Redes convolucionales

Métodos Generativos, curso 2025-2026

Guillermo Iglesias, guillermo.iglesias@upm.es
Jorge Dueñas Lerín, jorge.duenas.lerin@upm.es
Edgar Talavera Muñoz, e.talavera@upm.es

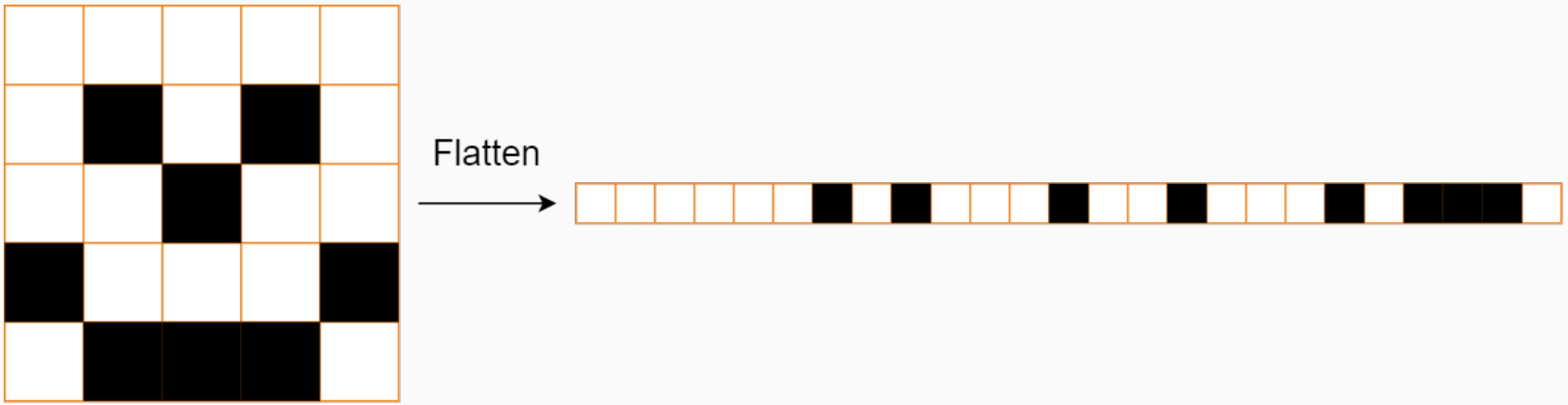
7 de octubre de 2025

Escuela Técnica Superior de Ingeniería de Sistemas Informáticos | UPM

Motivación

Problemas del perceptrón

Como anteriormente se ha visto, una arquitectura de perceptrón es capaz de tratar con imágenes. Para ello las matrices bidimensionales o tridimensionales son transformadas a un vector unidimensional con la operación de "flatten".



Problemas del perceptrón

El principal inconveniente de esta aproximación es que se pierde toda la información espacial de la imagen.

Esto hace que se pierdan las relaciones de distancia y color.

Otro problema es la enorme magnitud de las redes creadas de esta manera.

512x512x3 píxeles = 786.432 neuronas entrada

Redes convolucionales

Las redes neuronales convolucionales surgen para adaptar las redes neuronales al tratamiento de imágenes.

Los principales beneficios de su uso son los siguientes:

- Aprovechamiento de la información espacial.
- Reducción del número de parámetros.
- Invarianza aprendida de los datos (traslaciones, escalas, deformaciones).

Fundamentos de las redes convolucionales

Operación de convolución

La operación de convolución consiste en la combinación lineal de una ventana de píxeles de una imagen.

Para ello hay dos elementos fundamentales:

- Imagen de entrada: Una matriz bidimensional de datos (normalmente normalizada a [-1, 1] o [0, 1]).
- Filtro o kernel: Una matriz (normalmente de 3x3 o 5x5) con la que se realizará la combinación lineal de los elementos de la imagen.

Input

2	1	4	0
1	2	2	0
3	1	2	1
0	0	-1	1

Kernel

0	1	-1
0	1	2
0	1	0

Operación de convolución

La salida se calcula haciendo una combinación lineal de cada región de la imagen. De esta manera la salida contiene la activación de cada zona de la imagen.

Esta región que el kernel es capaz de observar se conoce como campo receptivo.

- Capa 1: kernel 3×3, campo receptivo 3x3; Capa 2: kernel 3×3 aplicado sobre la salida de la primera capa, campo receptivo de cada neurona es 5×5; ...

Input

2	1	4	0
1	2	2	0
3	1	2	1
0	0	-1	1

Kernel

0	1	-1
0	1	2
0	1	0

Output

4	8
5	5

Ejemplo de convolución 2-D

f

2	1	4	0
1	2	2	0
3	1	2	1
0	0	-1	1

g

0	1	-1
0	1	2
0	1	0

s

4	

Ejemplo de convolución 2-D

f

2	1	4	0
1	2	2	0
3	1	2	1
0	0	-1	1

g

0	1	-1
0	1	2
0	1	0

s

4	8

Ejemplo de convolución 2-D

f

2	1	4	0
1	2	2	0
3	1	2	1
0	0	-1	1

g

0	1	-1
0	1	2
0	1	0

s

4	8
5	

Ejemplo de convolución 2-D

f

2	1	4	0
1	2	2	0
3	1	2	1
0	0	-1	1

g

0	1	-1
0	1	2
0	1	0

s

4	8
5	5

Ejemplo de convolución 2-D

f

2	1	4	0
1	2	2	0
3	1	2	1
0	0	-1	1

g

0	1	-1
0	1	2
0	1	0

s

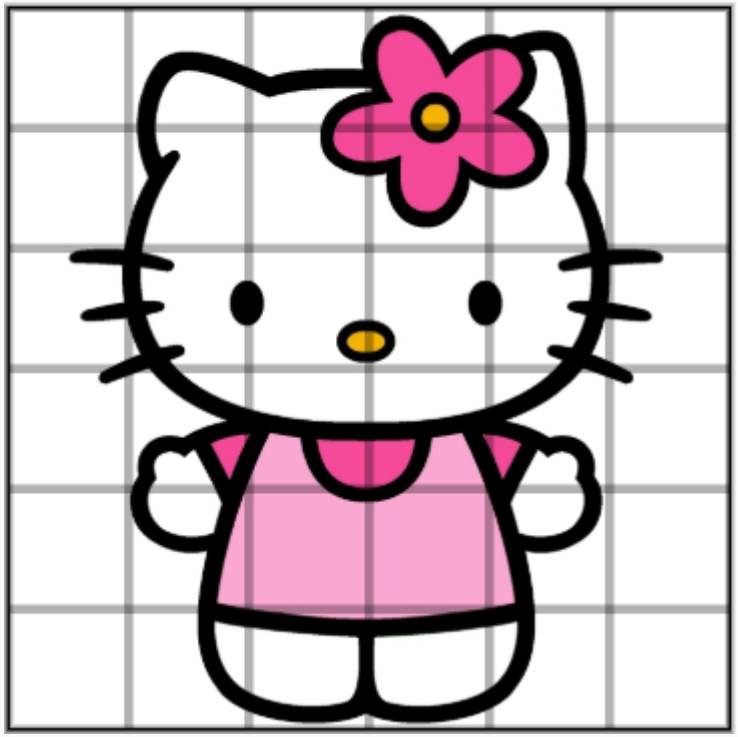
4	8
5	5

Campo receptivo

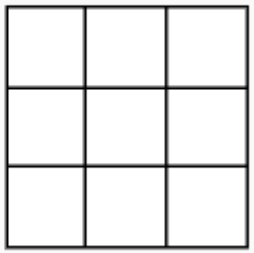
La salida de la operación tiene como objetivo la extracción de características de las distintas regiones de la imagen.

El campo receptivo de cada celda de la salida se activa cuando detecta una estructura de interés.

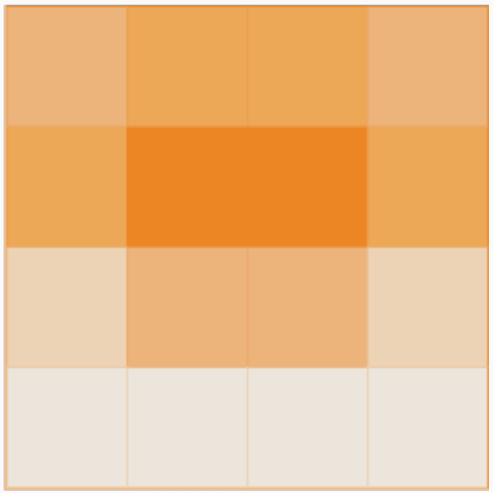
Input



Kernel



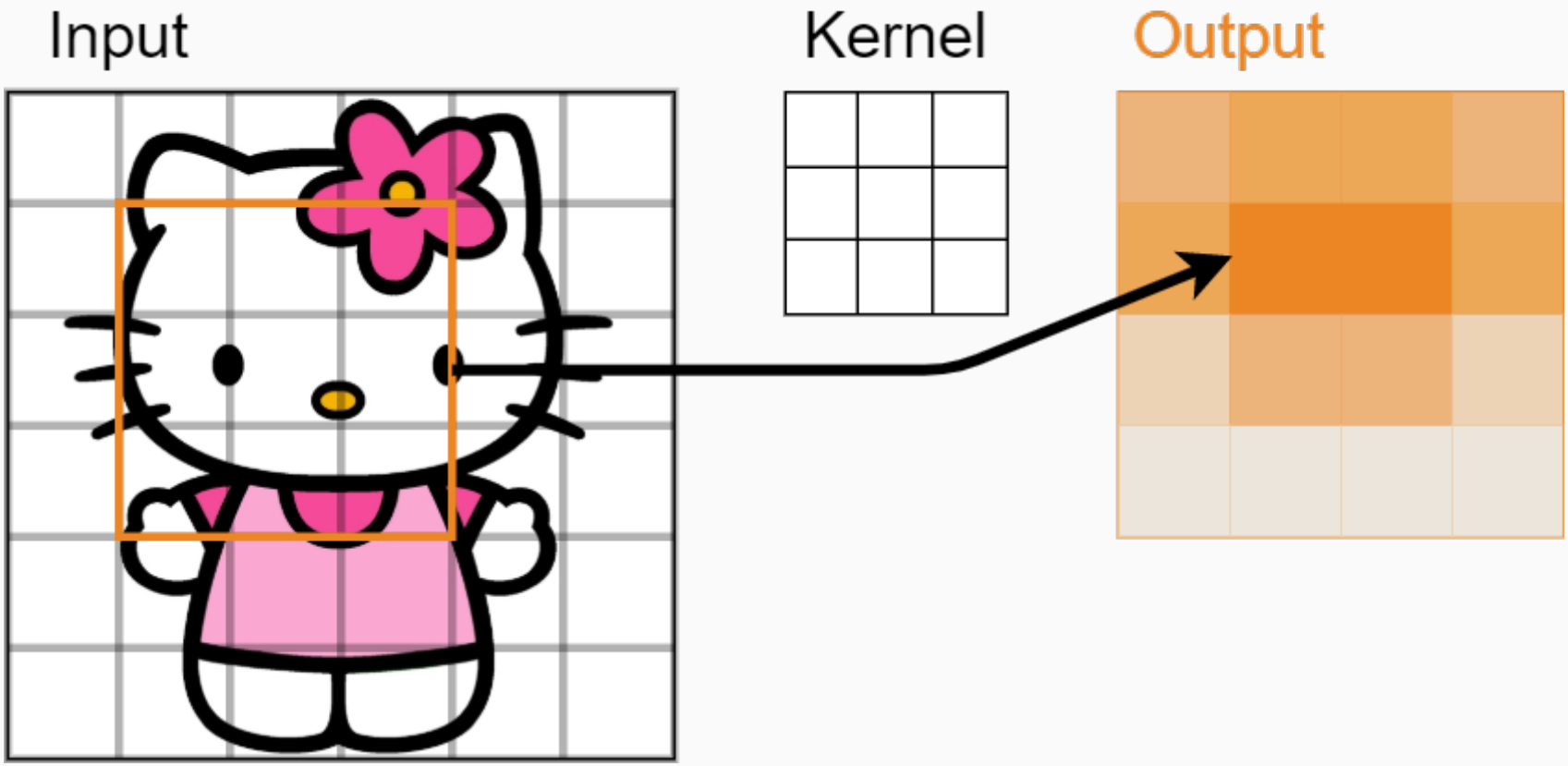
Output



Campo receptivo

La salida de la operación tiene como objetivo la extracción de características de las distintas regiones de la imagen.

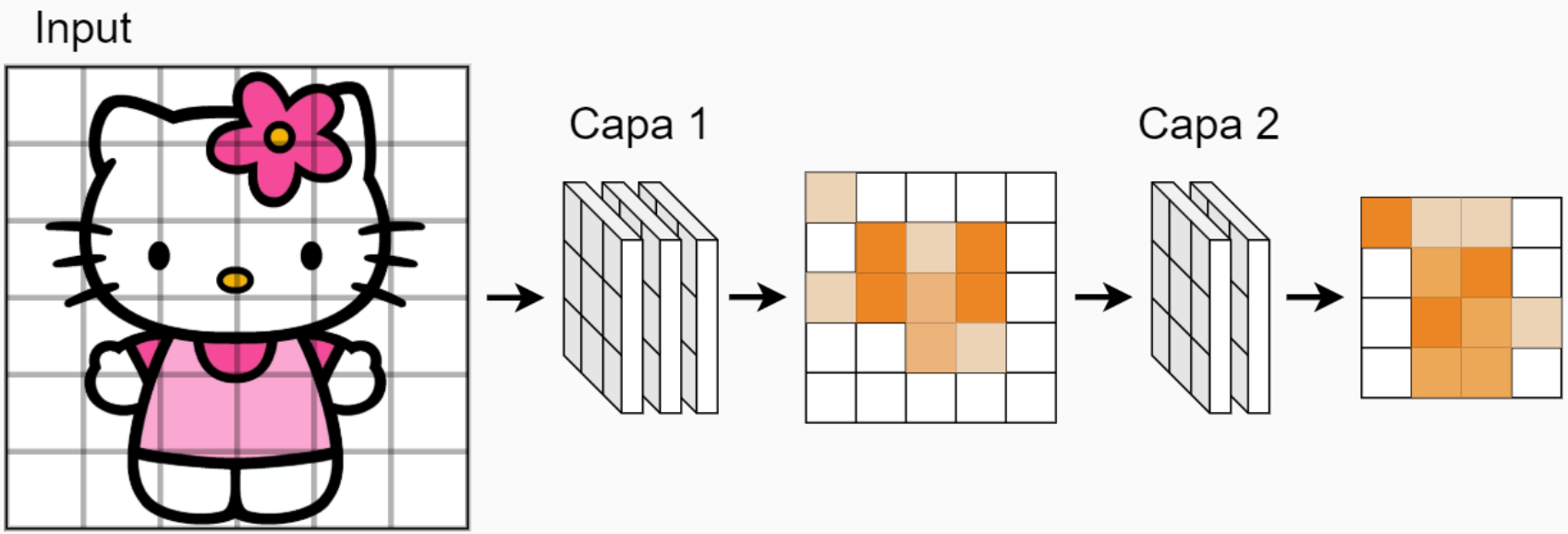
El campo receptivo de cada celda de la salida se activa cuando detecta una estructura de interés.



De neuronas a convoluciones

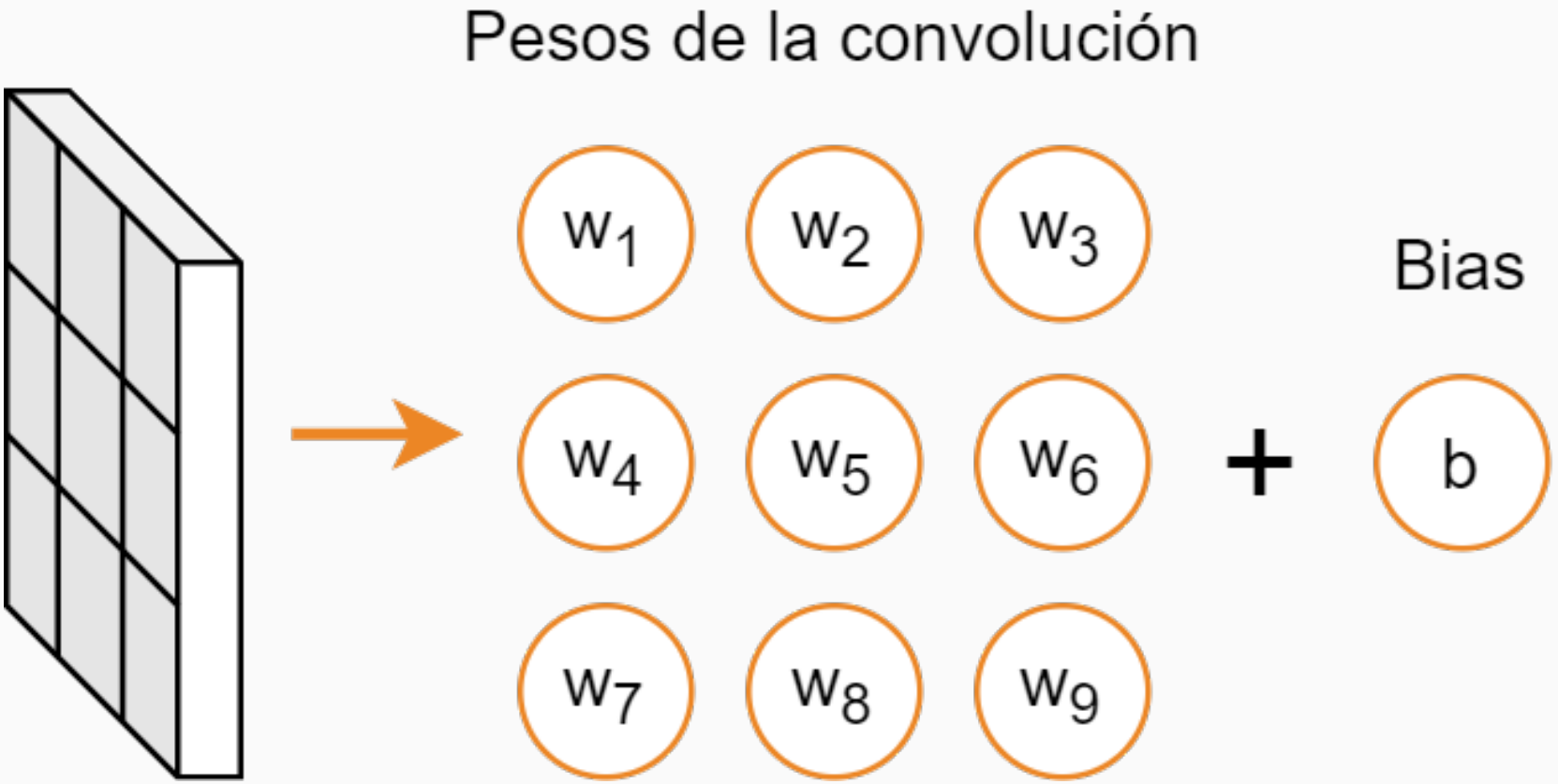
Una red de neuronas convolucional sustituye las capas densas por capas convolucionales.

Cada capa convolucional está compuesta por una serie de filtros de igual tamaño. Estos filtros se encargan de realizar el procesamiento de la información.



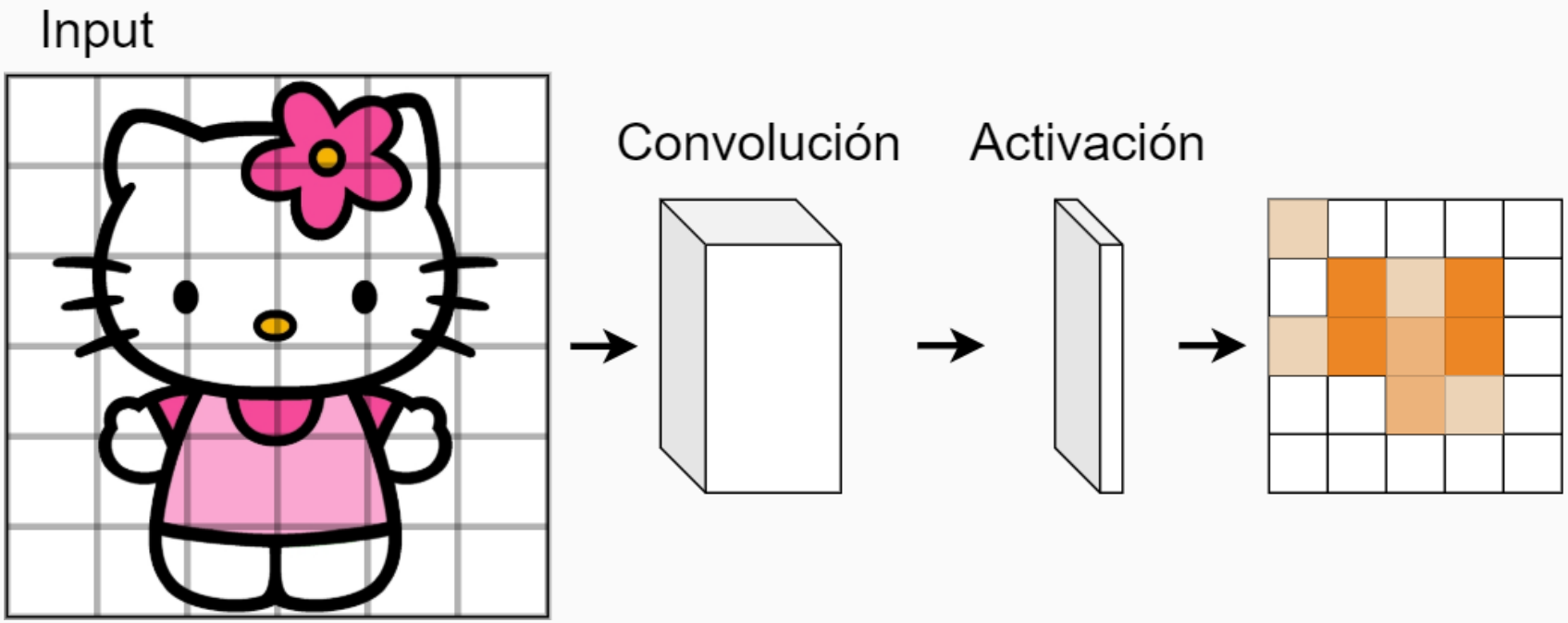
De neuronas a convoluciones

Cada filtro de la red está compuesto por una serie de neuronas.
Estas, igual que con las redes tradicionales tienen un peso asociado.
Este peso es el que regula cómo se realiza la convolución.



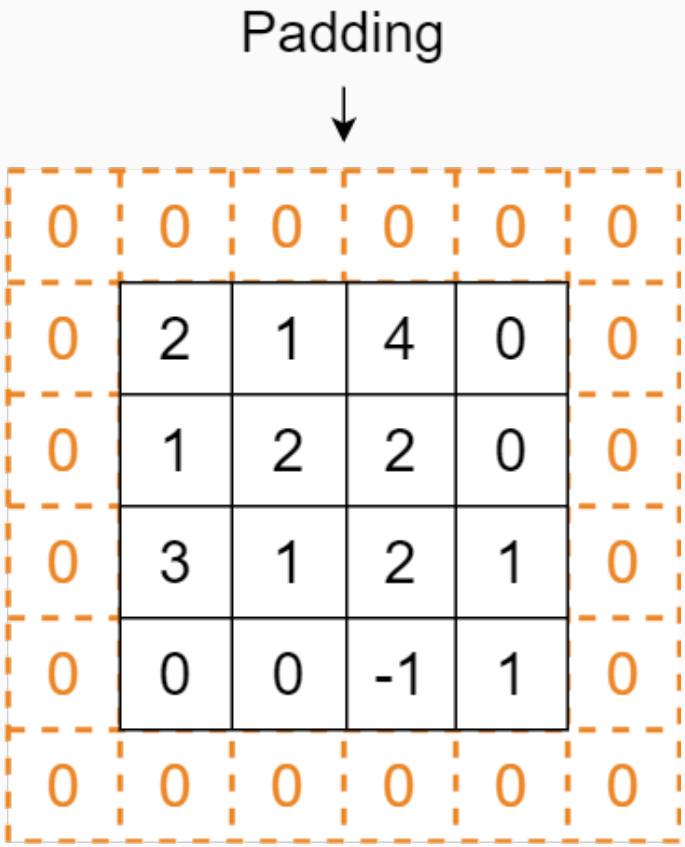
De neuronas a convoluciones

Tras haber realizado la convolución de unos datos de entrada, el resultado pasa por una activación a través de una función no lineal, tal y como sucede en las redes neuronales densas.



Padding en la convolución

Para controlar las dimensiones de salida de cada capa convolucional se aplica un *padding* a la imagen de entrada. Este consiste en un marco de "0" que evita la reducción dimensional.



Padding en la convolución

Existen dos configuraciones predominantes para la elección de padding en la librería keras:

- **Valid**: No se aplica ningún padding.
- **Same**: Se aplica un padding que haga que la dimensión de salida sea igual a la de entrada.

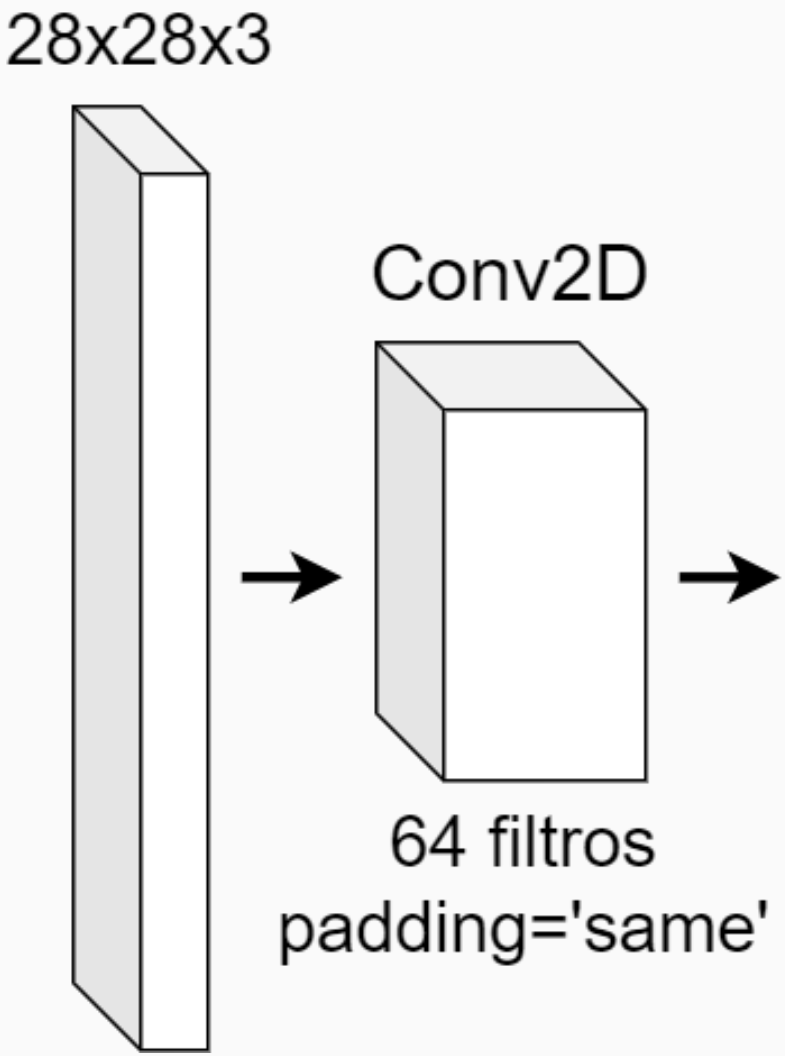
Ejemplo: Para una imagen de 16x16 píxeles y un filtro de 3x3, el padding "same" sería de 1 píxel.

Resultado de una capa convolucional

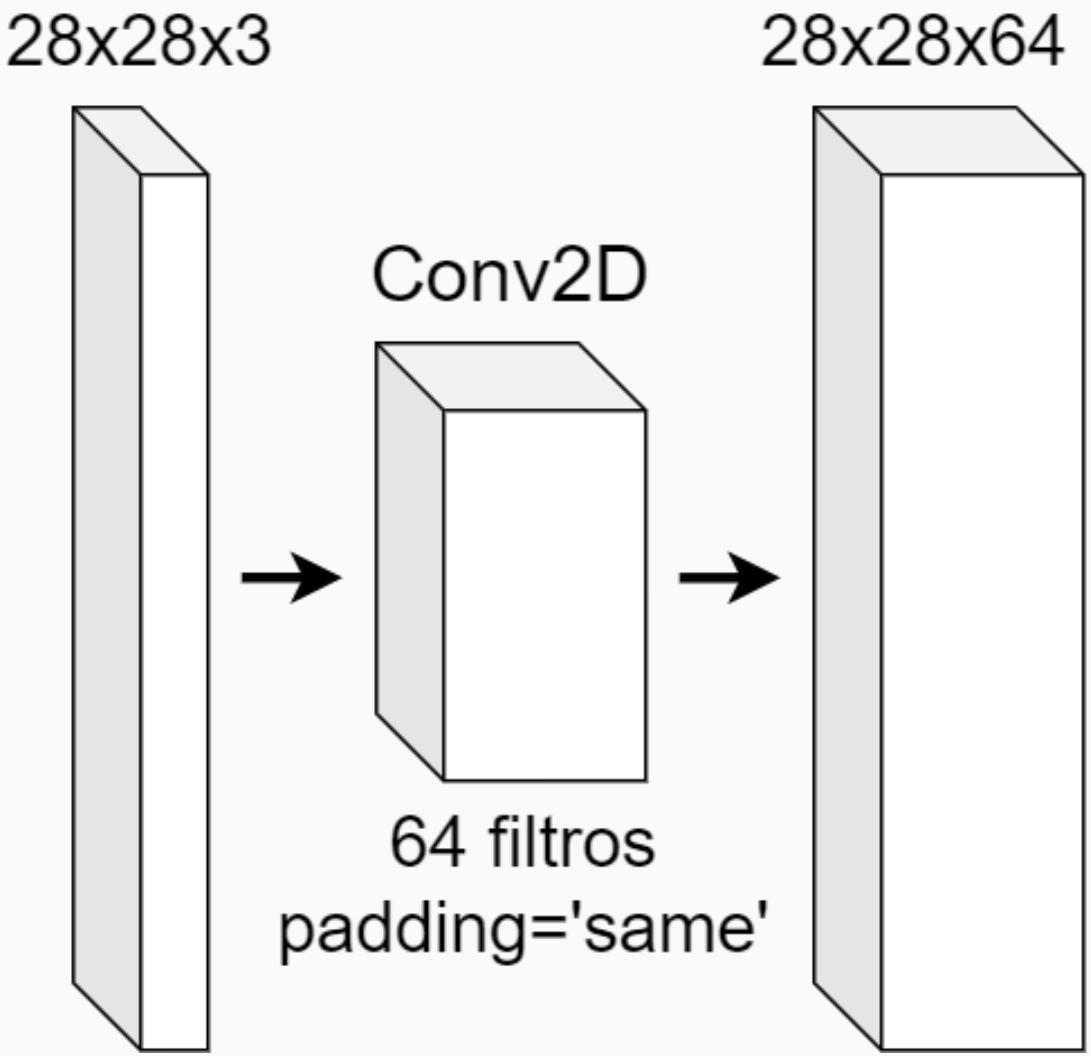
Una capa convolucional aplica varios filtros sobre la entrada y produce un volumen de salida cuyas dimensiones son:

- Alto y ancho: Determinados por las dimensiones de la entrada, el tamaño del kernel y el padding.
  - También influye el stride, se ve a continuación.
- Profundidad: Igual al número de filtros utilizados en la convolución.

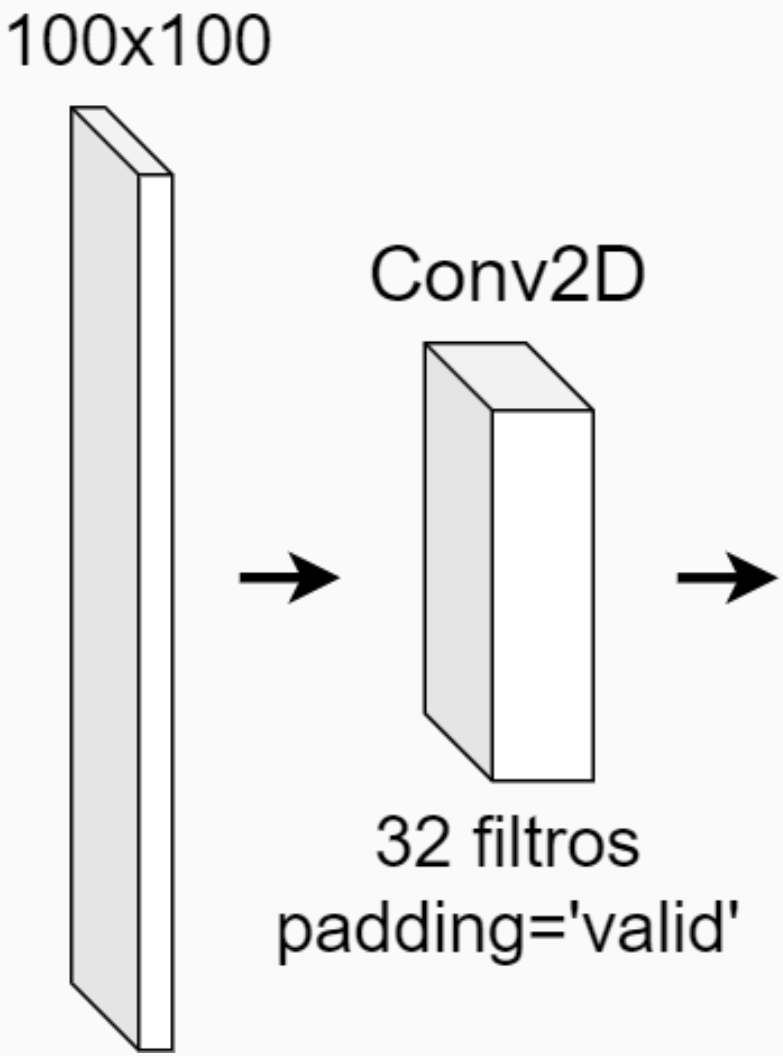
Resultado de una capa convolucional



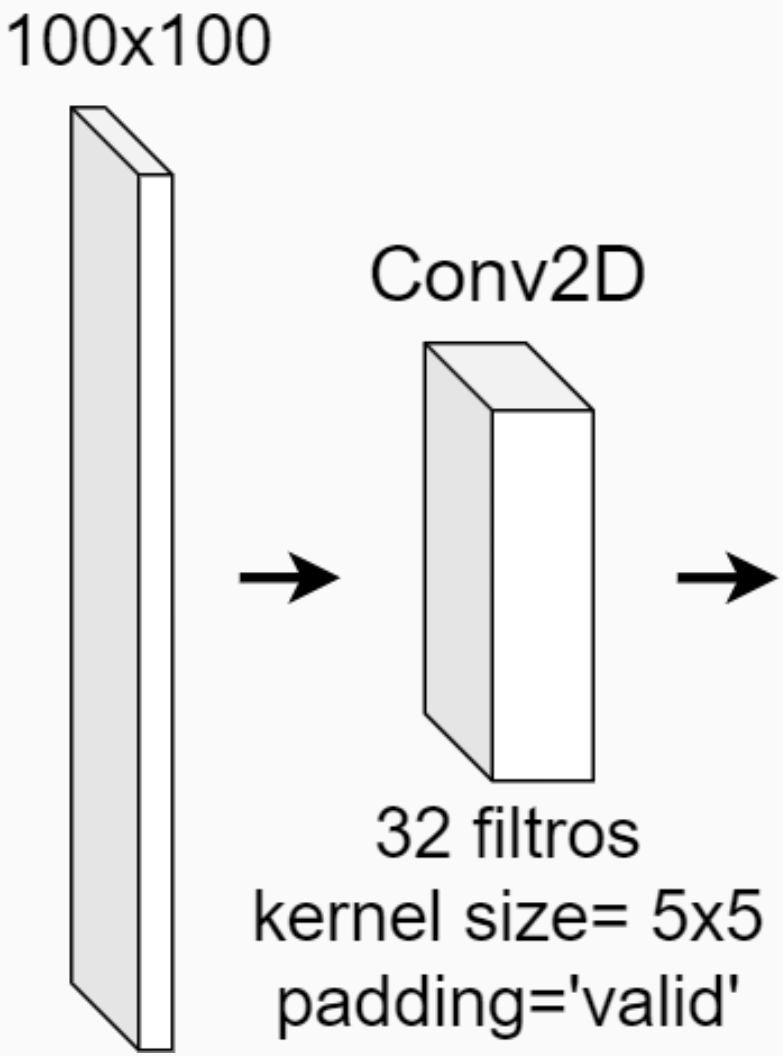
Resultado de una capa convolucional



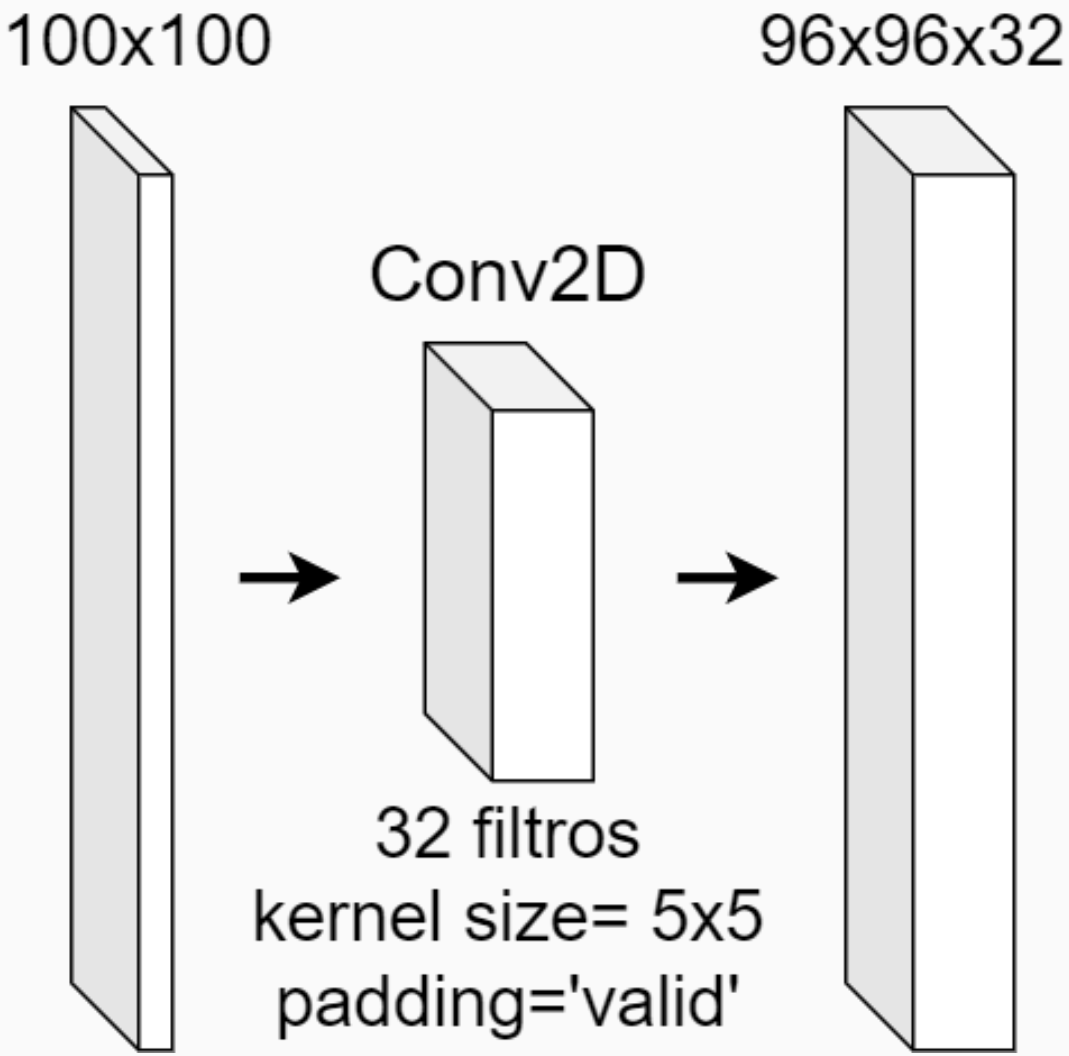
Resultado de una capa convolucional



Resultado de una capa convolucional



Resultado de una capa convolucional



Strides en la convolución

Los *strides* o pasos de una convolución corresponden con el número de casillas que se desplaza horizontal y verticalmente el filtro al realizar la convolución.

Stride = (2, 2)

f

2	1	4	0	2	3
1	2	2	0	1	2
3	1	2	1	0	1
0	0	-1	1	0	3
1	2	1	2	0	3
2	0	1	3	0	3

g

0	1	0
0	1	0
0	1	0

s

4	

Strides en la convolución

Los *strides* o pasos de una convolución corresponden con el número de casillas que se desplaza horizontal y verticalmente el filtro al realizar la convolución.

Stride = (2, 2)

f

2	1	4	0	2	3
1	2	2	0	1	2
3	1	2	1	0	1
0	0	-1	1	0	3
1	2	1	2	0	3
2	0	1	3	0	3

g

0	1	0
0	1	0
0	1	0

s

4	1

Strides en la convolución

Los *strides* o pasos de una convolución corresponden con el número de casillas que se desplaza horizontal y verticalmente el filtro al realizar la convolución.

Stride = (2, 2)

f

2	1	4	0	2	3
1	2	2	0	1	2
3	1	2	1	0	1
0	0	-1	1	0	3
1	2	1	2	0	3
2	0	1	3	0	3

g

0	1	0
0	1	0
0	1	0

s

4	1
3	4

Hiperparámetros de una convolución 2D

La capa Conv2D de la libería keras tiene una serie de hiperparámetros que permiten su configuración, dentro de los más importantes se encuentran:

- filters
- kernel_size
- strides
- padding
- activation

Hiperparámetros de una convolución 2D

- filters

Corresponden al número de filtros que se le aplican a los datos de entrada.

Se define con un integer.

- kernel_size

Determina el tamaño de los filtros que constituyen la capa.

Se define con un integer para filtros cuadrados, pero admite definir las dimensiones por separado en un vector (alto, ancho).

Hiperparámetros de una convolución 2D

- strides

Define el paso de la convolución a lo largo de los ejes.

Se define con un integer para un paso igual en ambos ejes, pero admite definir cada dimensión por separado en un vector (alto, ancho).

- padding

Determina el padding aplicado a los datos de entrada.

Se pueden definir las opciones "valid" y "same".

Hiperparámetros de una convolución 2D

- activation

Define la activación aplicada tras la convolución.

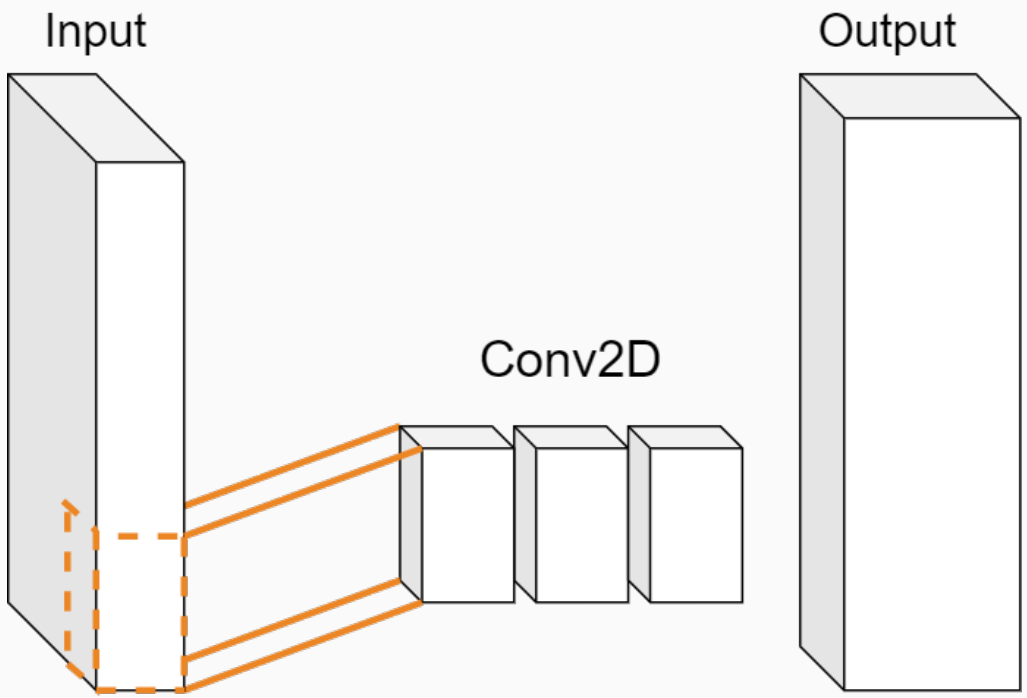
Dentro de las posibles activaciones se encuentran: relu, sigmoid, softmax, softplus, softsign, tanh, selu, elu, exponential.

Existe la posibilidad de aplicar otras activaciones así como activaciones custom. Por ejemplo para aplicar la función LeakyReLU.

Parámetros de una convolución

El número de parámetros de cada capa convolucional viene dado por el tamaño del filtros, el número de filtros y la profundidad de la información de la capa anterior:

$$((kernel_{height} * kernel_{width} * depth_{input}) + 1) * filters \tag{1}$$



Notebook de ejemplo, dimensiones de convolución

El siguiente notebook contiene un breve código para explorar las dimensiones de salida de una capa convolucional.

- 1.2_01-DimensionesConv2D.ipynb

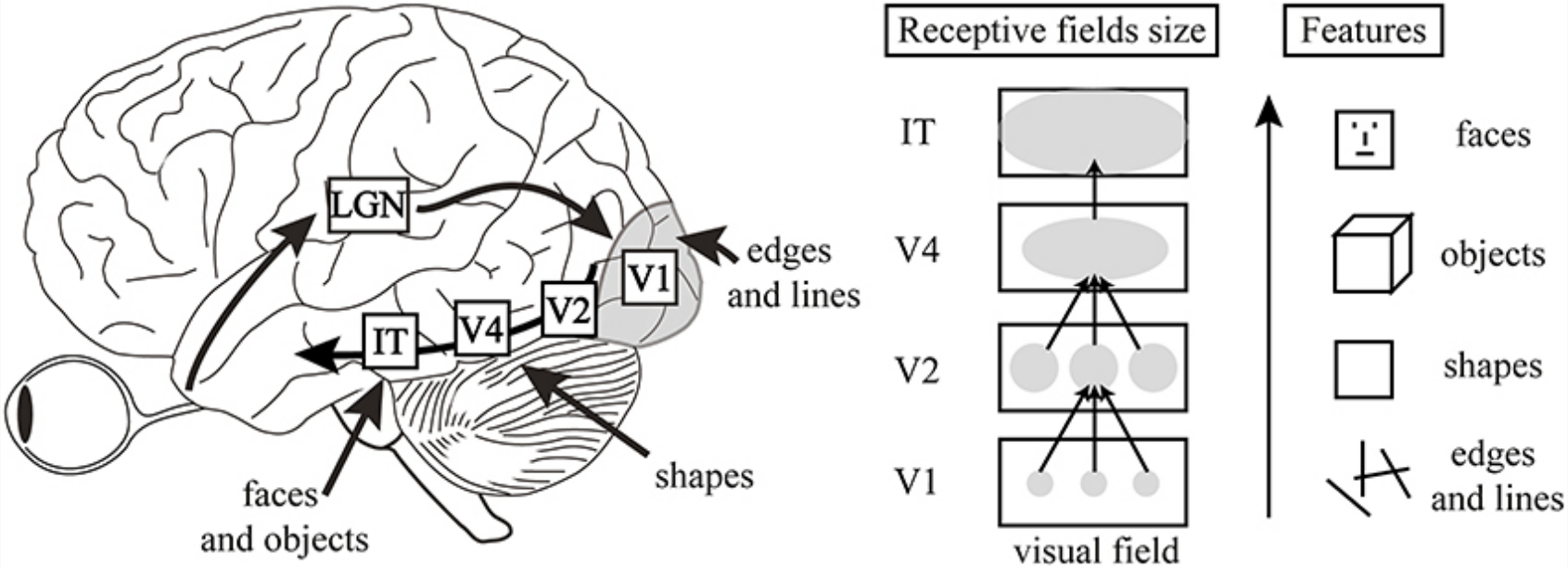
Aprendizaje de una red convolucional

A lo largo del tema se estudiarán distintas arquitecturas construidas con capas convolucionales, pero cabe destacar que la estructura por capas de estas redes consigue imitar el procesamiento del cortex cerebral del cerebro.

Las capas ocultas de las redes convolucionales contienen una jerarquía especializada en la tarea para la que se entrena.

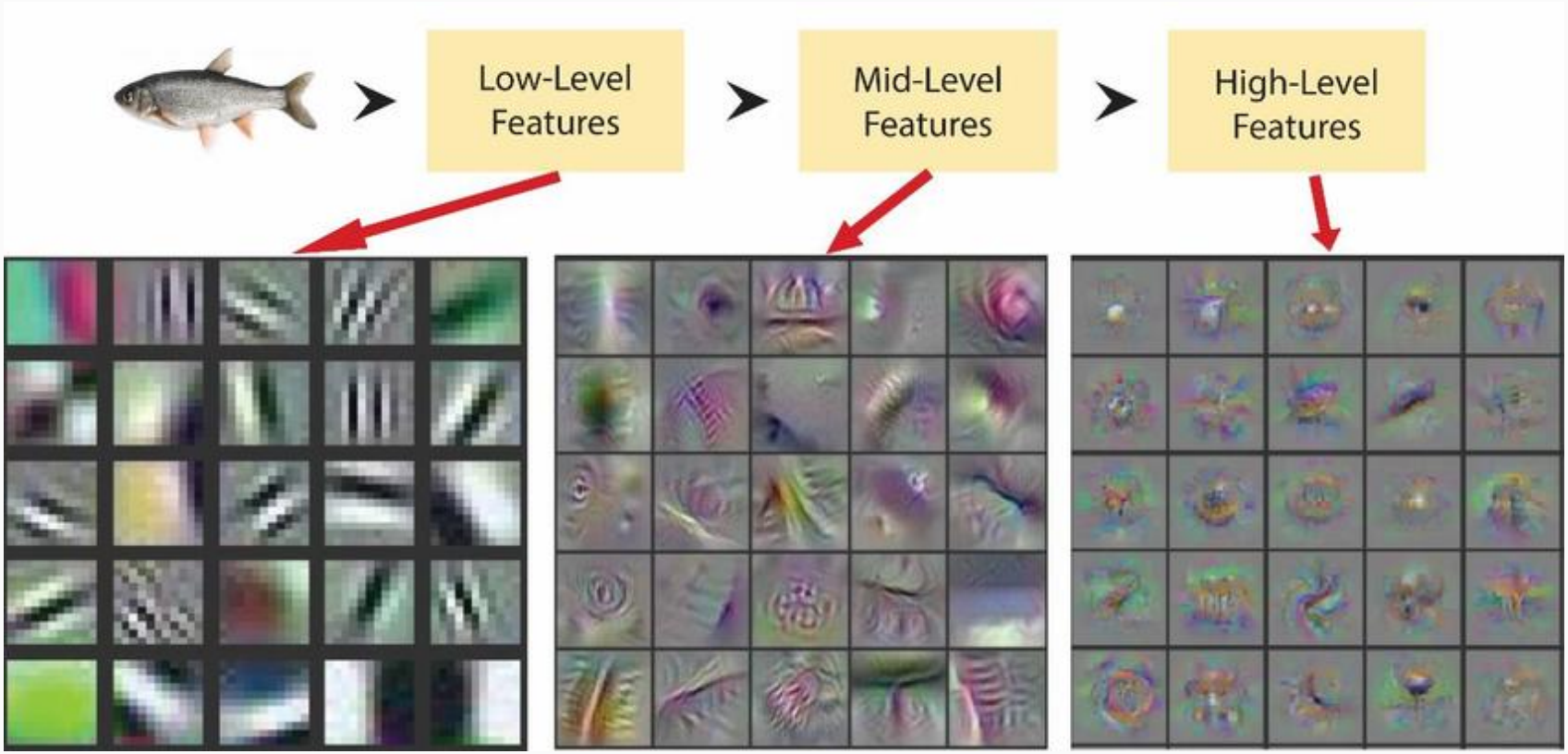
Esto se traduce en que, la primeras capas de la red se encargan de procesar información de bajo nivel, como líneas o curvas; mientras que las últimas capas se encargan de información de alto nivel, como una cara o la silueta de un animal.

Aprendizaje de una red convolucional



[1]

Aprendizaje de una red convolucional



[2]

Web interactiva con convoluciones

Web interactiva con convoluciones

- 👉CNN Explainer

Notebook de ejemplo, clasificador con redes convolucionales

El siguiente notebook contiene un ejemplo de clasificador redes convolucionales.

- 1.2_02-CNNImagenes.ipynb

Referencias i

[1] Michael H. Herzog and Aaron M. Clarke (frontiers).
Cortex image.
https://www.frontiersin.org/articles/10.3389/fncom.2014.00135/full.

[Online; accessed August, 2022].

[2] Shoaib Ahmed Siddiqui, Ahmad Salman, Muhammad Imran Malik, Faisal Shafait, Ajmal Mian, Mark R Shortis, and Euan S Harvey.
Automatic fish species classification in underwater videos: exploiting pre-trained deep neural network models to compensate for limited labelled data.
ICES Journal of Marine Science, 75(1):374–389, 2018.

Contribuciones de las diapositivas

- **Autor original de las diapositivas:** Guillermo Iglesias Hernández