Introducción al Deep Learning

Día 3: Procesamiento de imágenes y secuencias

Manuel Germán y David de la Rosa

Universidad de Jaén

Universidad de Jaén

SIMIDAT
INTELLIGENT SYSTEMS AND DATA MINING

(mgerman,drrosa)@ujaen.es

Cuestiones previas

1. ¿Qué es el *Deep Learning*?
2. ¿Qué es una neurona?
3. ¿Qué es una red neuronal?
4. ¿Cómo se programa una red neuronal?
5. ¿Cómo se evalúa una red neuronal?

Expectativas

Tras esta sesión, sabremos:

- Por qué es importante procesar imágenes.
- Qué es exactamente una imagen.
- Qué son los kernels.
- Cómo se introducen los kernel en una red neuronal.
- Qué es y cómo se construye una CNN (Convolutional Neural Network)
- Cómo implementar una CNN usando *Pytorch* y *Pytorch Lightning*.

1

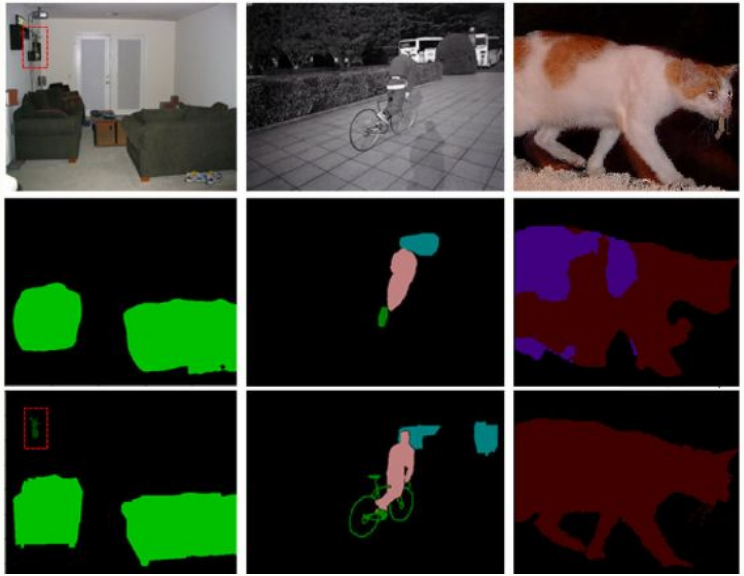
¿Por qué procesar imágenes?

¿Por qué procesar imágenes?

¿Qué animal aparece en la imagen?

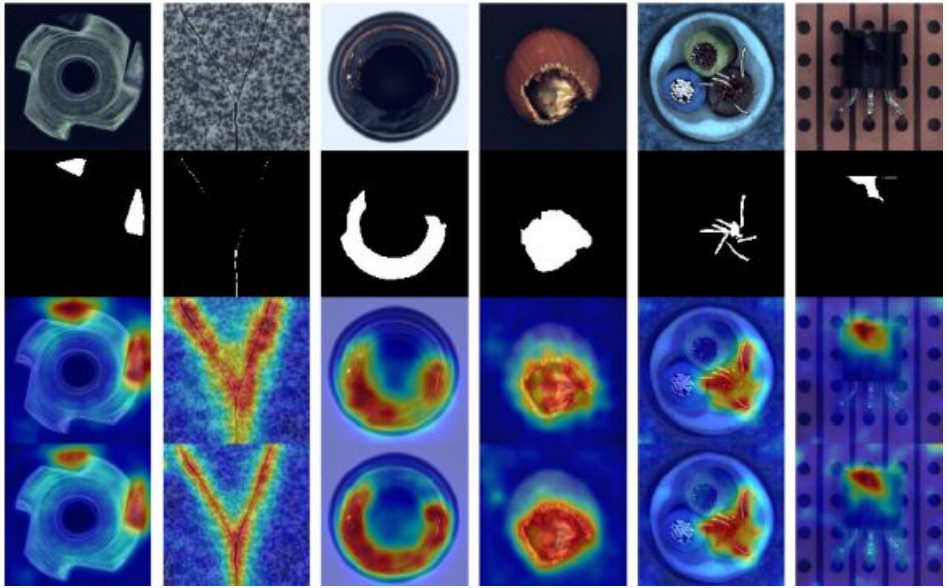
¿Por qué procesar imágenes?

¿Dónde están los objetos interesantes en estas imágenes?



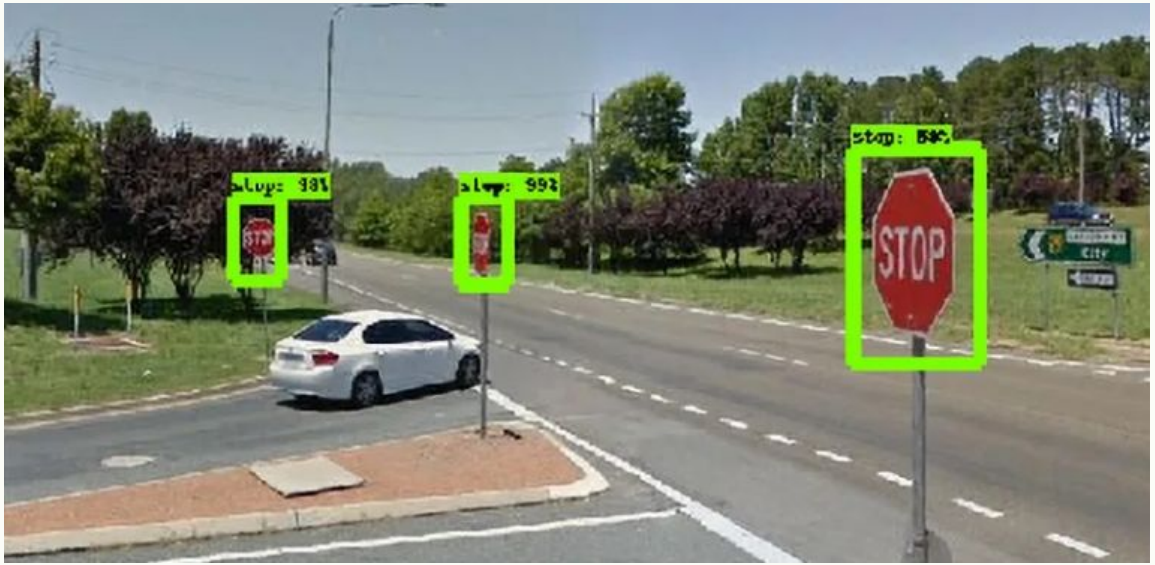
¿Por qué procesar imágenes?

¿Qué anomalías hay en estas imágenes?



¿Por qué procesar imágenes?

¿Qué señales de tráfico aparecen en estas imágenes?

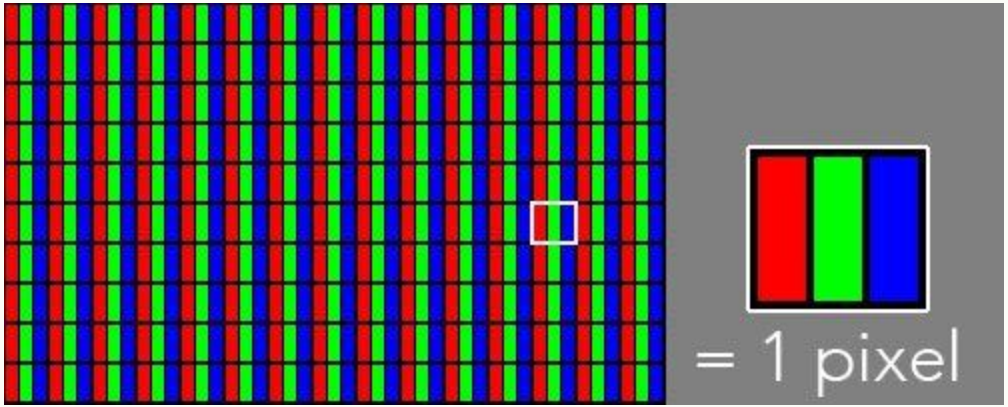


2

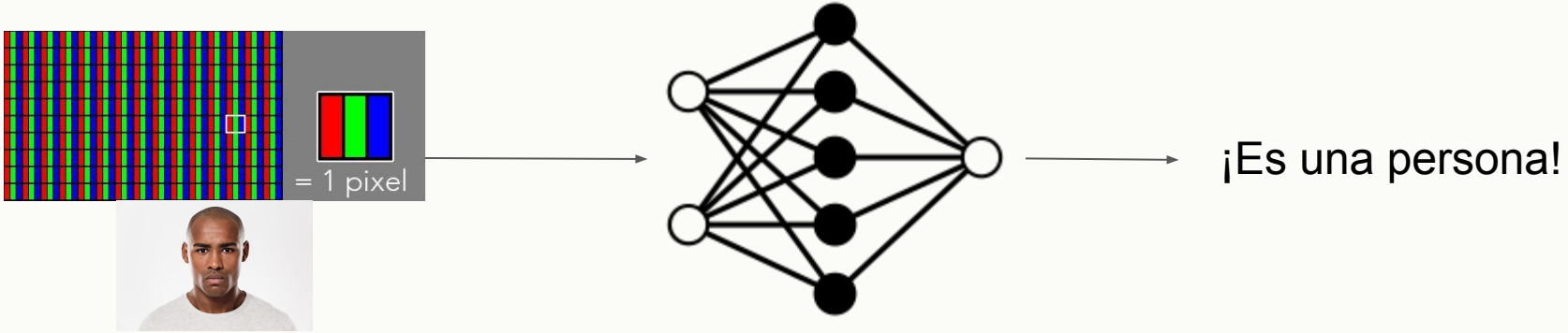
¿Qué es una imagen?

¿Qué es una imagen?

¿Qué es una imagen?



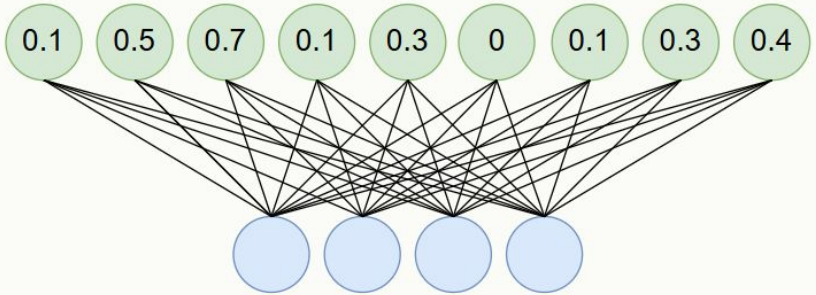
Idea: procesar con redes neuronales



Problemas…

Imagen 3x3
(en realidad son mucho más grandes)

0.1	0.5	0.7
0.1	0.3	0
0.1	0.3	0.4



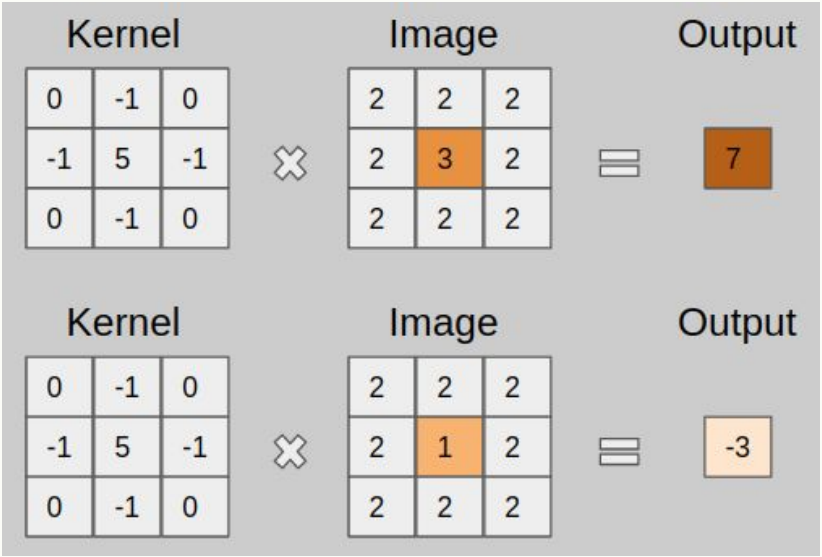
36 conexiones
Y si la imagen es más grande?
Y si la capa tiene más neuronas?

3

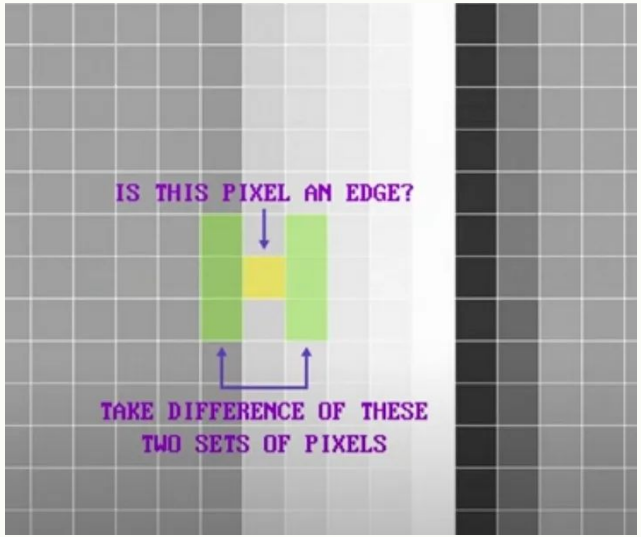
Solución: CNN

Kernel

Antes de entrar en detalle sobre qué es una CNN, hay que hablar de los **kernels**



¿Para qué sirven los kernels?



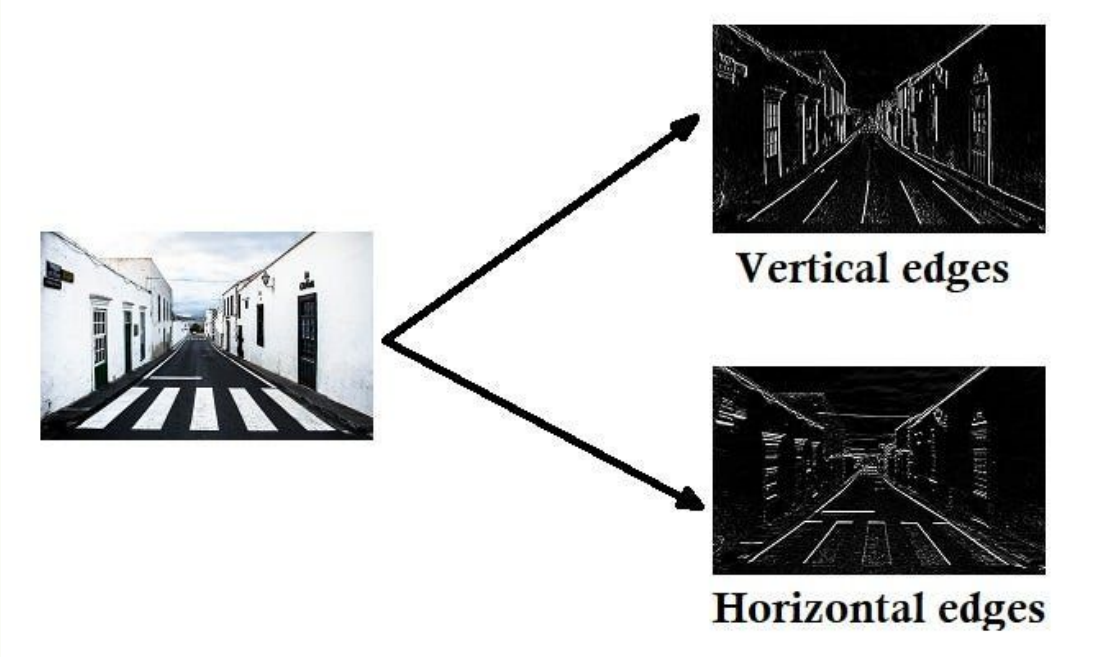
vertical

-1	0	1
-1	0	1
-1	0	1

horizontal

-1	-1	-1
0	0	0
1	1	1

¿Para qué sirven los kernels?



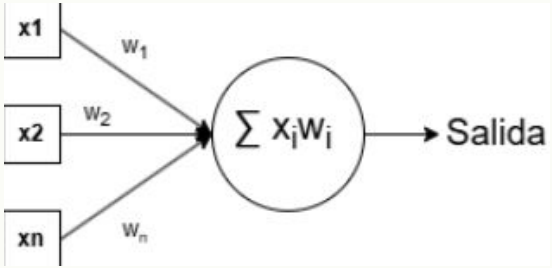
¿Qué relación tiene con las redes neuronales?

Kernel → Extraer características de las imágenes para utilizarlas a posteriori. Detección de bordes, de caras, de objetos…

Red neuronal → Extraer características de los datos en las capas intermedias para utilizarlas a posteriori.

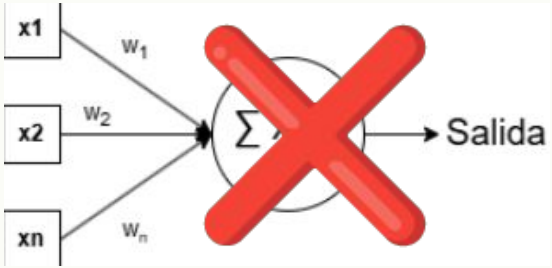
¿Y si lográramos que trabajaran juntos?

Se propone una solución



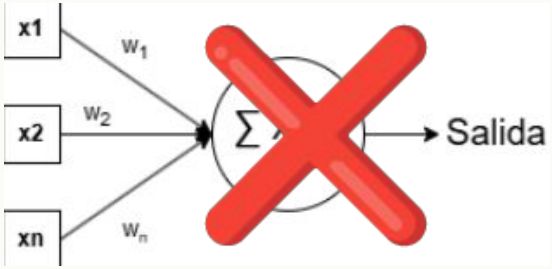
Backpropagation Applied to Handwritten Zip Code Recognition

Se propone una solución



Backpropagation Applied to Handwritten Zip Code Recognition

Se propone una solución



w1	w2	w3
w4	w5	w6
w7	w8	w9

Kernel con parámetros que se aprenden

Backpropagation Applied to Handwritten Zip Code Recognition

Redes neuronales convolucionales (CNN)

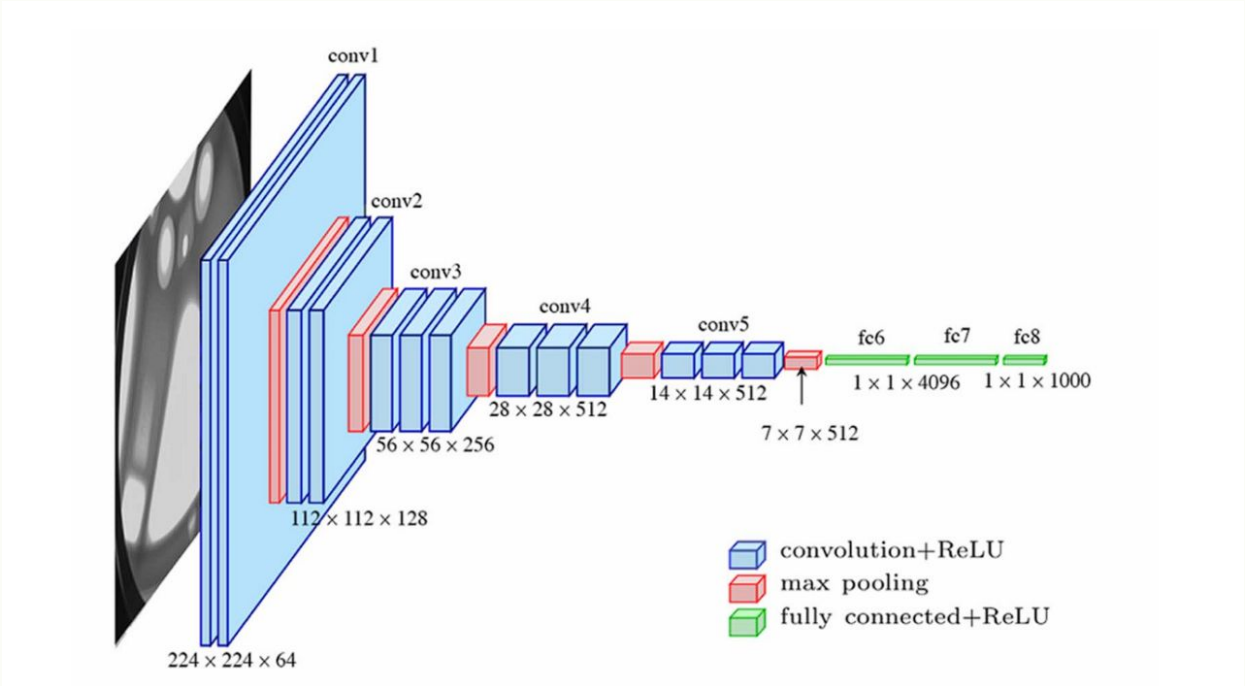
¿Recordamos los dos problemas que había? Eficiencia y capturar patrones entre píxeles

¡Ambos han desaparecido!

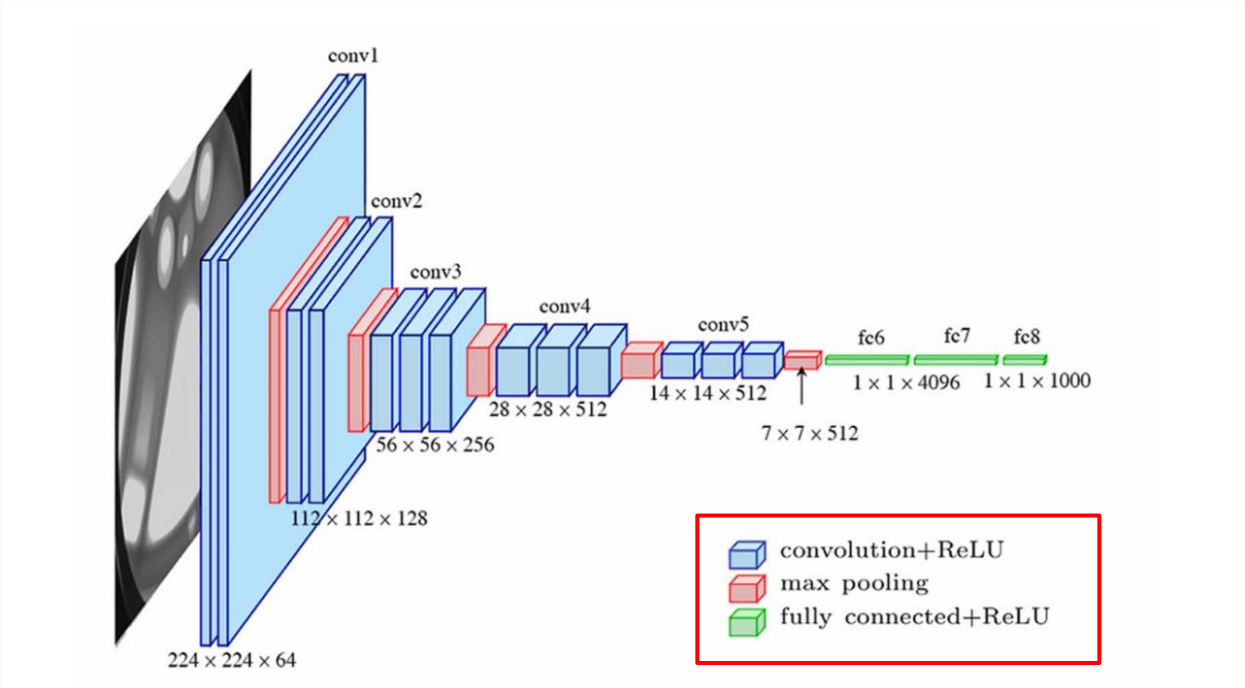
Eficiencia → Se usa el mismo kernel (y los mismos pesos) para procesar la imagen al completo

Patrones → Como los kernel actúan sobre grupos de píxeles, se puede capturar la relación entre píxeles de diferentes filas y columnas

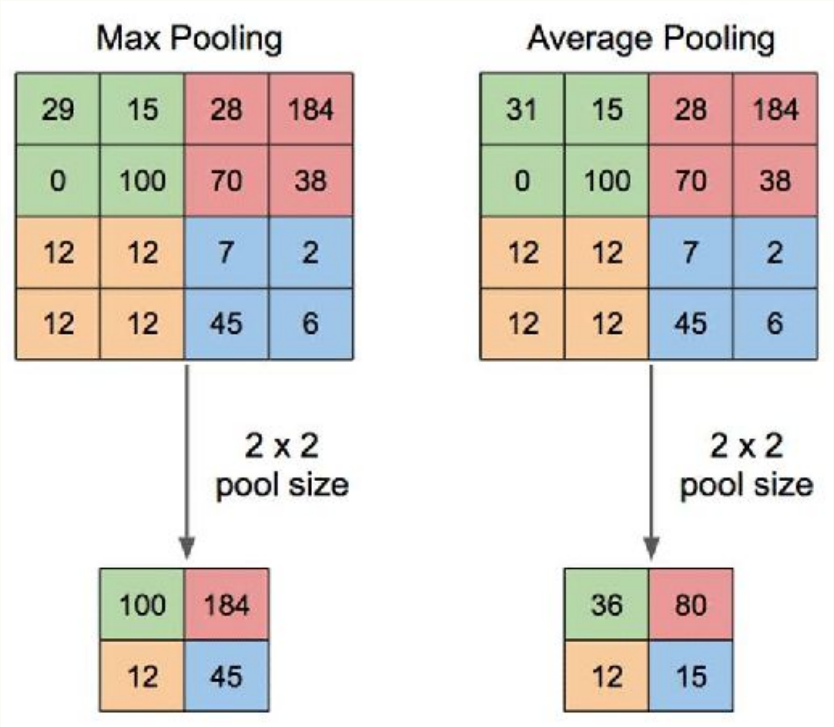
Redes neuronales convolucionales (CNN)



Redes neuronales convolucionales (CNN)



Estructura de una CNN



Hiperparámetros de una capa convolucional

Tres hiperparámetros clave:

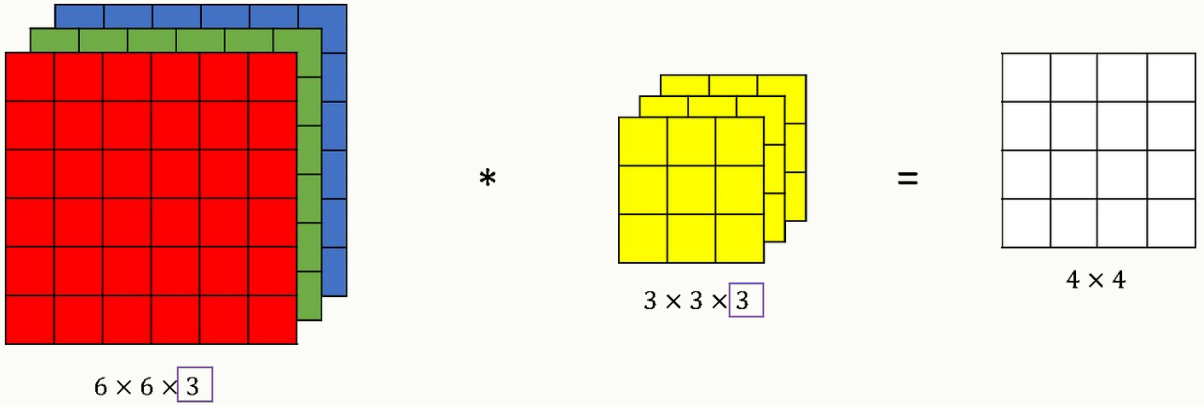
- Padding
- Kernel size
- Stride

Vamos a analizarlos de forma interactiva:
https://poloclub.github.io/cnn-explainer/

Convoluciones 2D y 3D

Cuando aplicamos una **convolución 2D** a una imagen, el filtro solo se mueve en **dos direcciones**.

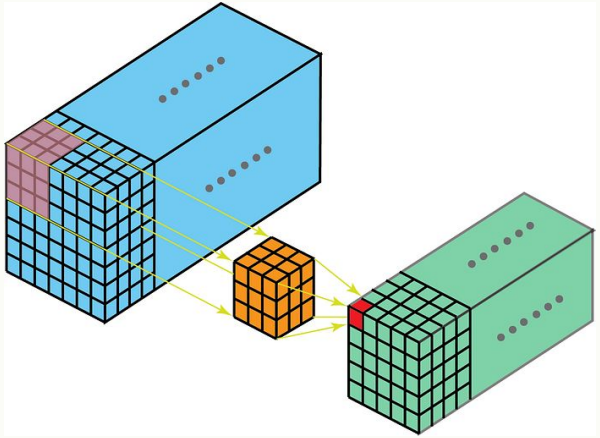
Dimensiones del filtro: [anchura, altura, canales (1 gris, 3 RGB)]



Convoluciones 2D y 3D

Cuando aplicamos una **convolución 3D** a una secuencia de imágenes, el filtro se mueve en **tres direcciones**.

Dimensiones del filtro: [anchura, altura, profundidad, canales (1 gris, 3 RGB)]



Vamos a programar una CNN!