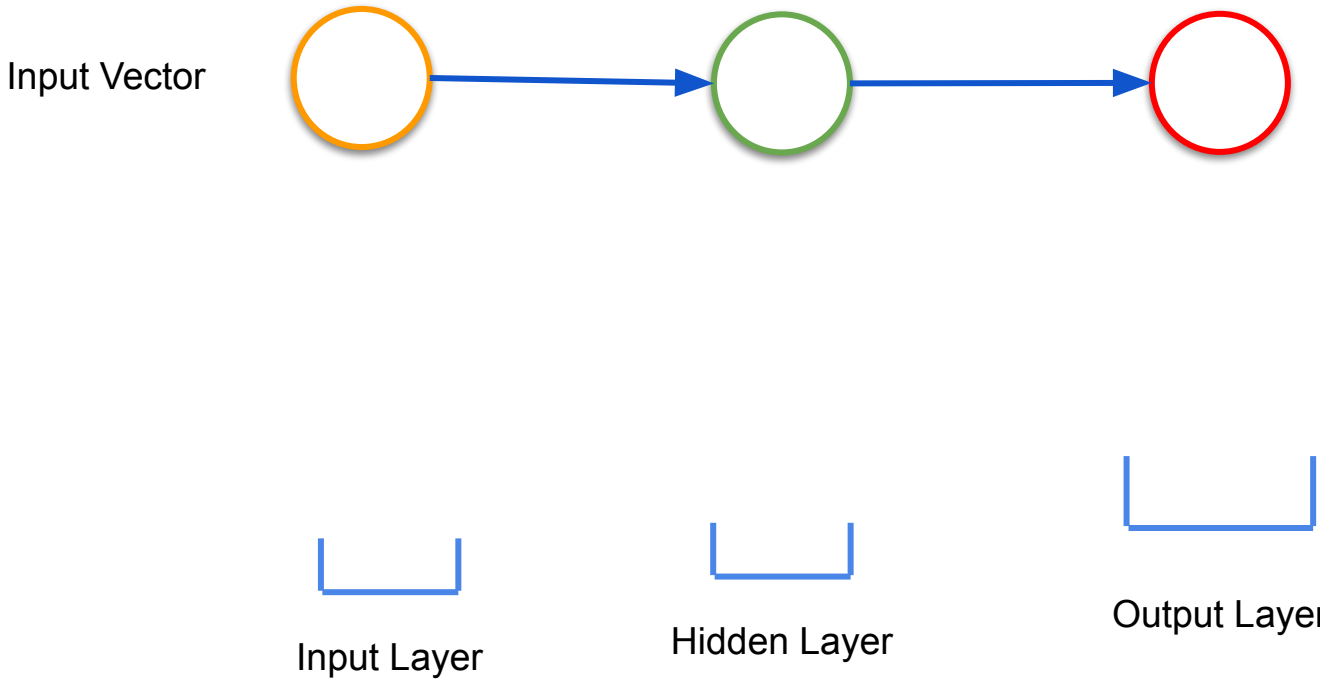
Input Vector

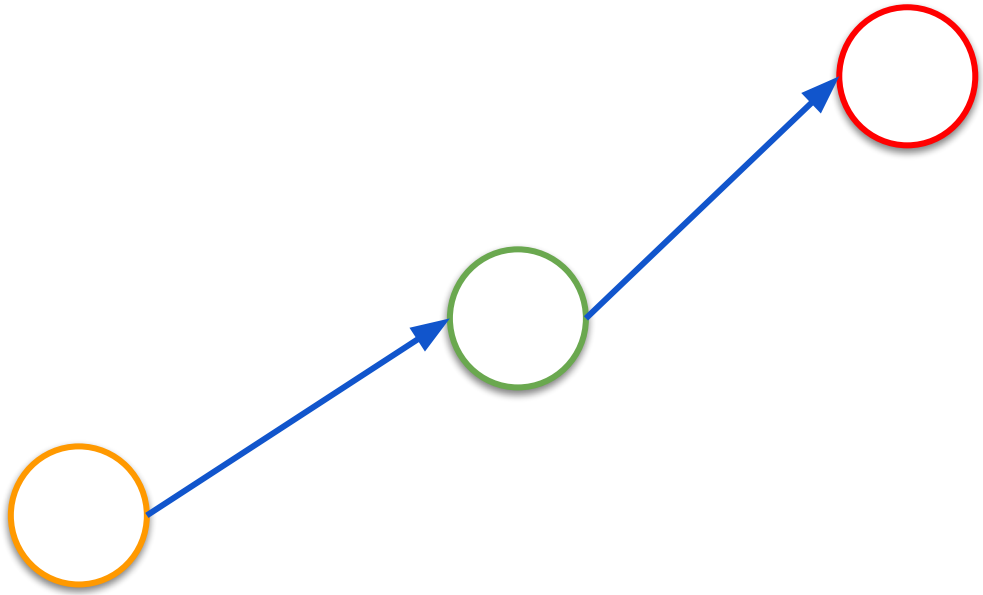


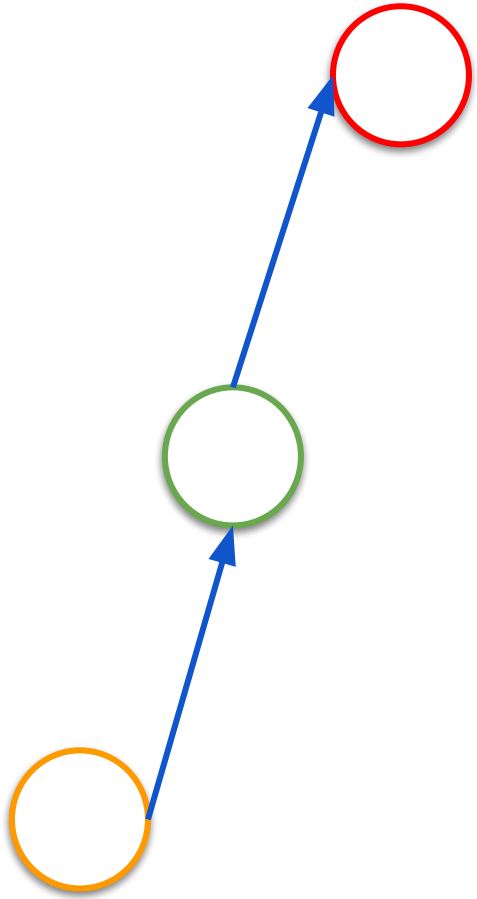
Input Layer

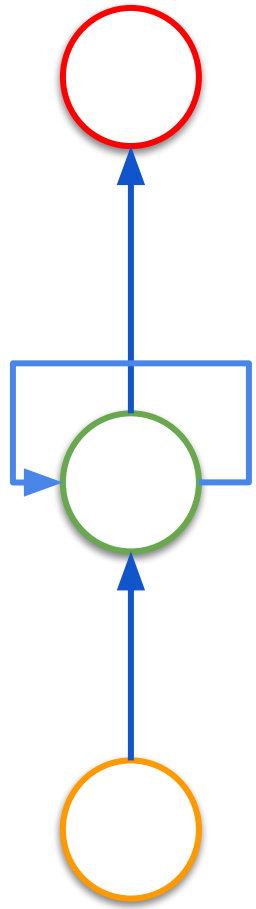
Hidden Layer

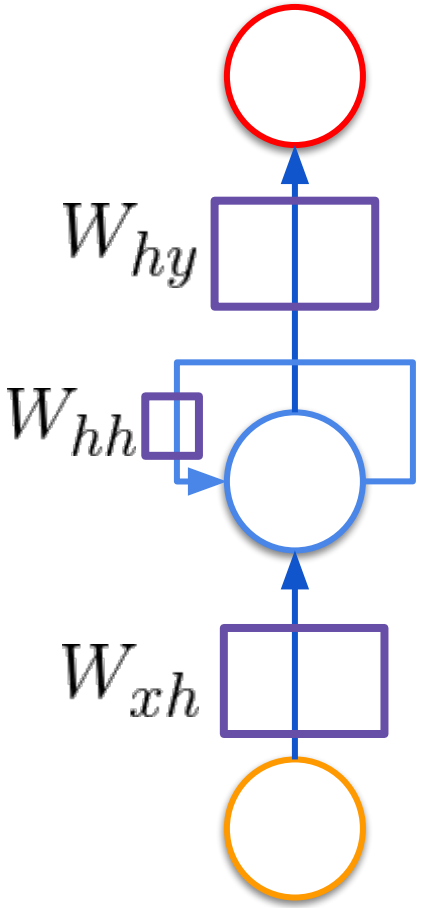
Output Layer

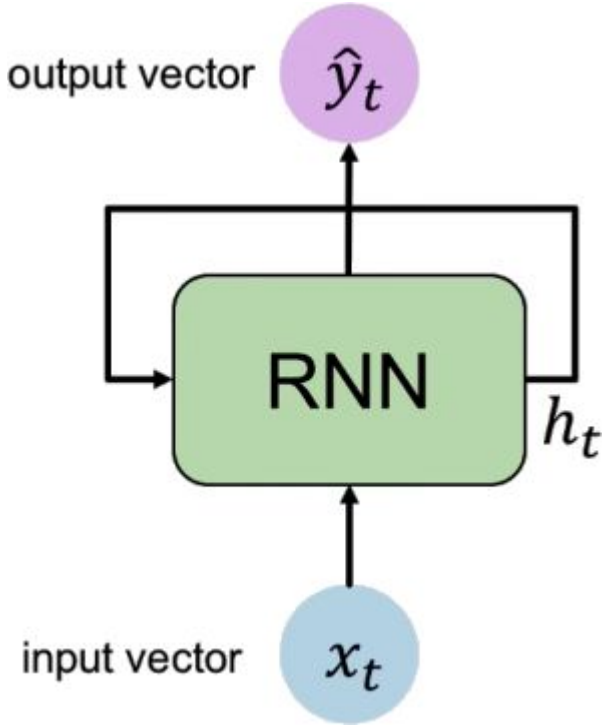

Input Vector
Input Layer
Hidden Layer
Output Layer



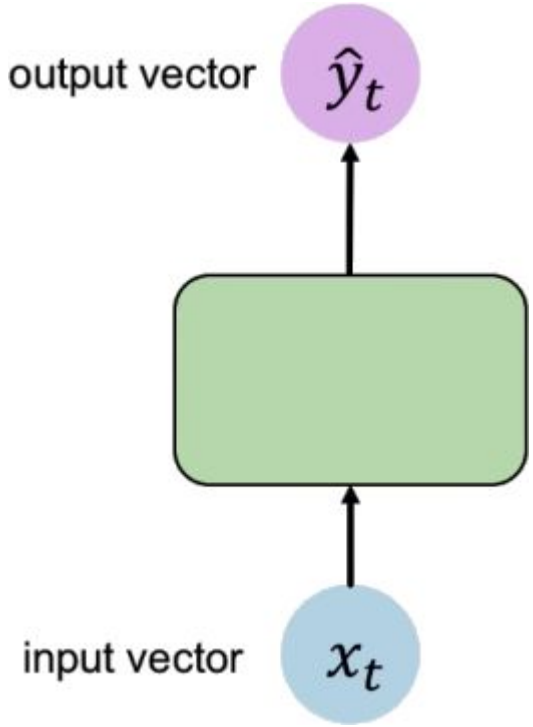


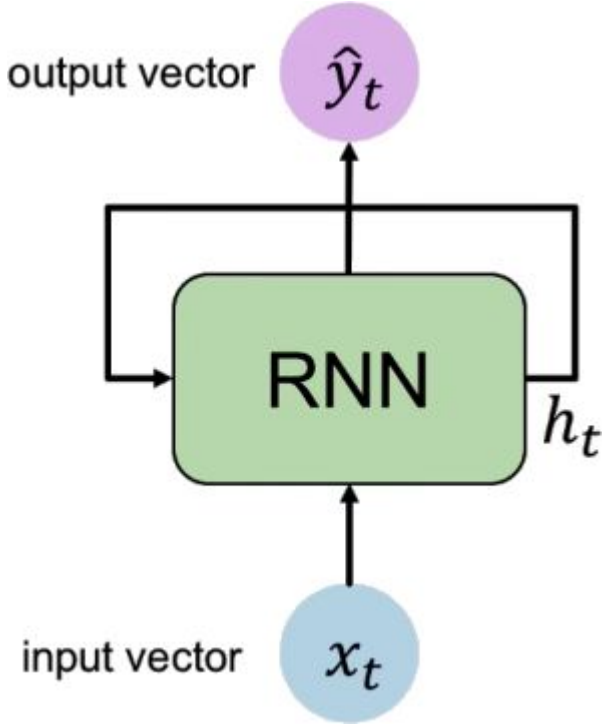



W_{hy}
W_{hh}
W_{xh}


output vector
$\hat{y}_t$
RNN
h_t
input vector
x_t

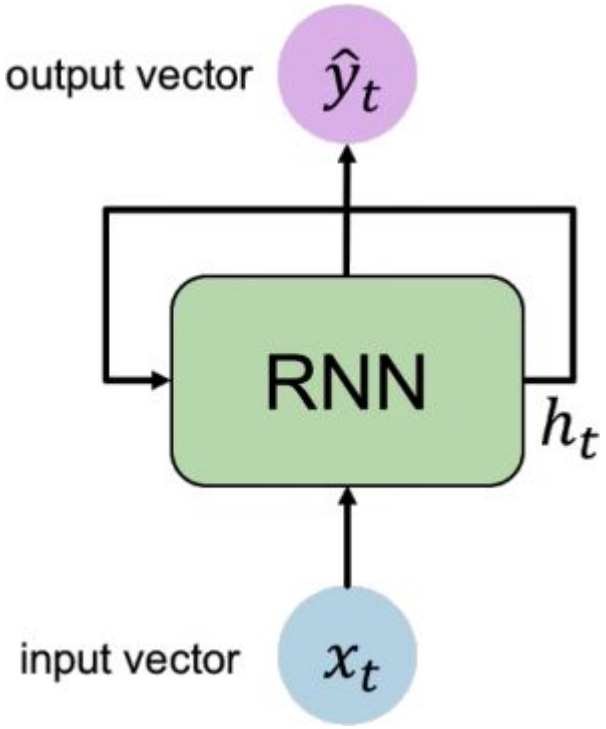
“Vanilla” FFNN vs RNN





RNN

- In each timeStamp, RNNs have as input:
  - Xt (current reading)
  - ht (internal state)
- and one output:
  - yt



Output Vector

$$\hat{y}_t = \boldsymbol{W}_{hy}^{T} h_t$$

Update Hidden State

$$h_t = \tanh(\boldsymbol{W}_{hh}^{T} h_{t-1} + \boldsymbol{W}_{xh}^{T} x_t)$$

Input Vector

$$x_t$$

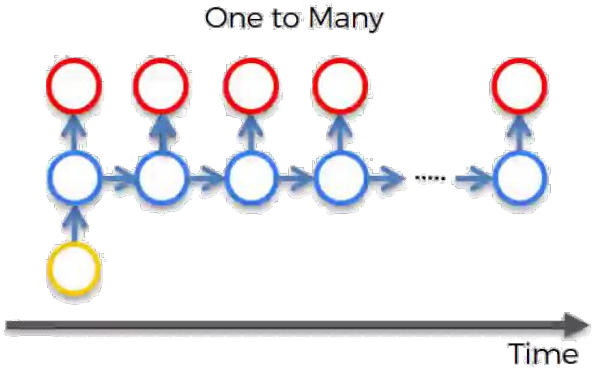
Wxh -> input to hidden
Whh -> hidden to hidden
Why -> hidden to output

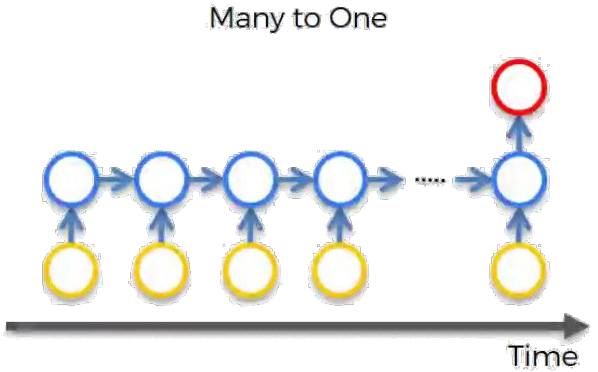
```
tf.keras.layers.SimpleRNN(rnn_units)
```

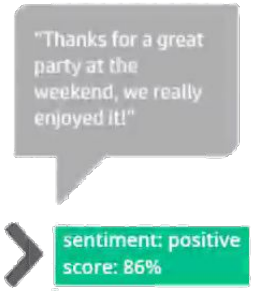
RNN Usages

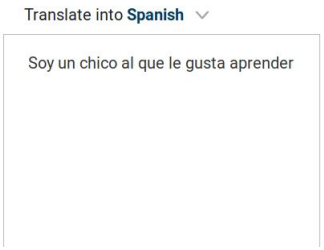
Mondragon Unibertsitatea
Faculty of Engineering

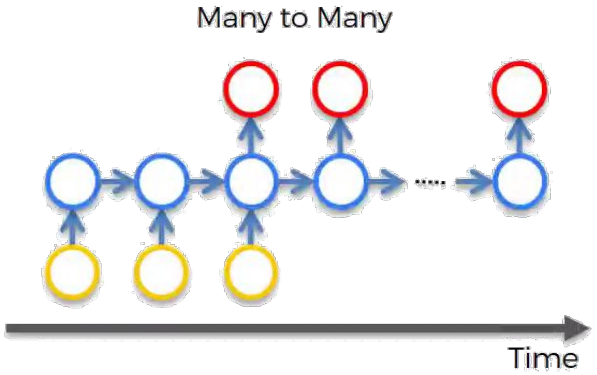
"black and white dog jumps over bar."

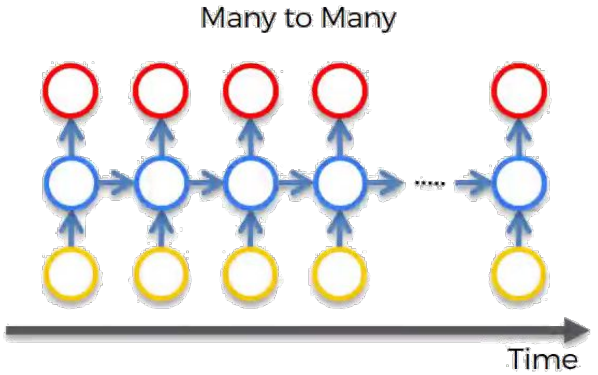


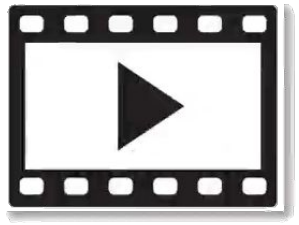








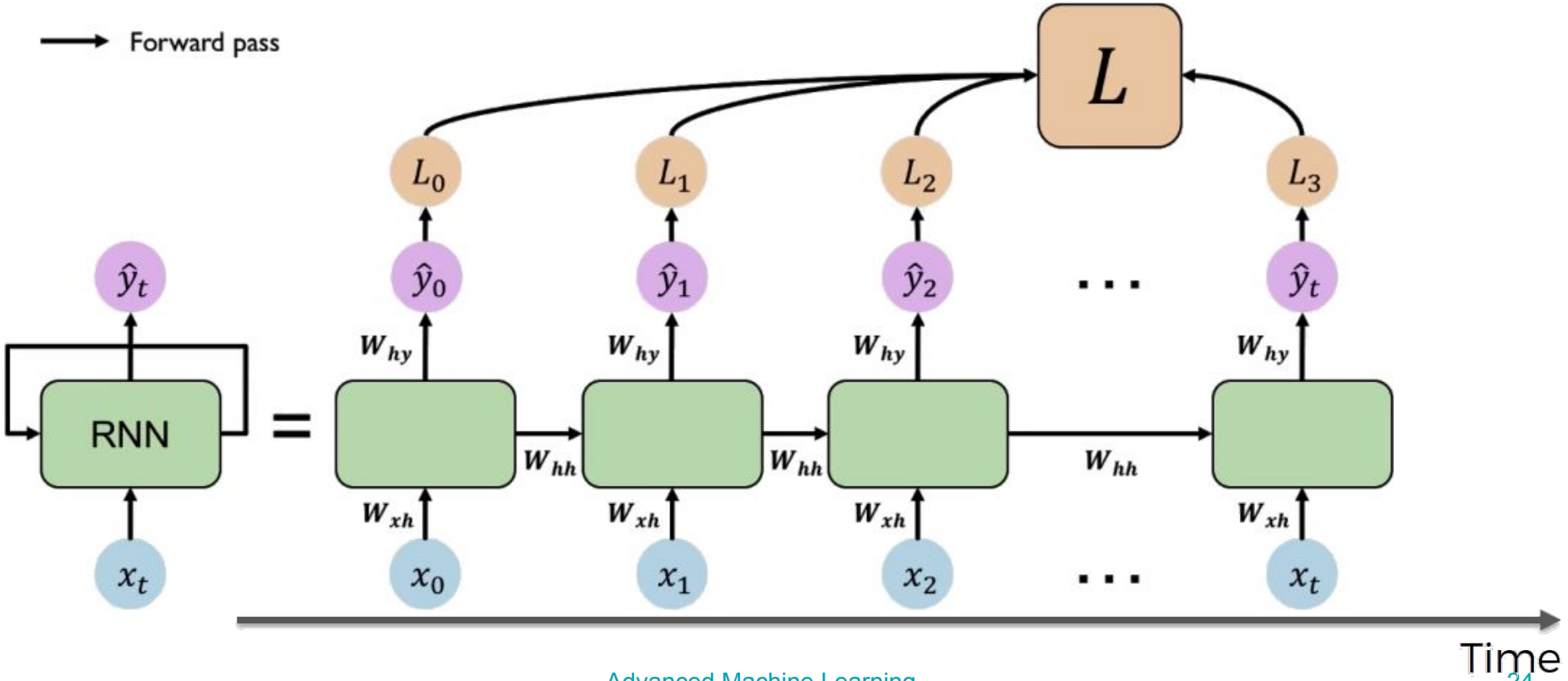




RNN, Forward pass and Loss calculation

Mondragon Unibertsitatea
Faculty of Engineering

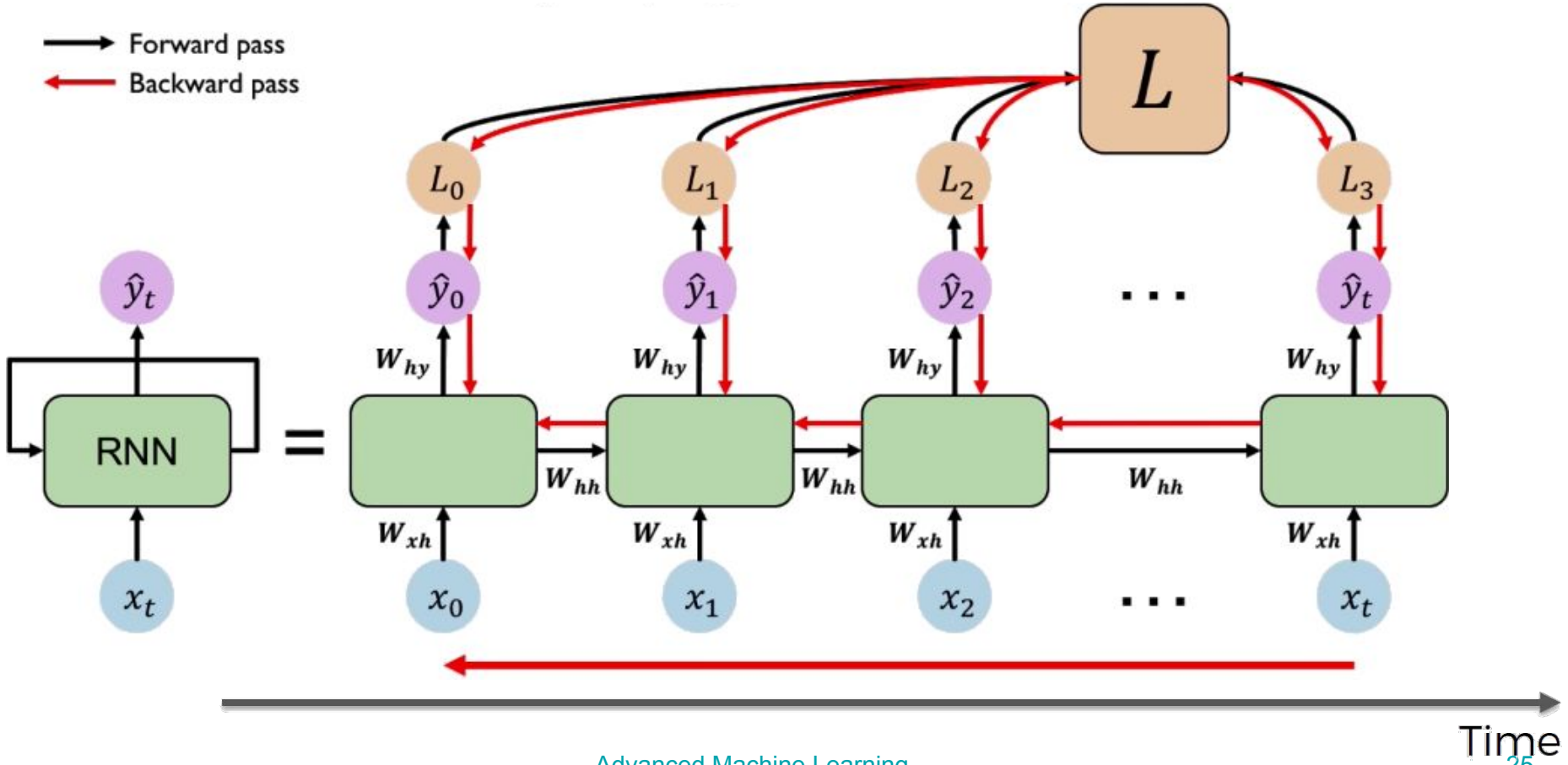
- We can depict RNNs as:
  - Multiple copies of themselves, that pass down a message through time (ht)
  - REMEMBER: we reuse the same Weight matrices at each time step
  - In the forward pass:
    - We can compute a loss for each time step, and the global loss will be the sum of all the losses



RNN-> Backpropagation Through Time

Mondragon Unibertsitatea
Faculty of Engineering

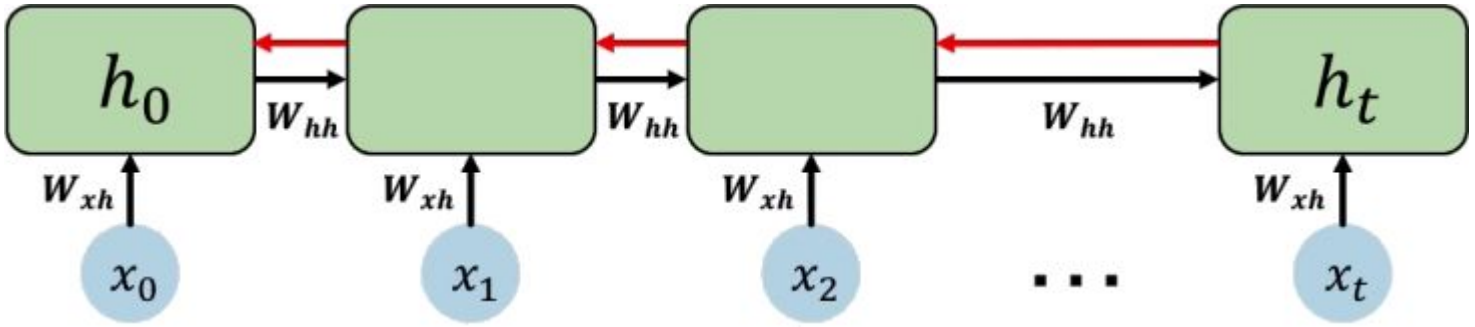
- BPTT
  - Error is Back Propagated
    - First though each state
    - Finally through Hidden states



Problems with RNNs

Mondragon Unibertsitatea
Faculty of Engineering

- The RNN : “Vanishing / Exploding Gradient”
- Computing the gradient of h0:
  - involves many “factors” (multiplications) of Whh and activation function derivatives



Razvan Pascanu et al. (2013)

Many values > 1:
exploding gradients

Gradient clipping to scale big gradients

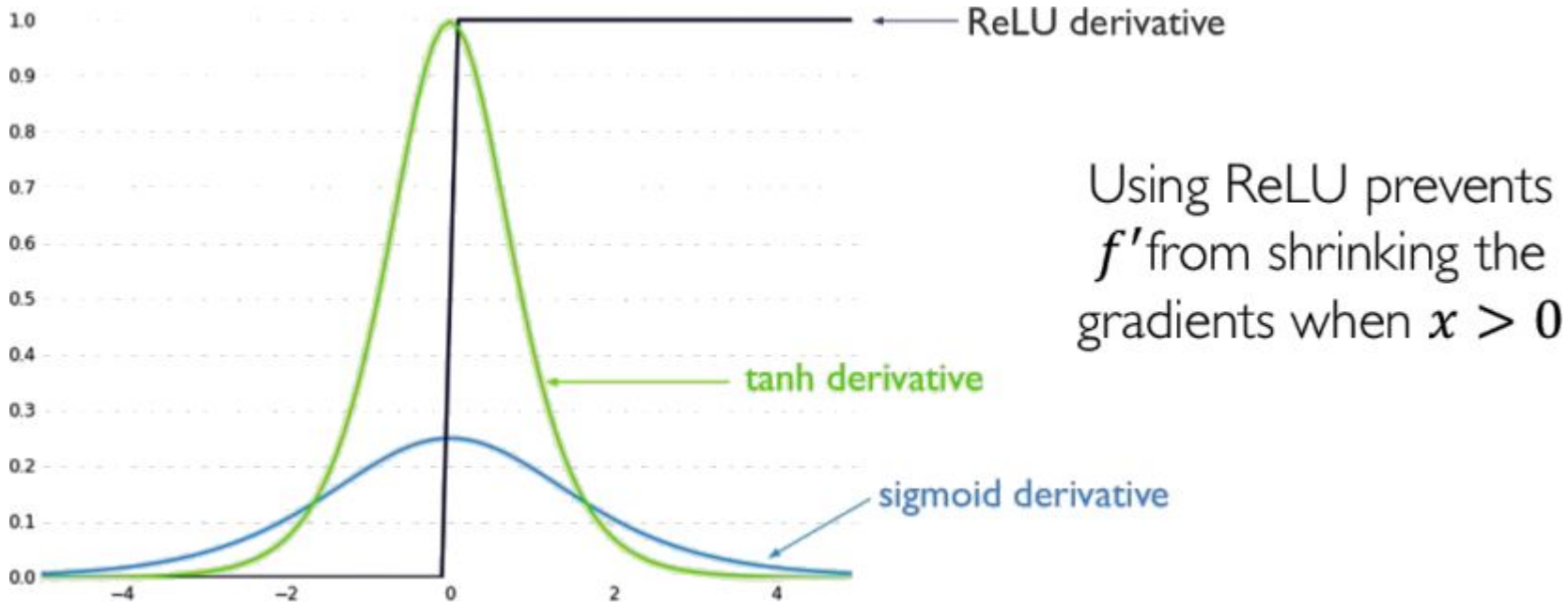
Many values < 1:
vanishing gradients

1. Activation function
2. Weight initialization
3. Network architecture

Tackling Vanishing Gradient Problem

Mondragon Unibertsitatea
Faculty of Engineering

- Strategy 1: **Choose activation function whose derivative is = 1**



Tackling Vanishing Gradient Problem

- Strategy 2: **Initialise the weights to identity matrix**
  - Usually, Weights are initialized to a number close to 0
  - Initialising them to identity matrix, helps from shrinking too quickly

$$I_n = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}$$

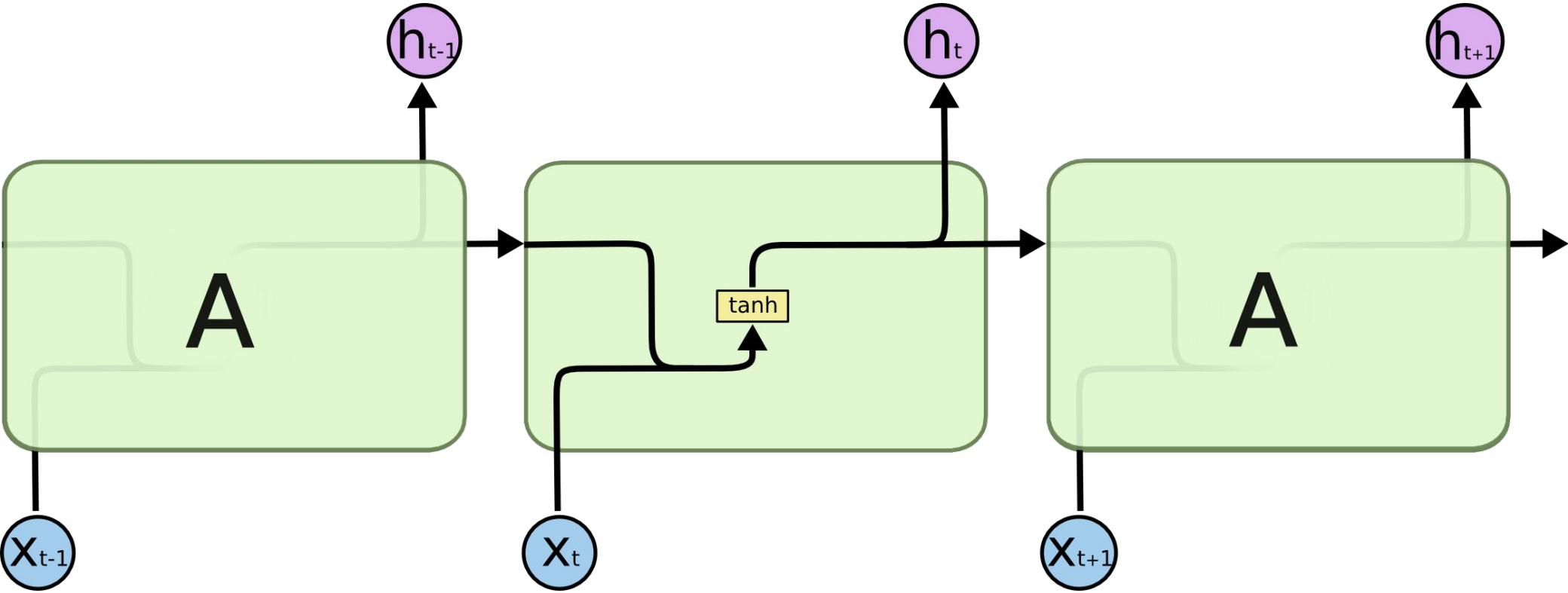
Tackling Vanishing Gradient Problem

- Strategy 3: **Using complex recurrent units with gates**
  - Instead of using vanilla RNN
  - These help control the information that is passing to the future

Long Short-Time Memory Networks (LSTM)

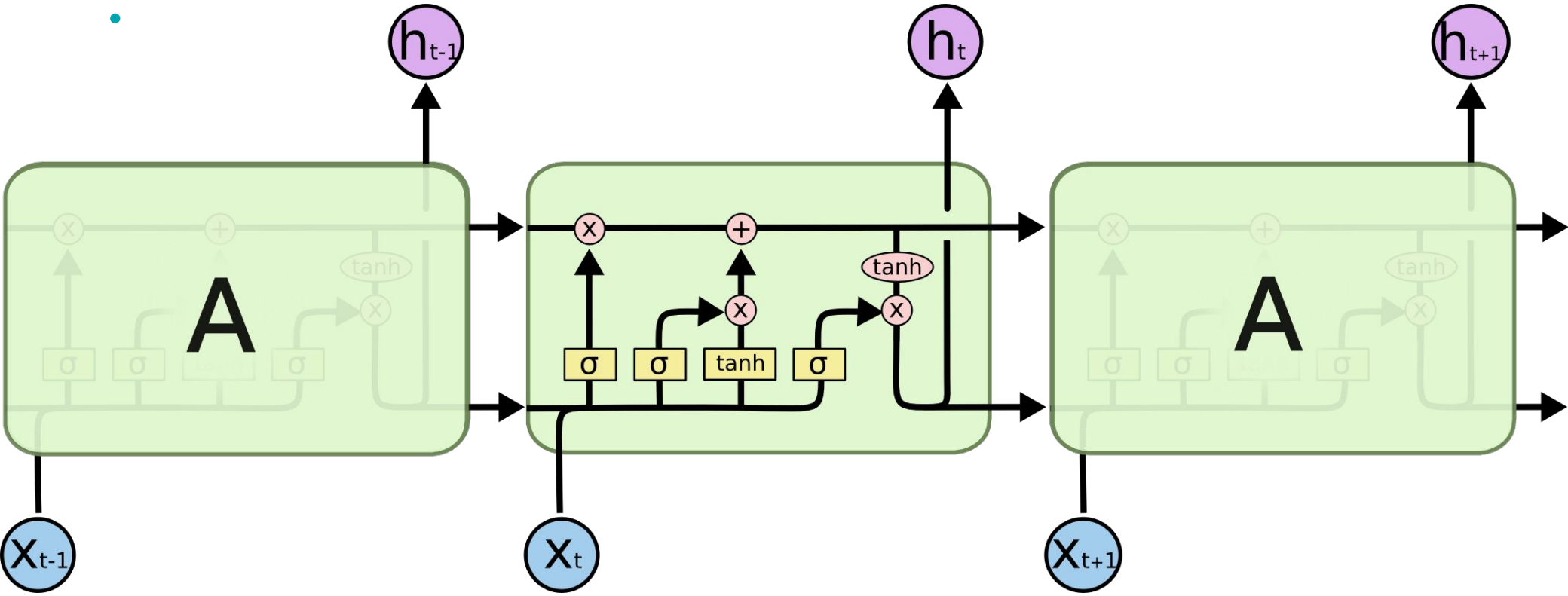
```
tf.keras.layers.LSTM(units)
```

Normal RNN



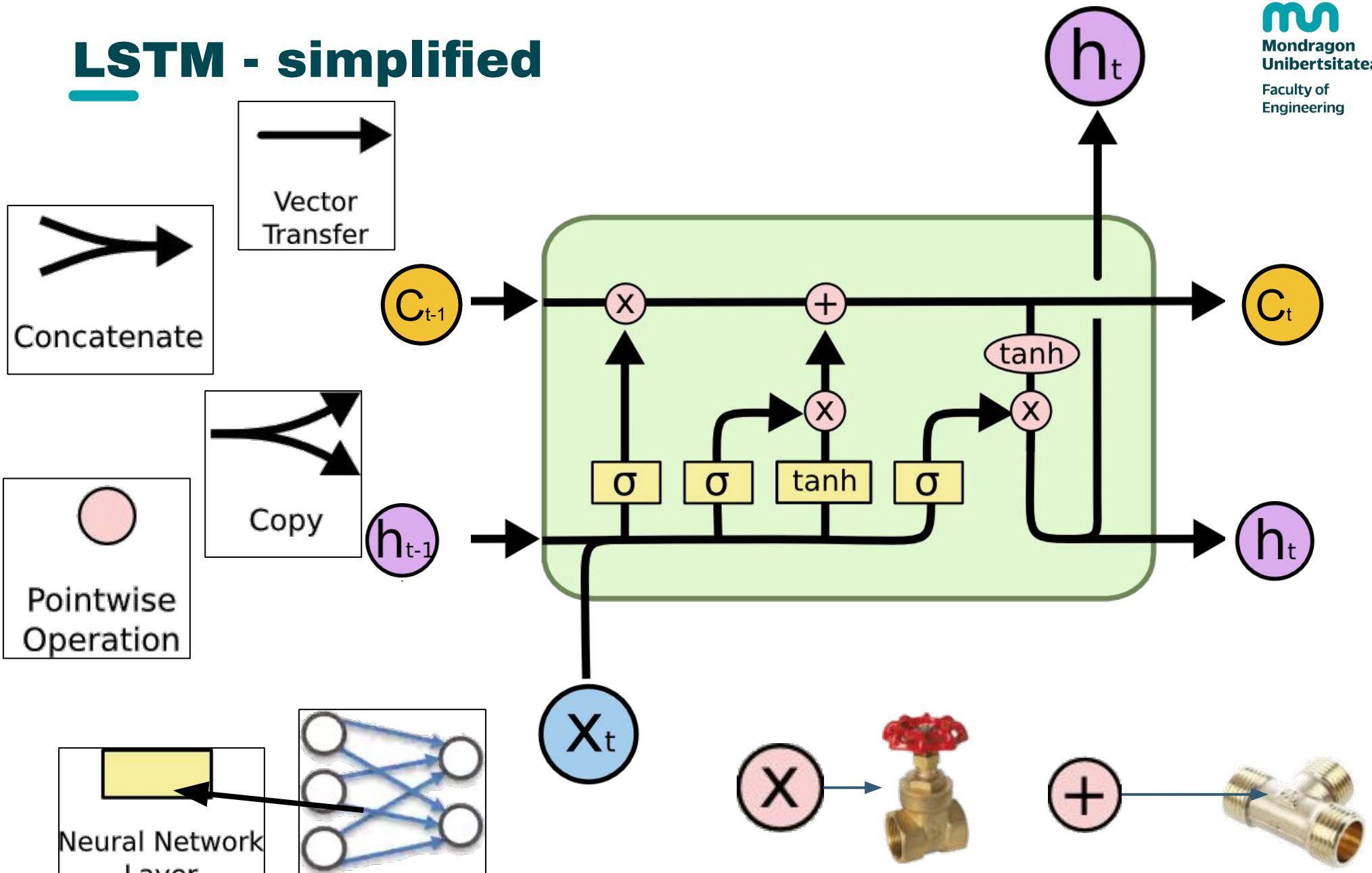
source: Christopher Olah blog

LSTM - simplified



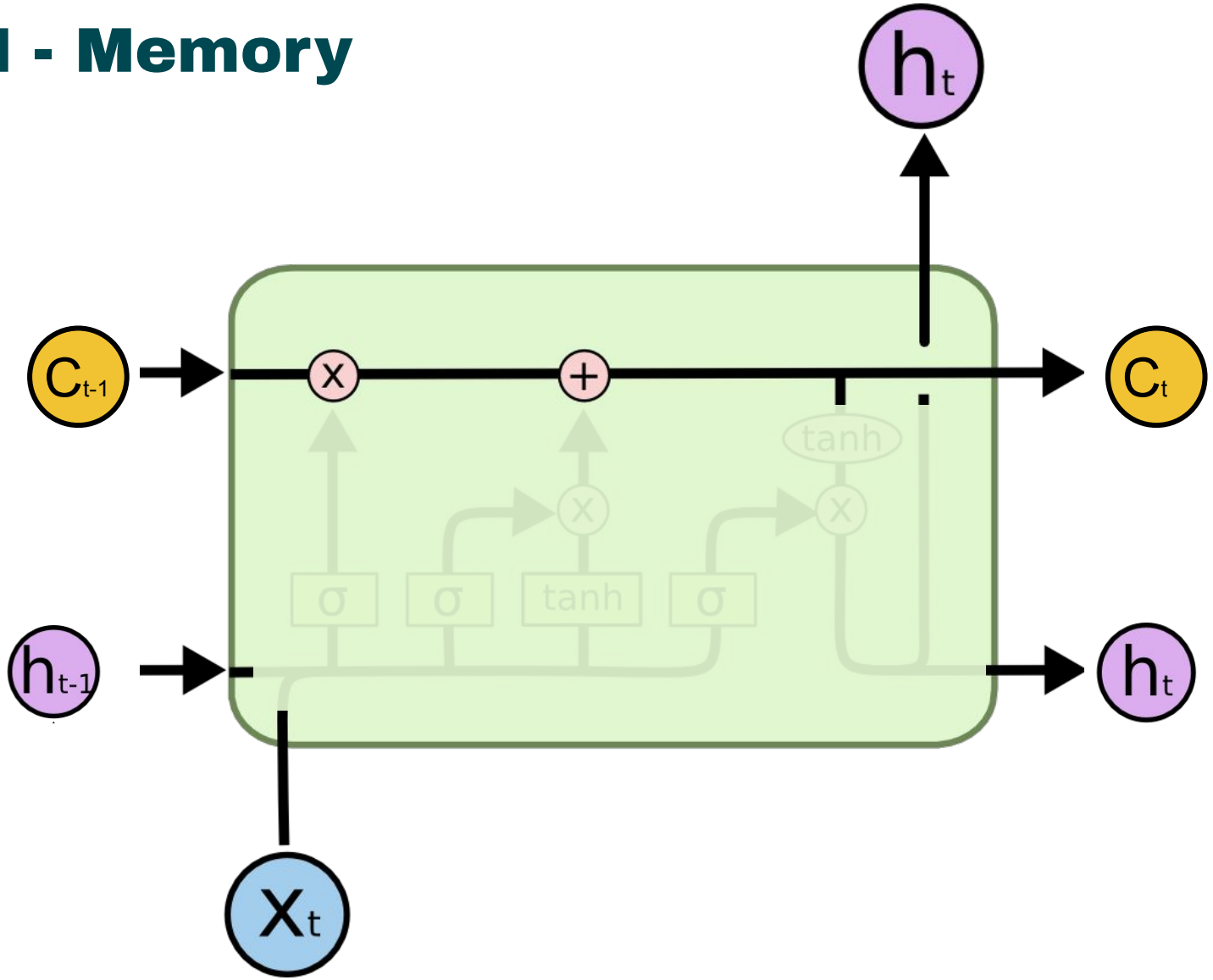
source: Christopher Olah blog

LSTM - simplified



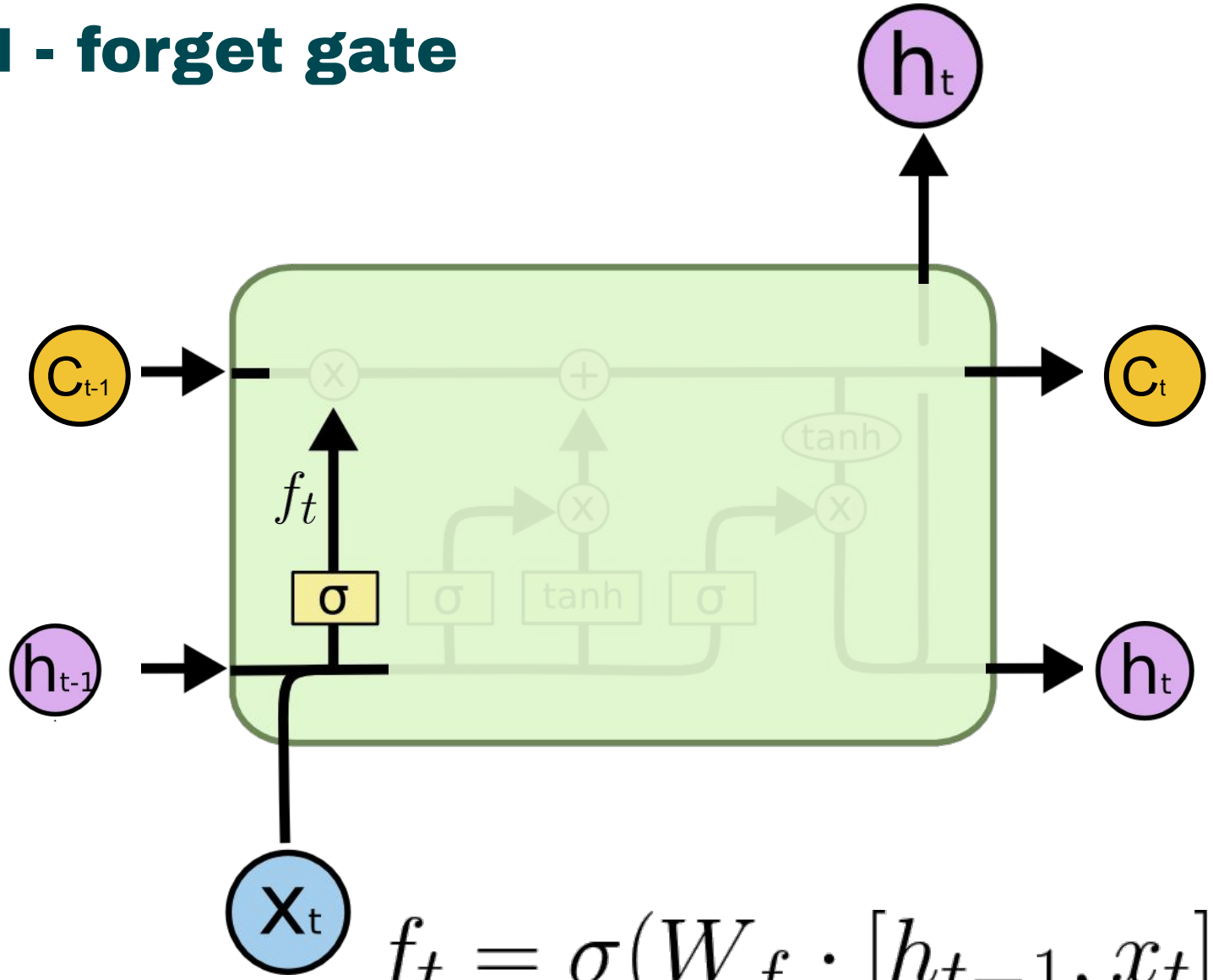
source: Christopher Olah blog

LSTM - Memory



source: Christopher Olah blog

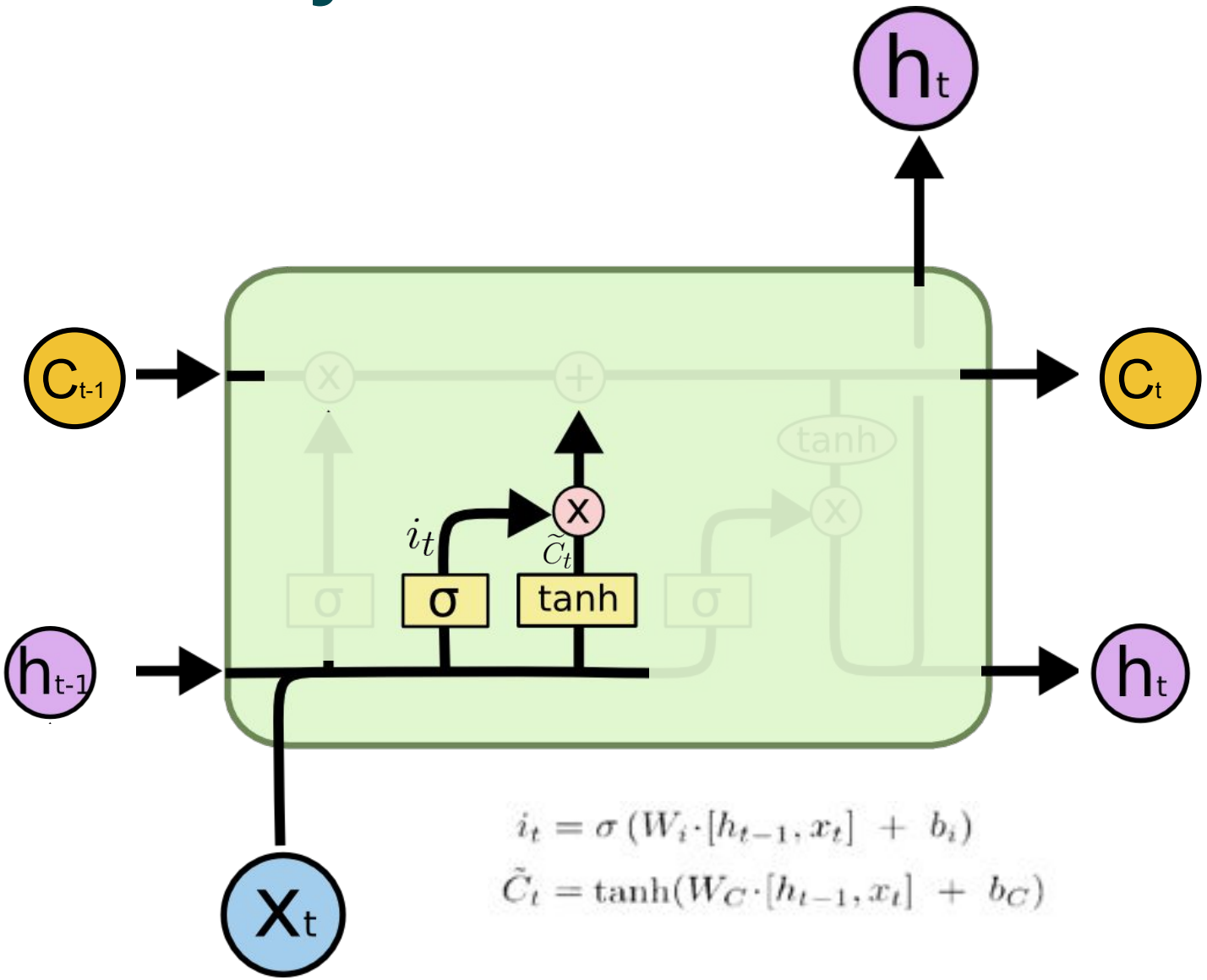
LSTM - forget gate



$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

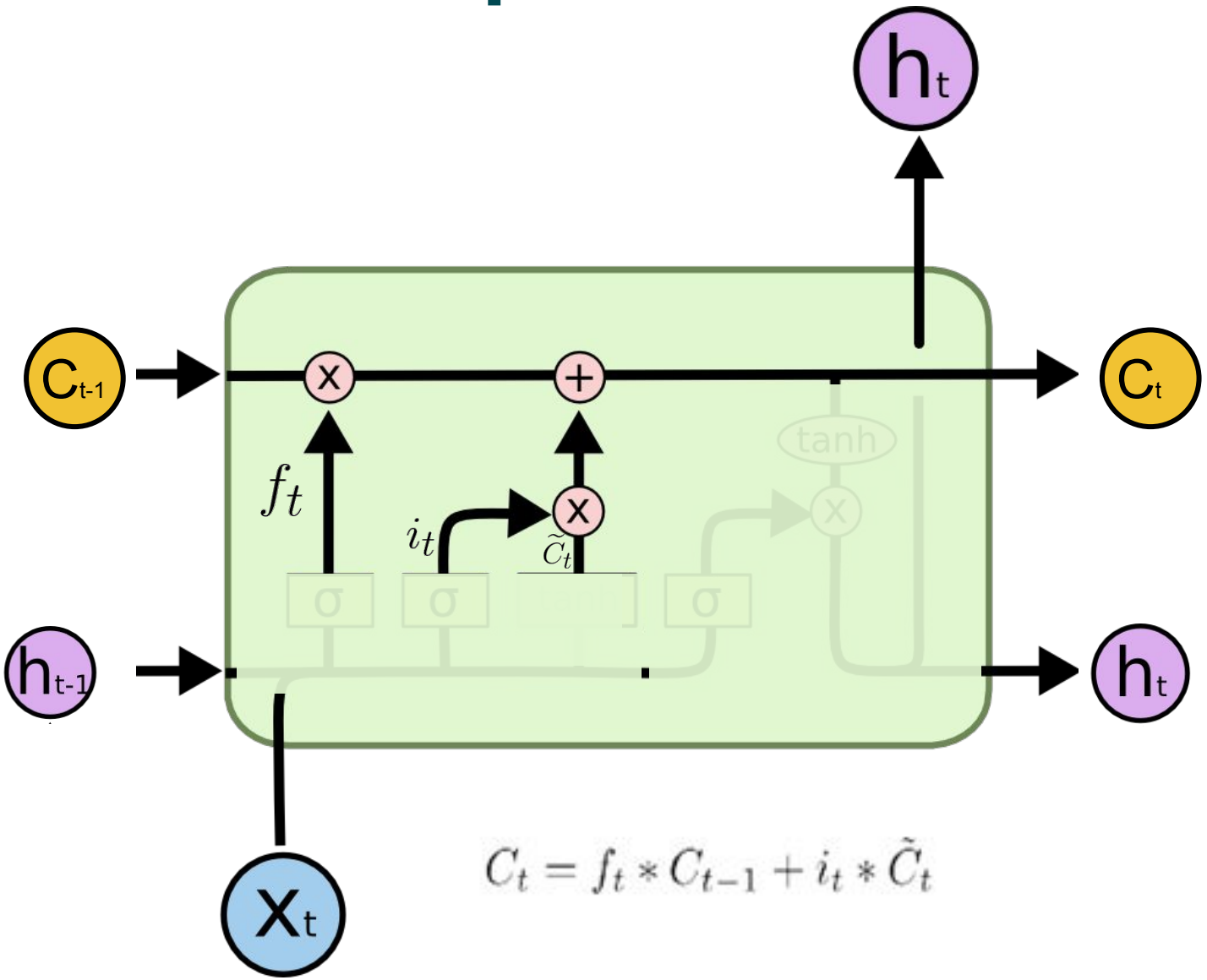
source: Christopher Olah blog

LSTM - Memory Gate



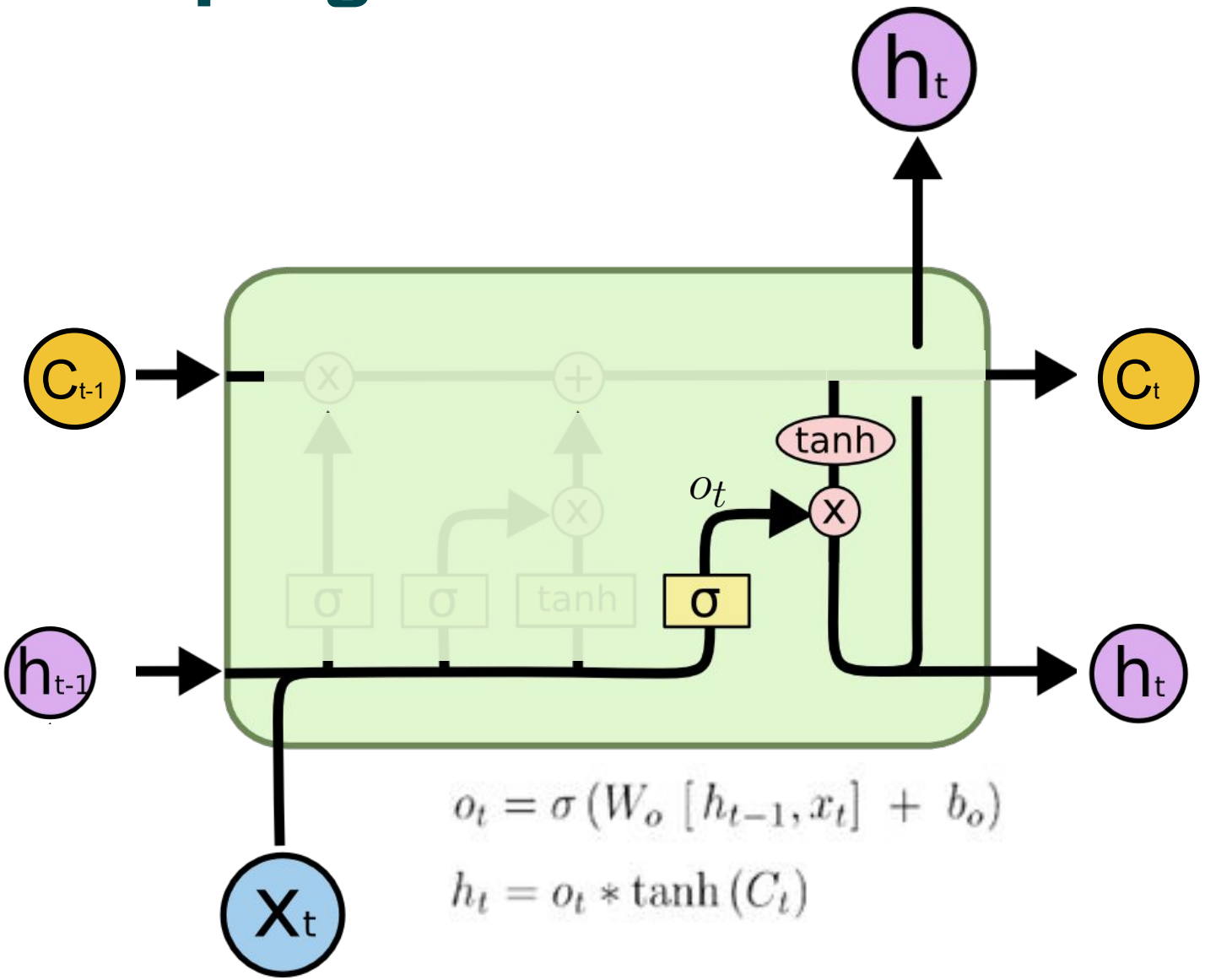
source: Christopher Olah blog

LSTM - Cell State Update



source: Christopher Olah blog

LSTM - Output gate



source: Christopher Olah blog

LSTM - Cosas a tener en cuenta

Mondragon Unibertsitatea
Faculty of Engineering

- El número de variables a optimizar es muy grande por cada LSTM:
  - pesos y bias del forget gate
  - pesos y bias del memory gate X2
  - Pesos y bias del output gate
- Todo eso, por cada neurona LSTM!

Son muy potentes, pero, el tiempo de aprendizaje es alto, hay que diseñarlos con cabeza, si no, perdemos mucho tiempo.

Gated Recurrent Unit (GRU)

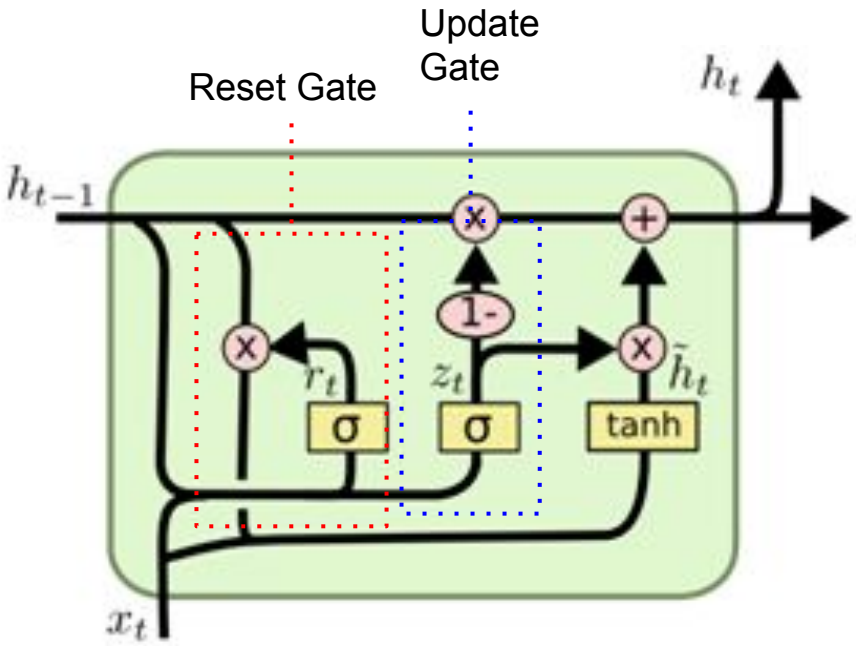
```
tf.keras.layers.GRU(units)
```

RNN - GRU

Mondragon Unibertsitatea
Faculty of Engineering

- Similar to LSTM
  - They combine the forget and input gate into only one gate, called the update gate
  - They also merge the Cell state and the hidden state into one

RNN - Gated Recurrent Unit



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

RNN - GRU

Mondragon Unibertsitatea
Faculty of Engineering

- The main advantage of GRU is **the reduced number of parameters** as compared to LSTM's without any compromise whatsoever which has resulted in **faster convergence** and a **more generalized** model.

Advanced use of RNNs

RNN - Recurrent Dropout

Mondragon Unibertsitatea
Faculty of Engineering

- RNNs also tend to Overfit.
- But applying dropout is not trivial for RNN
- The same pattern of dropped units should be applied every time step
  - Otherwise instead of helping learn you achieve the contrary *(Yarin Gal 2015)*
- Keras has 2 dropouts for RNNs
  - Dropout-> specifies dropout rate for input units of the layer
  - Recurrent_dropout-> specifies dropout rate for RNN units

```
model.add(layers.GRU(64, activation='relu',
                     dropout=0.1,
                     recurrent_dropout=0.5))
```

RNN - Stacking Recurrent Layers

Mondragon Unibertsitatea
Faculty of Engineering

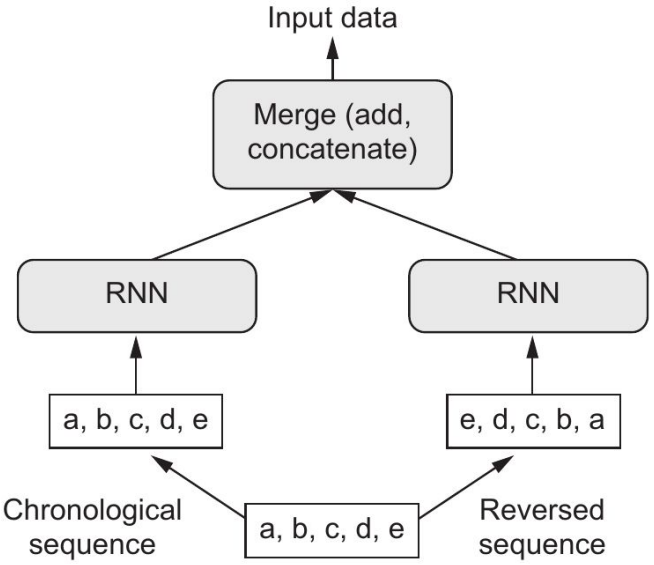
- We can stack several recurrent layers one after another
  - For example, in 2018 7 huge LSTM layers powered google translate
- In order to be able to do this, we need all intermediate layers to return their full sequence of outputs (a 3D tensor)
- This is done the next way:

```
model.add(layers.GRU(32,
                     dropout=0.1,
                     recurrent_dropout=0.5,
                     return_sequences=True,
                     input_shape=(None, float_data.shape[-1])))
```

RNN - Bidirectional RNNs

Mondragon Unibertsitatea
Faculty of Engineering

- The basic idea behind the bidirectional RNNs is to have 2 RNNs
  - One looking at the data in the given order (chronologically)
  - The other having the data reversed (anti-chronologically)
- Later, the both representations are merged
  - Helps to catch patterns that may have been overlooked

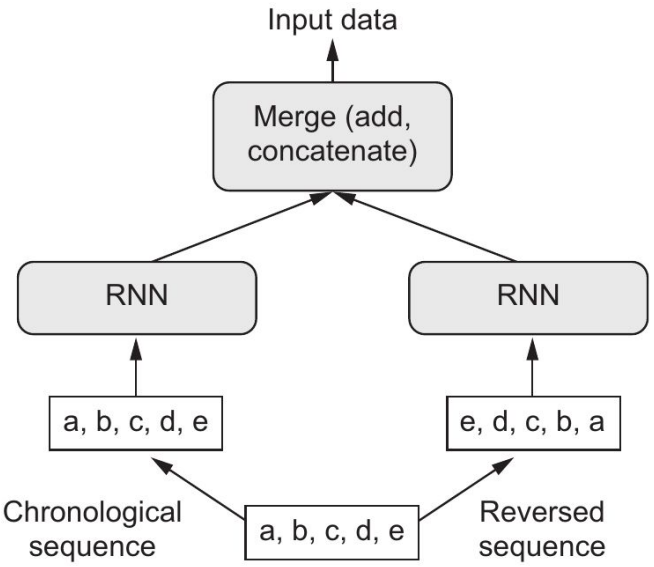



```
model = Sequential()
model.add(layers.Bidirectional(
    layers.GRU(32), input_shape=(None, float_data.shape[-1])))
```

RNN - Bidirectional RNNs

Mondragon Unibertsitatea
Faculty of Engineering

- The basic idea behind the bidirectional RNNs is to have 2 RNNs
  - One looking at the data in the given order (chronologically)
  - The other having the data reversed (anti-chronologically)
- Later, the both representations are merged
  - Helps to catch patterns that may have been overlooked




```
model = Sequential()
model.add(layers.Bidirectional(
    layers.GRU(32), input_shape=(None, float_data.shape[-1])))
```

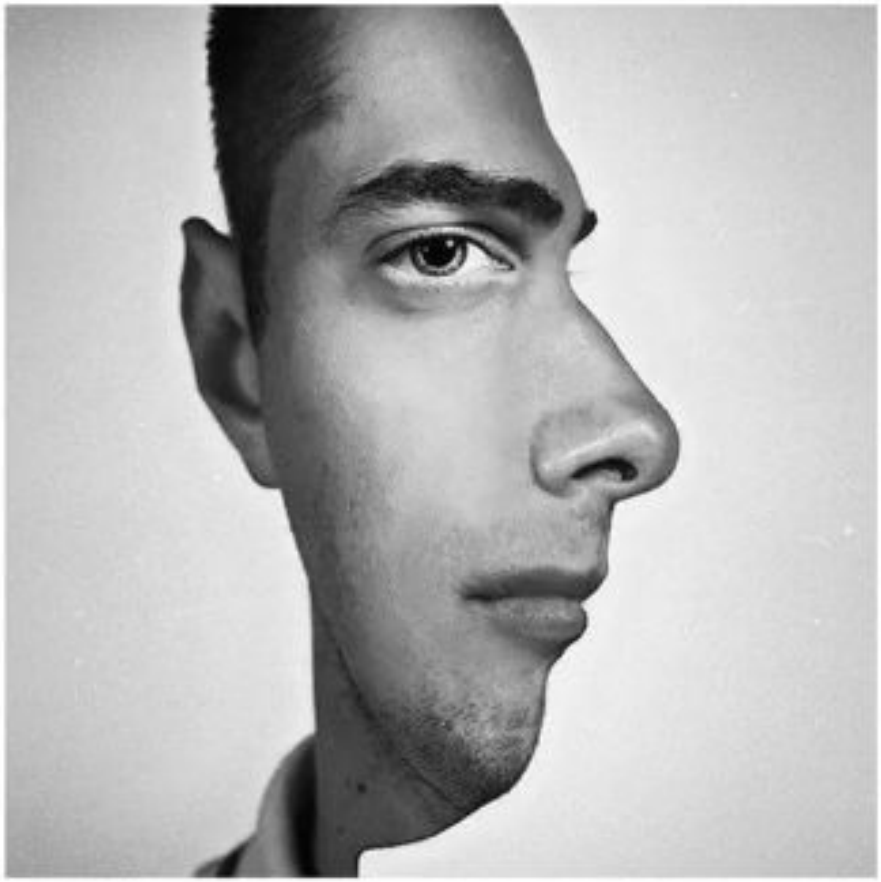
Exercises

Mondragon Unibertsitatea
Faculty of Engineering

- 2 rnn examples
- exercise 1
  - google stock prediction
- exercise 2
  - text generation with rnn, moby dick

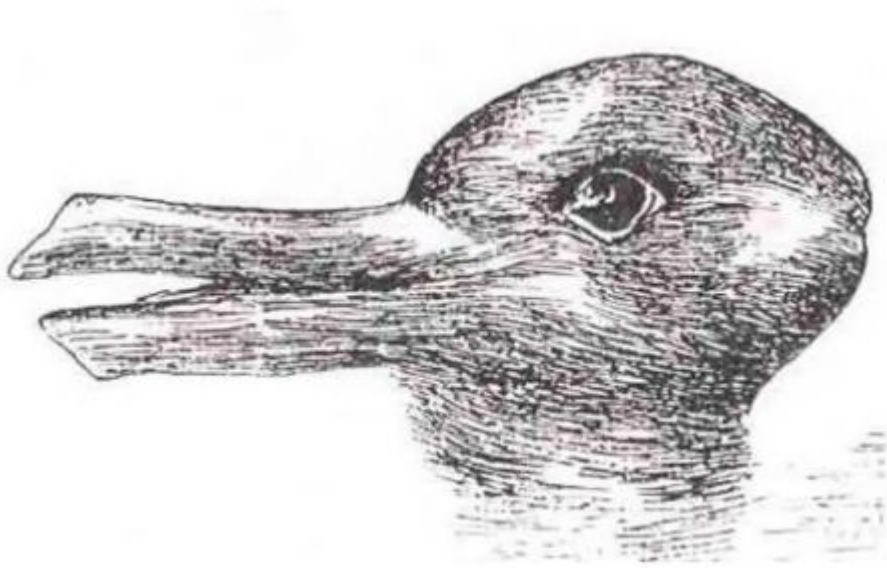
Convolutional Neural Networks

Features from images



Features from images

Mondragon Unibertsitatea
Faculty of Engineering



Examples from the test set
(with the network's guesses)

source: Geoffrey Hinton (charla)

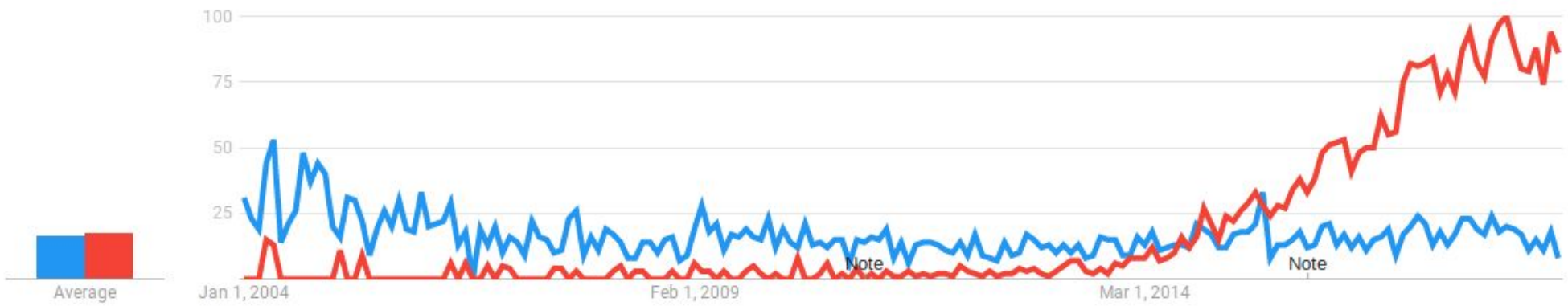
CNNs

Mondragon Unibertsitatea
Faculty of Engineering

artificial neural network
Search term

convolutional neural
Search term

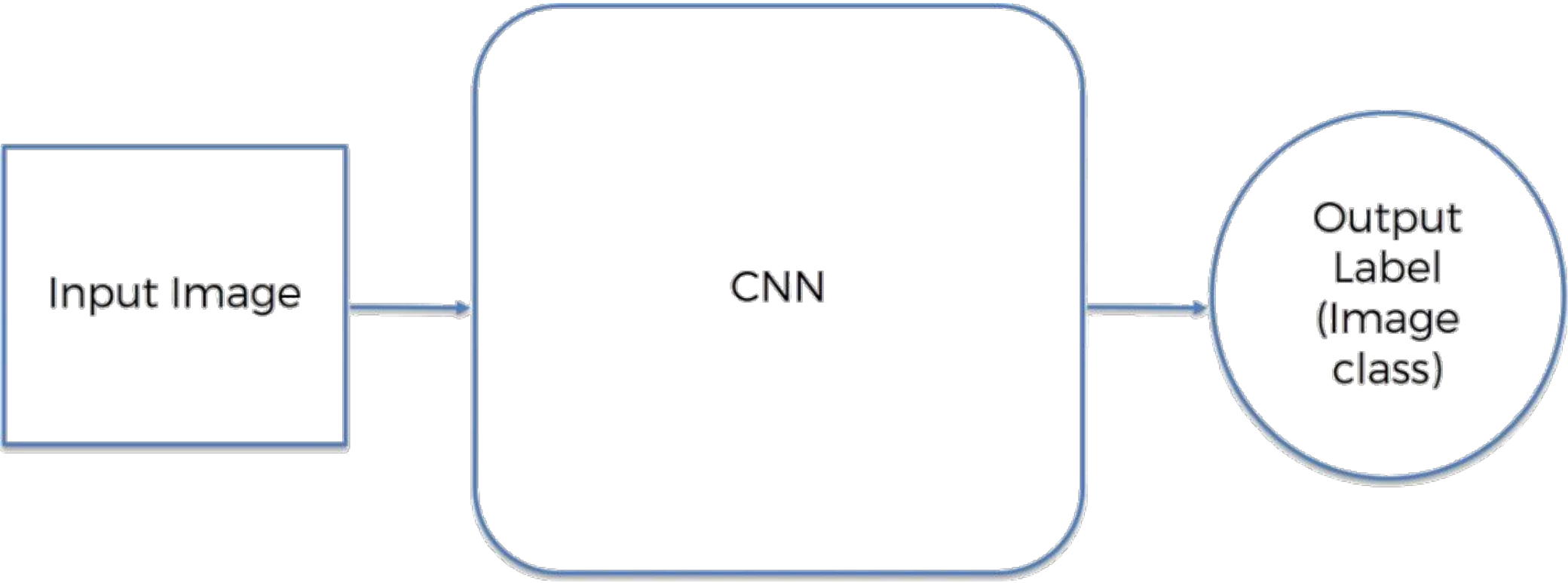
Interest over time

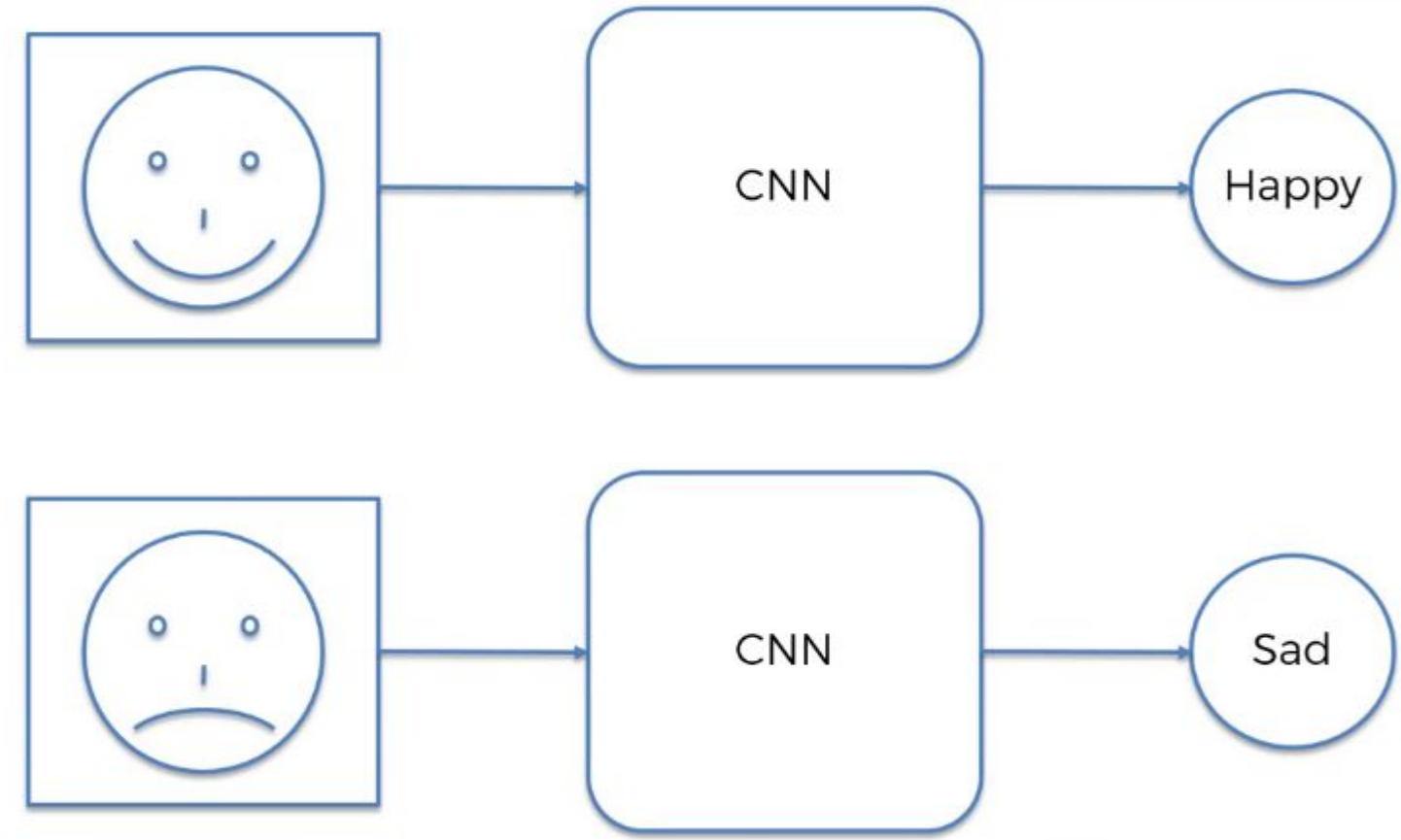


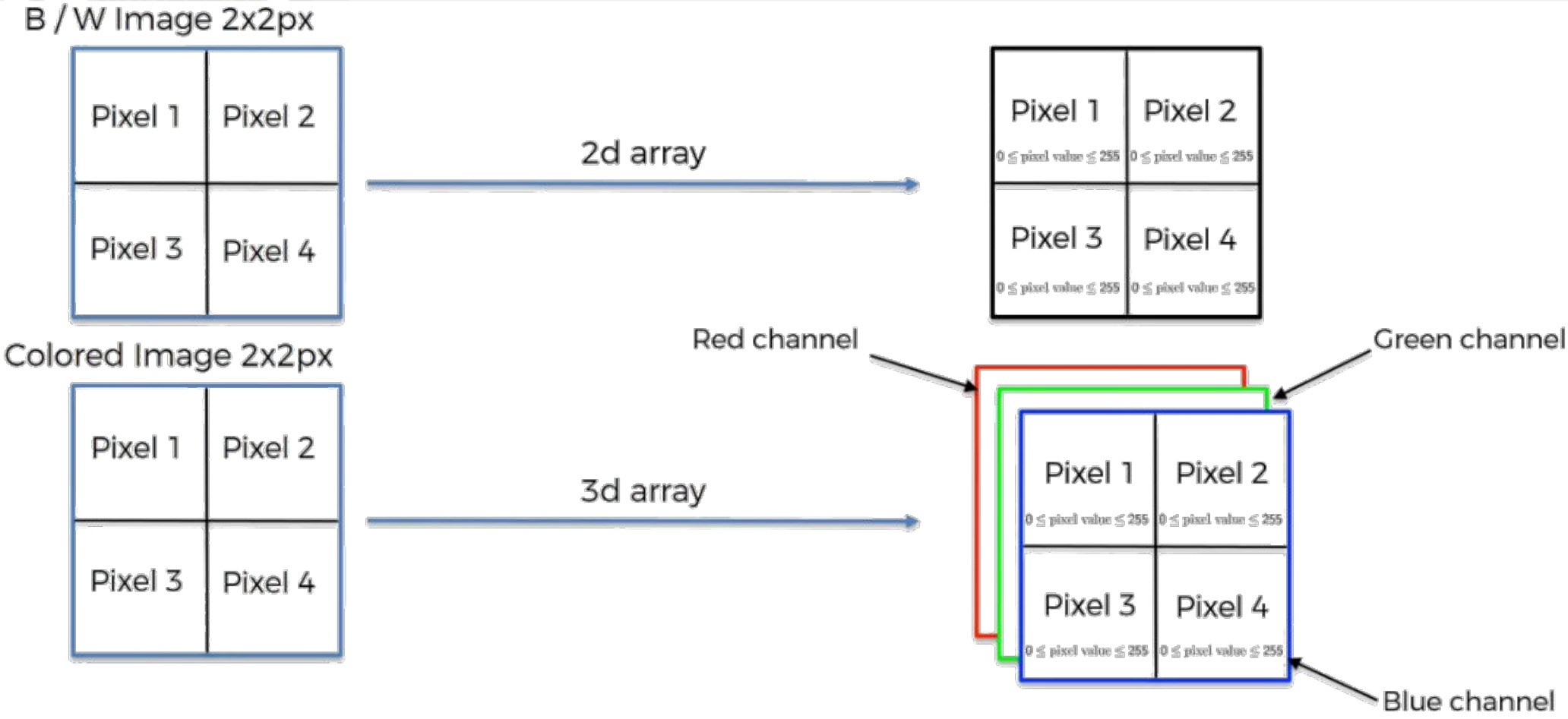
source: Google Trends

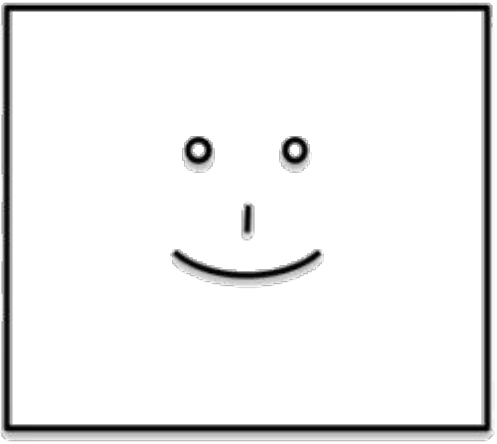
Yann Lecun

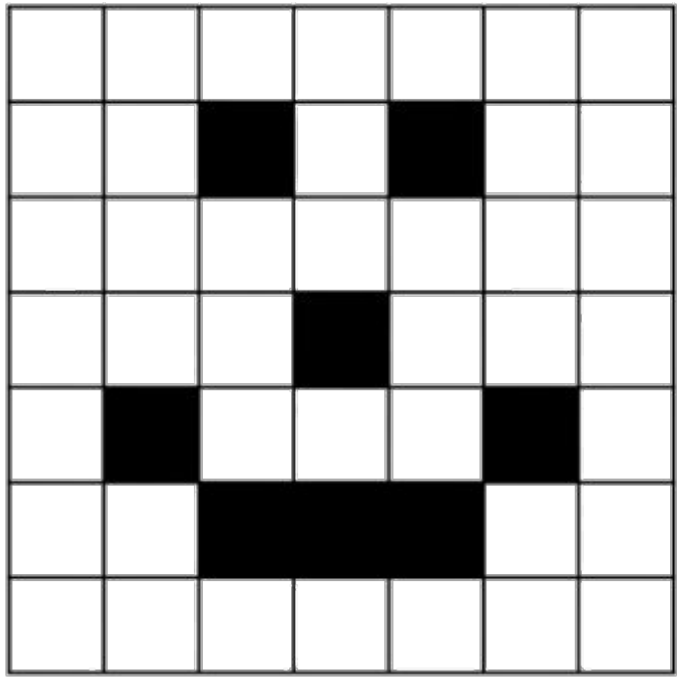
Google Facebook


Input Image
CNN
Output Label (Image class)


CNN
Happy
CNN
Sad


B / W Image 2x2px
Pixel 1
Pixel 2
Pixel 3
Pixel 4
2d array
Pixel 1
0 ≤ pixel value ≤ 255
Pixel 2
0 ≤ pixel value ≤ 255
Pixel 3
0 ≤ pixel value ≤ 255
Pixel 4
0 ≤ pixel value ≤ 255
Colored Image 2x2px
Pixel 1
Pixel 2
Pixel 3
Pixel 4
3d array
Red channel
Green channel
Blue channel
Pixel 1
0 ≤ pixel value ≤ 255
Pixel 2
0 ≤ pixel value ≤ 255
Pixel 3
0 ≤ pixel value ≤ 255
Pixel 4
0 ≤ pixel value ≤ 255





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

Steps of a CNN

1. Convolution
2. Max Pooling
3. Flattening
4. Full Connection

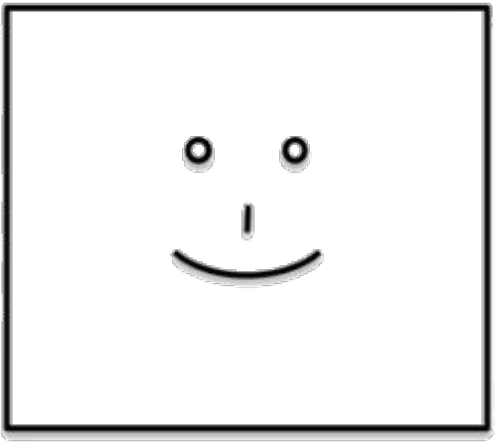
Step 1
Convolution

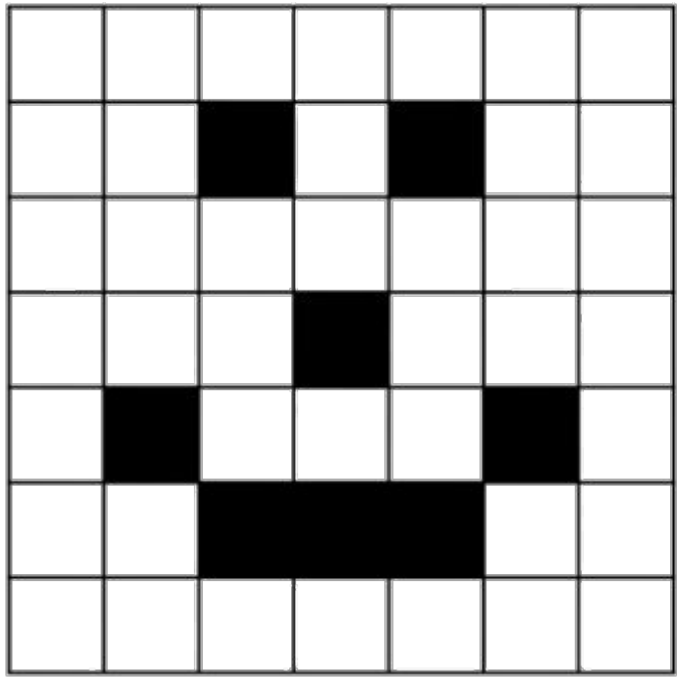
CNN Step 1: Convolution

Mondragon Unibertsitatea
Faculty of Engineering

$$(f * g)(t) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(\tau)\, g(t-\tau)\, d\tau$$

CNN Step 1: Convolution





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

CNN Step 1: Convolution

Mondragon Unibertsitatea
Faculty of Engineering

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

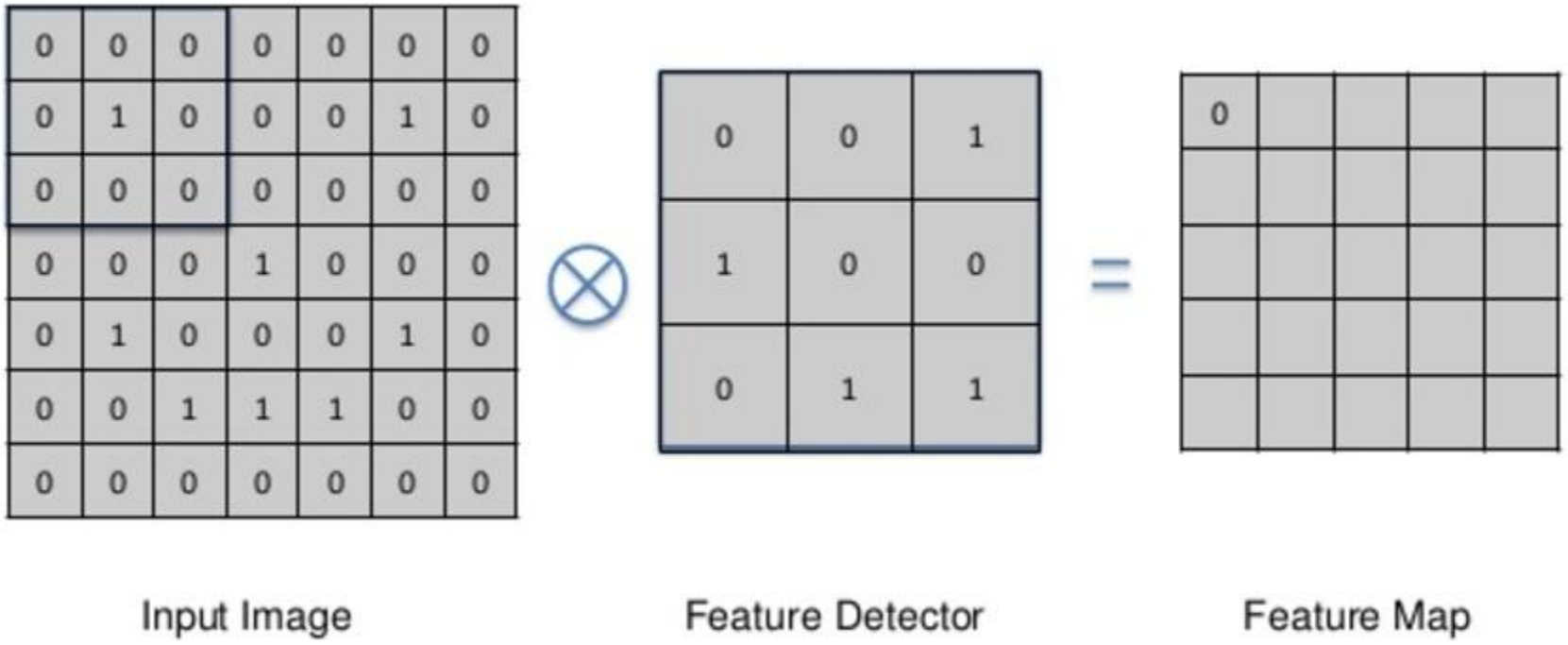
Input Image

0	0	1
1	0	0
0	1	1

Feature Detector

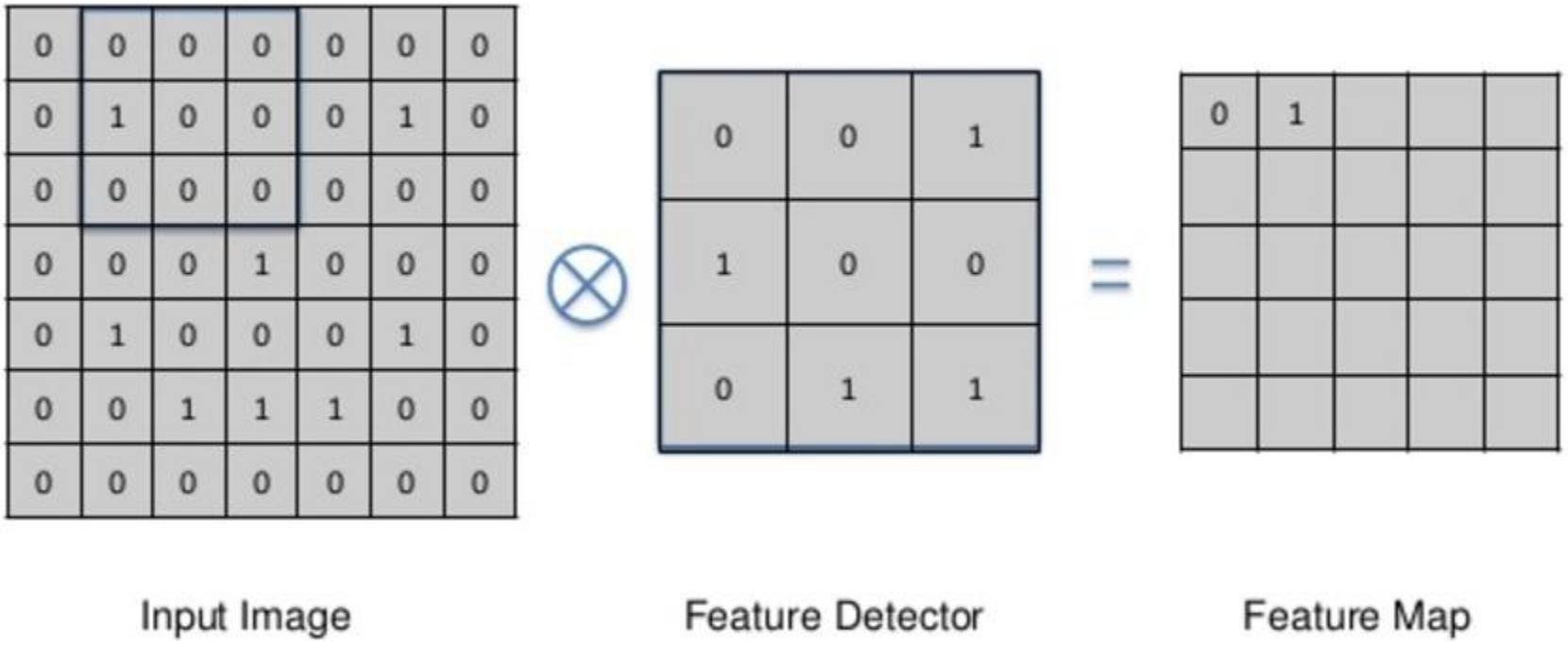
source: superdatascience

CNN Step 1: Convolution



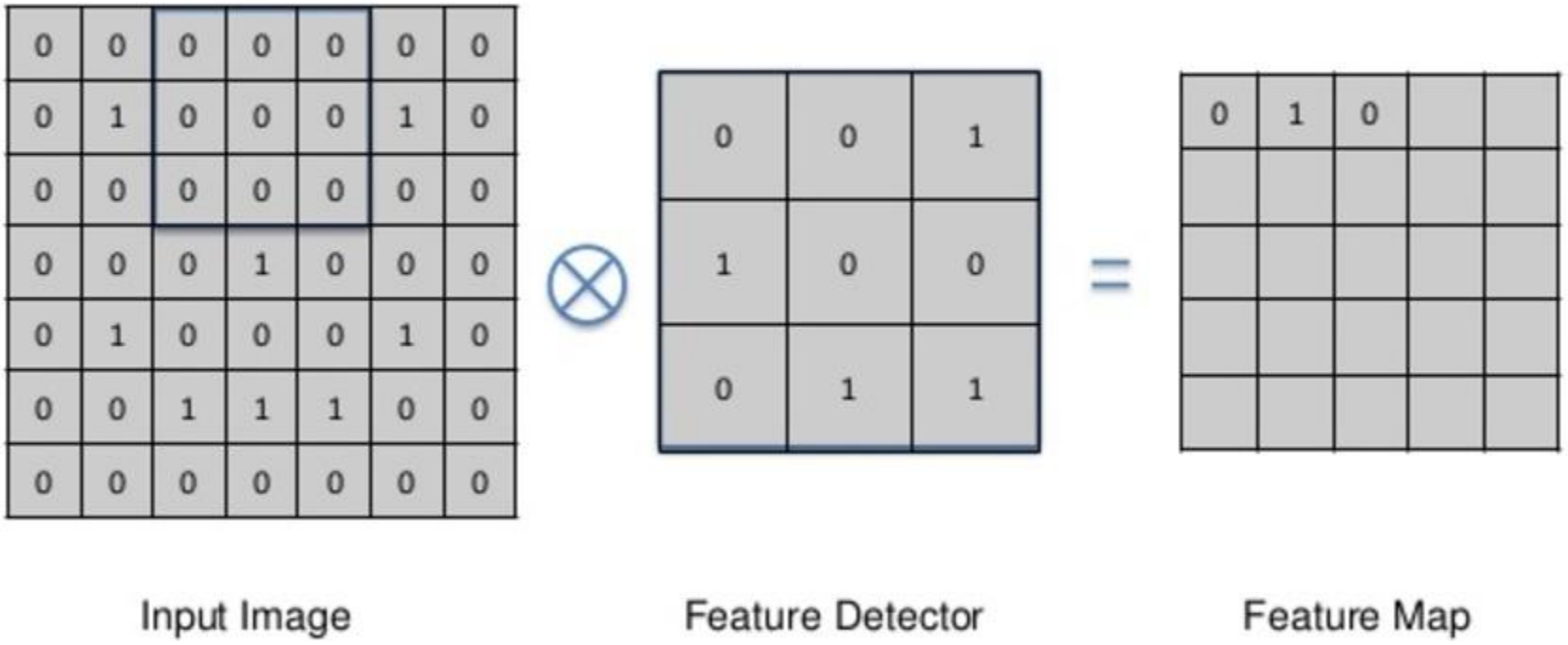
source: superdatascience

CNN Step 1: Convolution



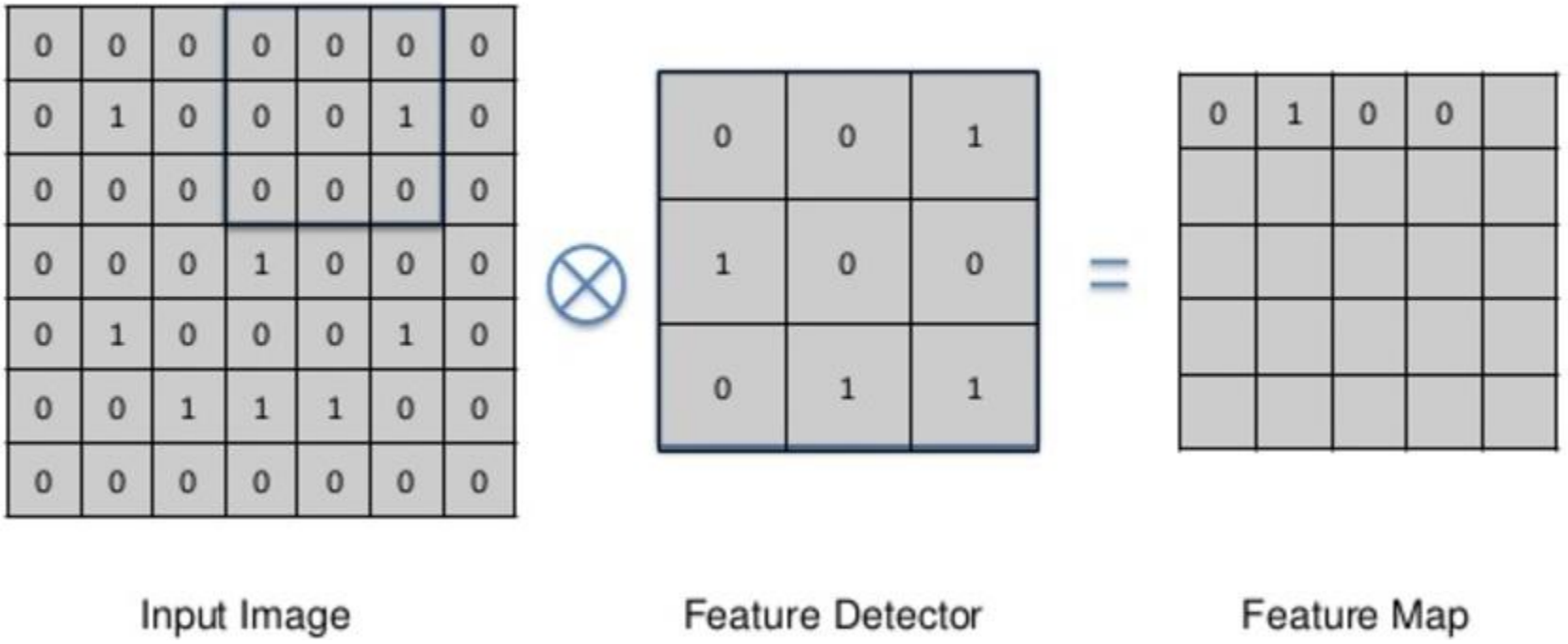
source: superdatascience

CNN Step 1: Convolution



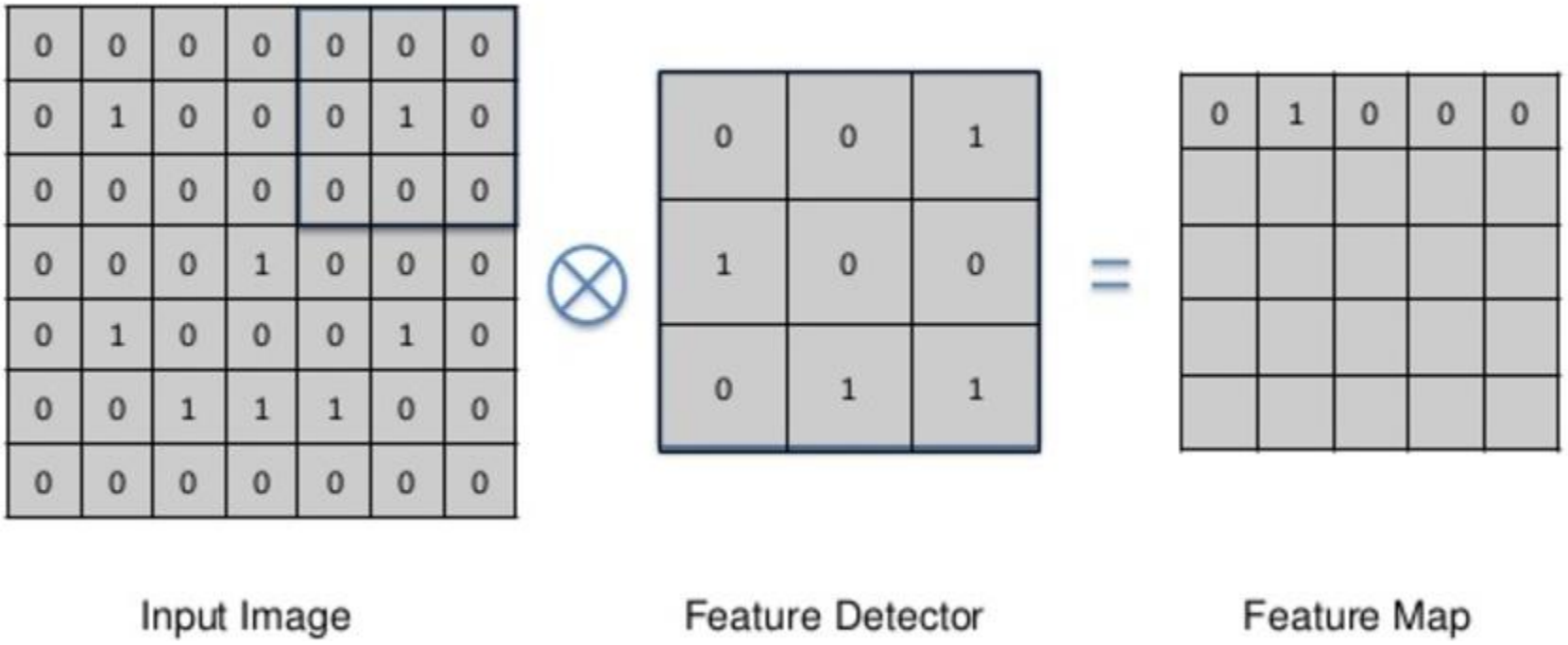
source: superdatascience

CNN Step 1: Convolution



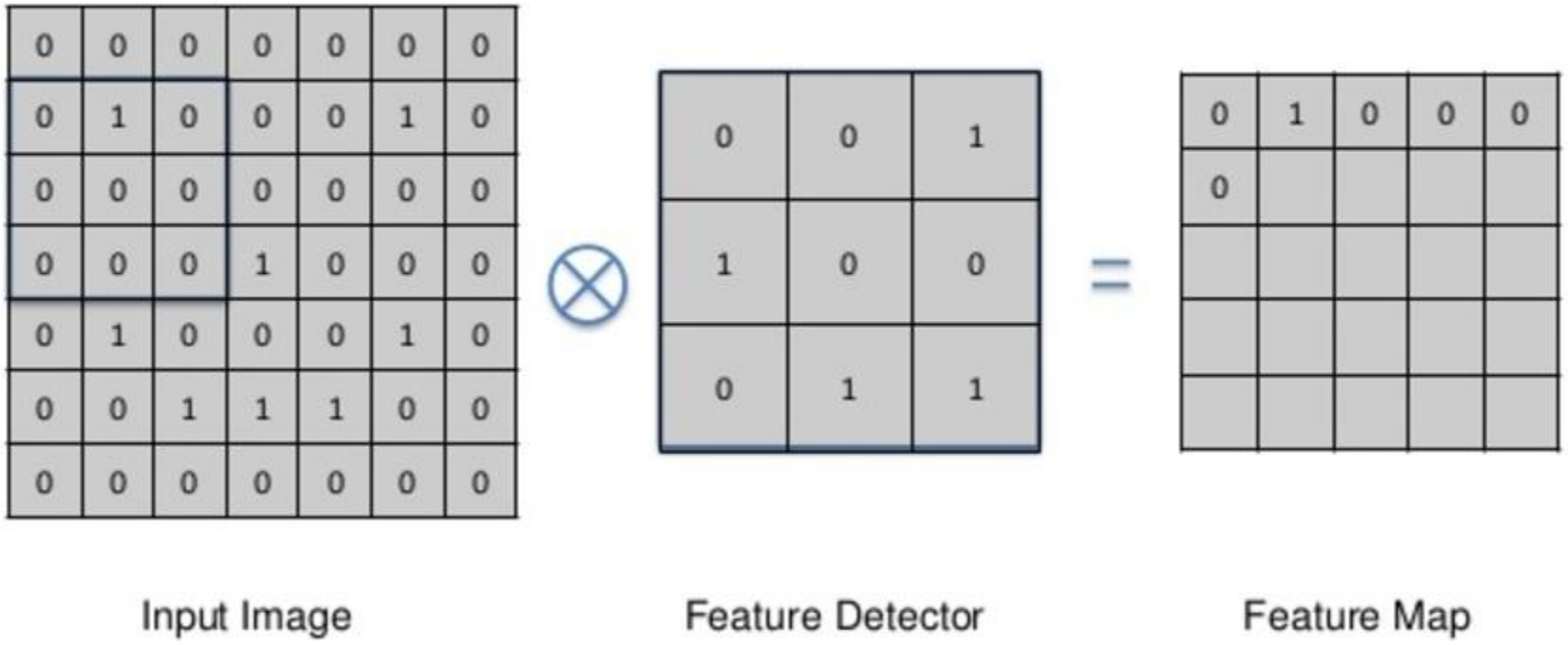
source: superdatascience

Convolution



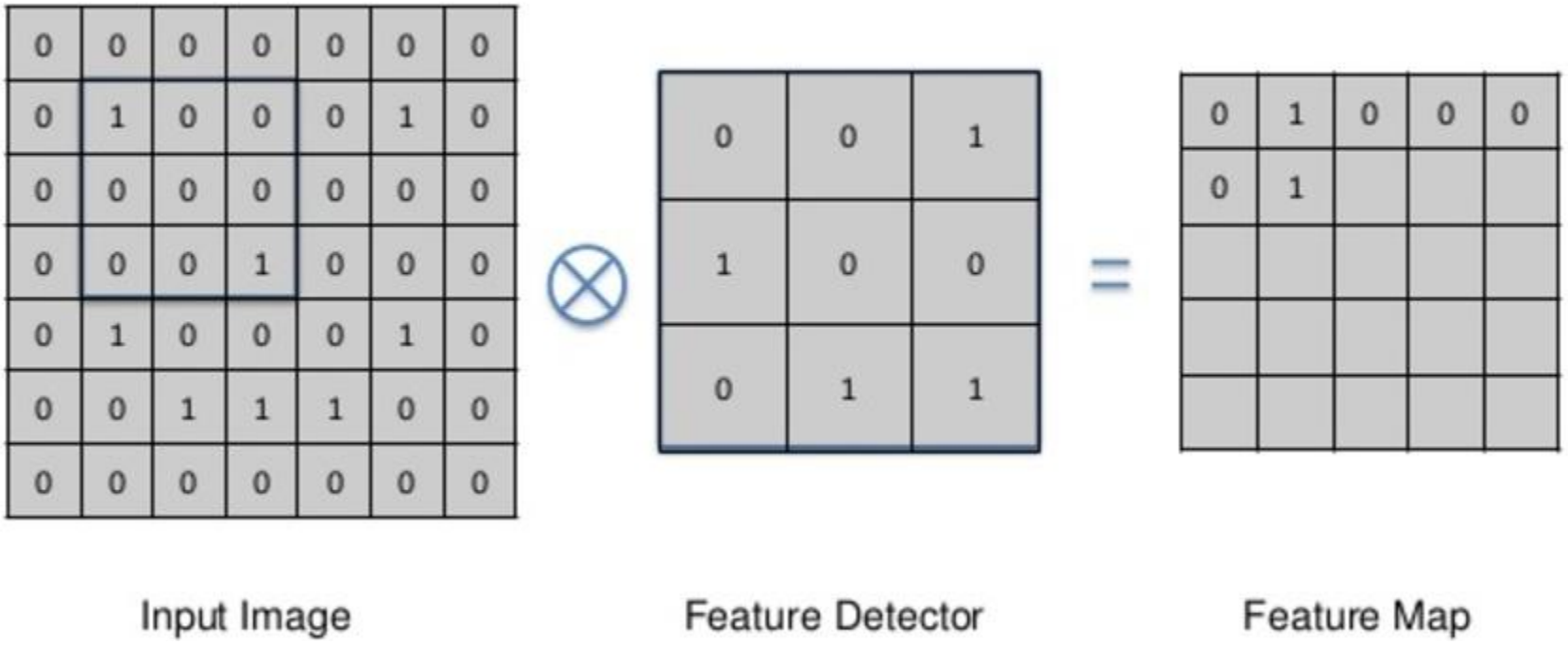
source: superdatascience

CNN Step 1: Convolution



source: superdatascience

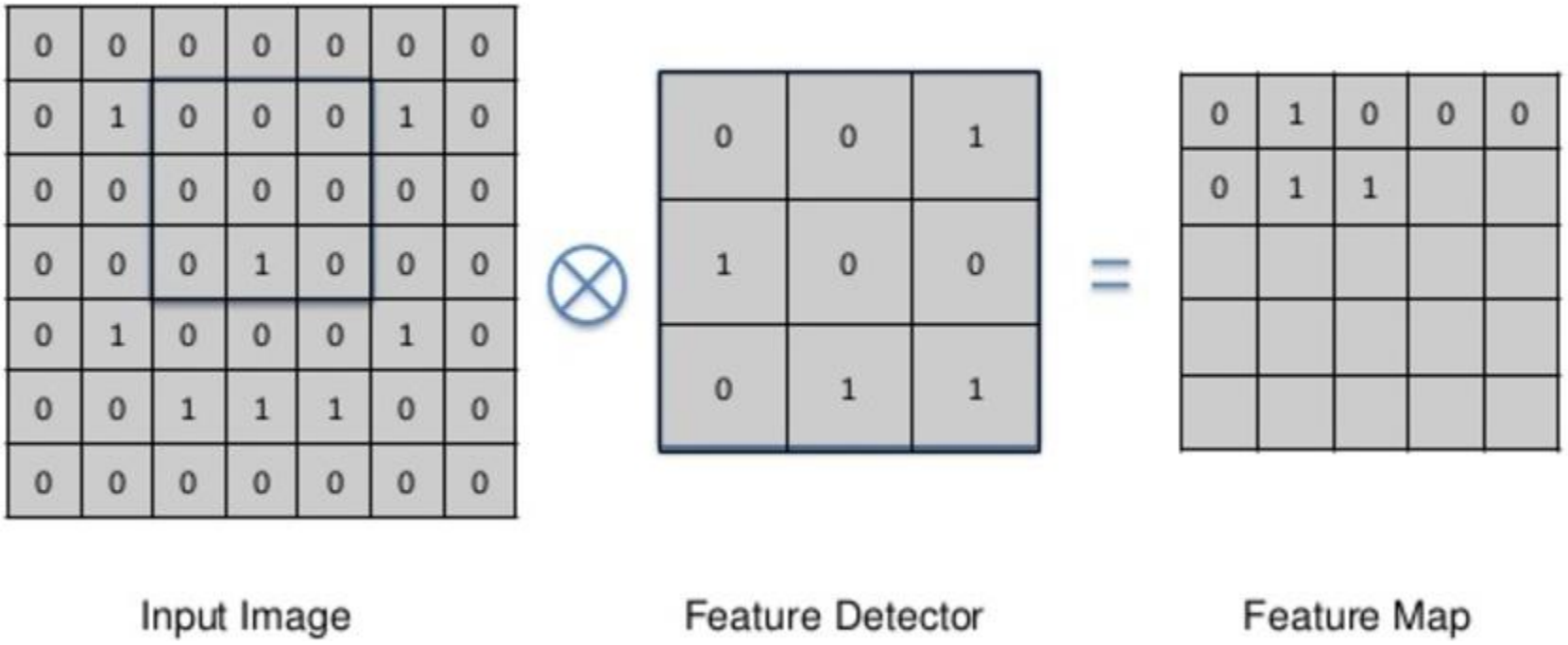
CNN Step 1: Convolution



source: superdatascience

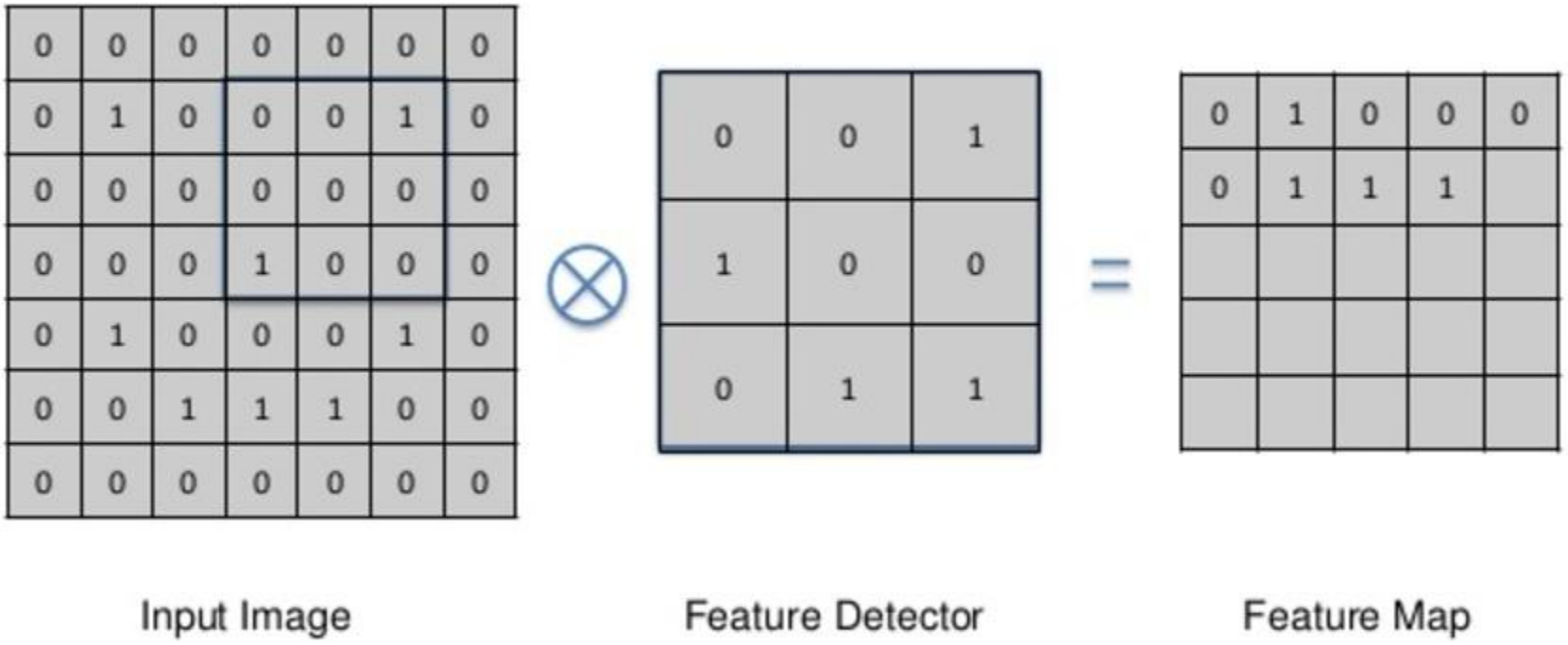
CNN Step 1: Convolution

Mondragon Unibertsitatea
Faculty of Engineering



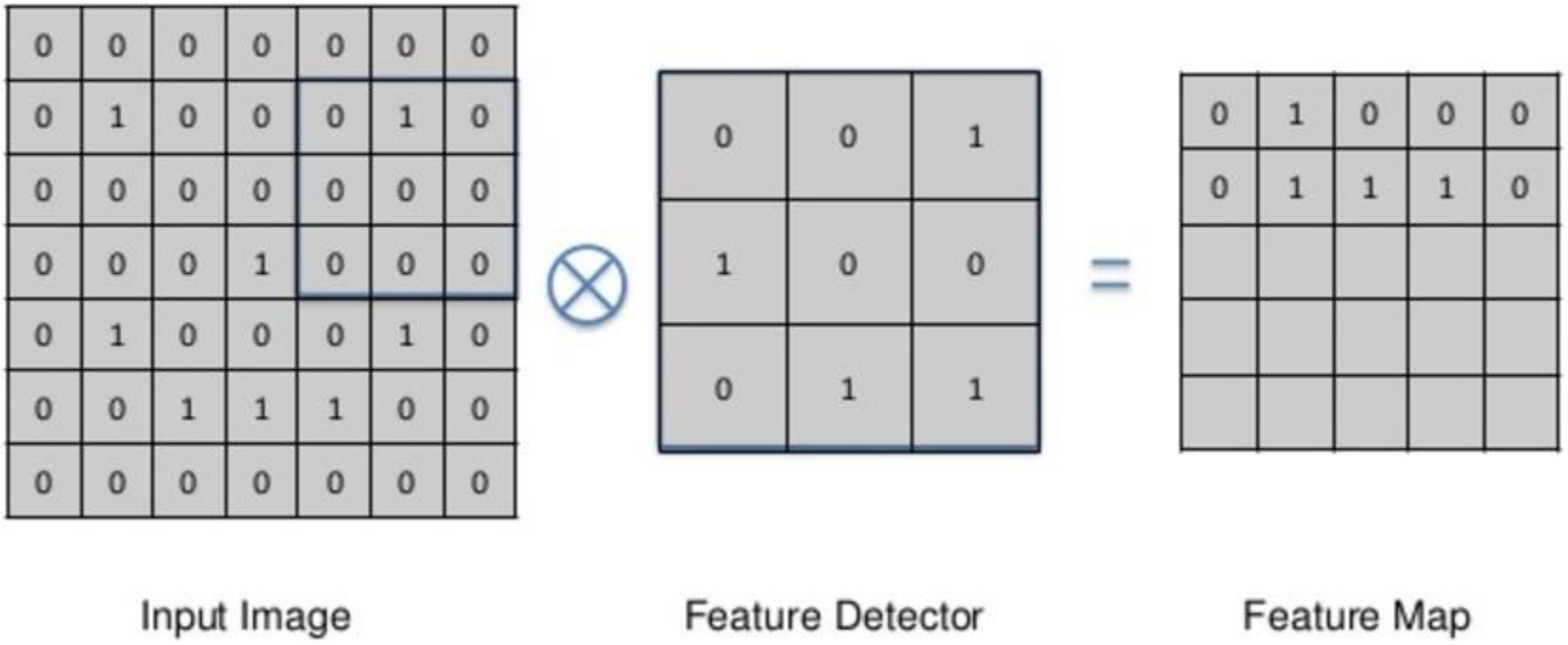
source: superdatascience

CNN Step 1: Convolution



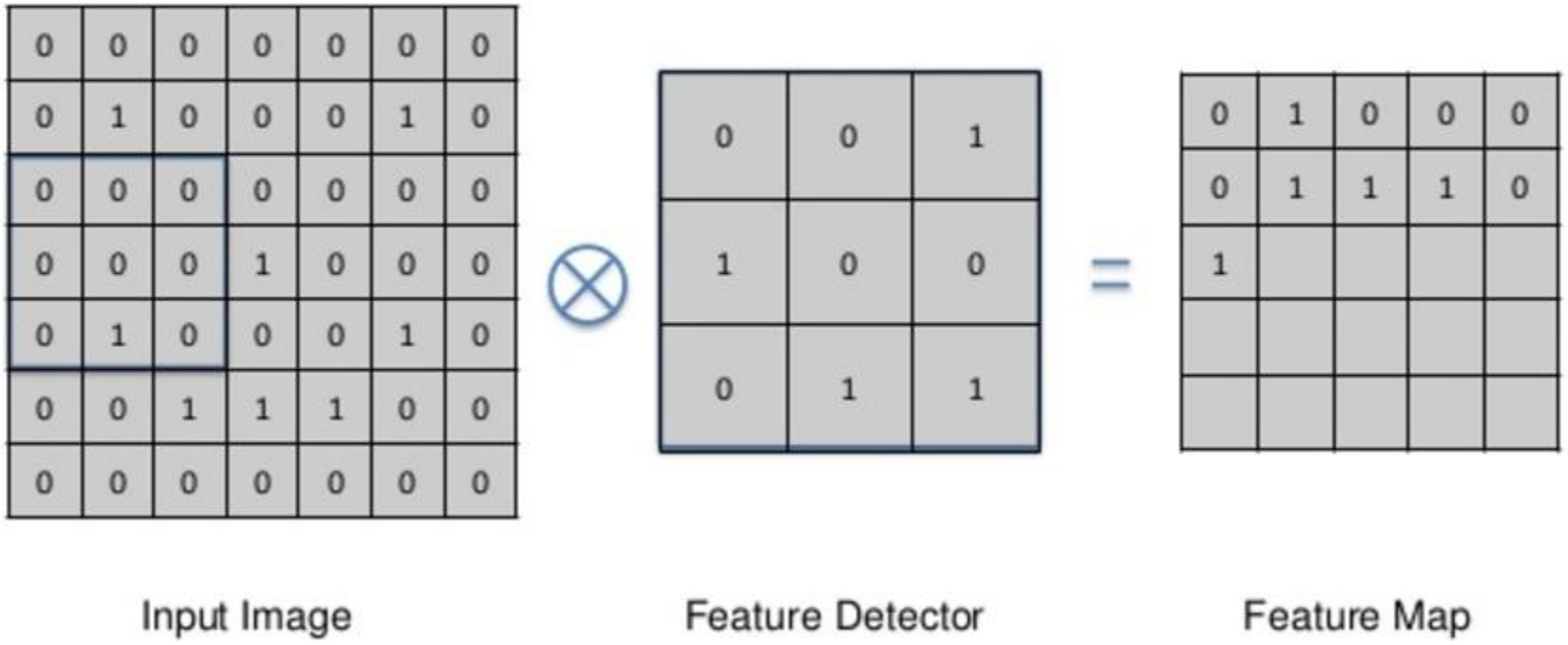
source: superdatascience

CNN Step 1: Convolution



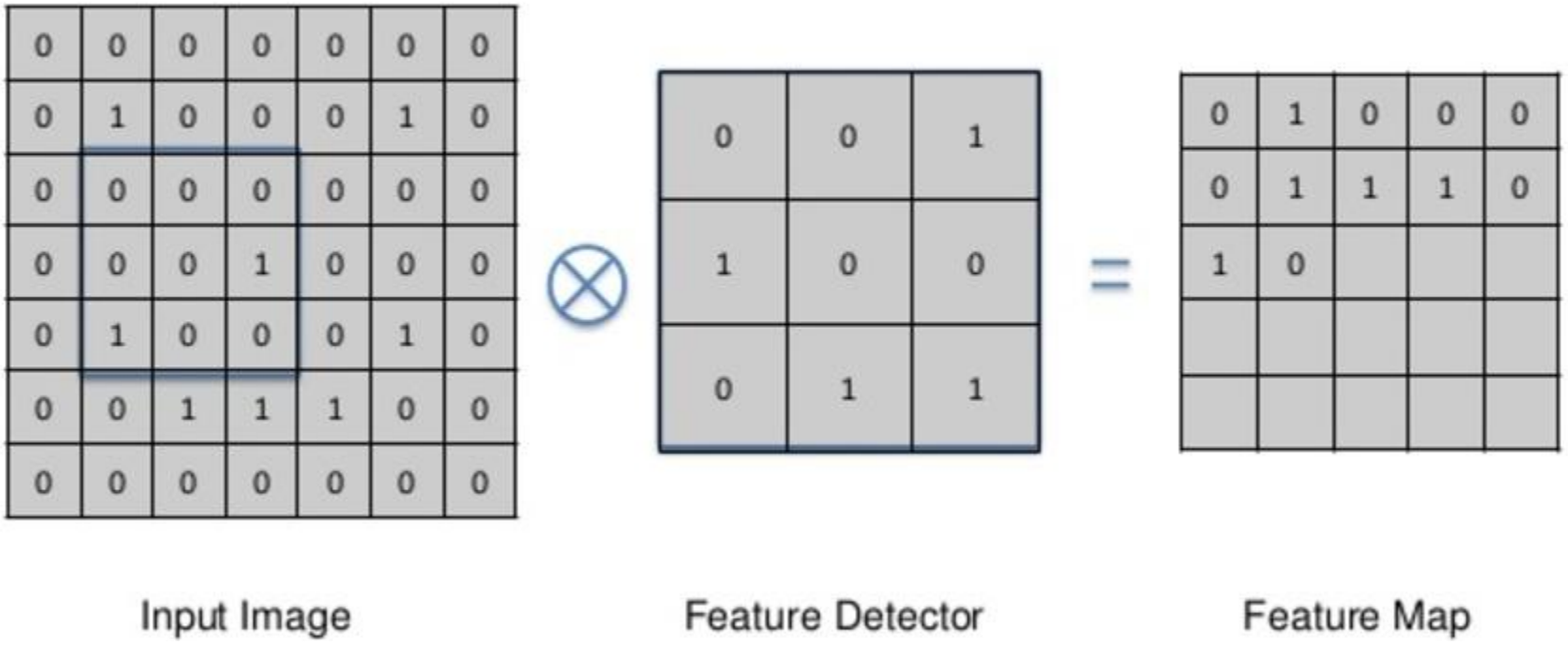
source: superdatascience

CNN Step 1: Convolution



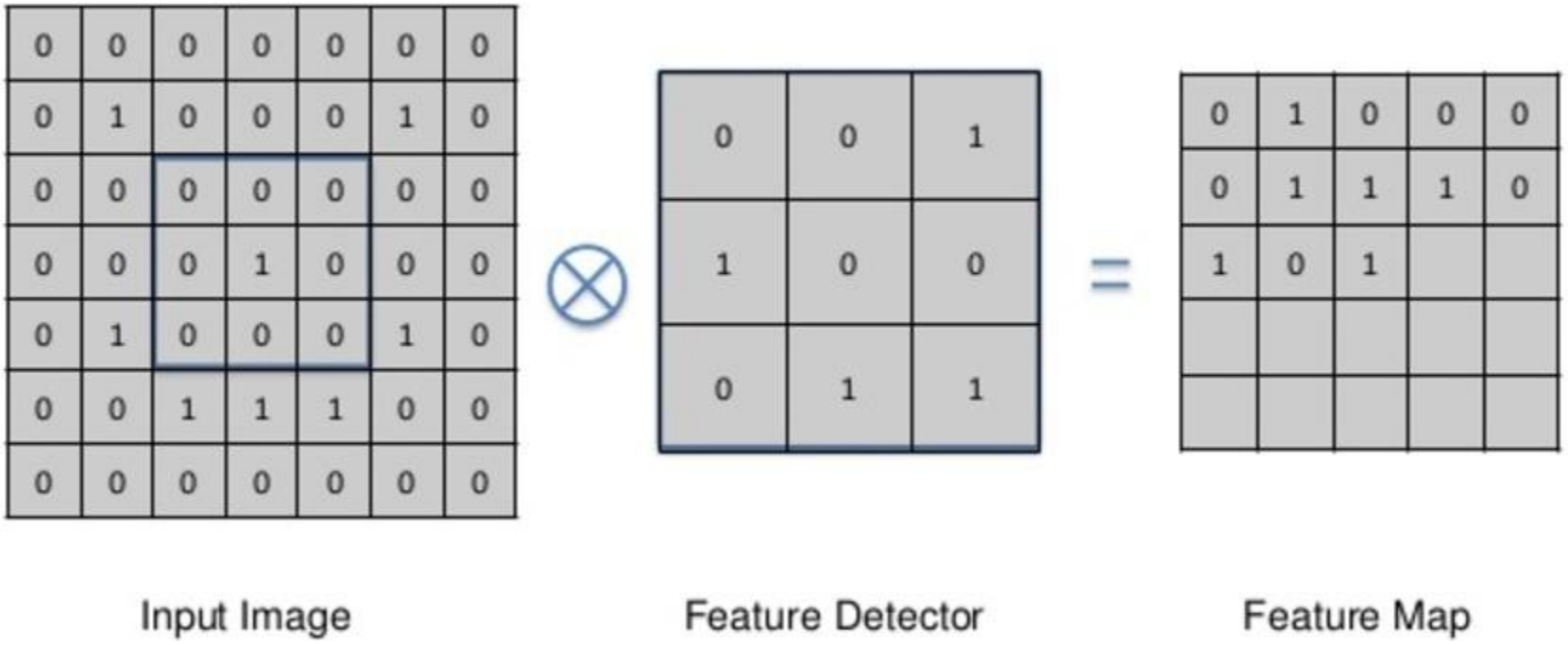
source: superdatascience

CNN Step 1: Convolution



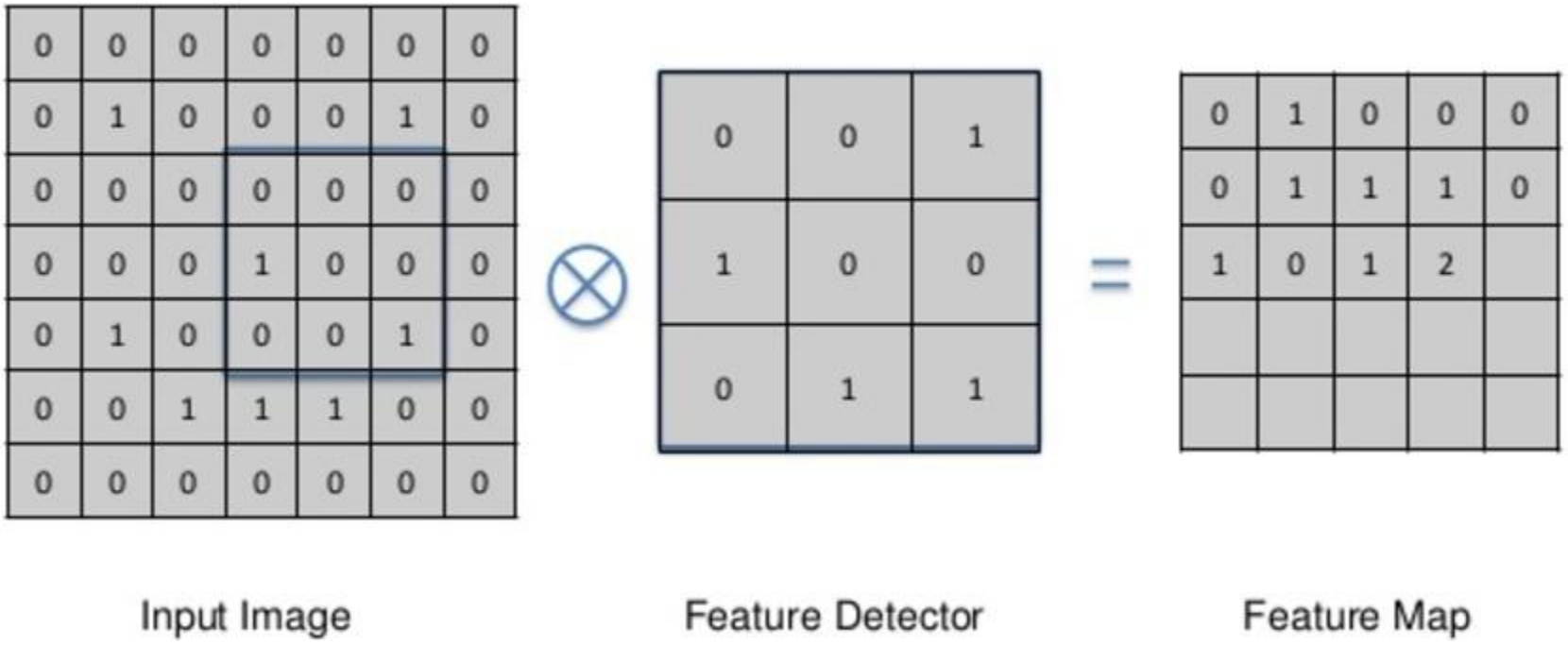
source: superdatascience

CNN Step 1: Convolution



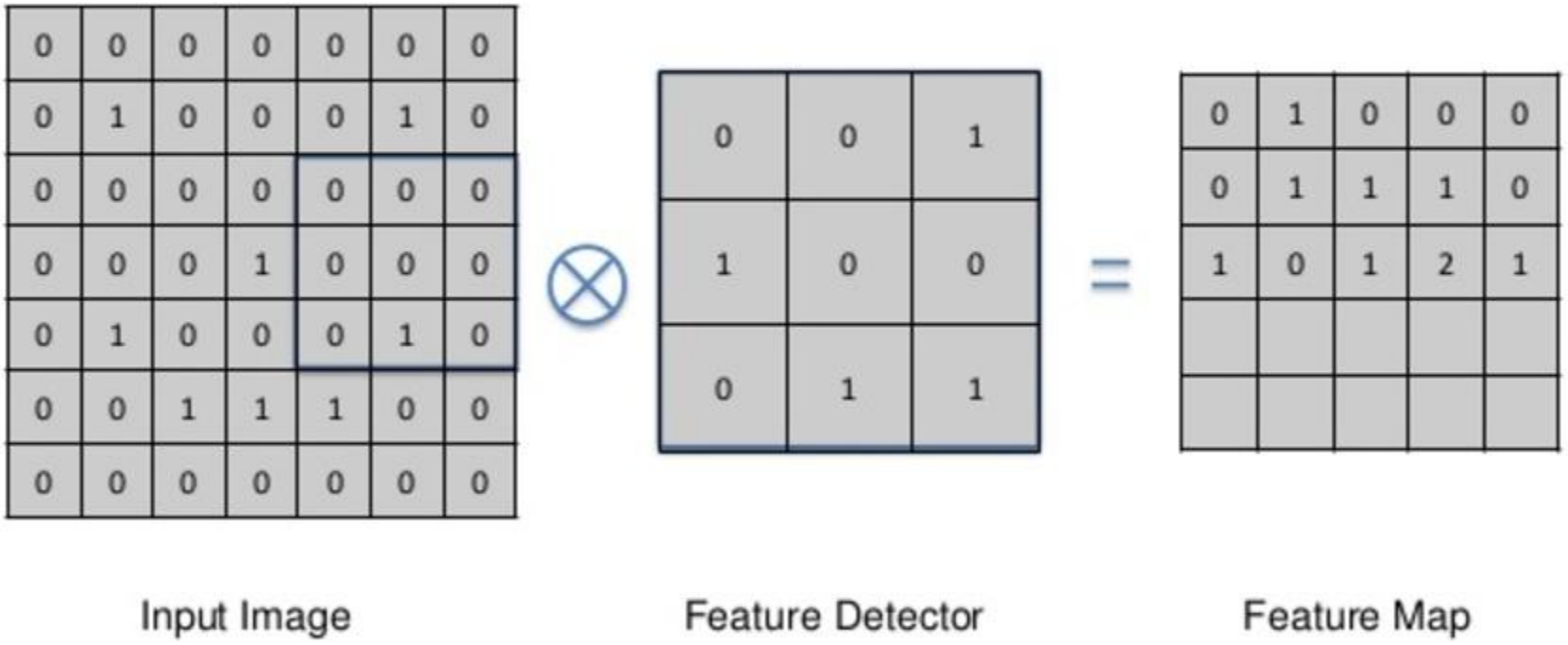
source: superdatascience

CNN Step 1: Convolution



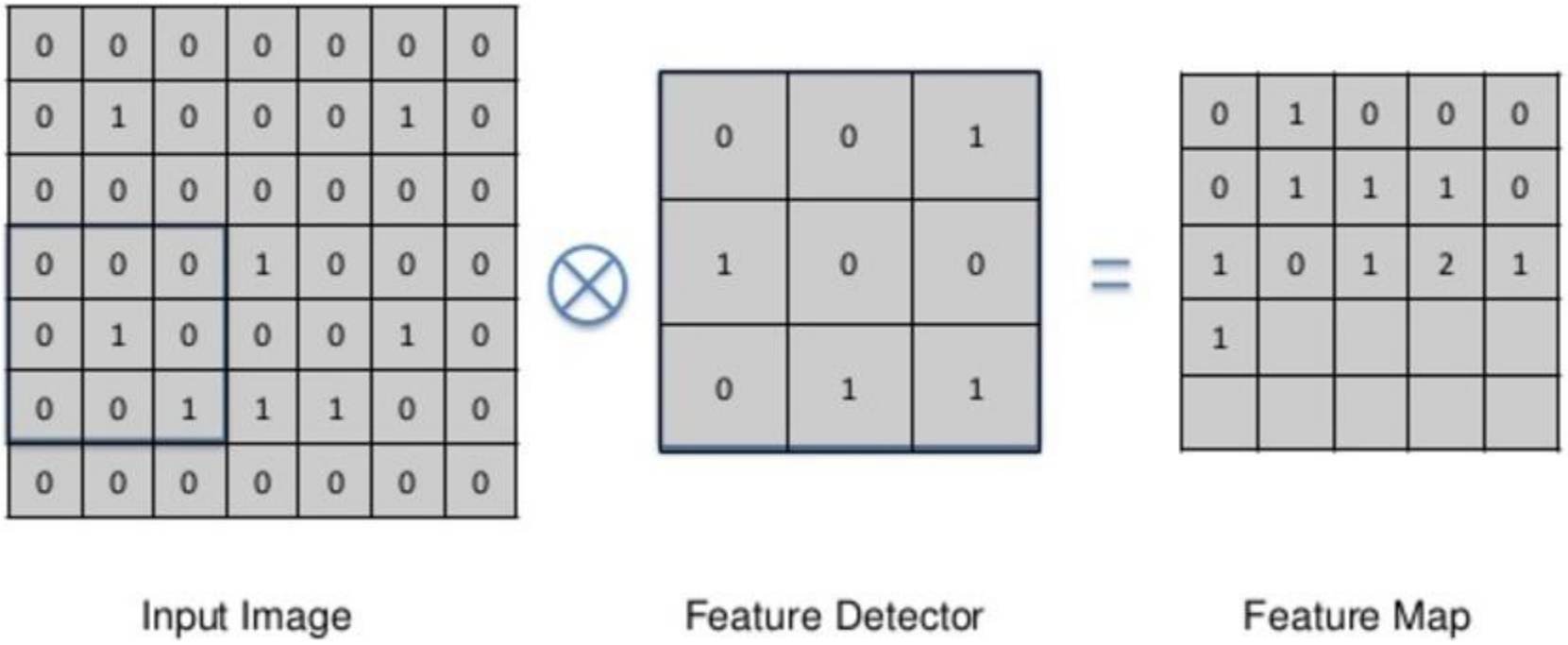
source: superdatascience

CNN Step 1: Convolution



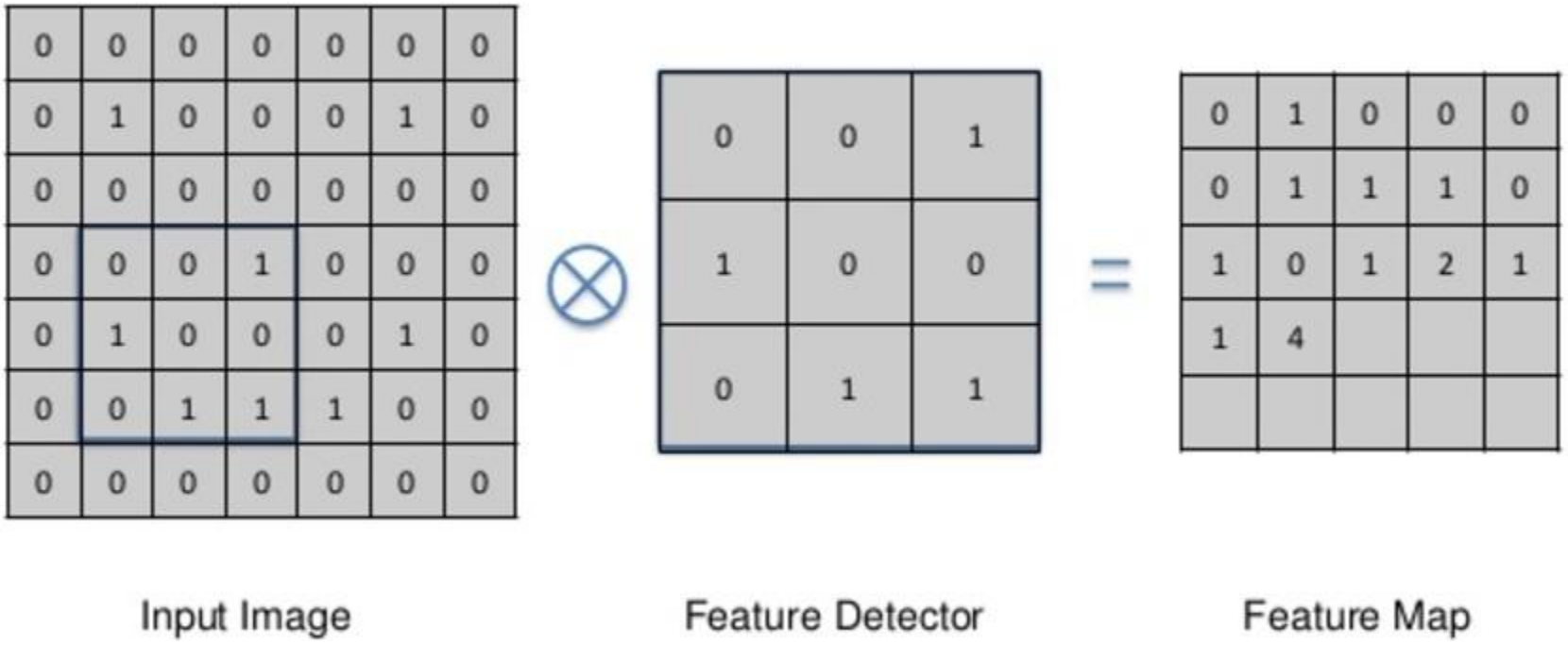
source: superdatascience

Convolution



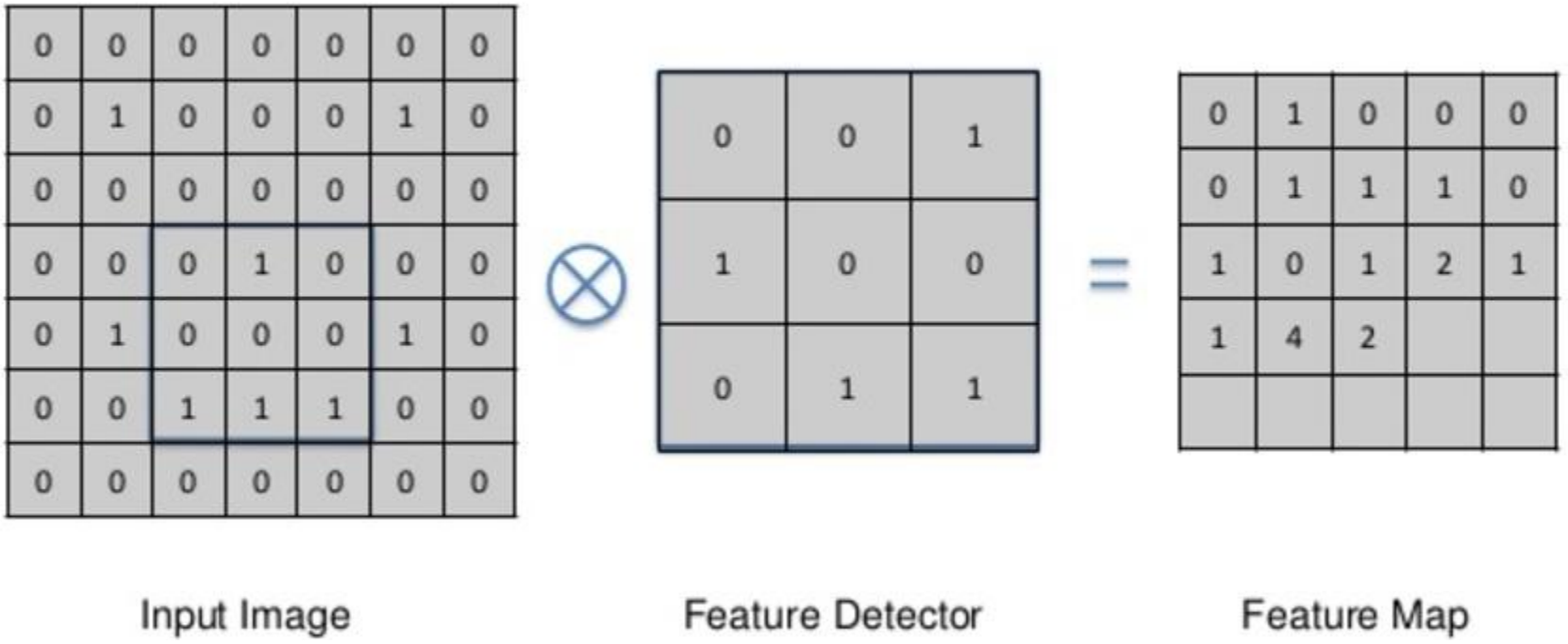
source: superdatascience

CNN Step 1: Convolution



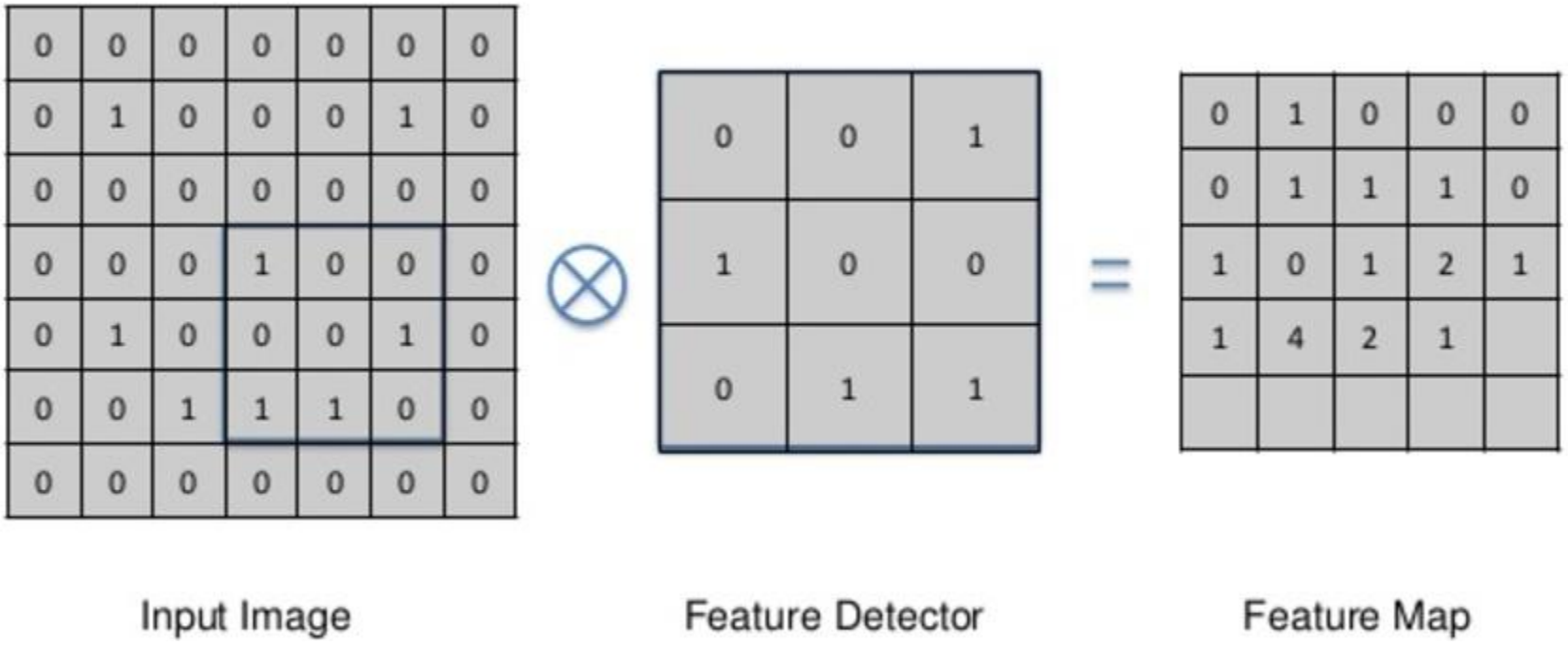
source: superdatascience

CNN Step 1: Convolution



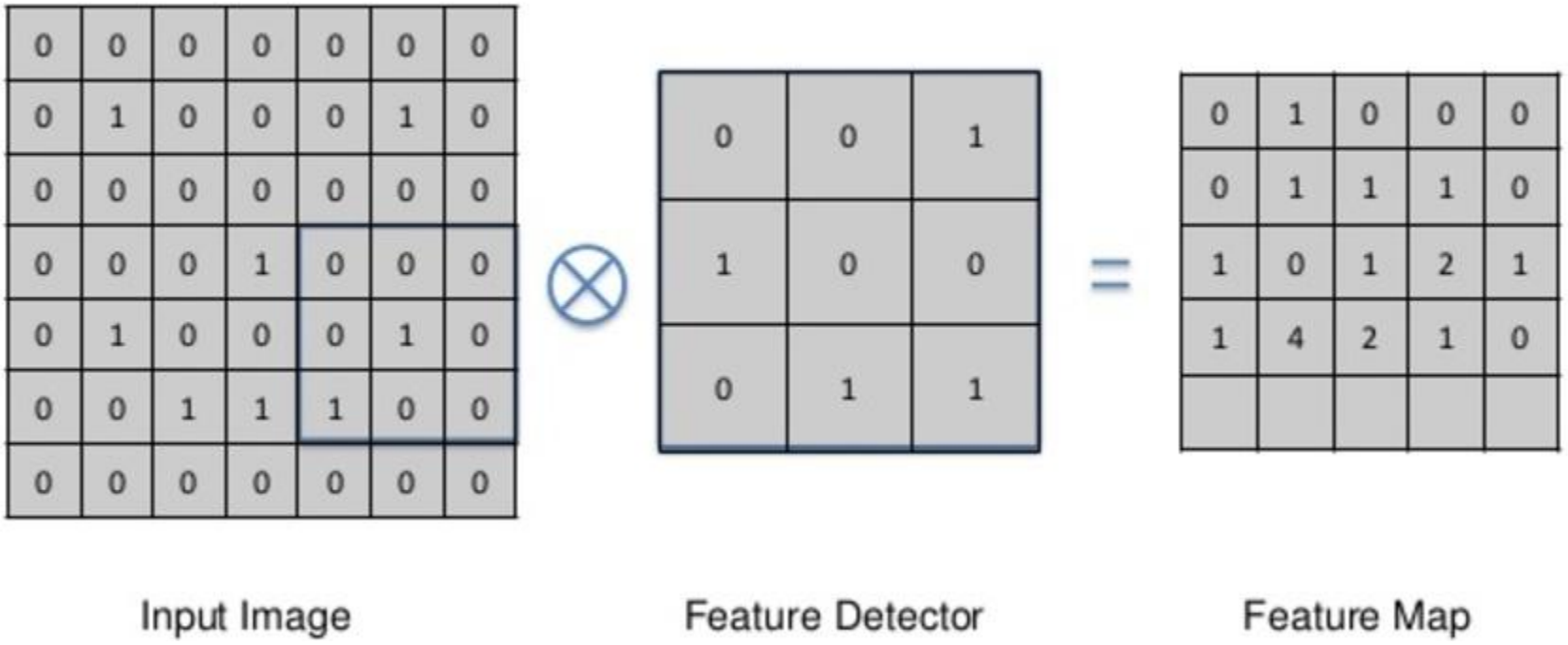
source: superdatascience

CNN Step 1: Convolution



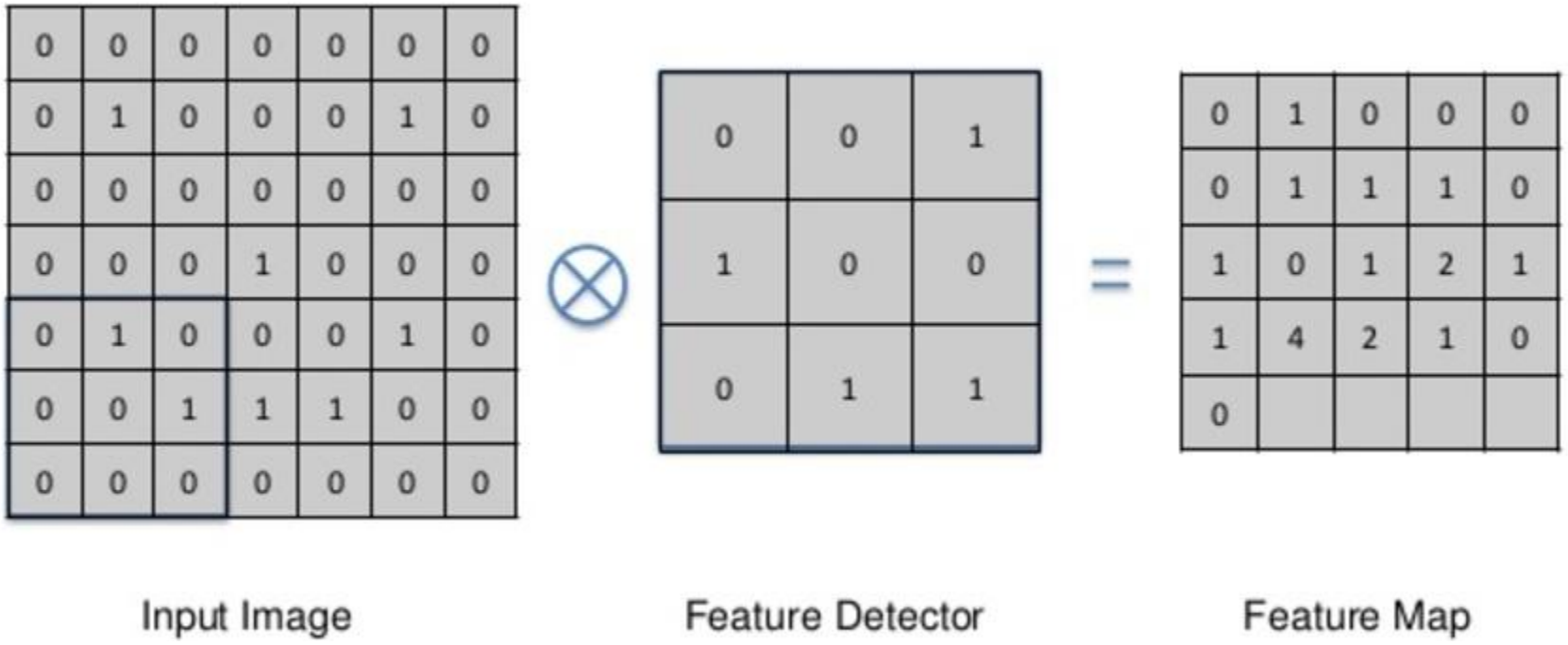
source: superdatascience

CNN Step 1: Convolution



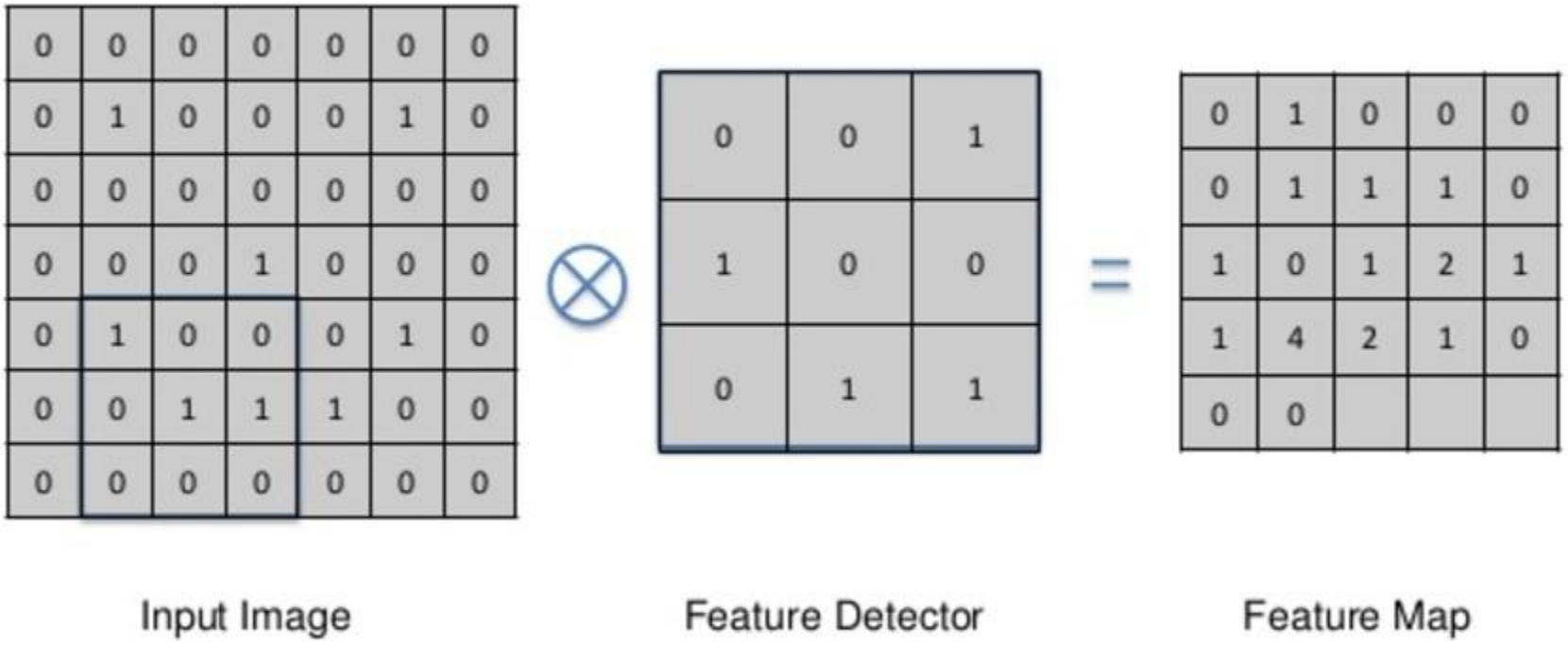
source: superdatascience

CNN Step 1: Convolution



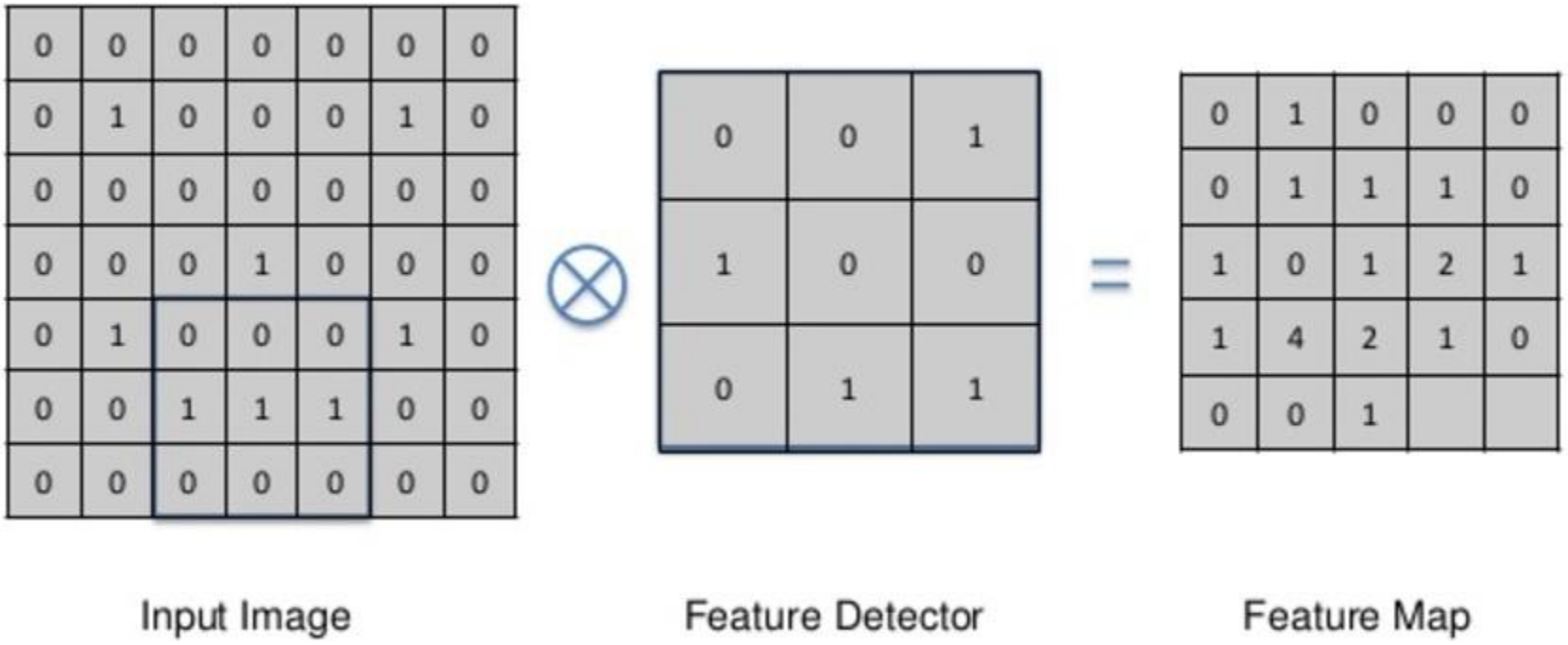
source: superdatascience

CNN Step 1: Convolution



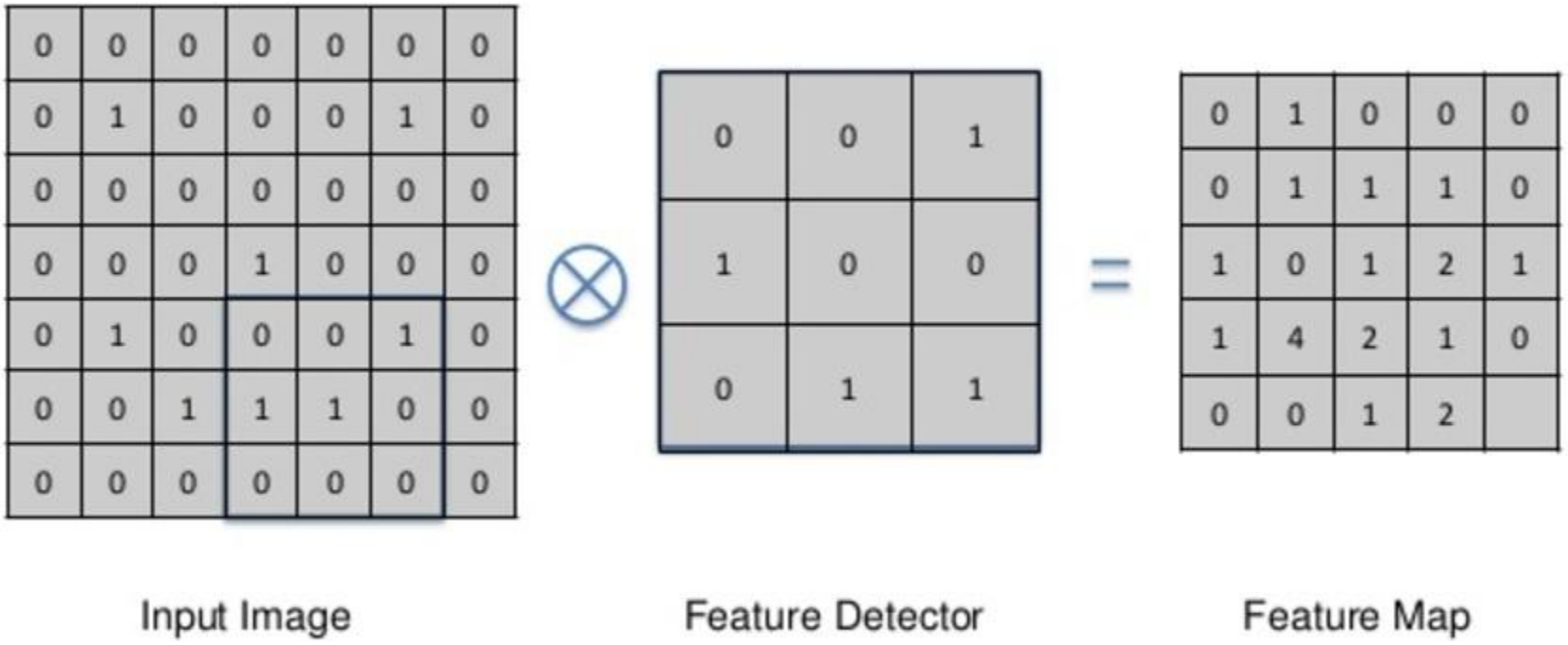
source: superdatascience

CNN Step 1: Convolution



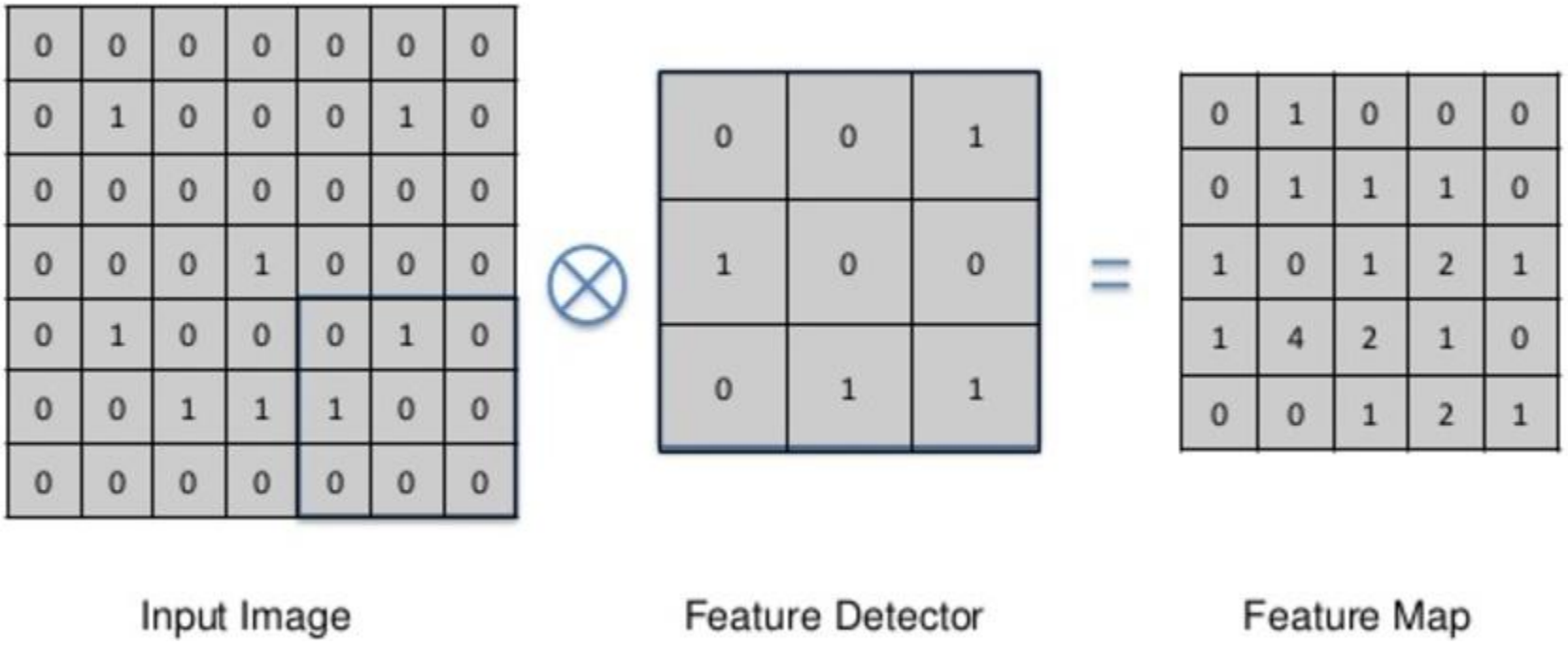
source: superdatascience

CNN Step 1: Convolution



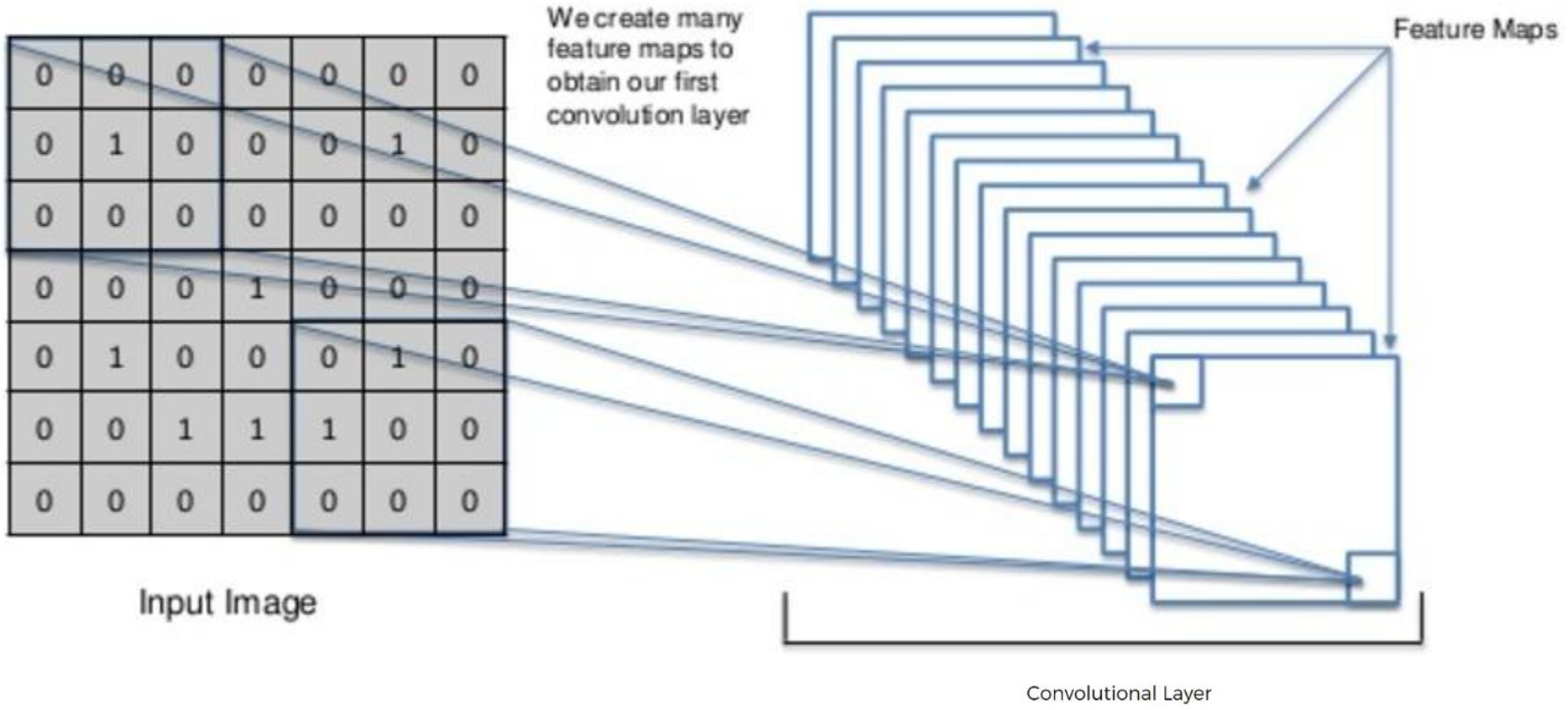
source: superdatascience

CNN Step 1: Convolution



source: superdatascience

CNN Step 1: Convolution



source: superdatascience

Convolution result examples

Mondragon Unibertsitatea
Faculty of Engineering

Sharpen

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

source: GIMP documentation

Convolution result examples

Mondragon Unibertsitatea
Faculty of Engineering

Blur

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

source: GIMP documentation

Convolution result examples

Mondragon Unibertsitatea
Faculty of Engineering

Edge Detect

	0	1	0	
	1	-4	1	
	0	1	0	

source: GIMP documentation

Convolution result examples

Mondragon Unibertsitatea
Faculty of Engineering

Emboss

	-2	-1	0	
	-1	1	1	
	0	1	2	

source: GIMP documentation

color images convolution

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				...
				...
				...
				...
...	...	...	...	...

source: stack Overflow

CNN Step 1: Convolution

Mondragon Unibertsitatea
Faculty of Engineering

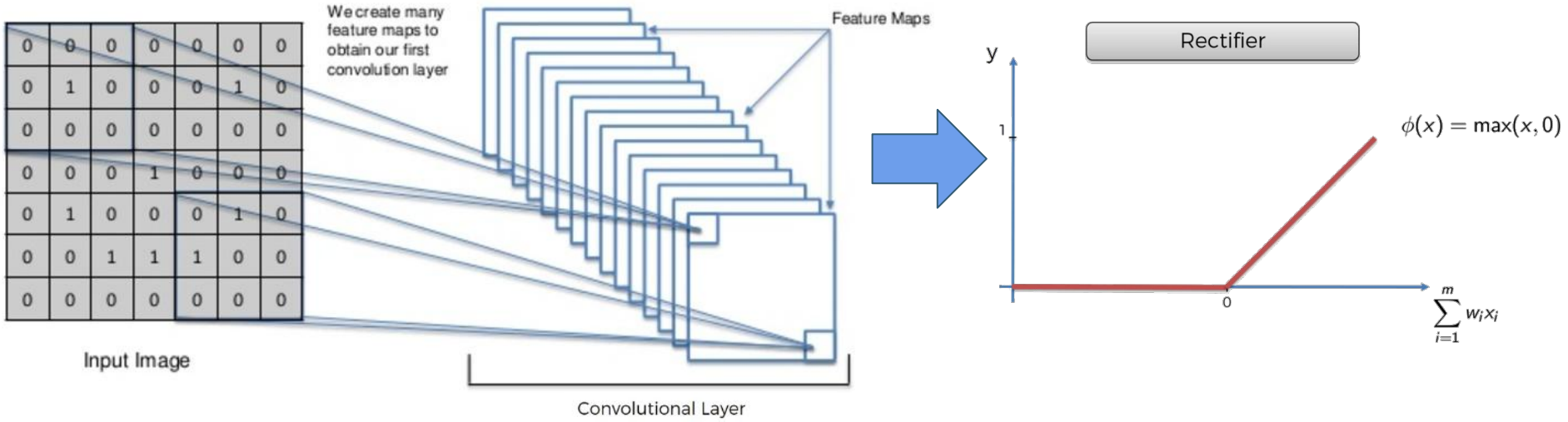
Searching for features in the images, using a feature detector, and putting them on a feature map that still preserves the spatial relationships between pixels.

Many times, we won't know what these characteristics mean, but they work.

source: superdatascience

Step 1(b)
ReLU Activation

Step 1b: ReLU layer



We add NonLinearity to the output

source: superdatascience

Step 2
Max Pooling

Step 2: Pooling

Step 2: Pooling

Mondragon Unibertsitatea
Faculty of Engineering

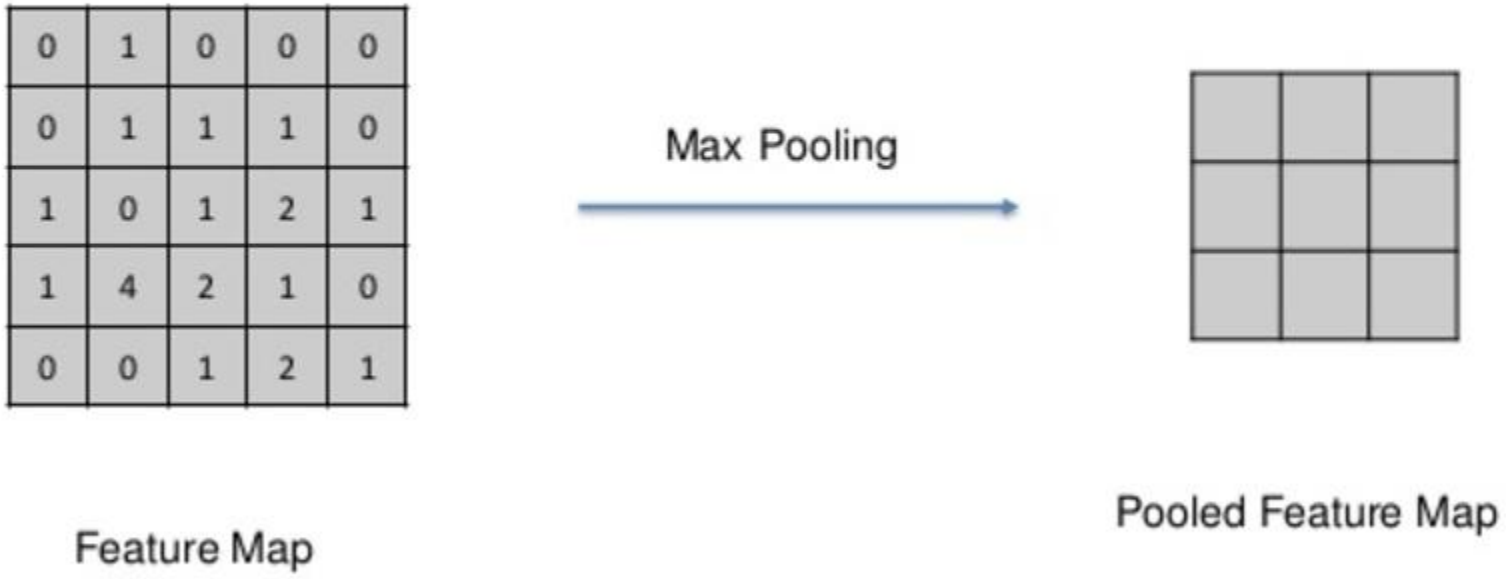
Step 2: MAX Pooling

Mondragon Unibertsitatea
Faculty of Engineering

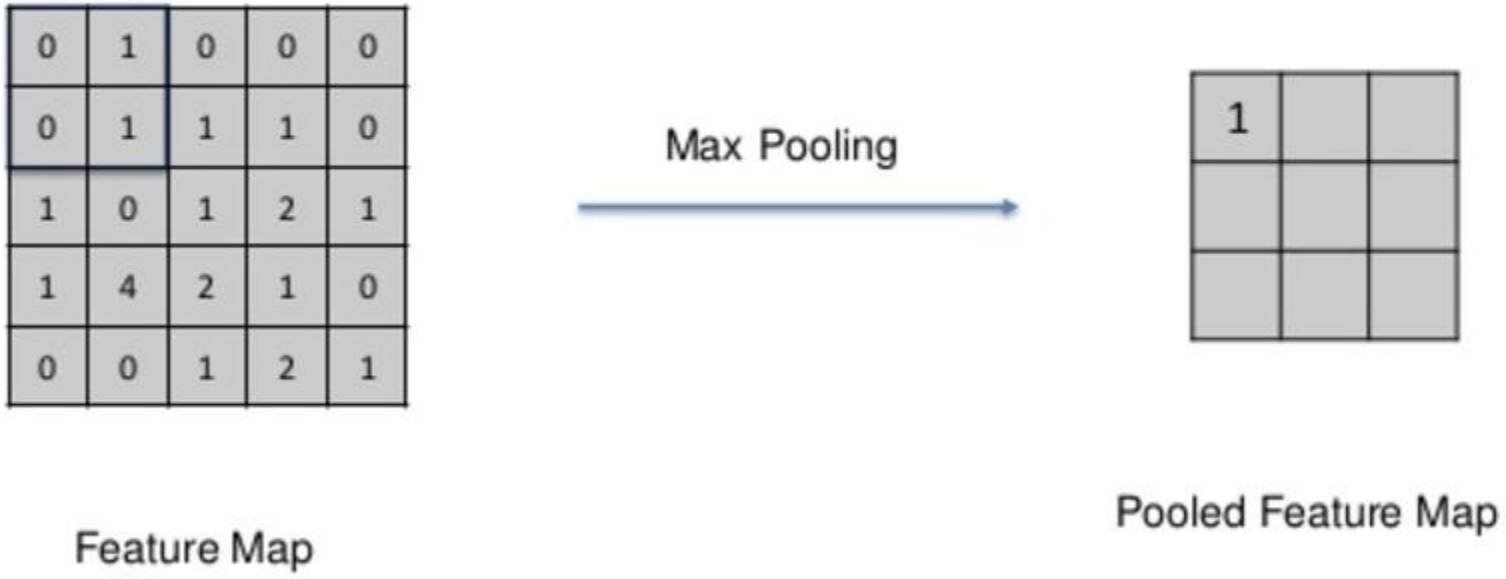
0	1	0	0	0
0	1	1	1	0
1	0	1	2	1
1	4	2	1	0
0	0	1	2	1

Feature Map

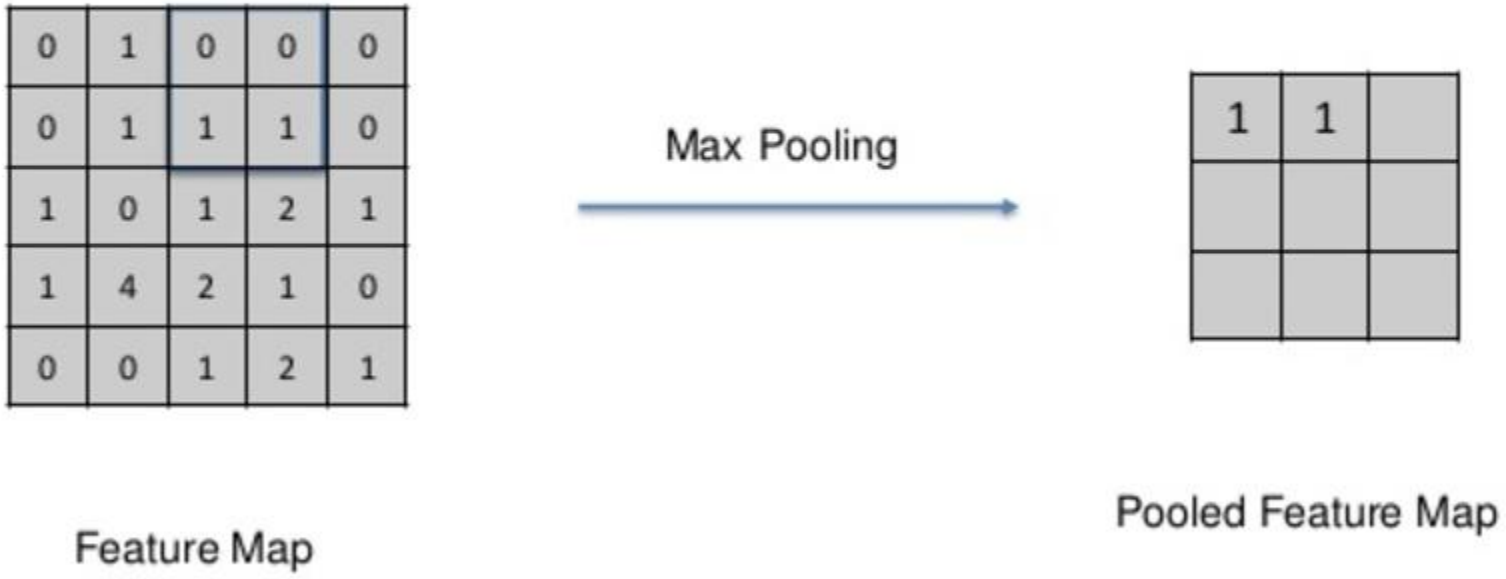
Step 2: MAX Pooling



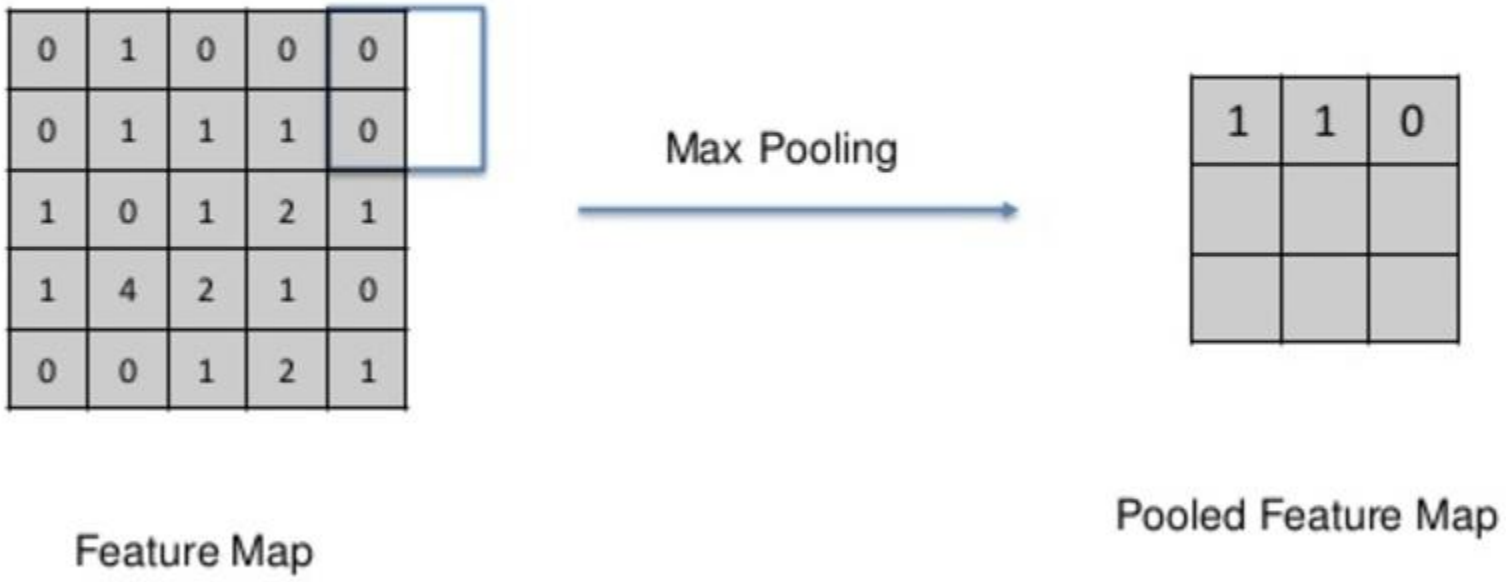
Step 2: MAX Pooling



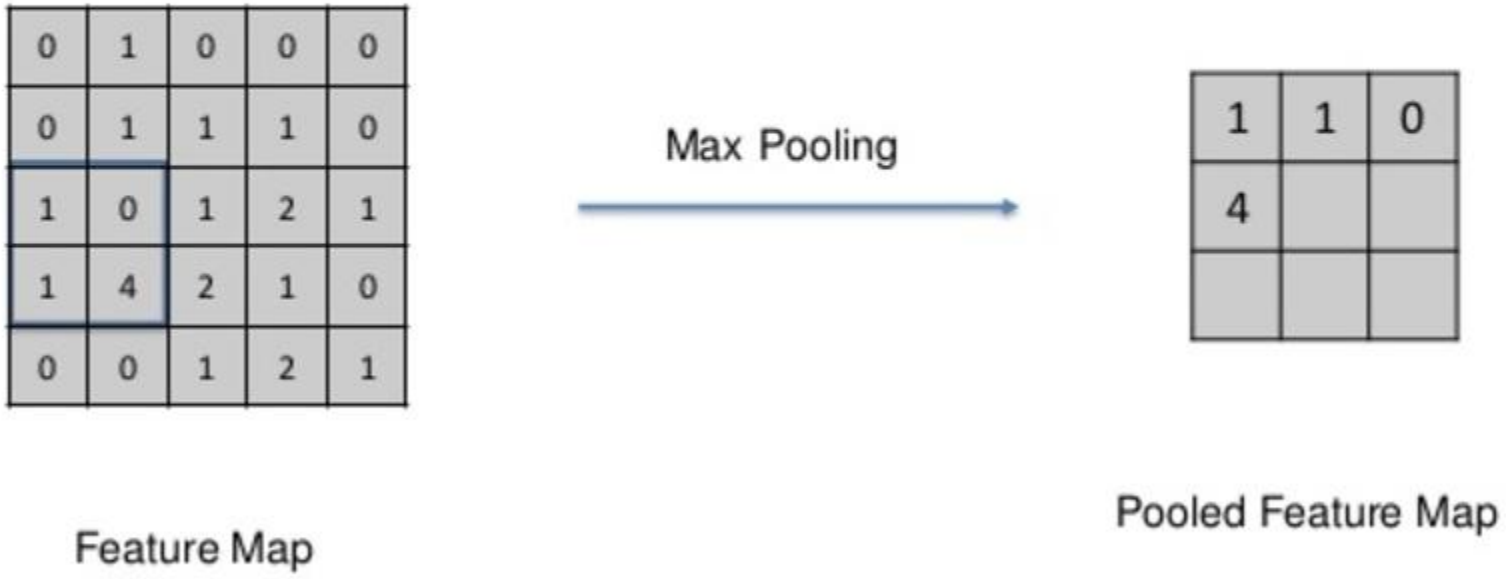
Step 2: MAX Pooling



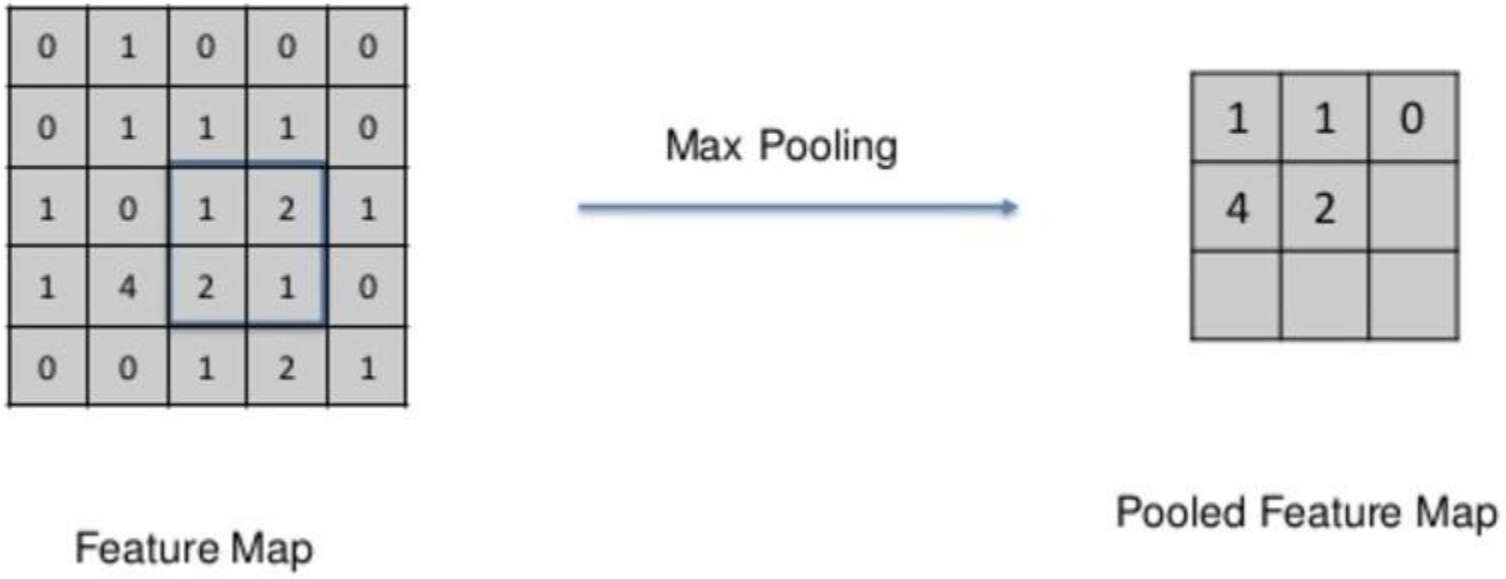
Step 2: MAX Pooling



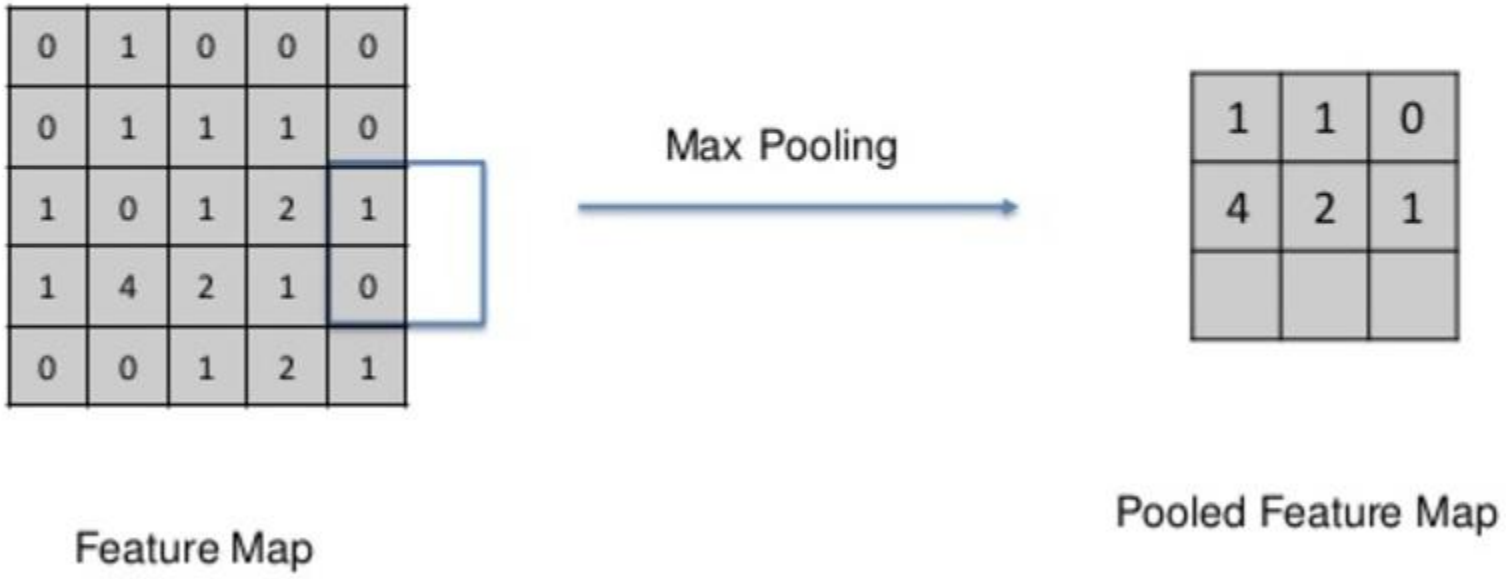
Step 2: MAX Pooling



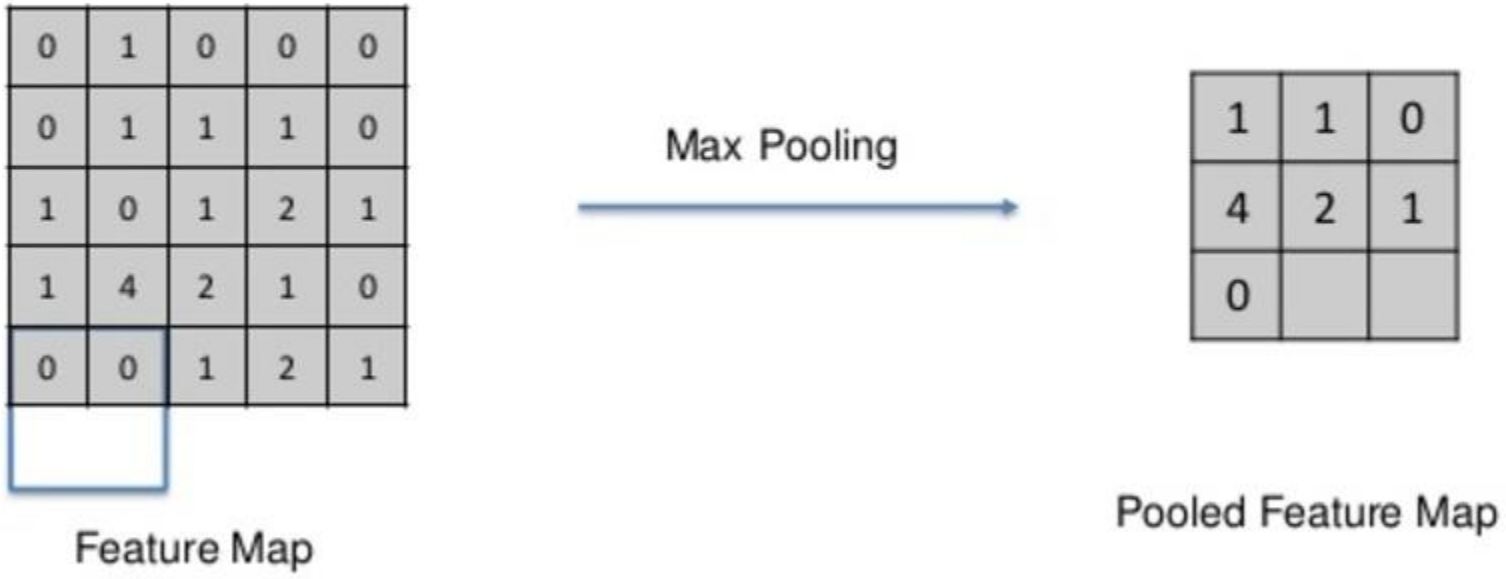
Step 2: MAX Pooling



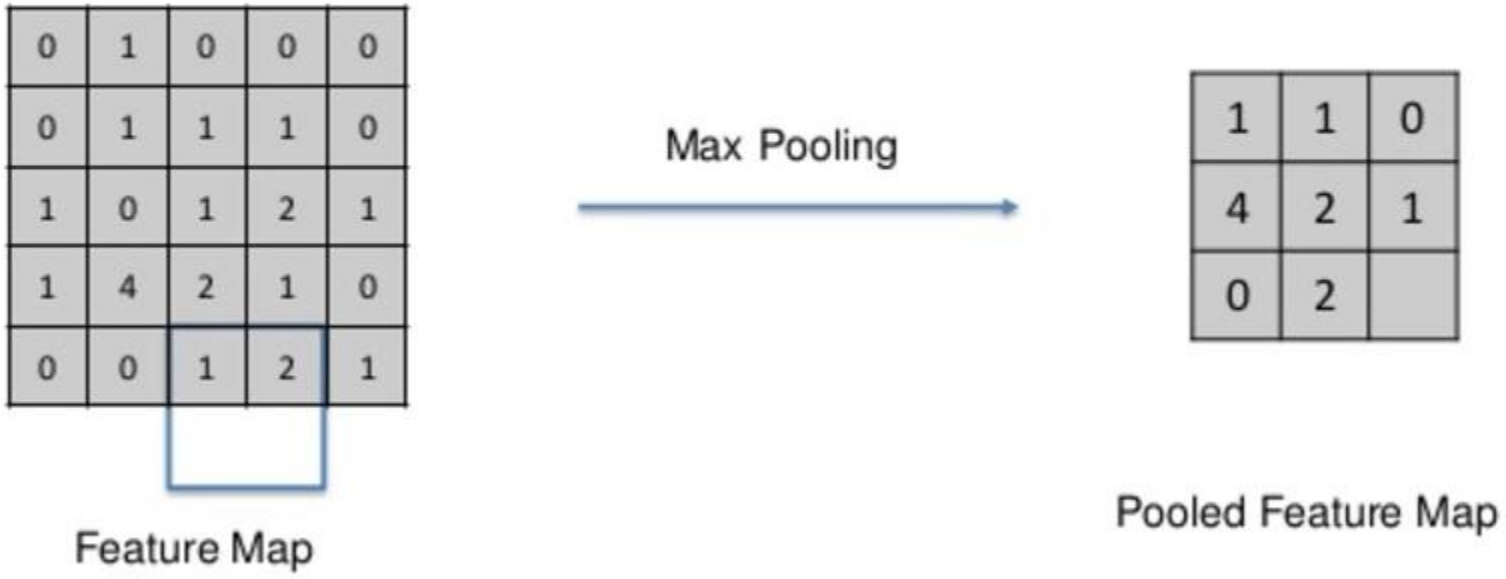
Step 2: MAX Pooling



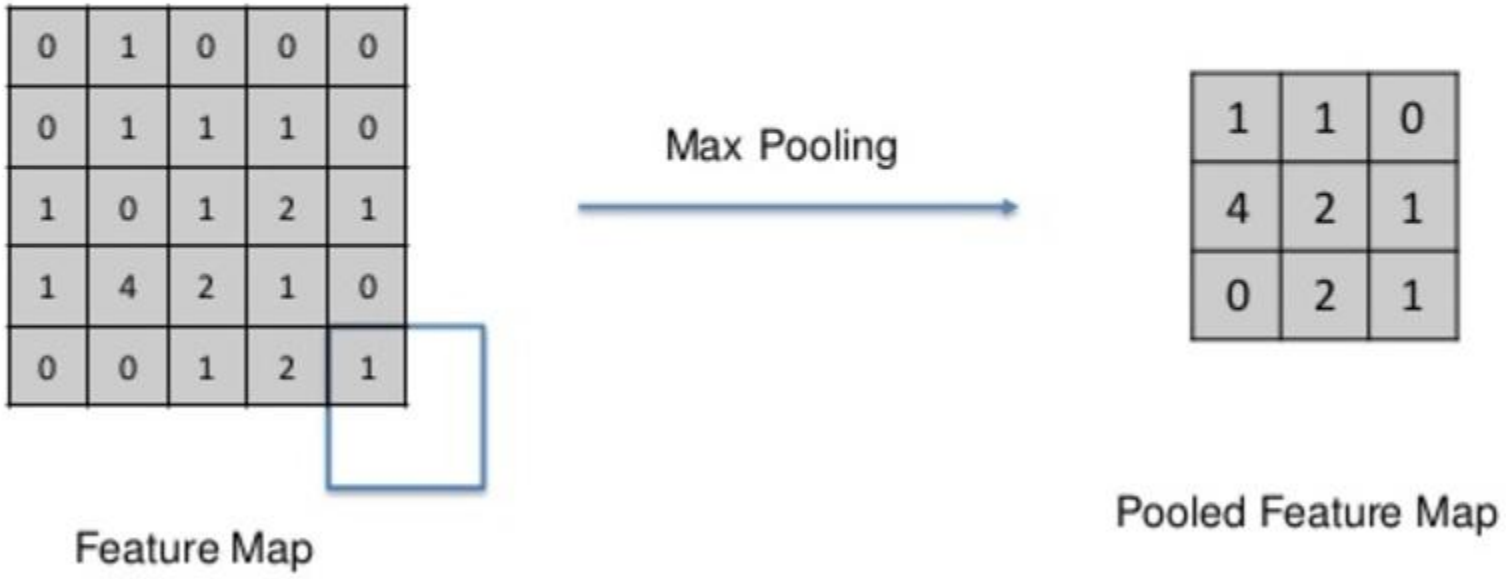
Step 2: MAX Pooling



MAX Pooling

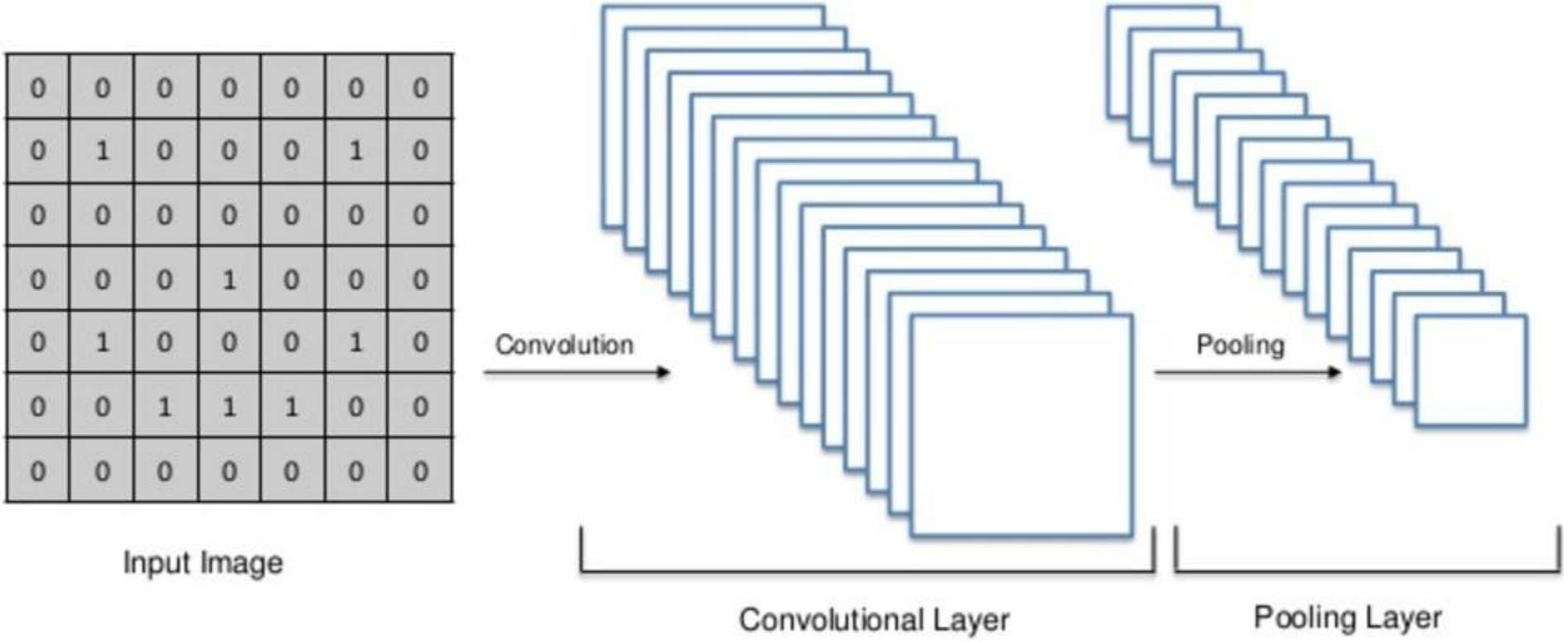

0 1 0 0 0
0 1 1 1 0
1 0 1 2 1
1 4 2 1 0
0 0 1 2 1
Feature Map
Max Pooling
1 1 0
4 2 1
0 2
Pooled Feature Map

Step 2: MAX Pooling



Step 1 and 2: Conv + pooling

Mondragon Unibertsitatea
Faculty of Engineering



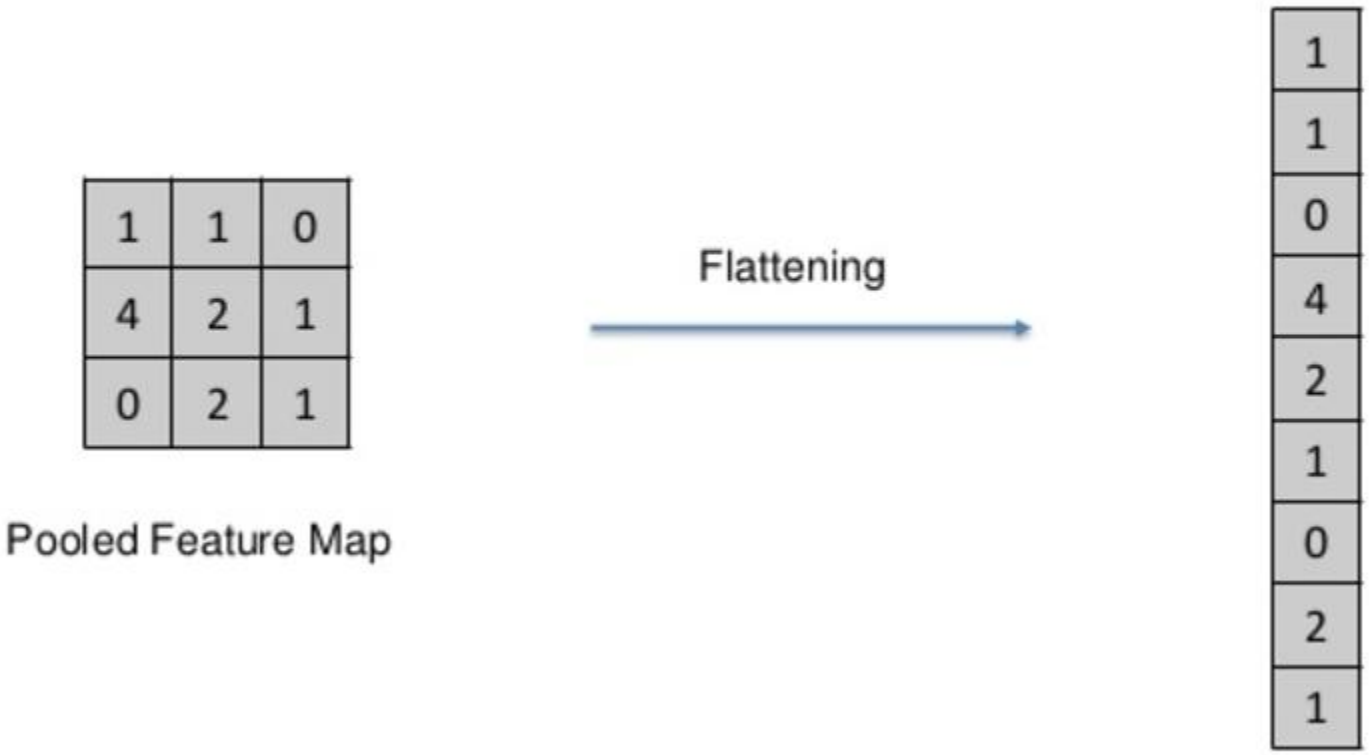
Step 1 and 2: Conv + pooling

Mondragon Unibertsitatea
Faculty of Engineering

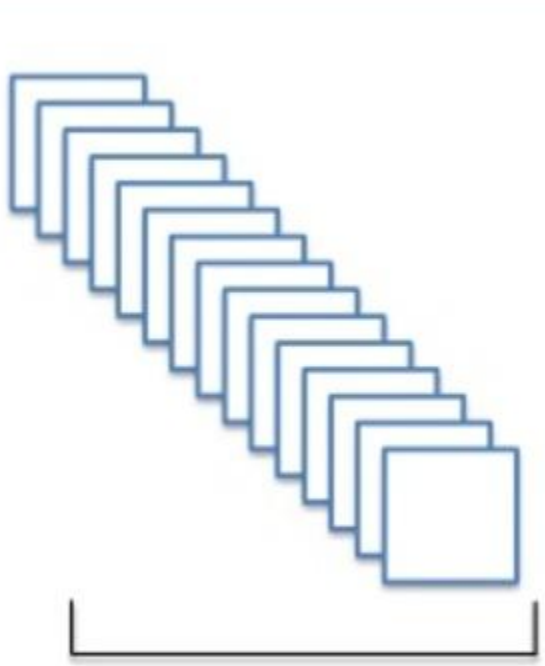
- Interesting paper about poolings: http://www.ais.uni-bonn.de/papers/icann2010_maxpool.pdf
- Convolution and pooling live! example: http://scs.ryerson.ca/~aharley/vis/conv/flat.html
-

Step 3
Flattening

Step 3: Flattening

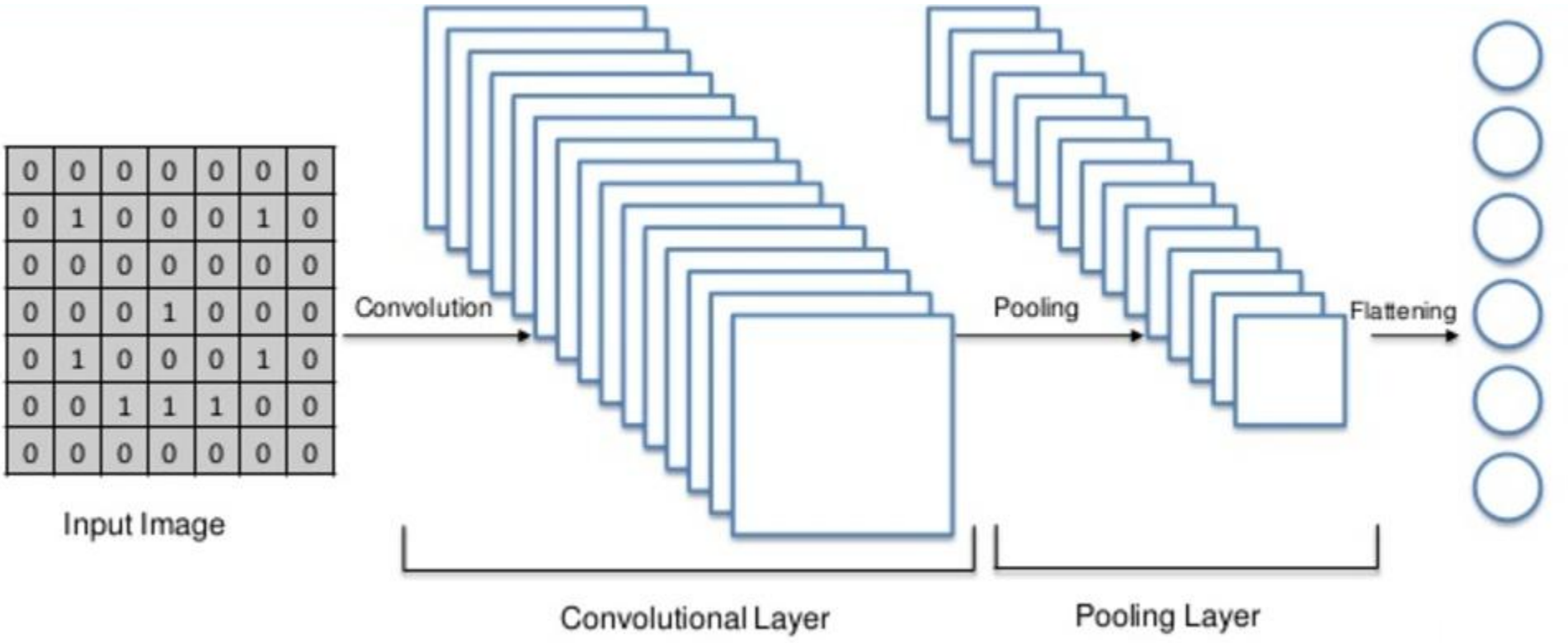


Step 3: Flattening



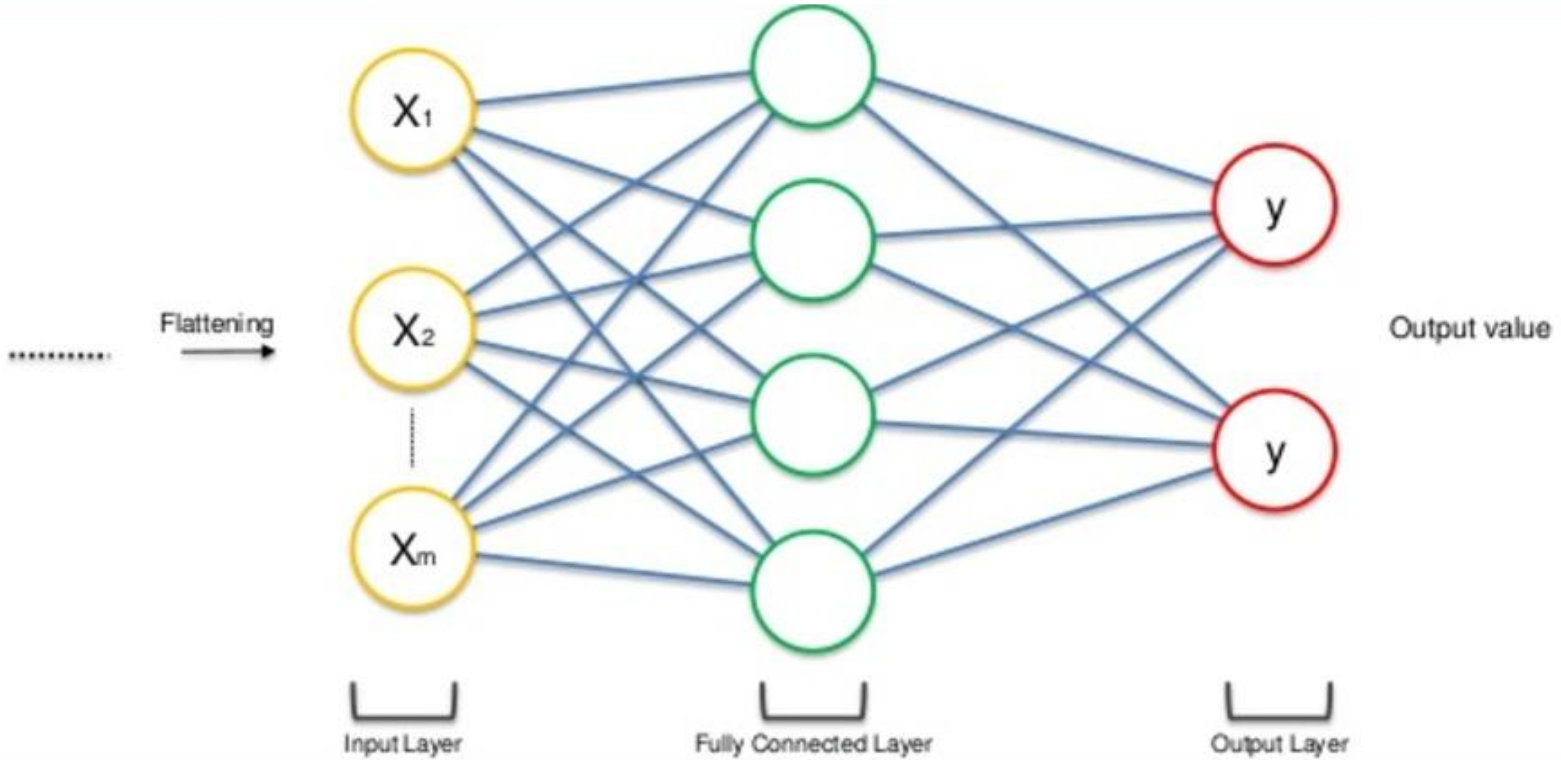
Entrada a un ANN

steps 1+2+3: conv.+pool.+flat.

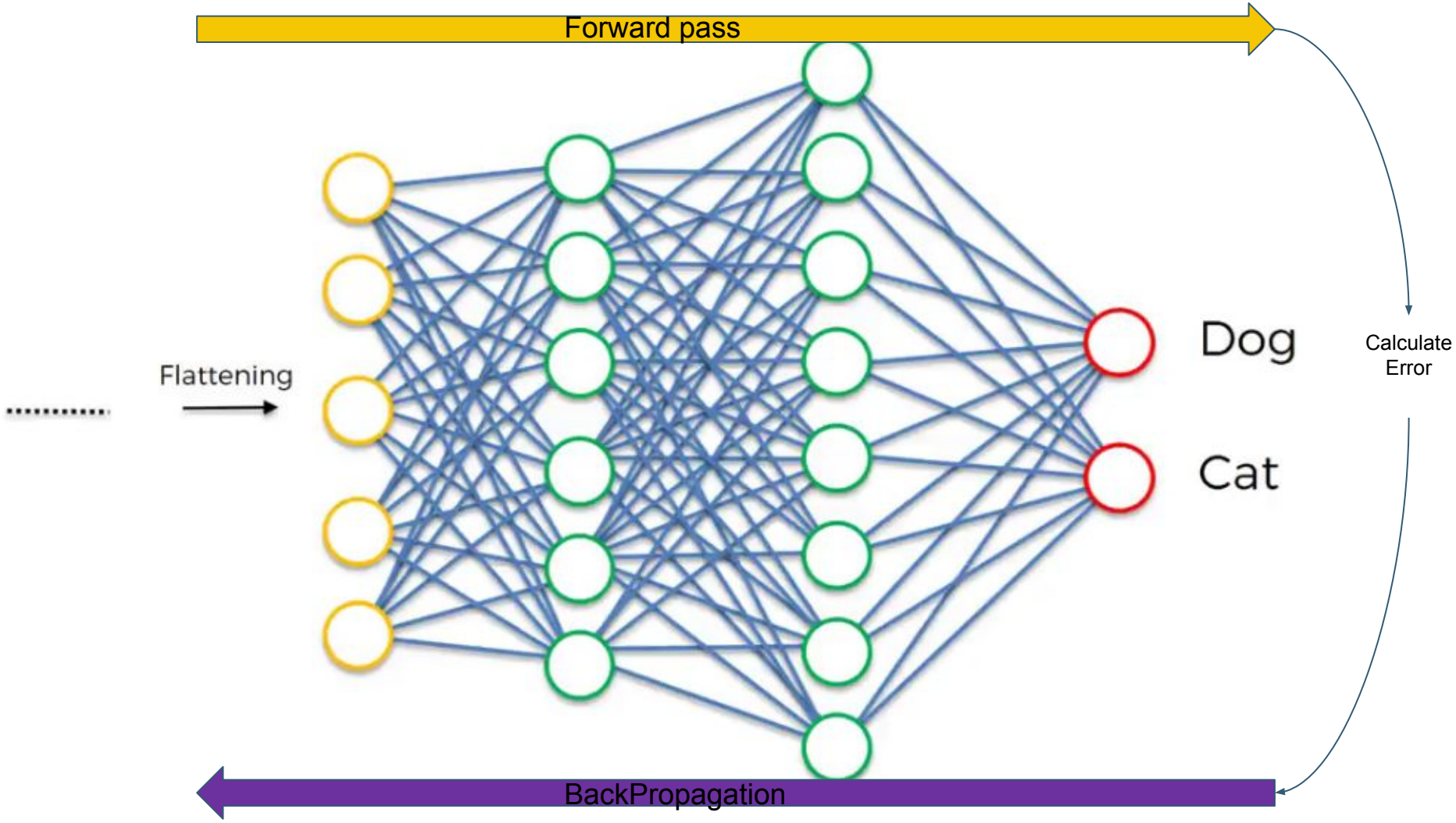


Step 4
Full Connection

Step 4: Fully connected ANN

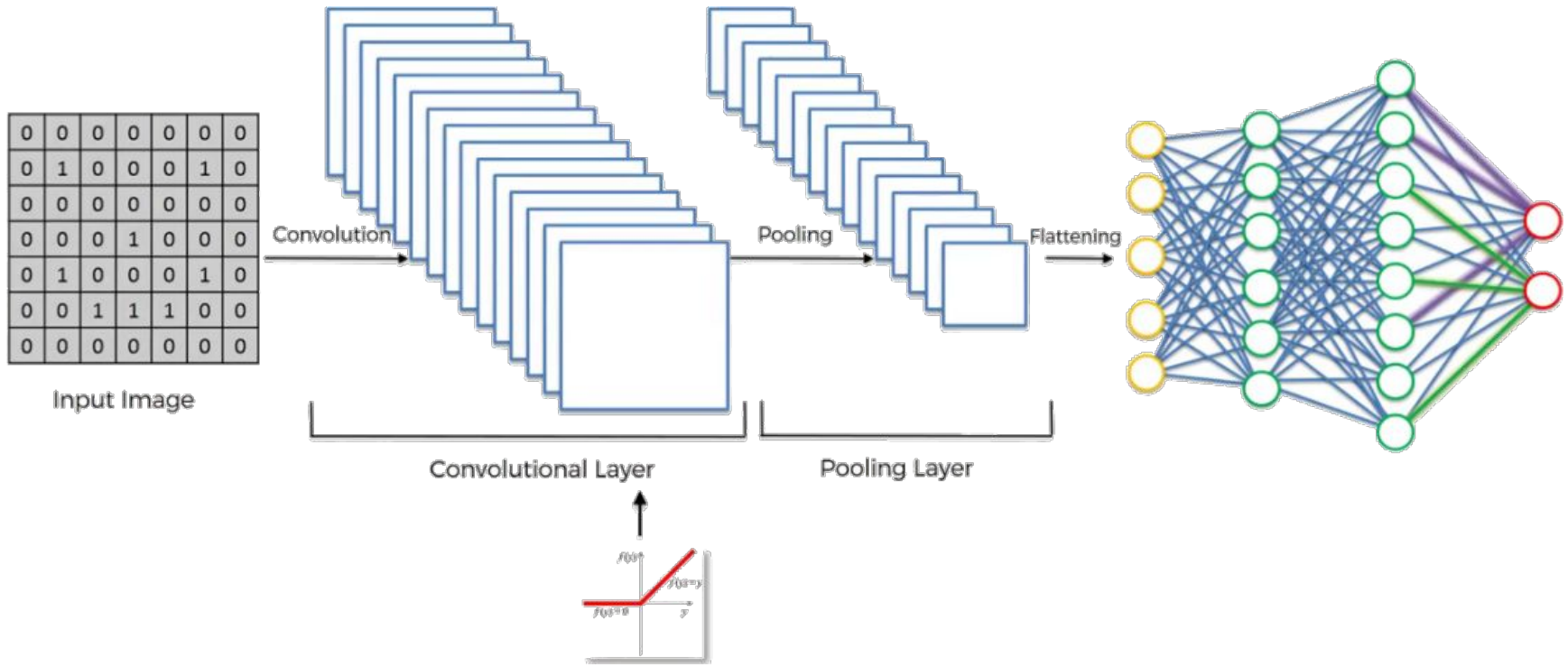


Step 4: Fully connected ANN



Summary

Summary


0 0 0 0 0 0 0
0 1 0 0 0 1 0
0 0 0 0 0 0 0
0 0 0 1 0 0 0
0 1 0 0 0 1 0
0 0 1 1 1 0 0
0 0 0 0 0 0 0
Input Image
Convolution
Pooling
Flattening
Convolutional Layer
Pooling Layer

CNN Tensorflow utils

Image preprocessing with TF

Mondragon Unibertsitatea
Faculty of Engineering

- Preprocessing steps:
  - Read the picture files
  - decode JPEG content to RGB pixels
  - Convert these into floating-point tensors
  - Rescale the pixels to be between [0,1]

- Image Agumentation
  - rotate
  - width/height shift
  - shear
  - zoom
  - horizontal/vertical flip
- Luckily, Keras has a ImageDataGenerator that makes
- This generator can also

Image preprocessing with TF


```
train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=40,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,)

test_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
        train_dir,
        target_size=(150, 150),
        batch_size=32,
        class_mode='binary')

validation_generator = test_datagen.flow_from_directory(
        validation_dir,
        target_size=(150, 150),
        batch_size=32,
        class_mode='binary')
```

Note that the validation data shouldn't be augmented!

Target directory

Resizes all images to 150 × 150

Because you use binary_crossentropy loss, you need binary labels.

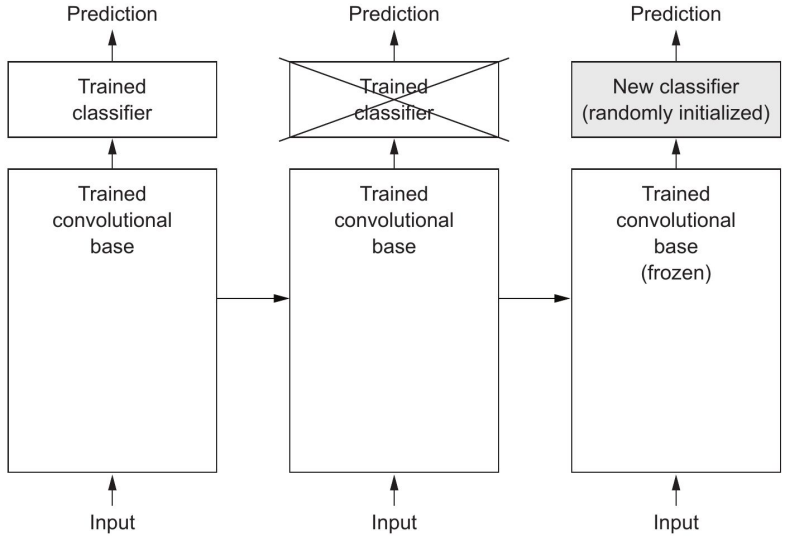
Using a pretrained covnet

Mondragon Unibertsitatea
Faculty of Engineering

- It is common to use a pretrained network to extract characteristics from images
  - A pretrained network is a network that was trained previously on a huge dataset
- Usually, the features learned by a pretrained network are general enough to use them in many different vision problems
- Usually, we change the “fully connected” layer and “freeze” the convolutional part
- There are several pretrained networks
  - VGG16
  - MobileNet
  - ResNet
  - ...

```
from keras.applications import VGG16

conv_base = VGG16(weights='imagenet',
                  include_top=False,
                  input_shape=(150, 150, 3))
```



Exercises

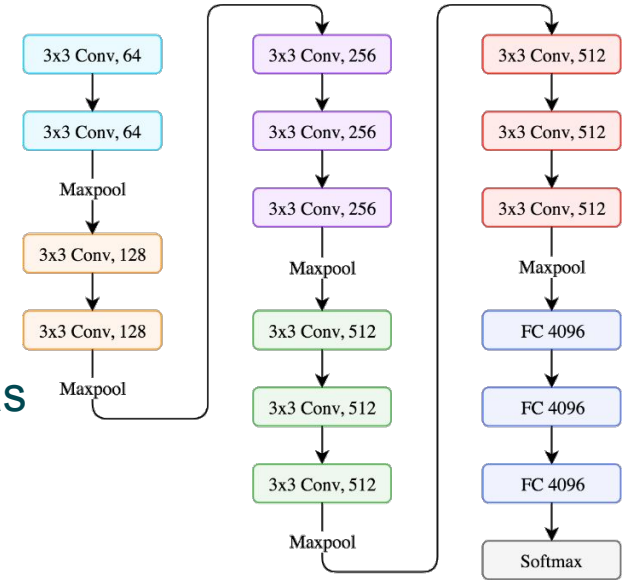
exercises

Mondragon Unibertsitatea
Faculty of Engineering

1. Crear y entrenar una red neuronal que diferencie perros de gatos con Keras y CNN
   1.1. Aprenderemos a crear todo lo explicado con las CNN
      1.1.1. Capa Convolucional + ReLU
      1.1.2. Max Pooling
      1.1.3. Flattening
      1.1.4. Red fully connected

2. Utilizar redes neuronales preentrenadas (VGG16)
   2.1. Aprenderemos a utilizar redes ya creadas y entrenadas
      2.1.1. VGG16

3. perros y gatos con resnet

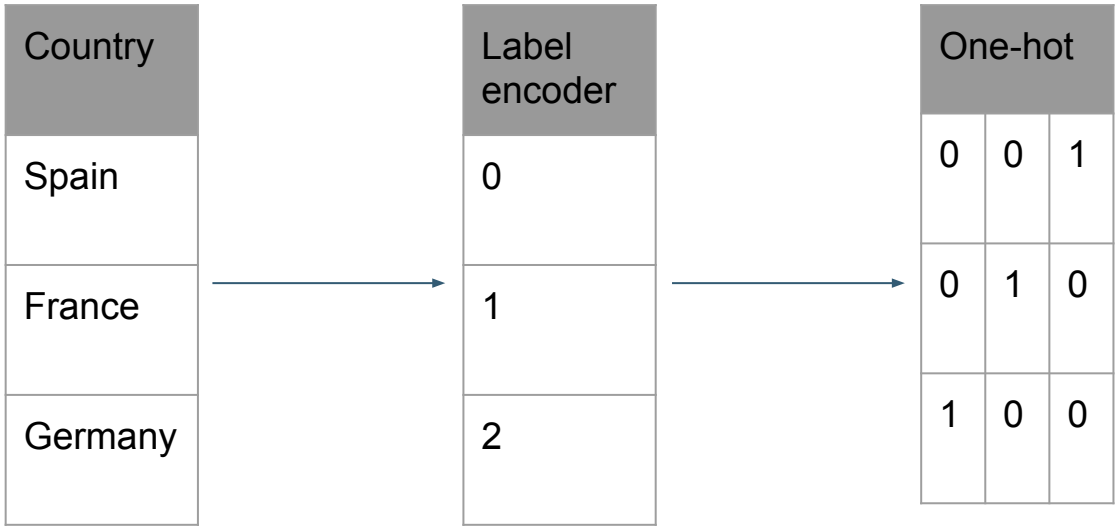


Apendix 1

Embedding Layer

Mondragon Unibertsitatea
Faculty of Engineering

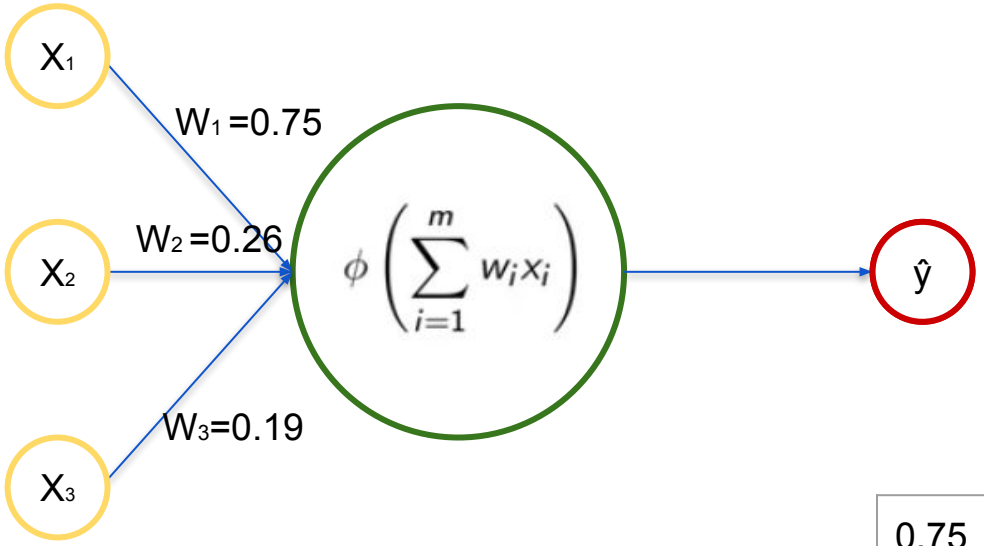
- Se utiliza cuando hay datos categóricos



Embedding Layer

Mondragon Unibertsitatea
Faculty of Engineering

One-hot		
0	0	1
0	1	0
1	0	0



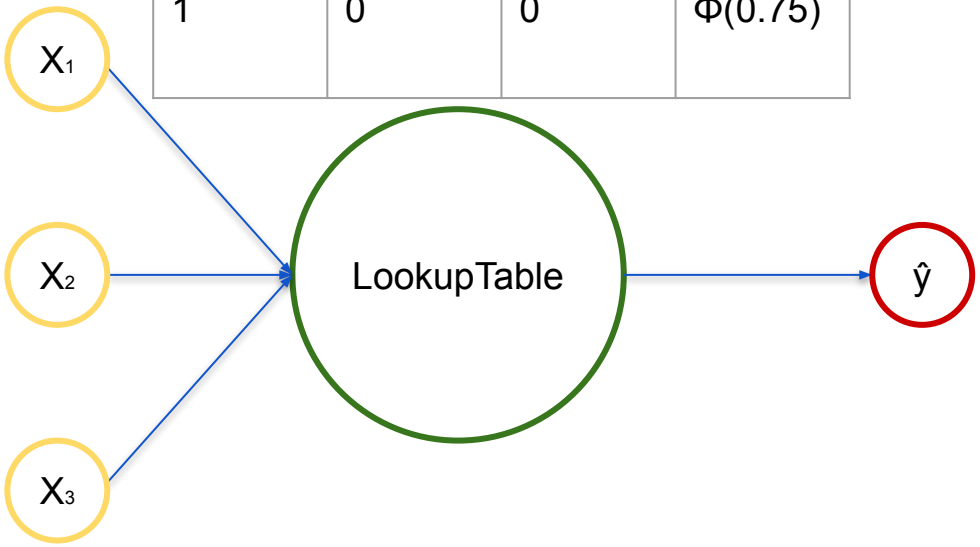
1	0	0	X	0.75 0.26 0.19	=	0.75		
0	0	1	X	0.75 0.26 0.19	=	0.19	0 1 0	X 0.75 0.26 0.19 = 0.26

Embedding Layer

Mondragon Unibertsitatea
Faculty of Engineering

- No es más que un “lookup table” para datos categóricos
- así ahorra computación

Input			output
0	0	1	Φ(0.19)
0	1	0	Φ(0.26)
1	0	0	Φ(0.75)



Mondragon Unibertsitatea

Faculty of Engineering

Eskerrik asko

Muchas gracias

Thank you

Ekhi Zugasti
ezugasti@mondragon.edu

Loramendi, 4. Apartado 23
20500 Arrasate – Mondragon
T. 943 71 21 85
info@mondragon.edu