Convolutional Neural Networks

UNED

ETS de Ingeniería Informática

Dr. Manuel Castillo-Cara
www.manuelcastillo.eu
Departamento de Inteligencia Artificial
Escuela Técnica Superior de Ingeniería Informática
Universidad Nacional de Educación a Distancia (UNED)

Preliminar

UNED

ETS de Ingeniería Informática

Índice

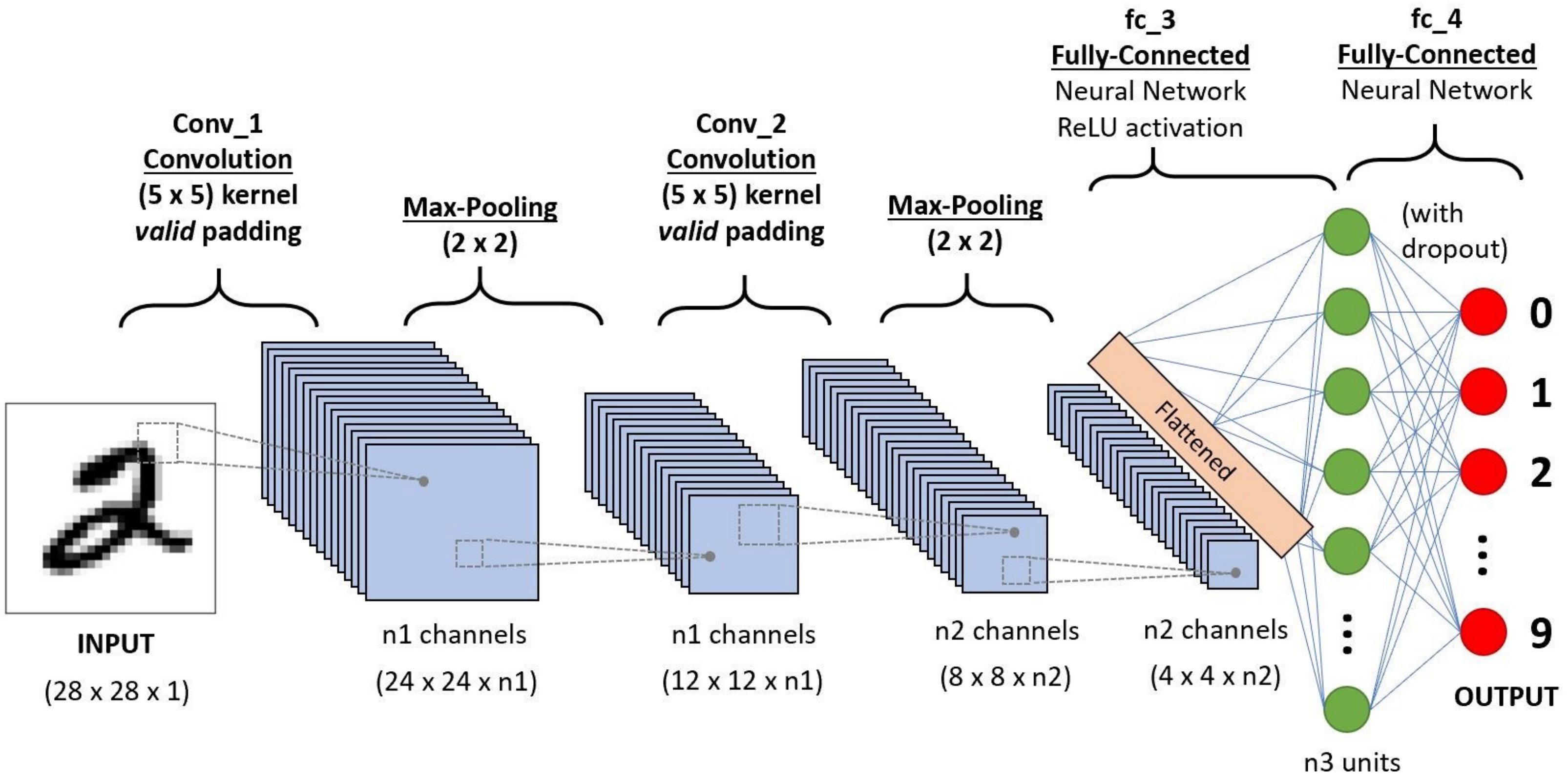
- Convolutional Neural Networks
- Operation Convolution
- Operation Pooling
- Towards standardization
- Transfer learning and finetunning
- CNN Vs RNN

UNED

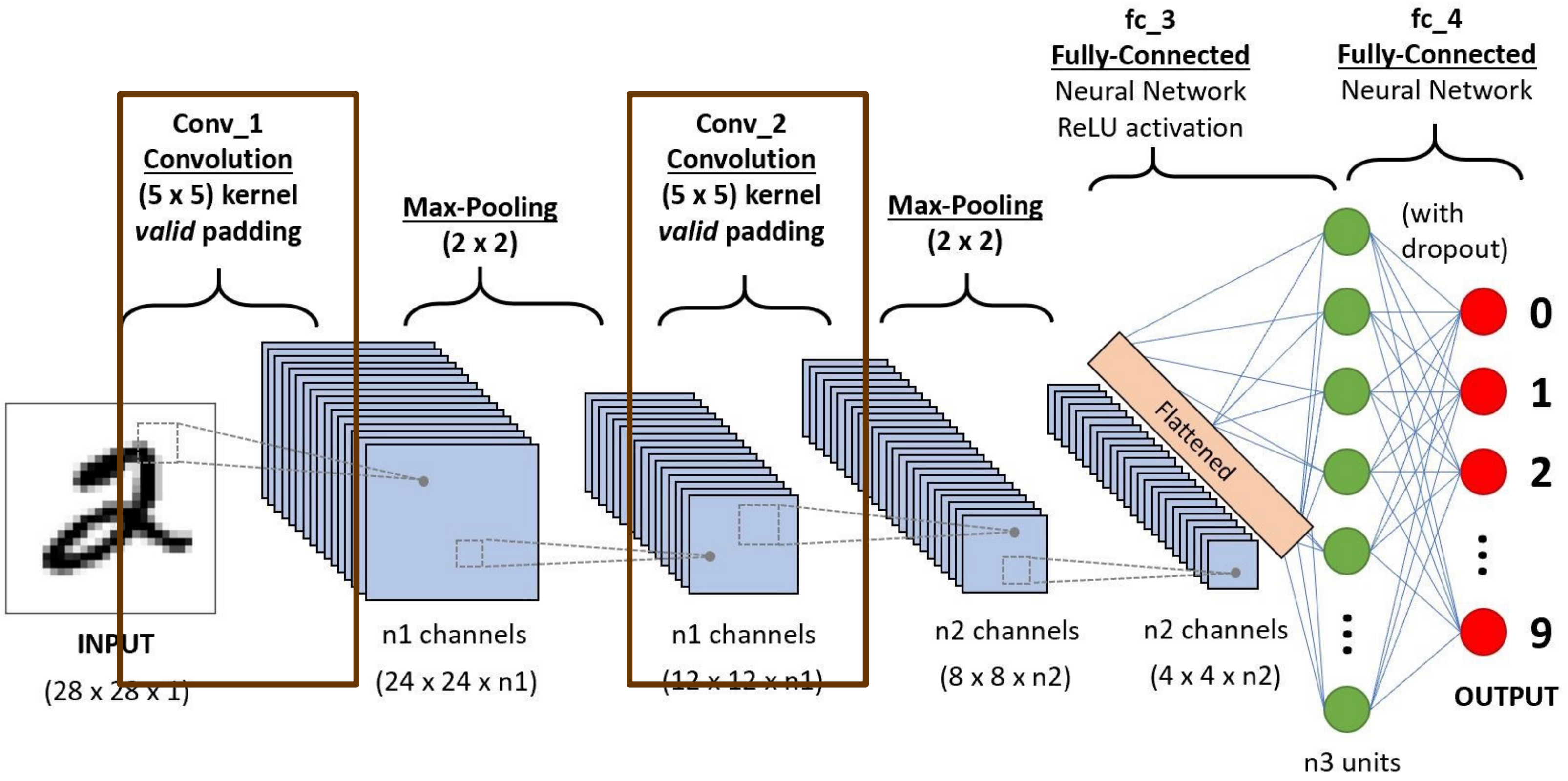
ETS de Ingeniería Informática

Convolutional Neural Networks

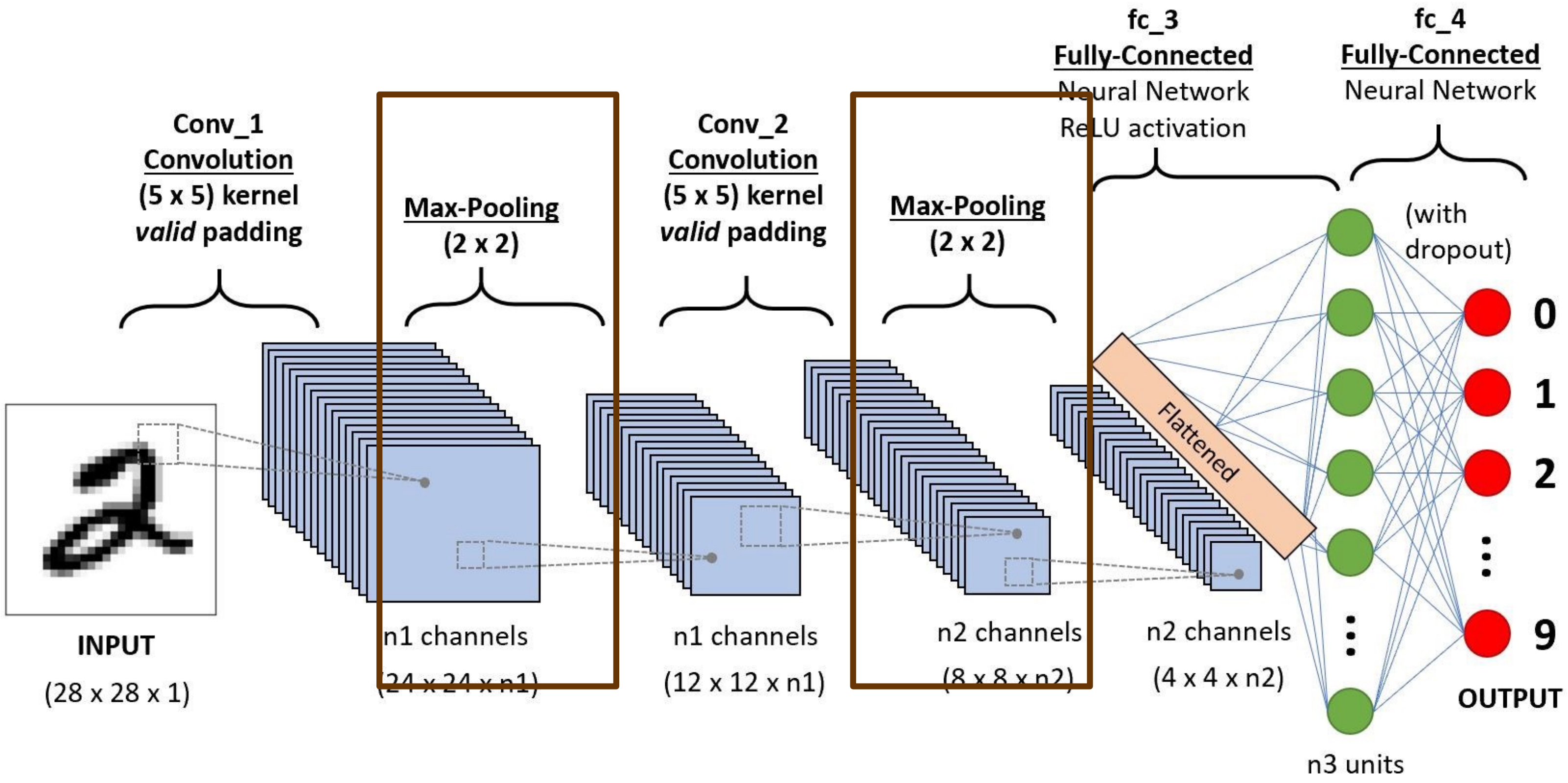
Background


Conv_1
Convolution
(5 x 5) kernel
valid padding
Max-Pooling
(2 x 2)
Conv_2
Convolution
(5 x 5) kernel
valid padding
Max-Pooling
(2 x 2)
fc_3
Fully-Connected
Neural Network
ReLU activation
fc_4
Fully-Connected
Neural Network
(with dropout)
Flattened
0
1
2
9
INPUT
(28 x 28 x 1)
n1 channels
(24 x 24 x n1)
n1 channels
(12 x 12 x n1)
n2 channels
(8 x 8 x n2)
n2 channels
(4 x 4 x n2)
n3 units
OUTPUT

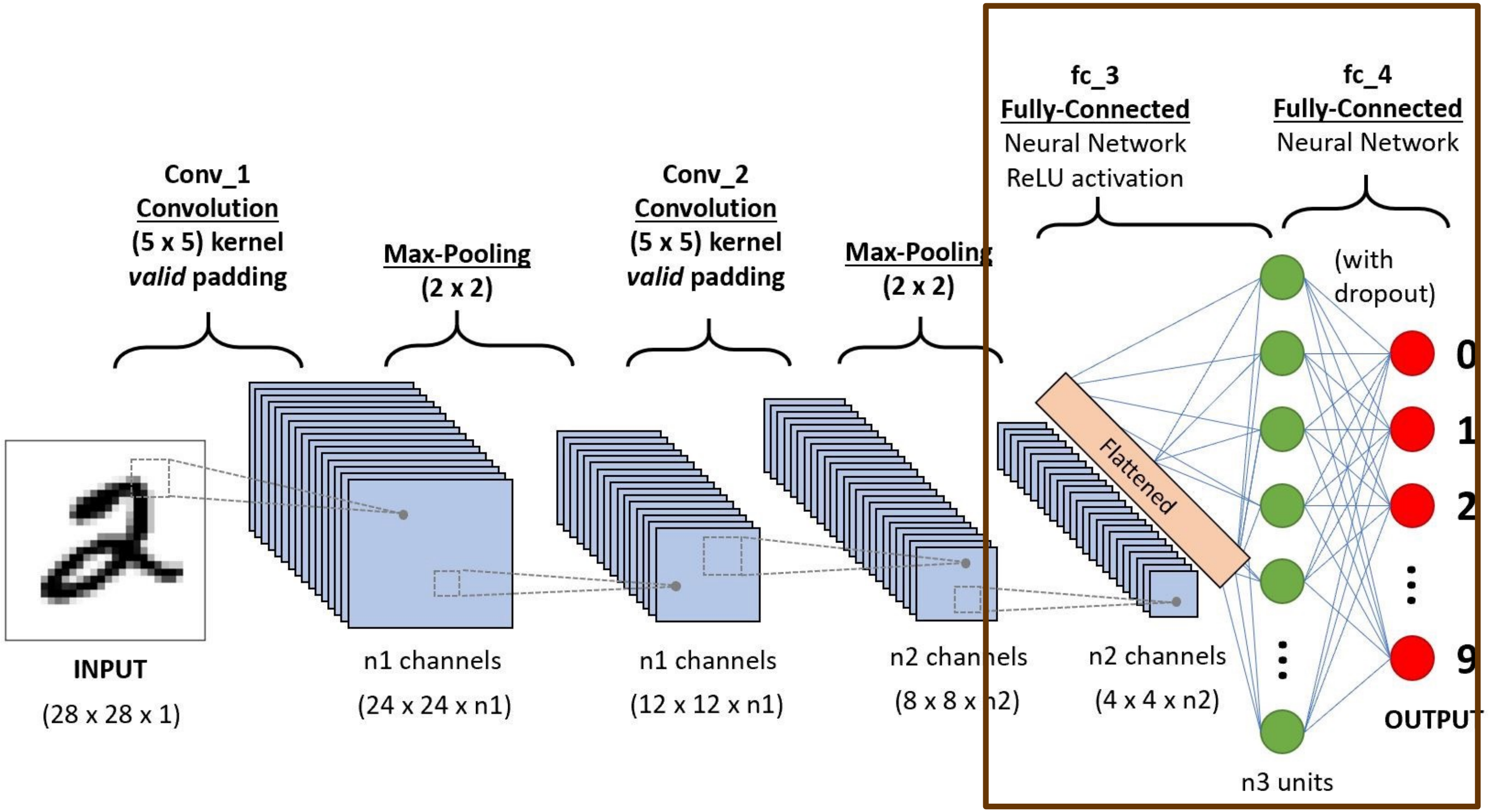
Convolutional layers



Pooling layers



Fully Connected Layers



Example

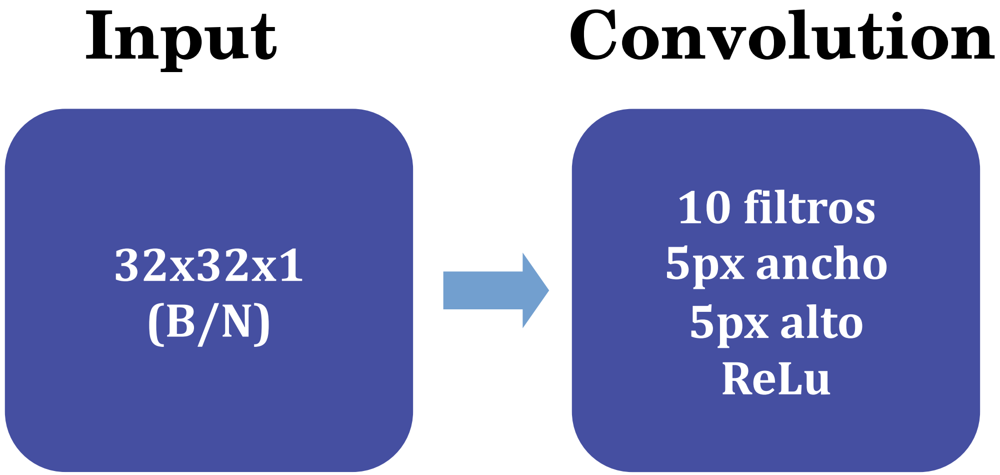
Input Image(32x32)

Input

32x32x1
(B/N)

Example

Input Image(32x32)



Input

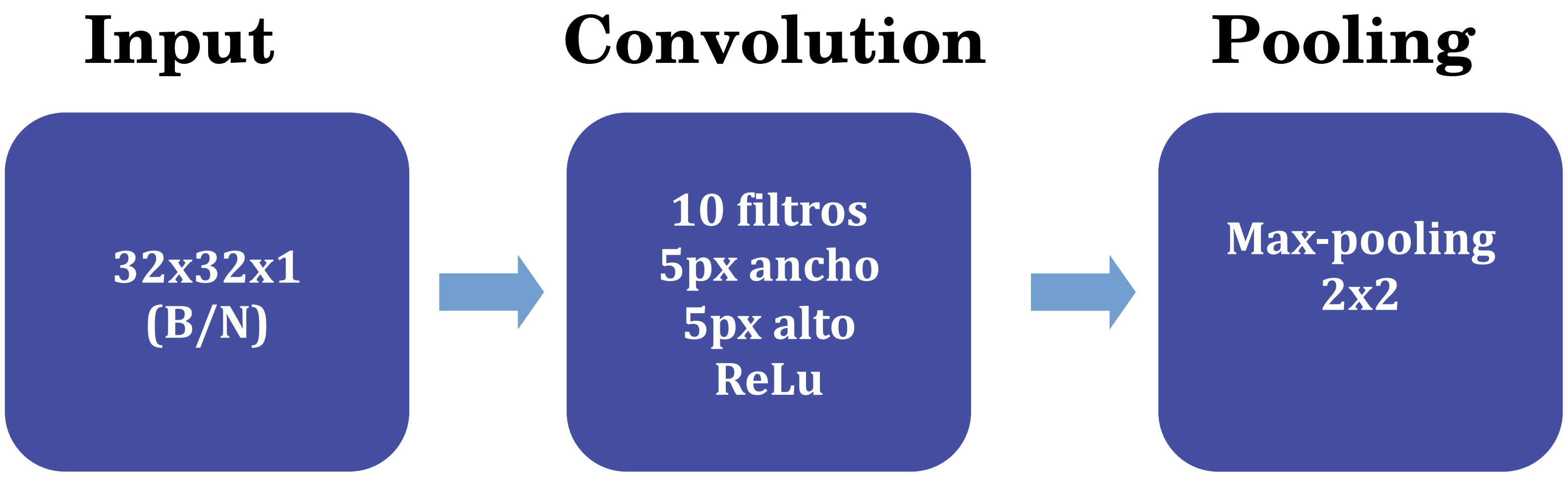
32x32x1
(B/N)

Convolution

10 filtros
5px ancho
5px alto
ReLu

Example

Input Image(32x32)



Input

32x32x1
(B/N)

Convolution

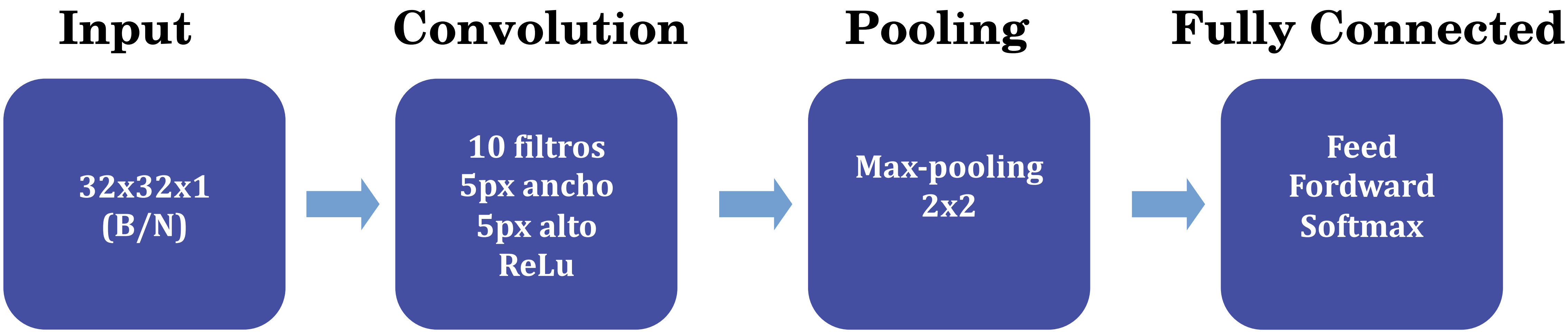
10 filtros
5px ancho
5px alto
ReLu

Pooling

Max-pooling
2x2

Ejemplo

Imagen de entrada (32x32)



Input

32x32x1
(B/N)

Convolution

10 filtros
5px ancho
5px alto
ReLu

Pooling

Max-pooling
2x2

Fully Connected

Feed
Fordward
Softmax

Tips

- Input Responsive Field Dimensions:
- Size of the receptive field
- Stride Width
- Number of filters
- Padding
- Pooling
- Data Preparation
- Pattern Architecture
- Dropout

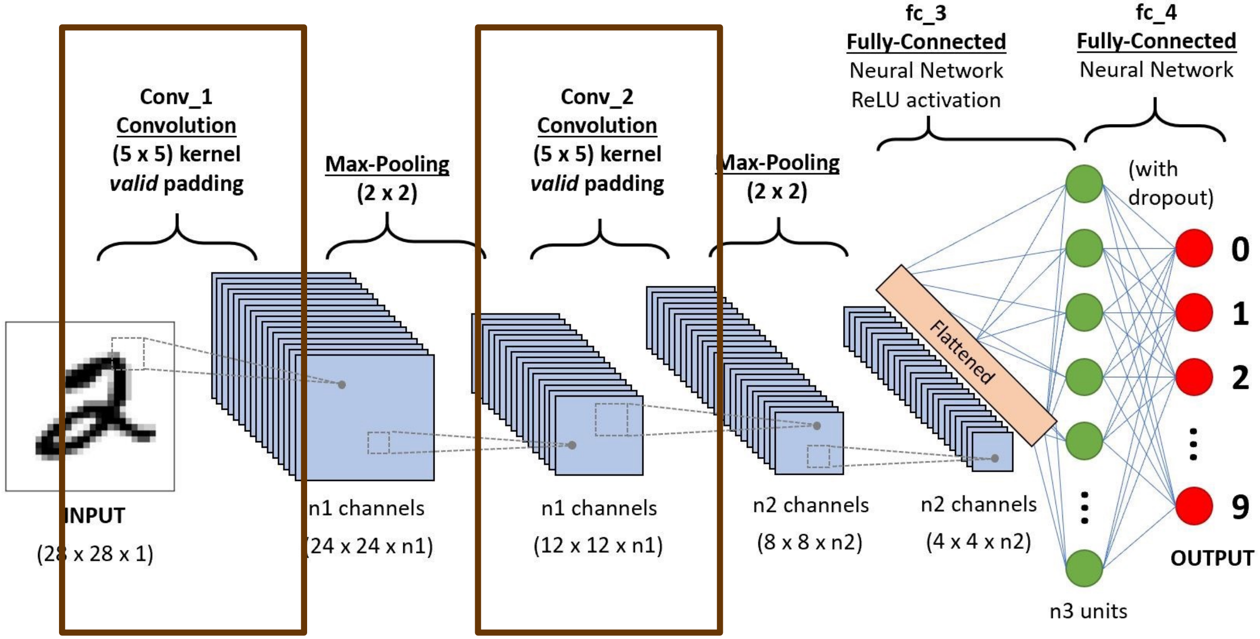
UNED

ETS de
Ingeniería
Informática

Operación Convolución

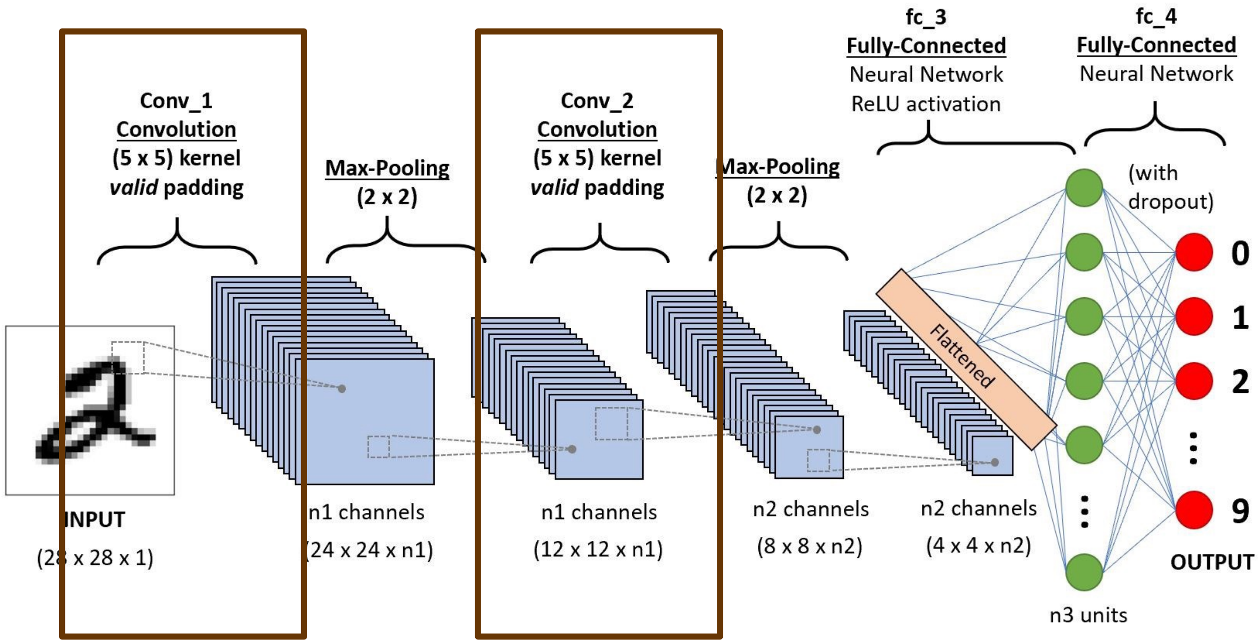
Convolution Operation(I)

- The first hidden layer of a CNN usually corresponds to a **convolutional layer**, which can be understood as the coding required for the application of filters (known as kernel functions)
- For a better understanding of the convolution operation, let's think of an input layer that encodes a binary image (B/W) and a **convolution** with which we want to represent an **edge detection filter.**



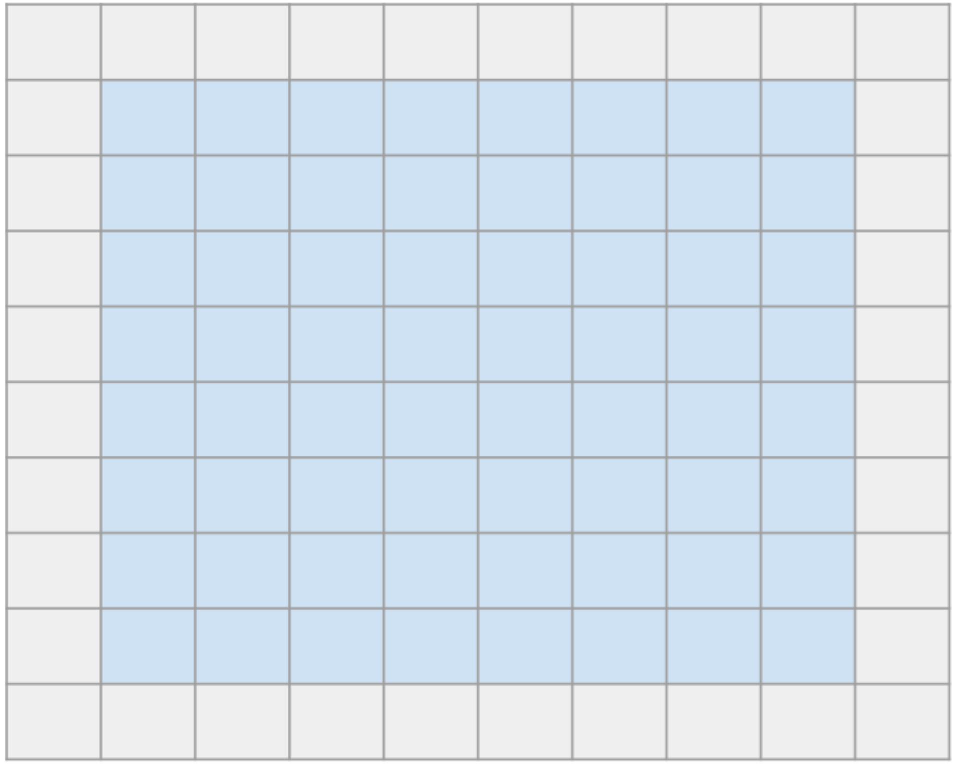
Convolution Operation(II)

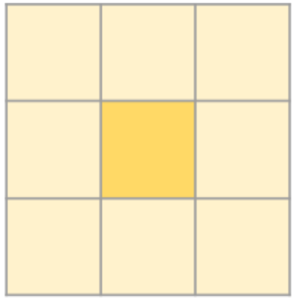
- Without going into more detail, edge search is based on looking for large differences between a pixel and the pixels around it
- We want to perform the same type of operation on the entire image, so a first quality appears: the use of **shared weights**
  - This means that the parameters/conditions for detecting an edge in one pixel of the image must be the same in **all other areas of the image**



Example

- Assuming a 10x10 pixel image, if the detection of an edge needs to work with the 8 pixels that surround it, we would have the following

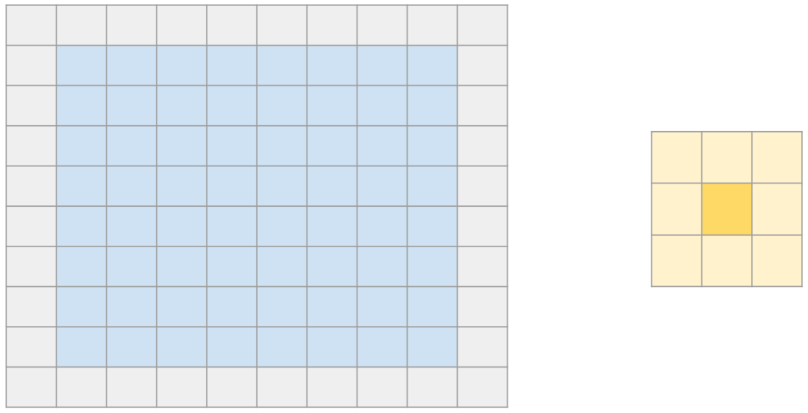




Example

Convolution Operation(I)

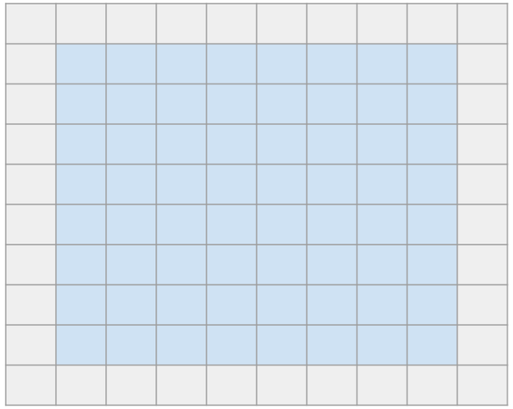
- In the image we show a filter that can only be applied to the highlighted areas

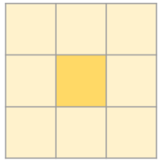


Example

Convolution Operation(I)

- In the image we show a filter that can only be applied to the highlighted areas
- Transferring the problem to be solved on the topology of a CNN, we would include an initial layer with 100 neurons (it can be visually displayed as a 10x10 array)

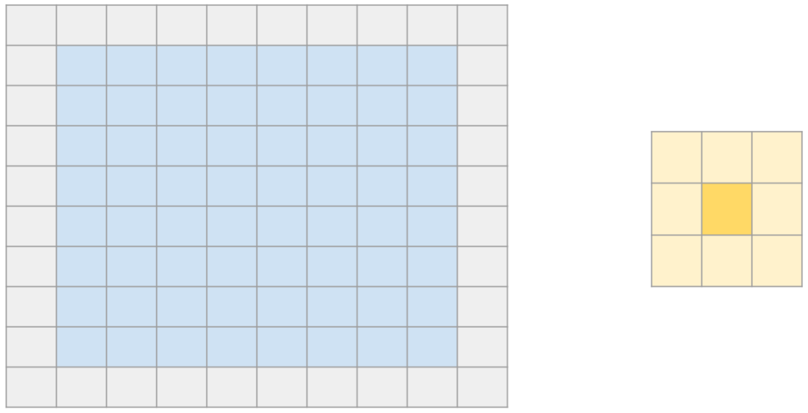




Example

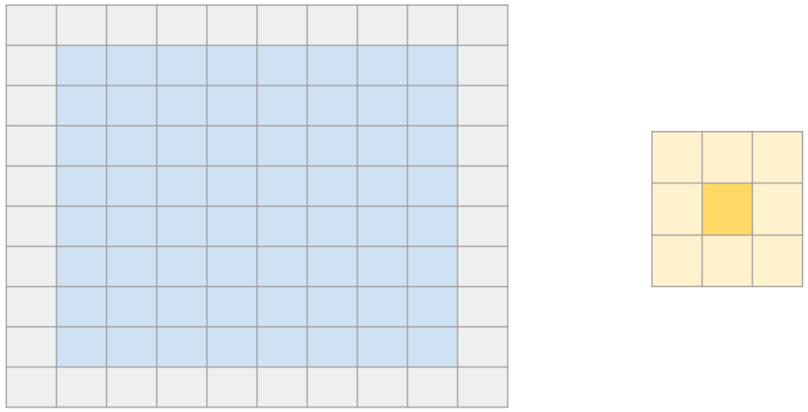
Convolution Operation(I)

- In the image we show a filter that can only be applied to the highlighted areas
- Transferring the problem to be solved on the topology of a CNN, we would include an initial layer with 100 neurons (it can be visually displayed as a 10x10 array)
- The first hidden layer will have **one neuron for each filter result** → will have 8x8 neurons



Example
Convolution Operation(I)

- Each of these 64 neurons will be connected to 9 neurons of the initial layer, but the **weights used** in this interaction w1… w9 will be **shared** by all neurons in this hidden layer

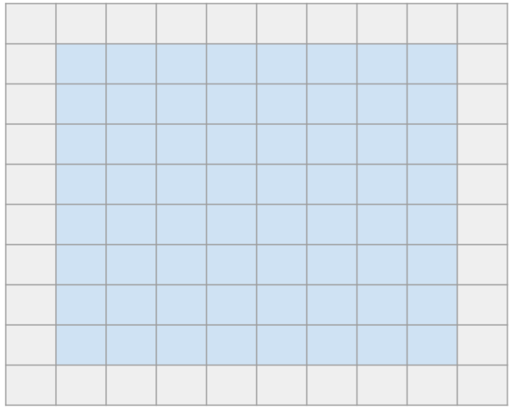


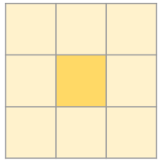
Example

Convolution Operation(I)

- Each of these 64 neurons will be connected to 9 neurons of the initial layer, but the **weights used** in this interaction w1... w9 will be **shared** by all neurons in this hidden layer

- This **simplifies** learning enormously, as well as giving **coherence** to the filter to be carried out



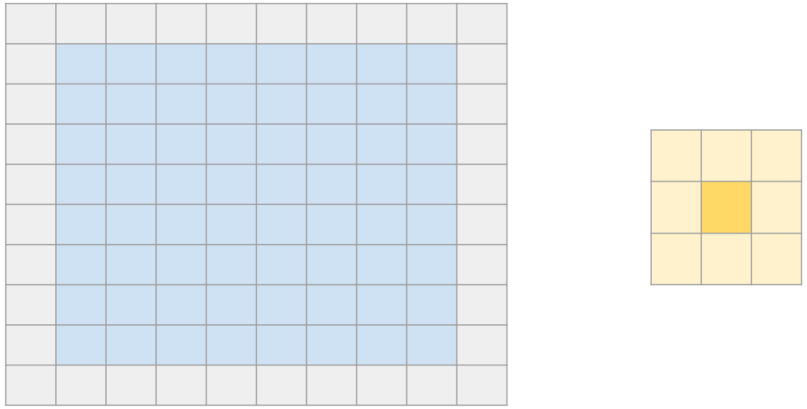


Example
Convolution Operation(I)

- Each of these 64 neurons will be connected to 9 neurons of the initial layer, but the **weights used** in this interaction w1... w9 will be **shared** by all neurons in this hidden layer

- This **simplifies** learning enormously, as well as giving **coherence** to the filter to be carried out

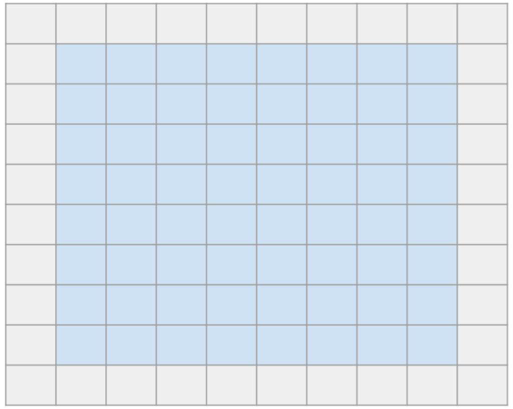
- Once learned, the activation function (usually ReLU) will let us know which pixel in the image **corresponds or does not correspond to an edge**

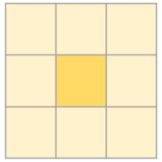


Example

Convolution Operation(I)

- In the proposed example, convolutions involving both the pixel under study and the **8 pixels around it (3x3) are proposed**



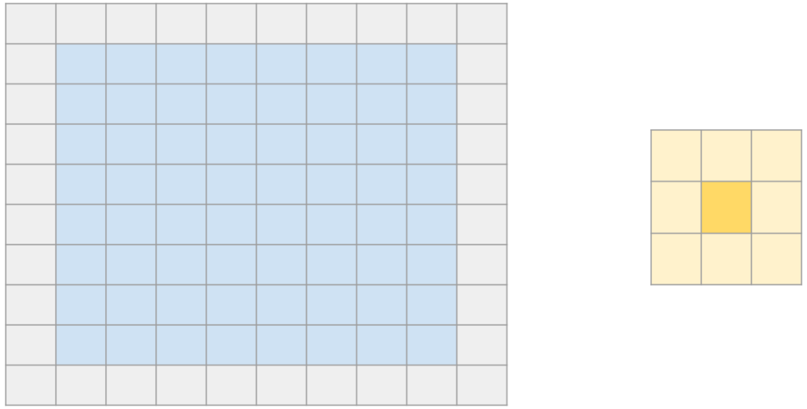


Example
Convolution Operation(I)

- In the proposed example, convolutions involving both the pixel under study and the **8 pixels around it (3x3) are proposed**

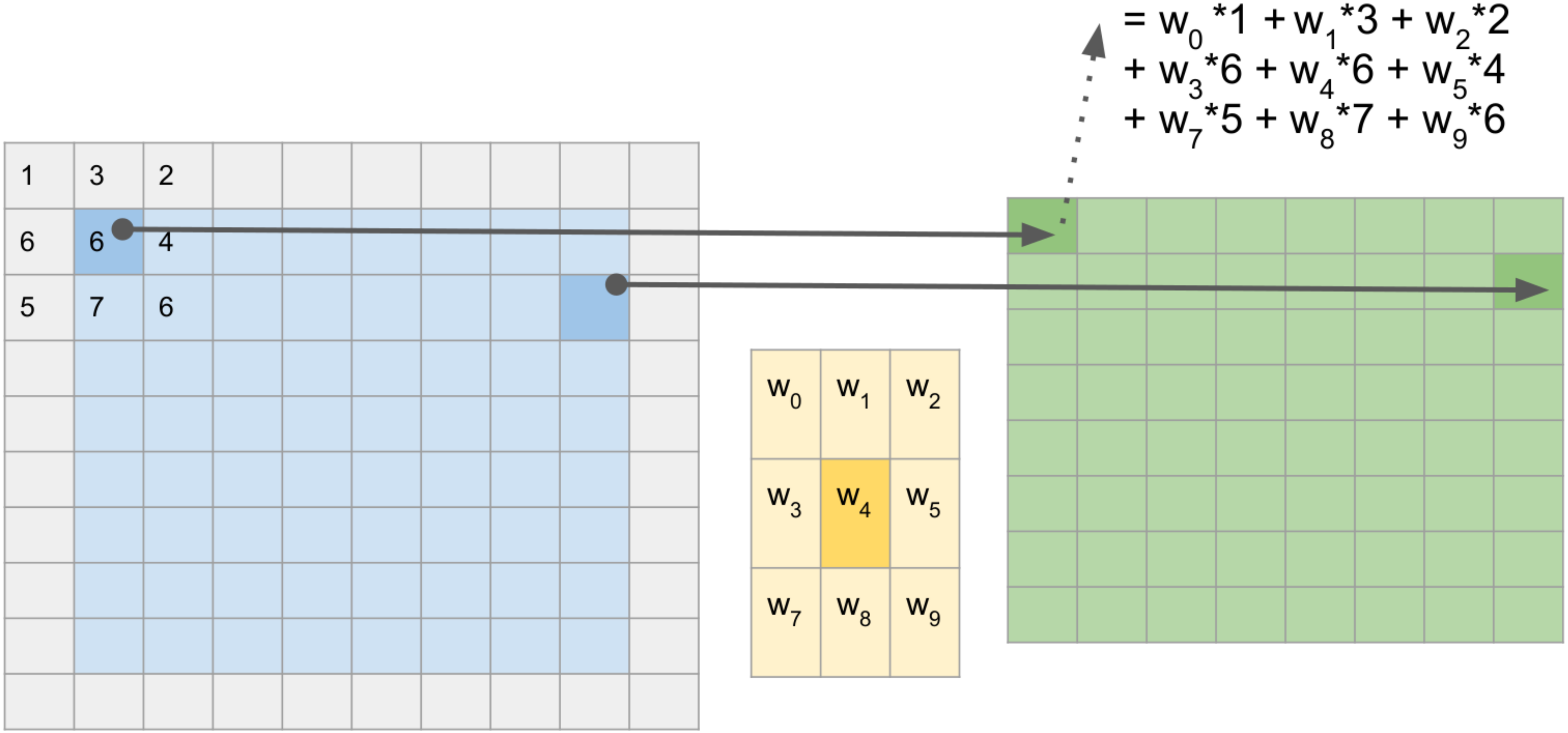
- In addition, it is assumed that this operation is performed one by one, but only on the pixels of the image that is possible (except rows/columns 0 and 9), resulting in a **layer of smaller size** (10x10 → 8x8)

- → 2 parameters define this configuration: **padding and stride**



Example

Convolution Operation(I)



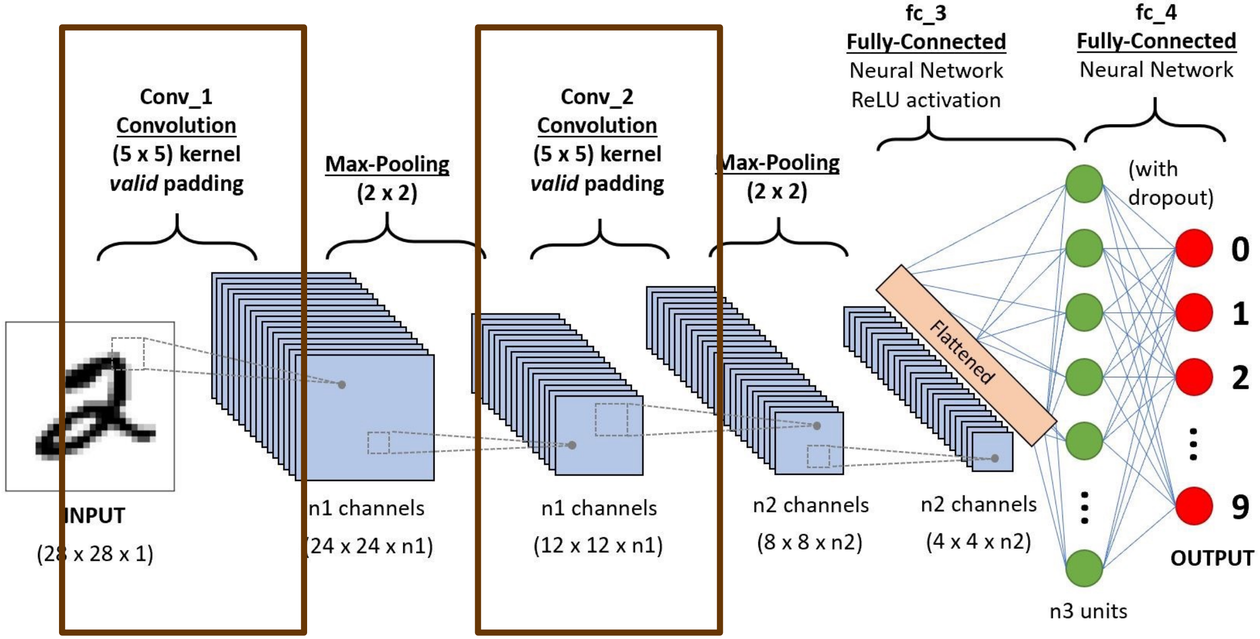
$$= w_0 *1 + w_1 *3 + w_2 *2 + w_3 *6 + w_4 *6 + w_5 *4 + w_7 *5 + w_8 *7 + w_9 *6$$

w_0	w_1	w_2
w_3	w_4	w_5
w_7	w_8	w_9

Padding

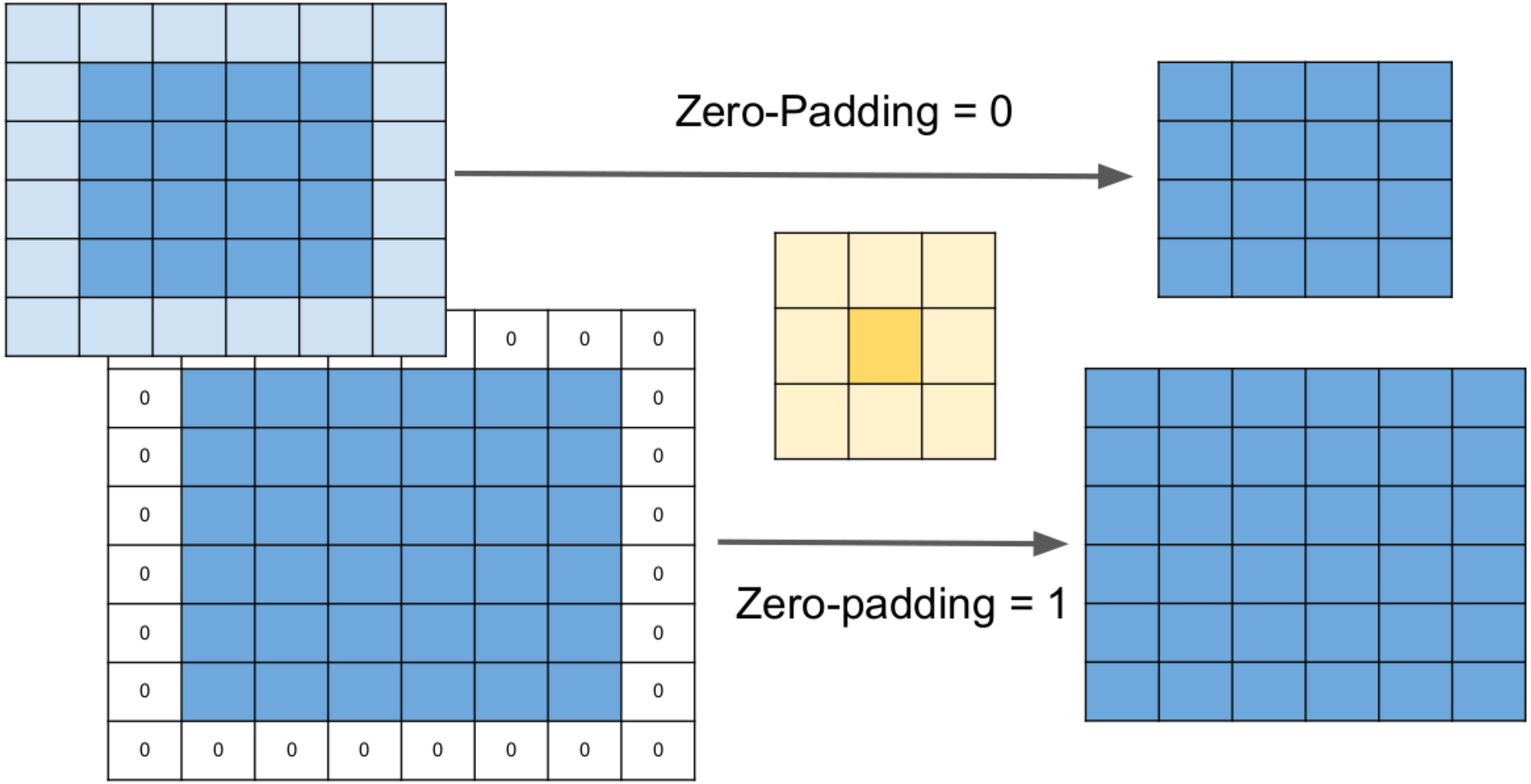
Definition

- The padding parameter (optional) defines a series of **synthetic neurons** used as filling, so that convolutions can be performed without the resulting layer being smaller than the input layer
- These neurons must take a value to perform the operations**, with zero** being the most commonly used value.
  - In this case we refer to this parameter as **zero-padding** N (N = number of additional neurons)



Padding
Example

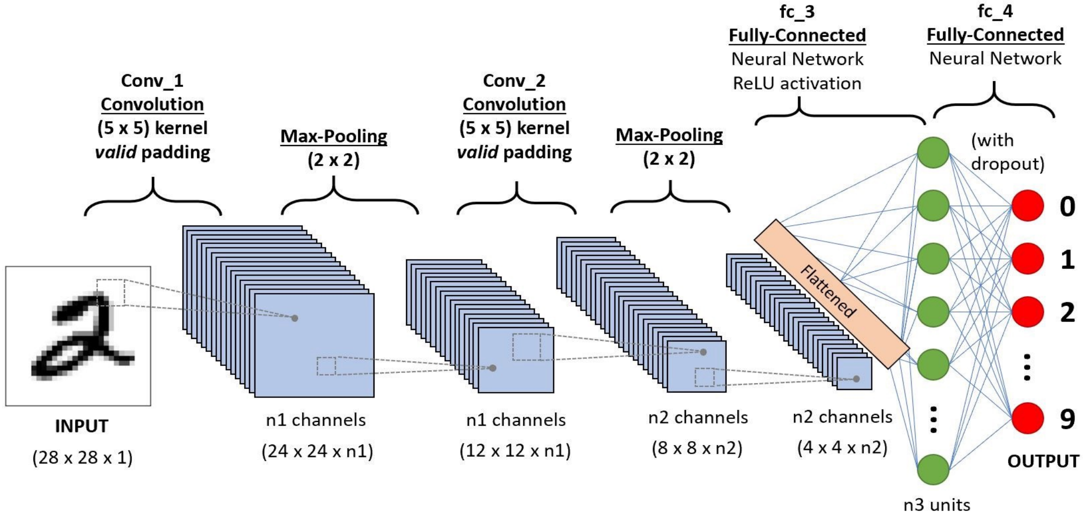
- Let's look at an example with 6x6 images and a 3x3 filter



Stride

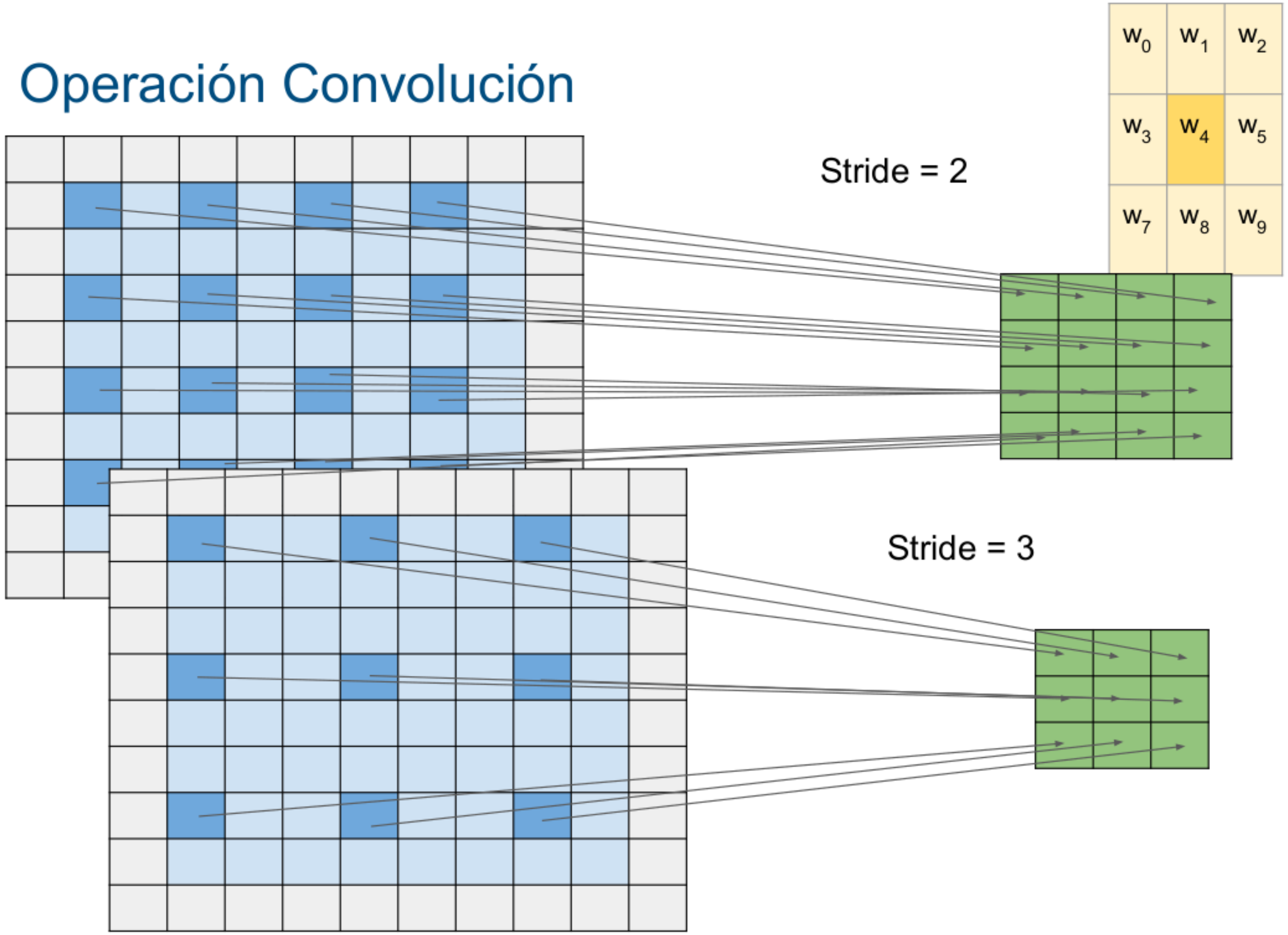
Definition

- The **stride** parameter tells us the number of neurons to advance after the application of a convolution
- To understand this parameter, we turn to spatial organization: Using a two-dimensional matrix, a value of **stride 2** will tell us that after applying a convolution:
  - The next one will be made two **columns to the right**, and
  - After the current row is finished, the next row to be processed will be selected after **moving down two rows**



Stride
Example

- Let's look at an example with 6x6 images and a 3x3 filter



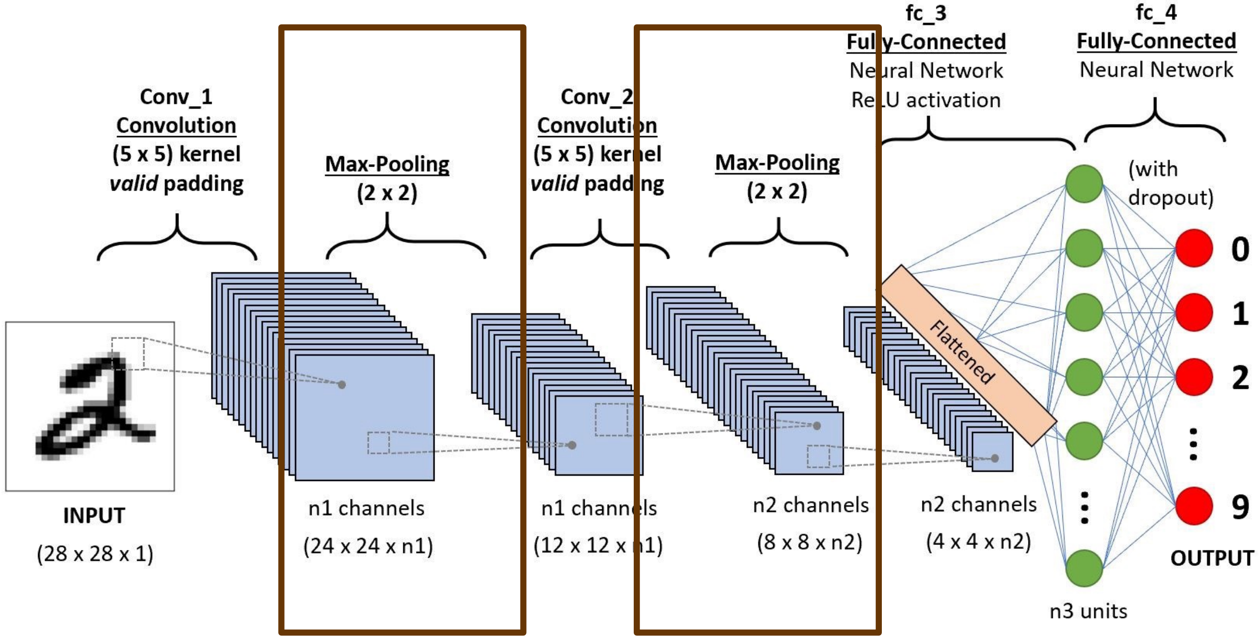
UNED

ETS de
Ingeniería
Informática

Operation Pooling

Definition

- The pooling operation aims to **reduce the dimensionality** of the layers generated after the application of convolution operations
- Remembering that the layers have three dimensions (width x height x depth), the reduction would be applied only to the **width and height dimensions**, without affecting the depth of these layers



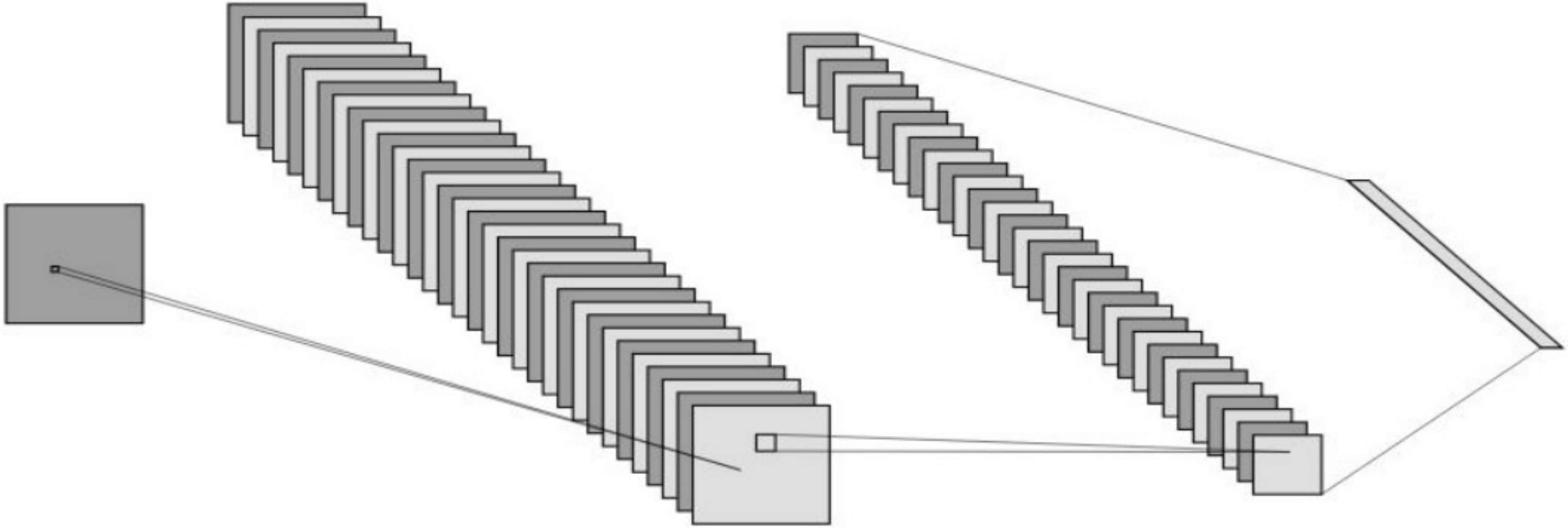
Example

- On a simple example, let's assume that we work with **640x480** (B/W) images and that the first convolution applies **32 3x3 filters** (stride 1, zero-padding 1)
- The result will be a **640x480x32** middle layer

Example

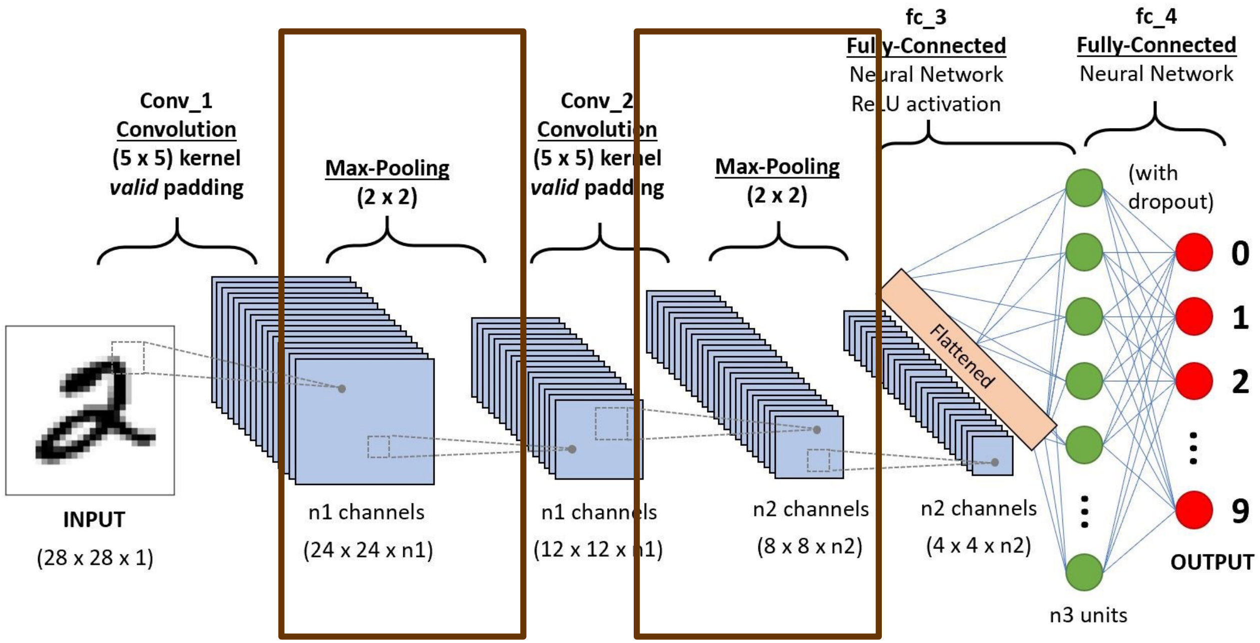
- On a simple example, let's assume that we work with **640x480** (B/W) images and that the first convolution applies **32 3x3 filters** (stride 1, zero-padding 1)
- The result will be a **640x480x32** middle layer
- A pooling operation **reducing the dimensionality** to 25% of the original would obtain a new intermediate layer of 320x240x32

Example



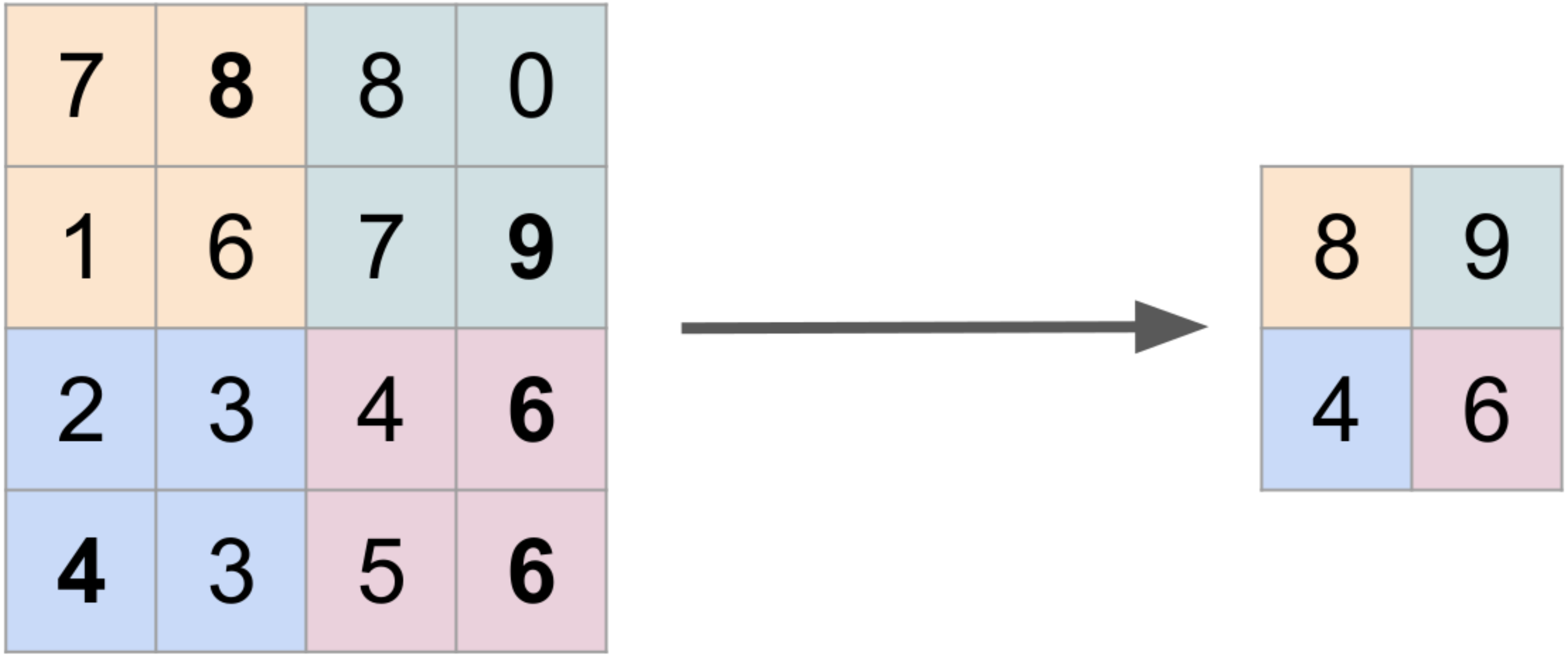
Max-pooling

- The most common pooling operation is max-pooling, where each operation computes the **maximum** for a series of **spatially connected neurons**
- It is applied through an NxM size filter, usually M=N (defined as spatial extent).
  - The stride parameter is also used, which will determine the **dimensionality reduction factor**



Max-pooling

Example of max-pooling 2x2 with stride=2

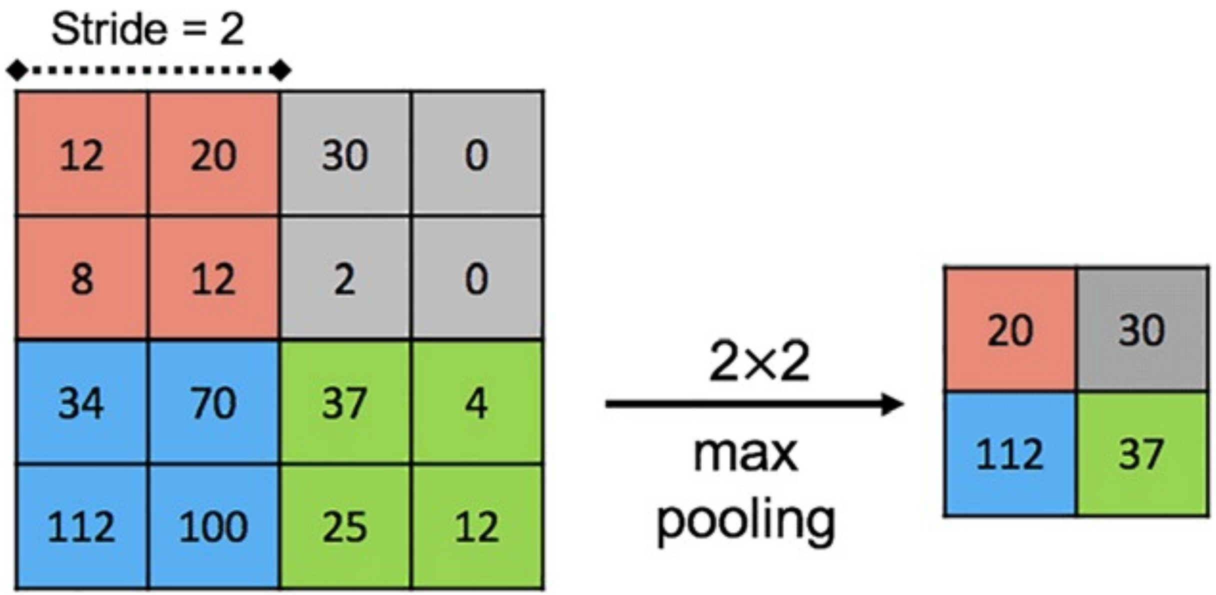


7	**8**	8	0
1	6	7	**9**
2	3	4	**6**
4	3	5	**6**

8	9
4	6

Considerations

- On a practical level, the use of filters greater than 3x3, or with a stride value greater than 2, usually generates poor results → we advise against their use
- As **alternatives** to max-pooling, we can replace the maximum function with other functions such as the **average**.
  - However, experience has shown the greater power of the **maximum** function



12	20	30	0
8	12	2	0
34	70	37	4
112	100	25	12

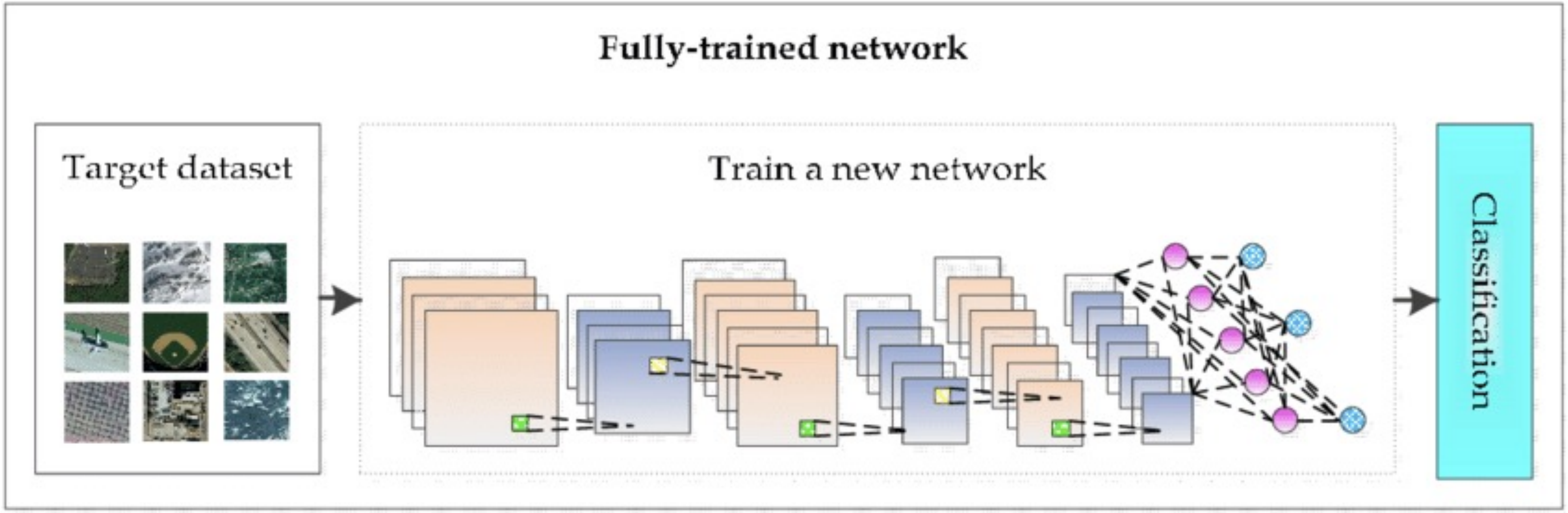
20	30
112	37

UNED

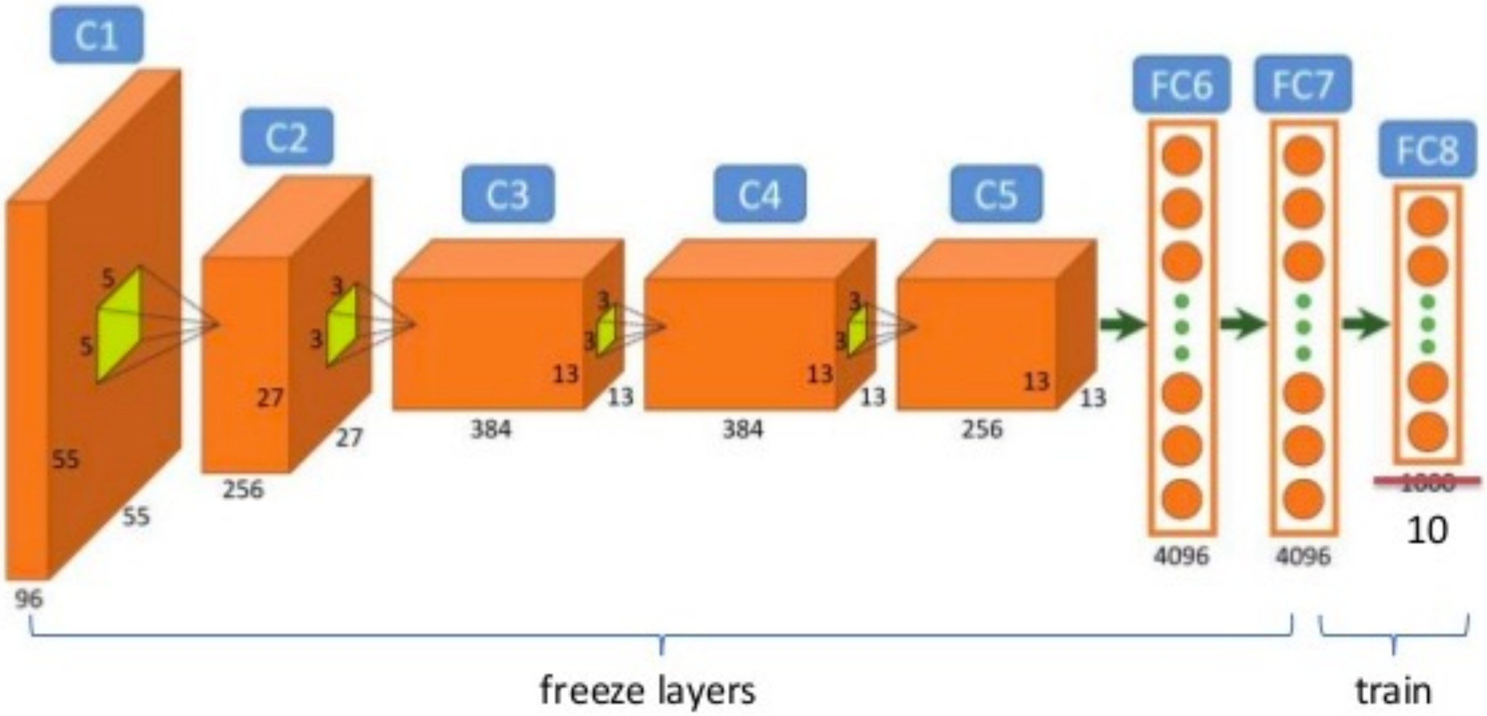
ETS de
Ingeniería
Informática

Hacia la estandarización

1. Transfer Learning



2. Finetunning

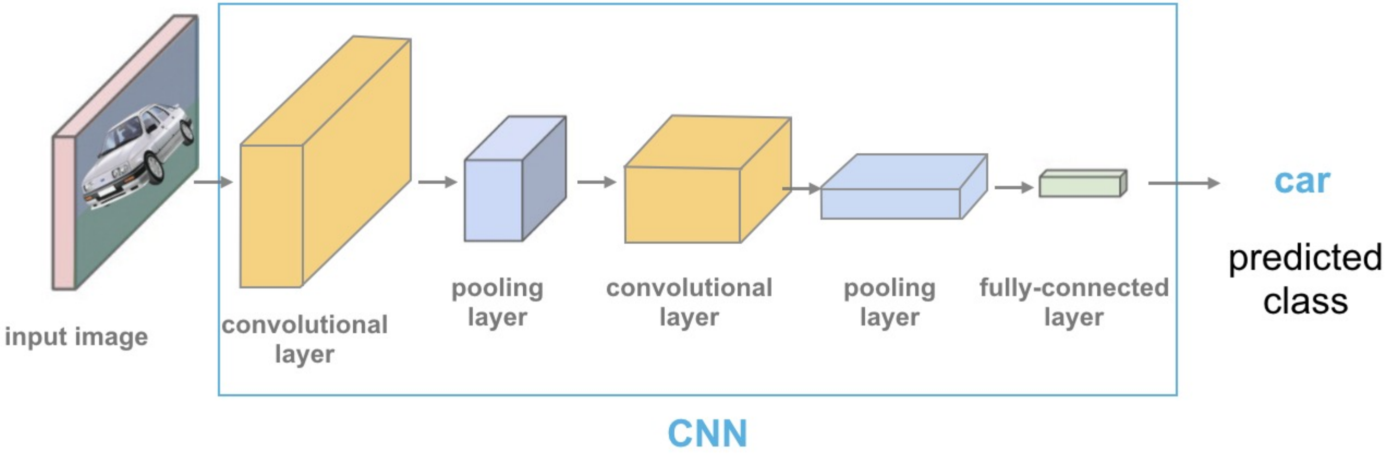


UNED

ETS de
Ingeniería
Informática

CNN Vs RNN

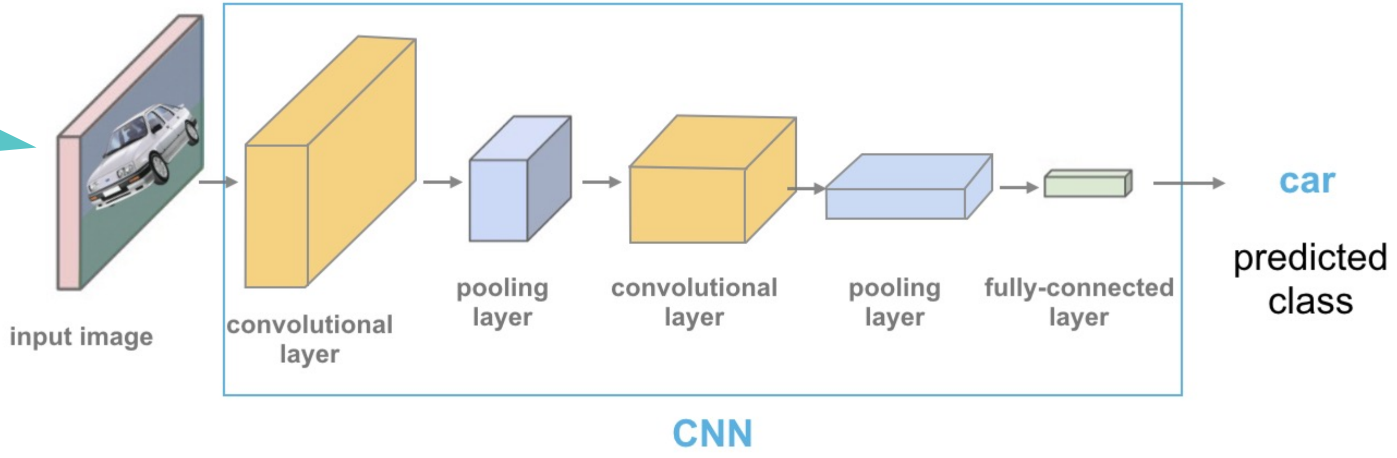
Background
CNN vs. RNN



Background

CNN vs. RNN

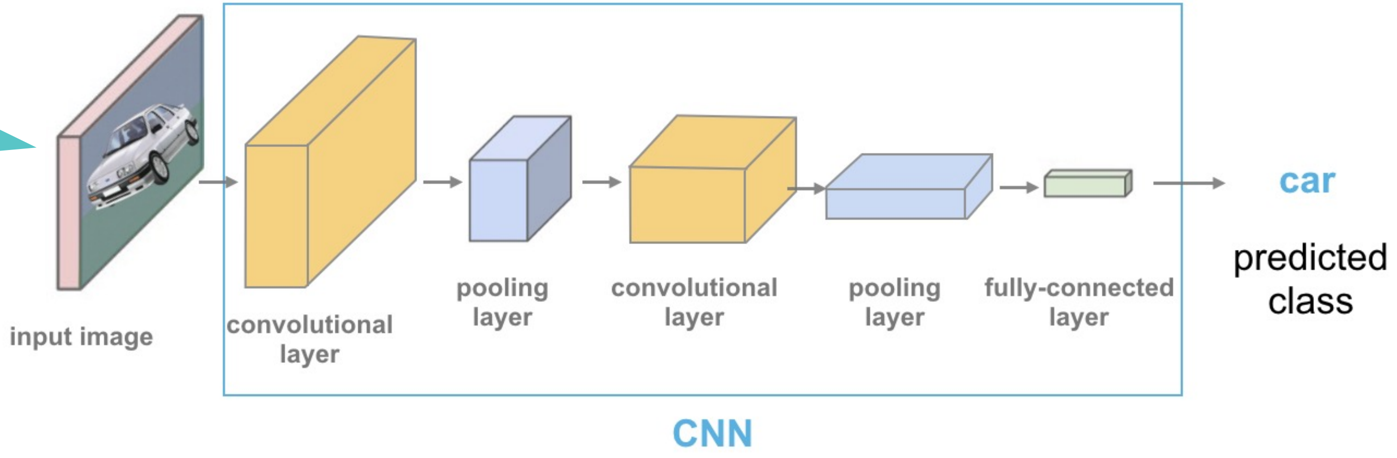
Data (image, sequence, word, etc.) to be classified
What happens if you enter a sequence?

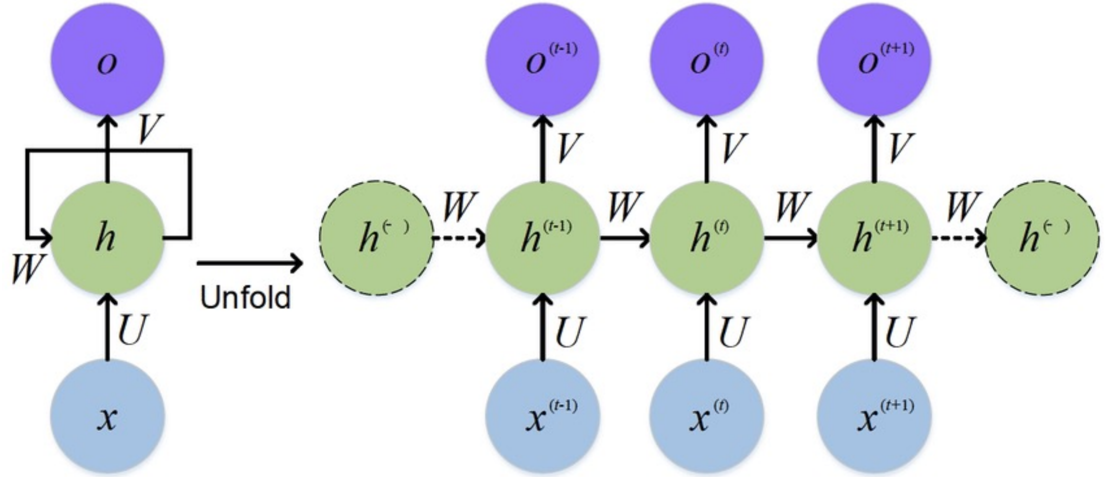


Background

CNN vs. RNN

Data (image, sequence, word, etc.) to be classified
What happens if you enter a sequence?

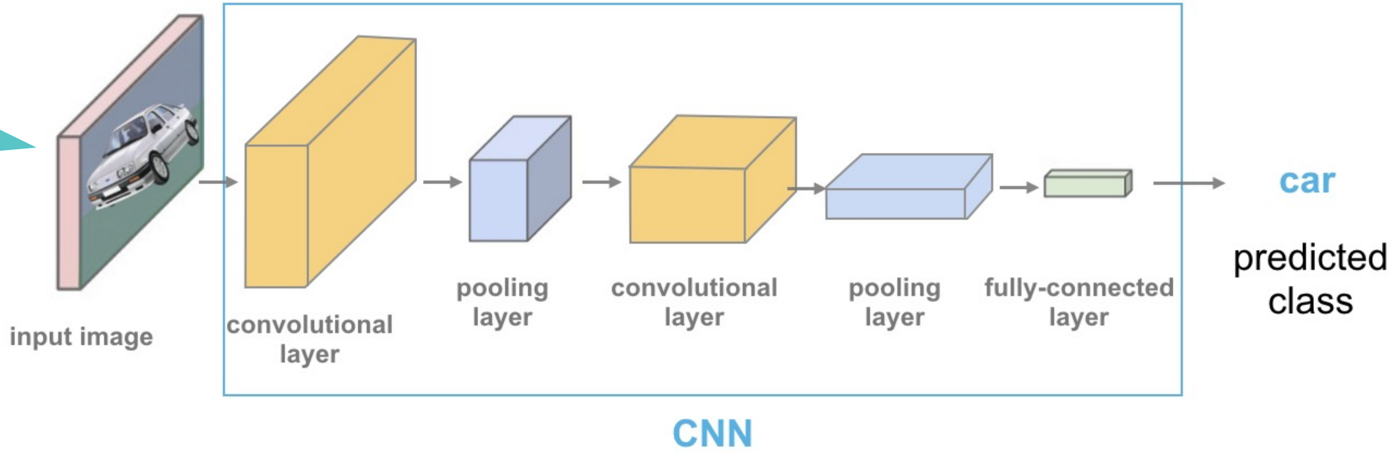




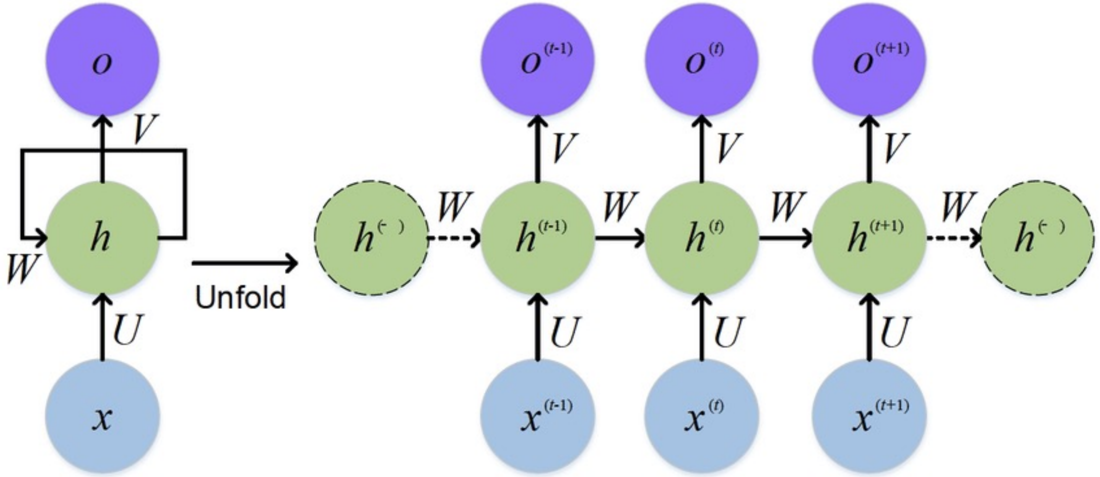
Background

CNN vs. RNN

Data (image, sequence, word, etc.) to be classified
What happens if you enter a sequence?



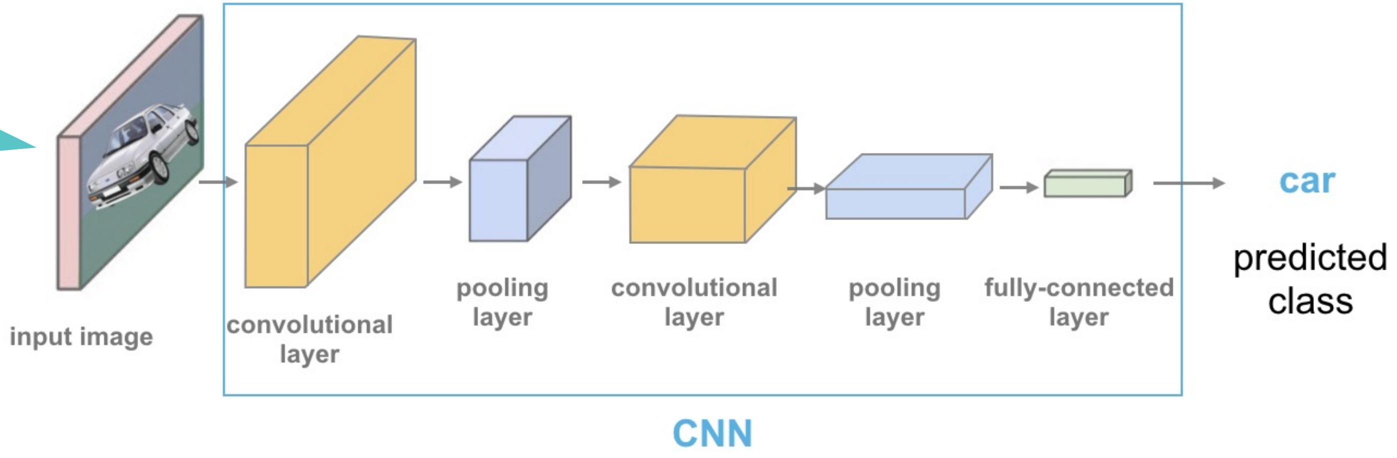
Sequence (conversation, text, video) of data with order
The data is correlated depending on the previous text



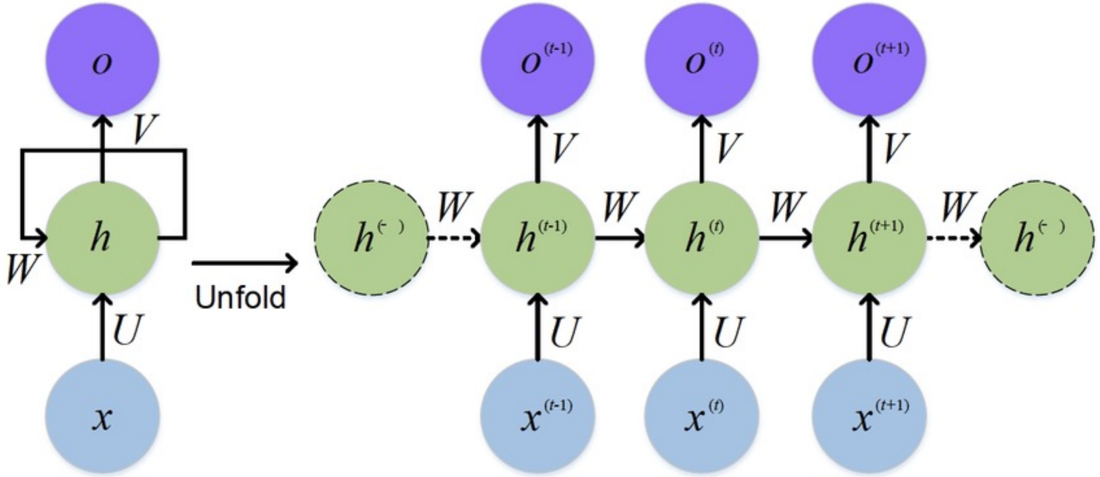
Background

CNN vs. RNN

Data (image, sequence, word, etc.) to be classified
What happens if you enter a sequence?



To do this, use the
from the previous iteration
to give new results,
i.e., has a memory



¡Gracias!

Dr. Manuel Castillo-Cara
www.manuelcastillo.eu
Departamento de Inteligencia Artificial
Escuela Técnica Superior de Ingeniería Informática
Universidad Nacional de Educación a Distancia (UNED)