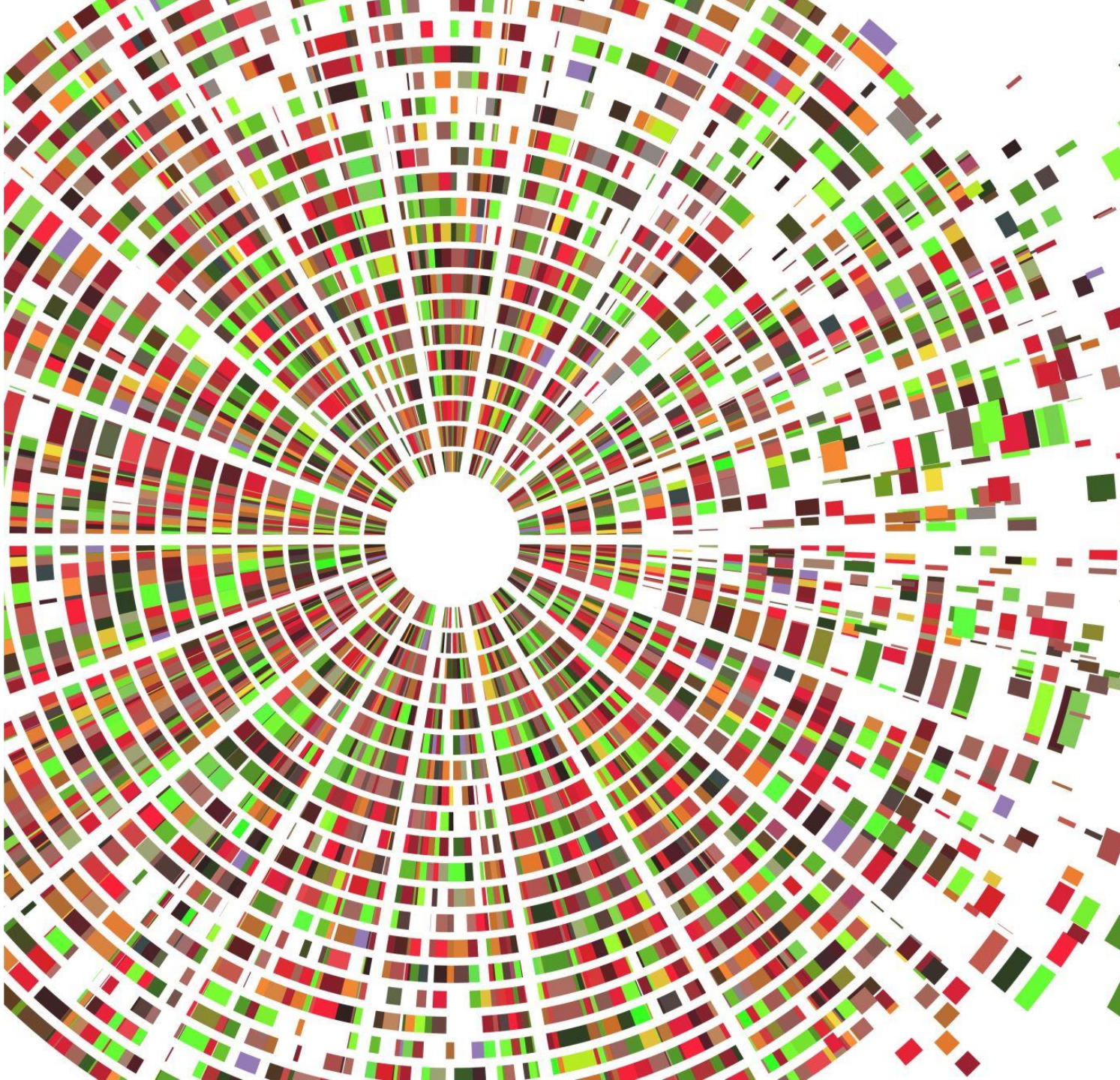
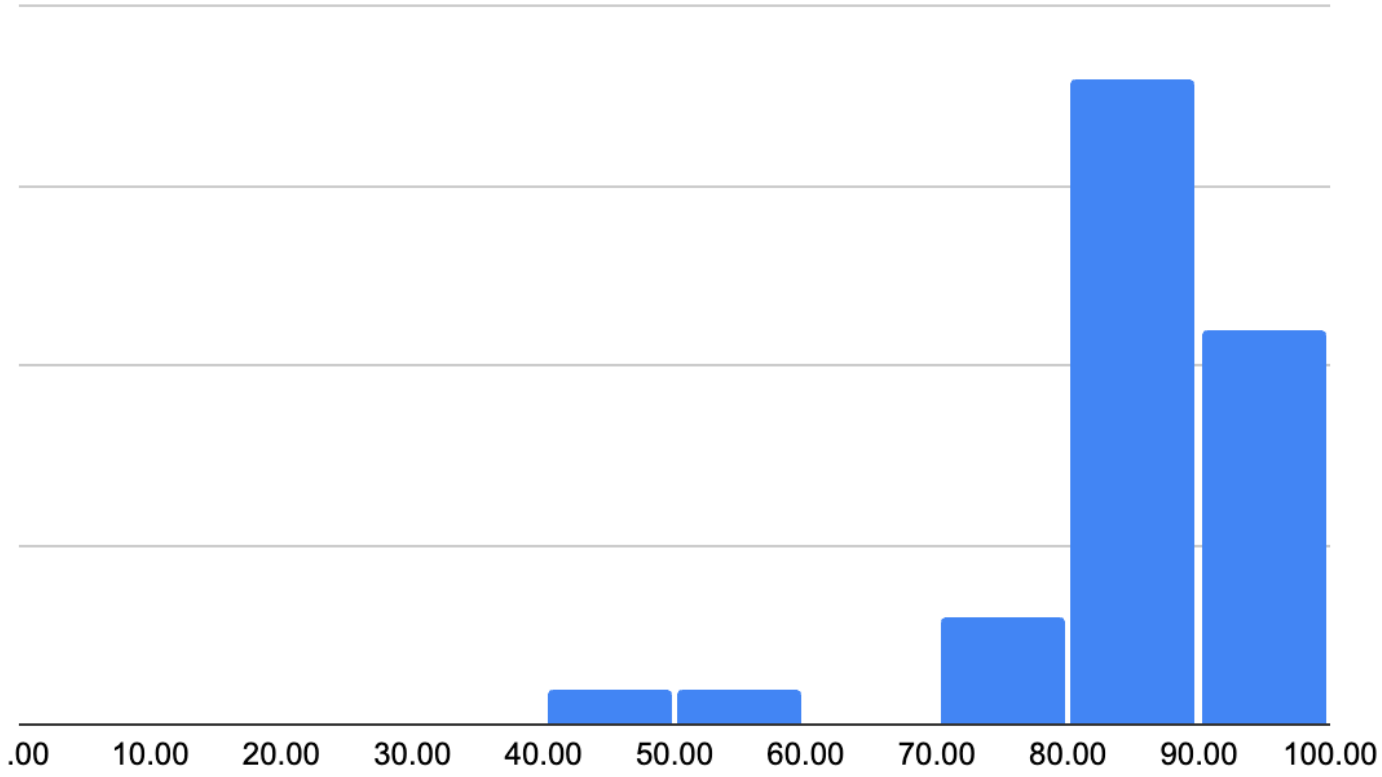
Métodos avanzados de ciencia de datos

Prof. Emily Díaz



Sobre el examen



Continuación de bases de redes neuronales convolucionales (CNN)

Ejemplo matemático

Imagen

1	2	3
4	5	6
7	8	9

Kernel

1	0
0	-1

2 Cálculo del primer valor

1*1 + 2*0 + 4*0 + 5*-1 = -4	

Ejemplo matemático

Imagen

1	2	3
4	5	6
7	8	9

Kernel

1	0
0	-1

3

Cálculo del segundo valor

-4	2*1 + 3*0 + 5*0 + 6*-1 = -4

Ejemplo matemático

Imagen

1	2	3
4	5	6
7	8	9

Kernel

1	0
0	-1

4 Cálculo del tercer valor

-4	-4
4*1 + 5*0 + 7*0 + 8*-1 = -4	

Ejemplo matemático

Imagen

1	2	3
4	5	6
7	8	9

Kernel

1	0
0	-1

4 Cálculo del cuarto valor

-4	-4
-4	5*1 + 6*0 + 8*0 + 9*-1 = -4

Con padding/relleno

Kernel/Filtro

0	1	0
0	1	1
1	0	0

Imagen + Relleno

0	0	0	0	0	0
0	1	2	3	4	0
0	5	6	7	8	0
0	9	10	11	12	0
0	13	14	15	16	0
0	0	0	0	0	0

Resultado

3	10	13	11
12	24	28	23
24	40	44	35
36	39	42	28

- Añadimos bordes de 0 como relleno para preservar el tamaño de la imagen y no perder información de los bordes.
- El tamaño del relleno depende del tamaño del kernel y del stride/paso. Para que quede el mapa del mismo tamaño que el original, la fórmula es:

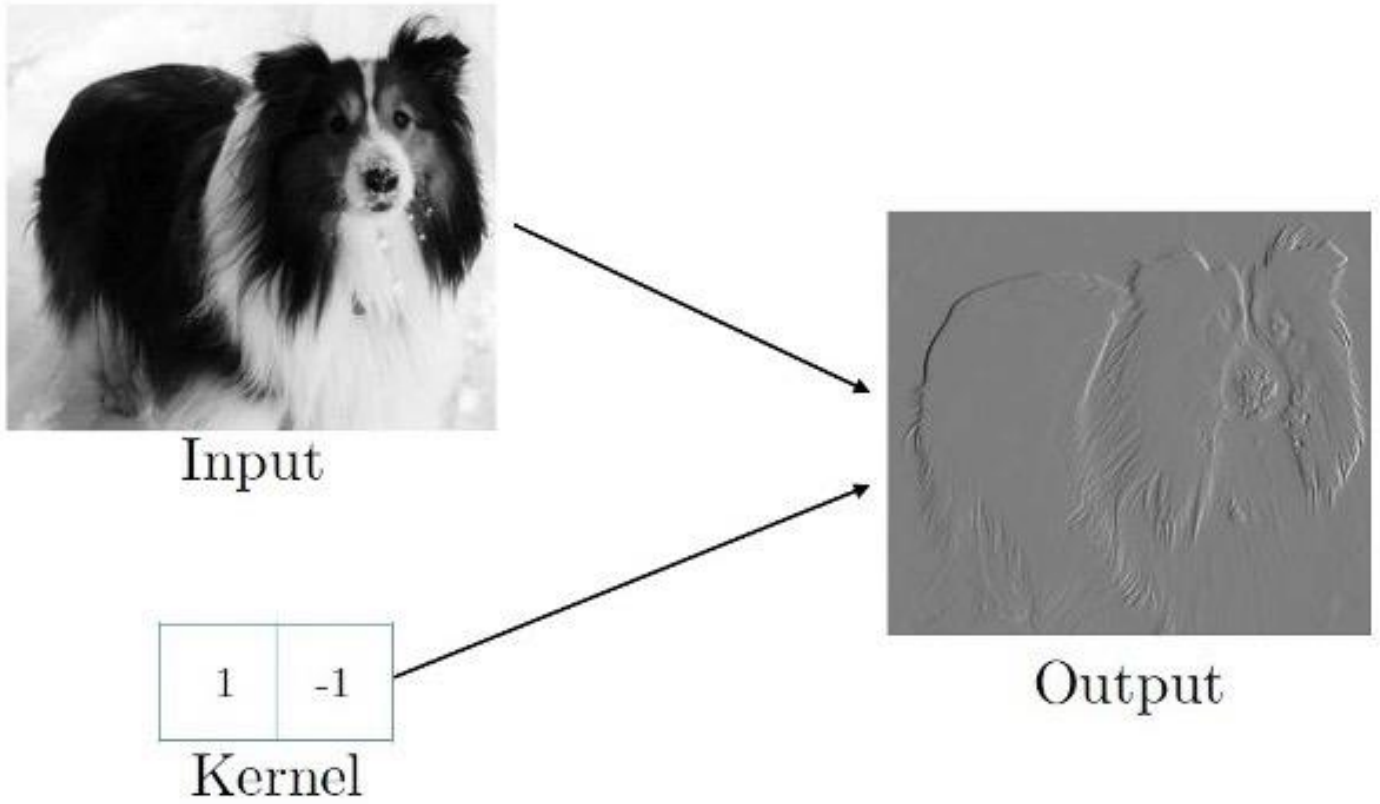
$$P = \frac{S * (H - 1) - H + F}{2}$$

- Donde H es la altura (asumiendo que la imagen es cuadrada), F el tamaño del filtro y S el stride
- Ejemplo con paso 1

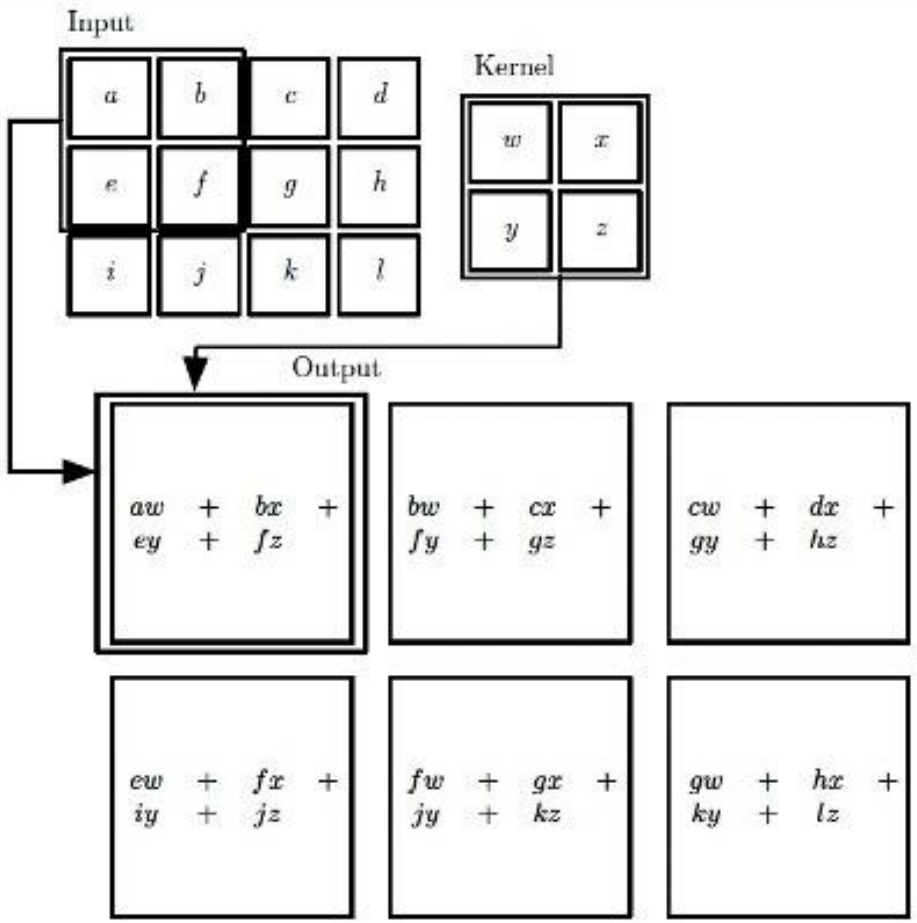
Imagen original

$$P = \frac{1 * (4 - 1) - 4 + 3}{2} = \frac{3 - 4 + 3}{2} = 1$$

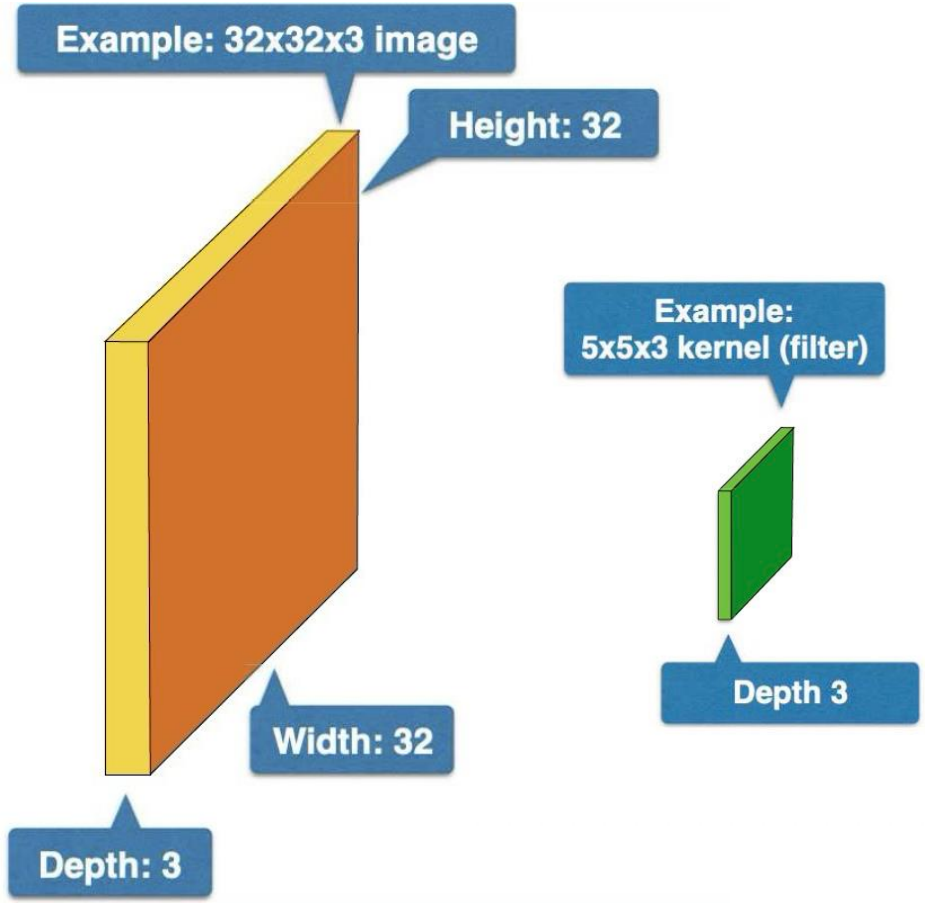
Los filtros sirven como detectors de distintas características dependiendo de sus coeficientes



Convolución en 2D- general

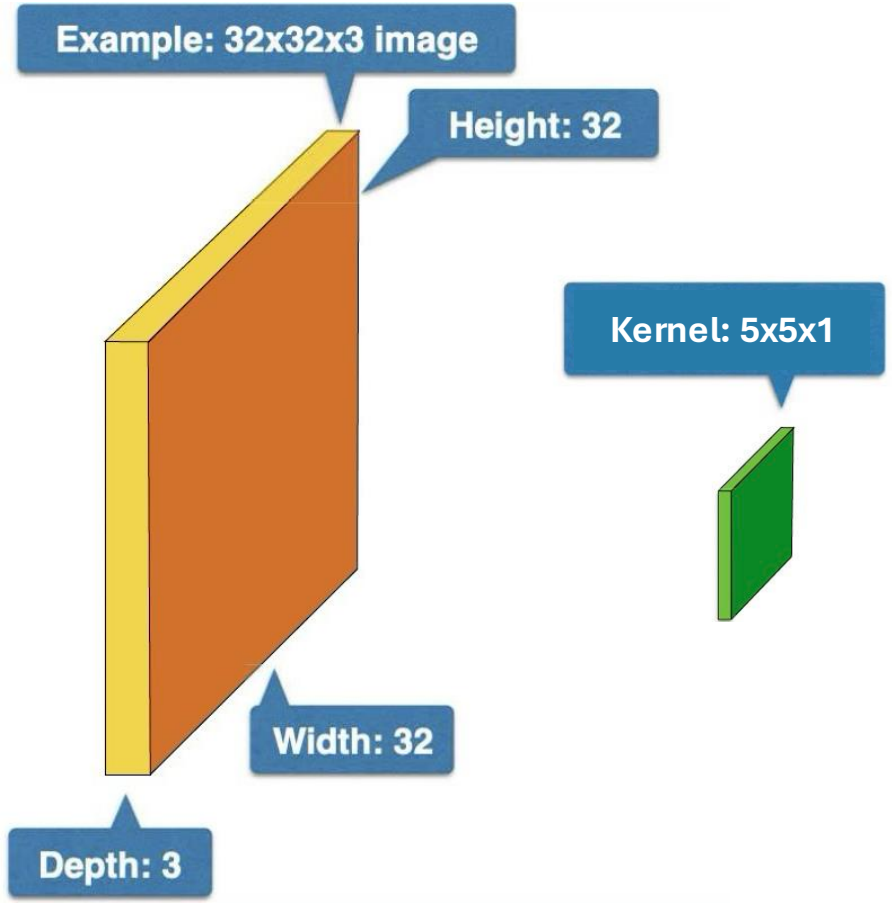


Convolución cuando tenemos 3 canales de color



- Al utilizar un kernel 5x5x3, cada posición en el mapa de características se calcula como **una suma ponderada de 5x5x3** = 75 valores (5x5 valores espaciales de cada uno de los 3 canales).
- El resultado es un **único valor por posición** en el mapa de características.
- Por lo tanto, el mapa de características resultante tiene un **único canal.**
- Resultado final asumiendo un paso de 1 y sin relleno: **28x28x1**

Convolución cuando tenemos 3 canales de color



- Al utilizar un **kernel 5x5x1**, significa que es una convolución en profundidad, lo que significa que el kernel 5x5 **se aplica independientemente a cada canal**.
- Cada canal de la entrada (32x32) se convoluciona con un kernel 5x5 independiente.
- Esto da como resultado 3 mapas de características independientes de 28x28 (asumiendo un paso 1 y sin relleno)
- Estos 3 mapas de características se pueden combinar de diferentes maneras según el método de convolución específico, como sumando o concatenando, pero si se interpretan como canales independientes, el resultado sería **28x28x3**.

Cómo se ve una capa de convolución en Keras

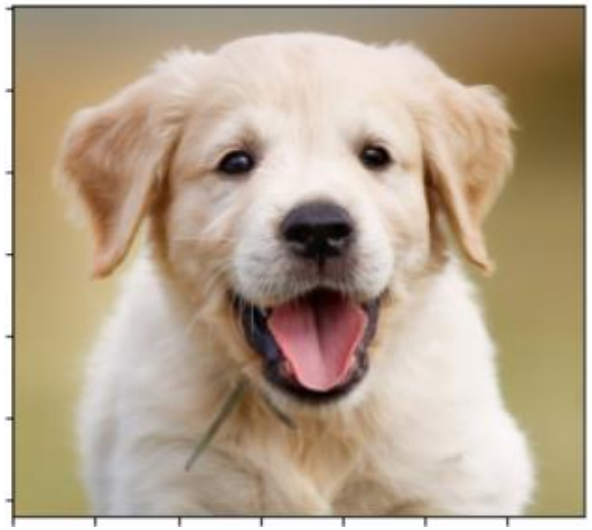
```
model.add(Conv2D(filters=8,
          kernel_size=(5, 5),
          input_shape=(32, 32, 3)))
```

Número de kernels a aplicar

Tamaño del kernel (5x5)

Tamaño de la imágenes. Solo necesario en la primera capa, las otras infieren el tamaño.

Ejemplo aplicando convoluciones



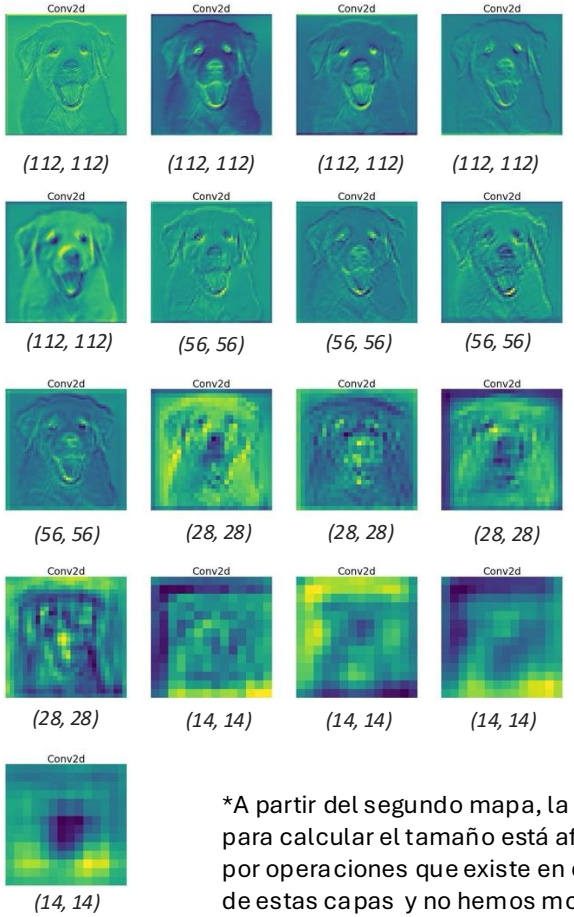
Resolución: 224 x 224

Modelo con 17 capas convolucionales

```
[    Conv2d(3, 64, kernel_size=(7, 7), stride=(2, 2), padding=(3, 3), bias=False),
   Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False),
   Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False),
   Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False),
   Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False),
  Conv2d(64, 128, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False),
Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False),
Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False),
Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False),
Conv2d(128, 256, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False),
Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False),
Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False),
Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False),
Conv2d(256, 512, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False),
Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False),
Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False),
Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)]
```

3904 mapas de características en total

Visualizando el mapa de características promedio por capa

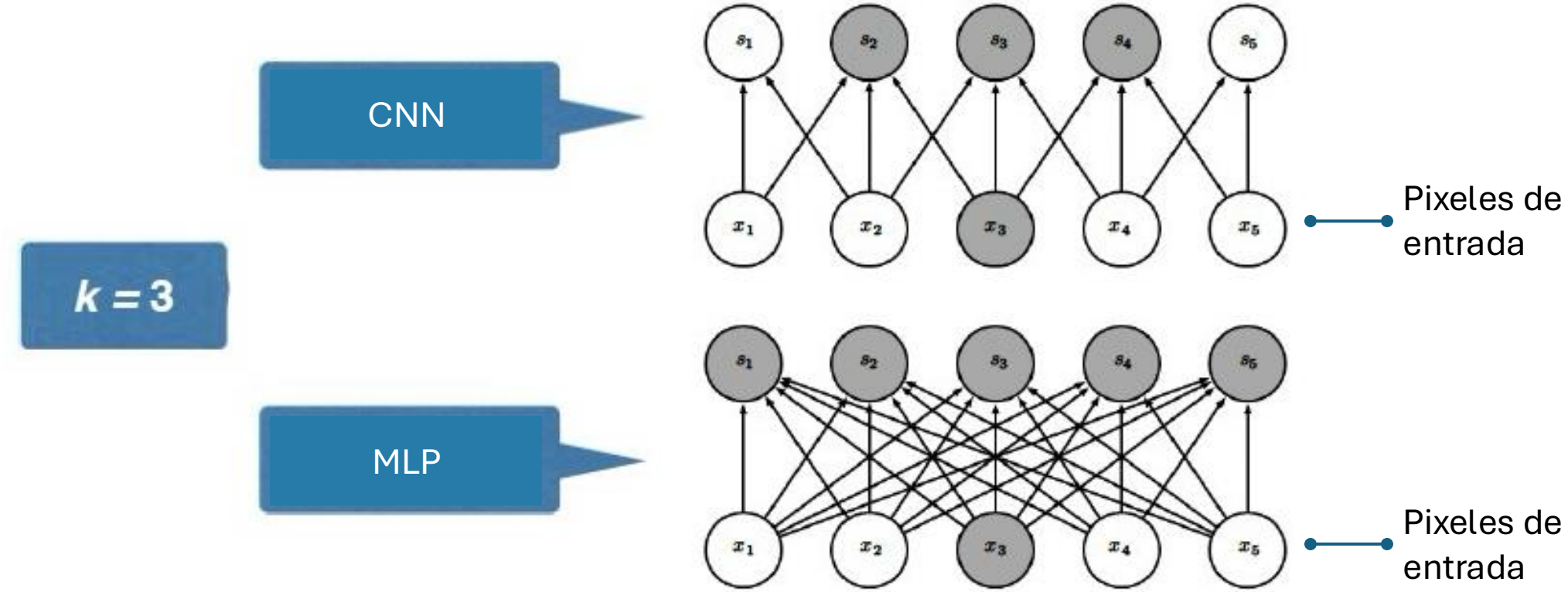


*A partir del segundo mapa, la fórmula para calcular el tamaño está afectada por operaciones que existe en el medio de estas capas y no hemos mostrado (como capas de agrupamiento/poling)

$$Tamaño\ del\ mapa = \frac{Imag\ Orig\ - Filtro + 2\ * Relleno}{Paso} + 1$$

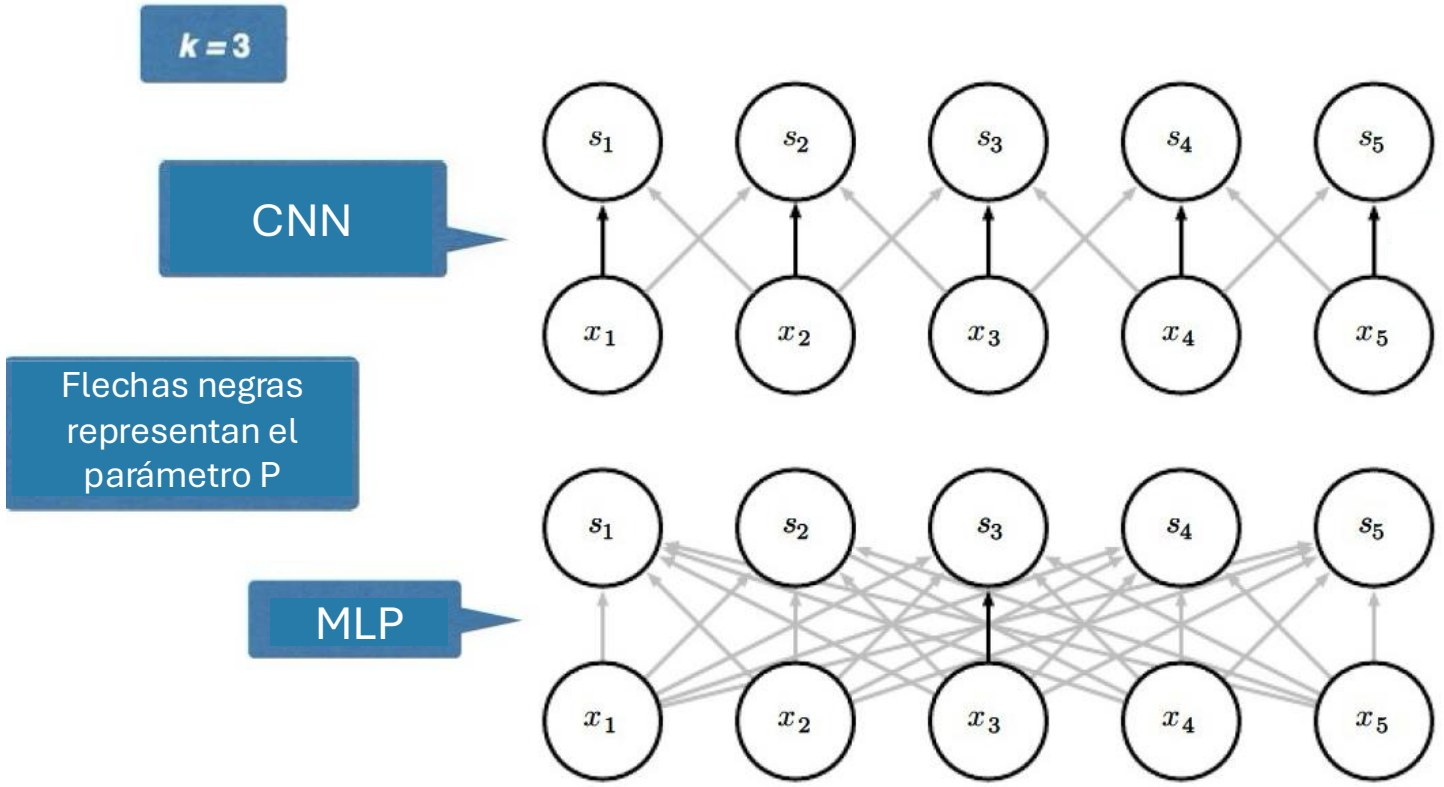
Vista gráfica de propiedades de las CNN

Interacciones dispersas con k= 3 (kernel 3x3)



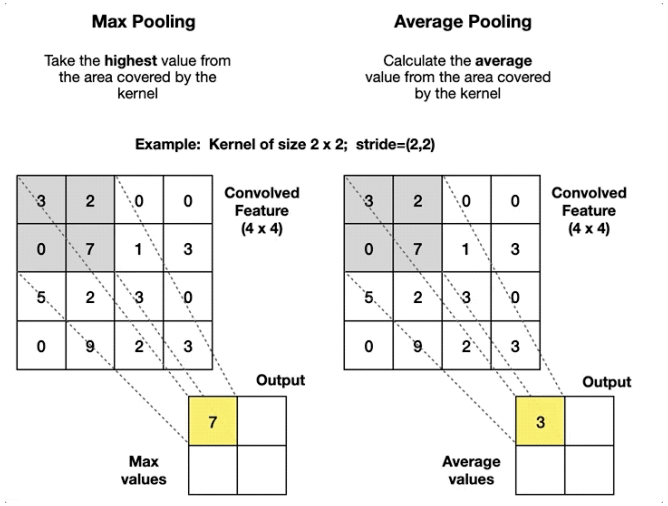
Vista gráfica de propiedades de las CNN

Parámetros compartidos

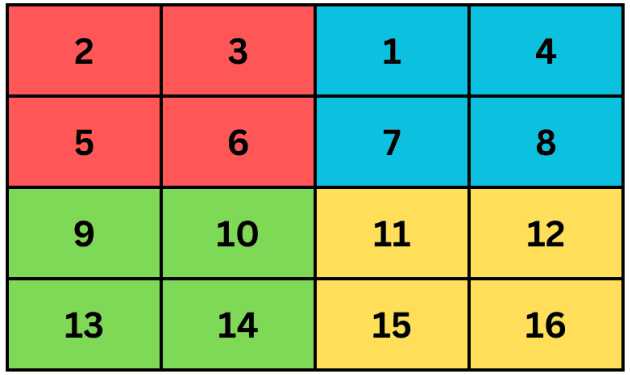


Otro tipo de capa: pooling o de agrupamiento

- La operación de agrupamiento implica deslizar un filtro bidimensional sobre cada canal del mapa de características y **resumir las características que se encuentran dentro de la región cubierta por el filtro.**
- Lógica detrás: Abordar la sensibilidad de la ubicación de las características mediante la reducción de la resolución de los mapas de características. Esto tiene el efecto de hacer que los mapas de características resultantes sean más resistentes a los cambios en la posición, lo que se conoce como *"invariancia de la traslación local"*
- Existe utilizando el máximo o el promedio como operación
- También se debe definer el stride/paso
- Padding/relleno típicamente no es usado para este tipo de capa
- Típicamente se aplican luego de realizar la convolución



Ejemplo de cálculo de agrupamiento/pooling



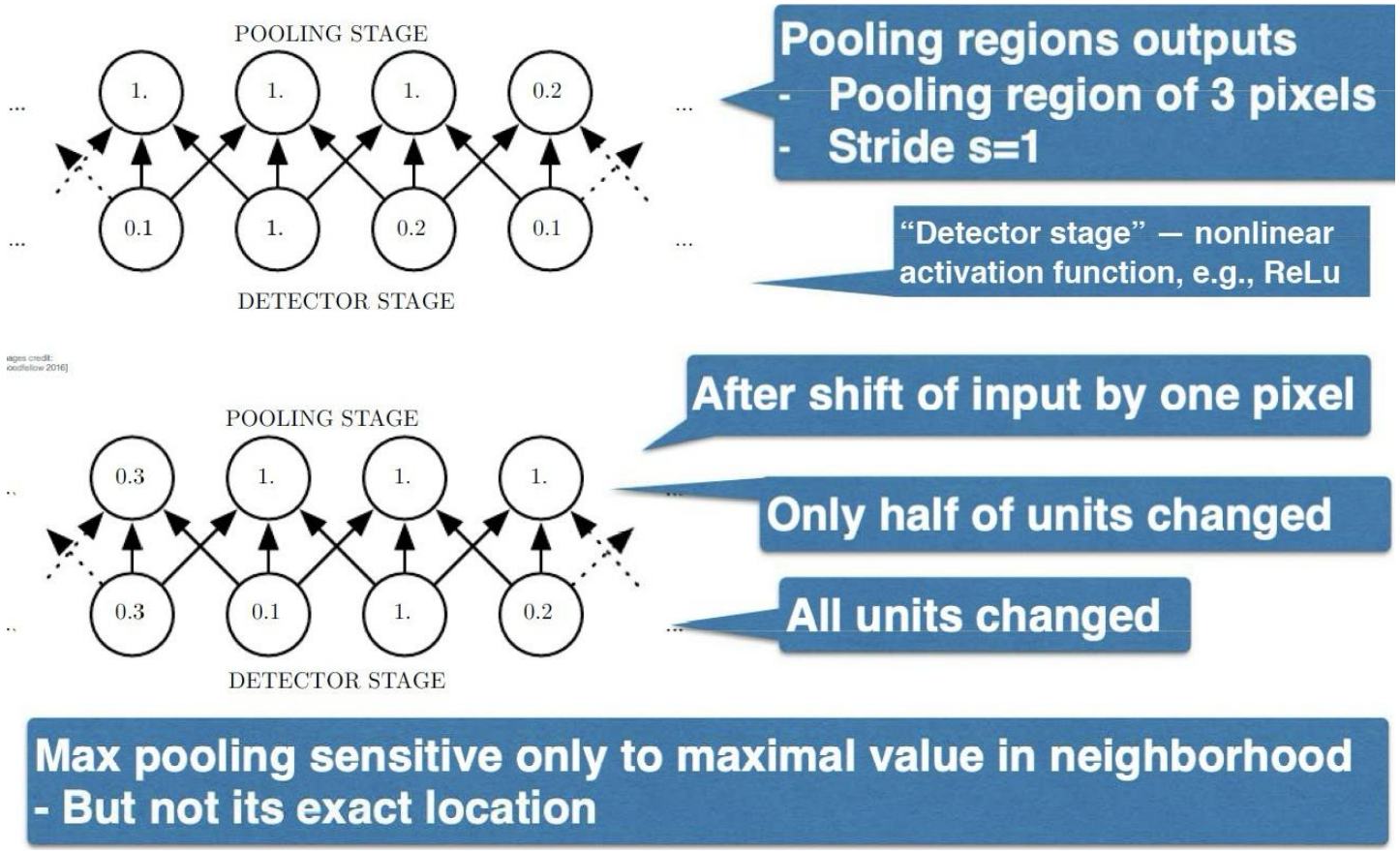
Con paso=2, con un filtro 2x2

Resultados

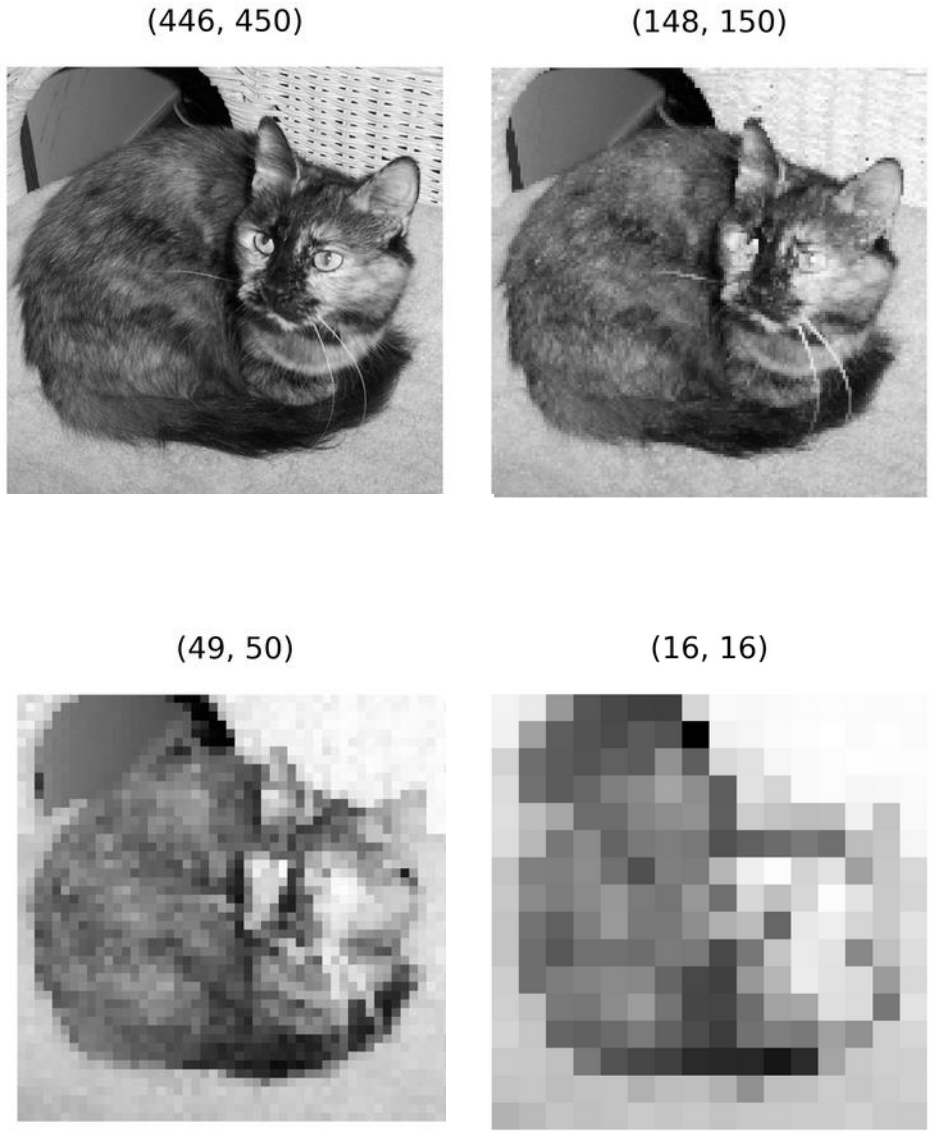
Con Promedio

Con Máximo

Visualización de la invarianza



- El agrupamiento máximo es sensible únicamente al máximo valor del vecindario pero **no a su ubicación exacta**
- Al mover los valores un pixel hacia el lado podemos ver en el ejemplo que **solo algunas unidades han cambiado.** Contrario a si no usaramos agrupamiento, donde todos cambiarían.
- Esto se debe a que **cada píxel es relevante** en la convolución, y el desplazamiento de un píxel cambia las entradas del filtro de convolución en cada paso.

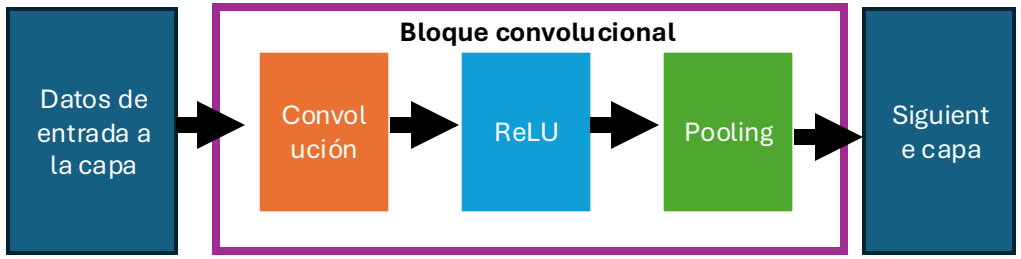


Pooling e invarianza local

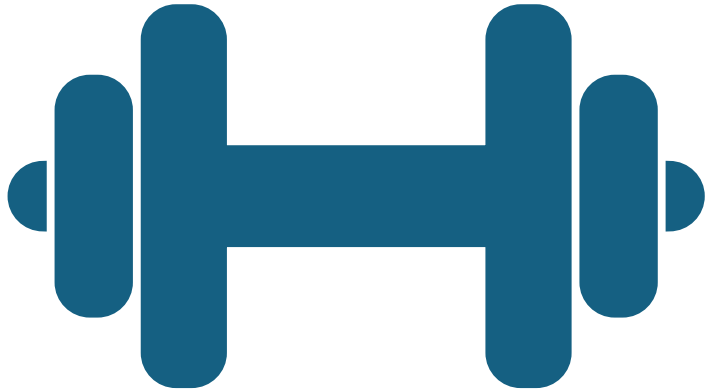
- Recordar que pooling hace **reducción de muestreo**, **no es una capa que aprende** aspectos de los datos, es una operación para ayudarnos con la invarianza espacial
- Por ende, **aquí no tenemos distintos filtros por capa**, solo uno que define el tamaño de la operación a aplicar

Arquitectura general de CNN

- Típicamente las CNN incluyen distinto número de las siguientes operaciones:
  - **Convolución**
  - **Detector:** Aplicación de no-linealidad (ReLU). Esto permite aprender patrones complejos. También se pueden utilizar otras funciones de activación, como la sigmoid o tanh, pero ReLU es la más común en la práctica.
  - **Pooling/agrupamiento**
- Las anteriores encapsulan lo que es conocido como bloque/capa de convolución (aunque sea un conjunto de capas) y se tiende a construir una arquitectura con muchas de estas agrupaciones una tras otra aprendiendo distintas representaciones, cada vez a nivel más profundo/abstracto.
- Las primeras capas pueden aprender características simples como bordes, mientras que las capas más profundas aprenden características más complejas como formas y objetos.



Entrenamiento



Diferencias en el entrenamiento de CNN con respecto a MLP

- El entrenamiento de redes neuronales convolucionales (CNN) y perceptrones multicapa (MLP) implica **principios similares**, como el **uso de descenso de gradiente y retropropagación**. Sin embargo, existen algunas diferencias clave en sus procesos de entrenamiento debido a sus arquitecturas distintas y la naturaleza de los datos que suelen manejar.
- En las CNN, **los gradientes se calculan para cada filtro** en las capas convolucionales en función de las **ubicaciones espaciales a** las que se aplican.
- El proceso de retropropagación implica el cálculo de gradientes para pesos compartidos, lo que puede ser más eficiente que los MLP. **Los gradientes de los pesos compartidos se promedian en diferentes ubicaciones espaciales.**
- Las actualizaciones de peso en las CNN implican la **actualización de filtros completos en lugar de pesos individuales.**

En la práctica, cómo resolver problemas con el entrenamiento de CNN

- **Si el entrenamiento falla:**
  - Problema con los datos de entrada
  - Tasa de aprendizaje alta
  - Problemas con la retropropagación y sus gradients
  - Problemas de inicialización
  - Sobreajuste temprano
- **Si el rendimiento es muy bajo:**
  - Función de pérdida no es la apropiedad
  - Falta de ajuste, intentar redes más complejas
  - Problemas con la retropropagación
  - Datos insuficientes o no representativos
  - Desbalance de clases
- **Mucho tiempo entrenando:**
  - Red excesivamente grande
  - Necesidad de GPUs
  - Incorporación de técnicas de optimización
  - Datos no procesados adecuadamente

En general:

- Evaluar continuamente las métricas de pérdida de entrenamiento y validación
- Ajustar hiperparámetros (tasa de aprendizaje, tamaño de lote, arquitectura, algoritmo de optimización, etc)

Ejercicios

Para la siguiente imagen, calcule...

- Convolución de 2x2 con paso/stride = 1 y sin relleno
- Seguido de max pooling/agrupamiento máximo de 2x2
- De qué tamaño va a ser el resultado final?
- Muestre el mapa de características resultante

$$Mapa(conv) = \left(\frac{Tamaño\ original - Tamaño\ del\ kernel + 2 * relleno}{paso}\right) + 1$$

$$Mapa\ (pool) = \left(\frac{Tamaño\ original - Tamaño\ del\ kernel}{paso}\right) + 1$$

Imagen original

10	**20**	**10**	**5**	**10**
4	**3**	**10**	**5**	**20**
2	**15**	**18**	**2**	**10**
8	**6**	**16**	**22**	**10**
2	**0**	**1**	**0**	**5**

Kernel

5	**2**
4	**1**

Para la siguiente imagen, calcule…

- El resultado de convolución va a ser: $\left(\frac{5-2}{1}\right) + 1 = 4 \rightarrow 4x4$
- El resulta del max pooling: $\left(\frac{4-2}{1}\right) + 1 = 3 \rightarrow 3x3$

Cálculo

Información dada

10	20	10	5	10
4	3	10	5	20
2	15	18	2	10
8	6	16	22	10
2	0	1	0	5

5	2
4	1

Convolución

10*5 + 20*2 + 4*4 + 3*1 = 109	20*5+10*2 +4*3+10*1 = 142	105	85
4*5 + 3*2 + 4*2 + 15*1 = 49	113	134	83
78	151	180	128
60	63	128	135

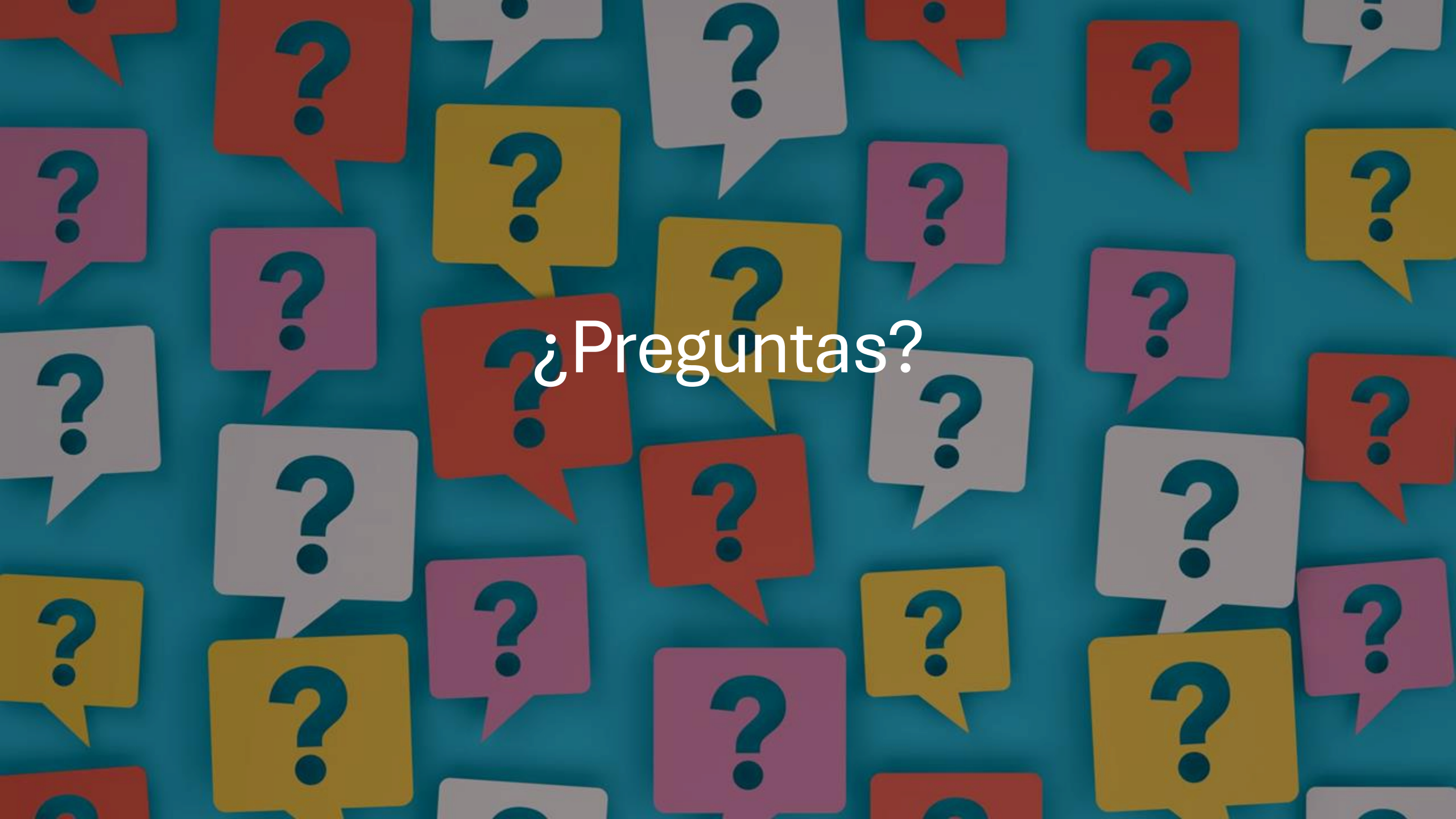
142	142	134
151	180	180
151	180	180

Agrupamiento máximo

Resultado final

Tarea moral

- Hacer el ejercicio con padding/relleno para la convolución y obtener el resultado. Calcular cuanto tiene que ser el padding/relleno basado en la fórmula dada
- Realizarlo con agrupamiento medio en lugar de máximo


¿Preguntas?