Redes convolucionales I

Visión por Computador, curso 2024-2025

Silvia Martín Suazo, silvia.martin@u-tad.com

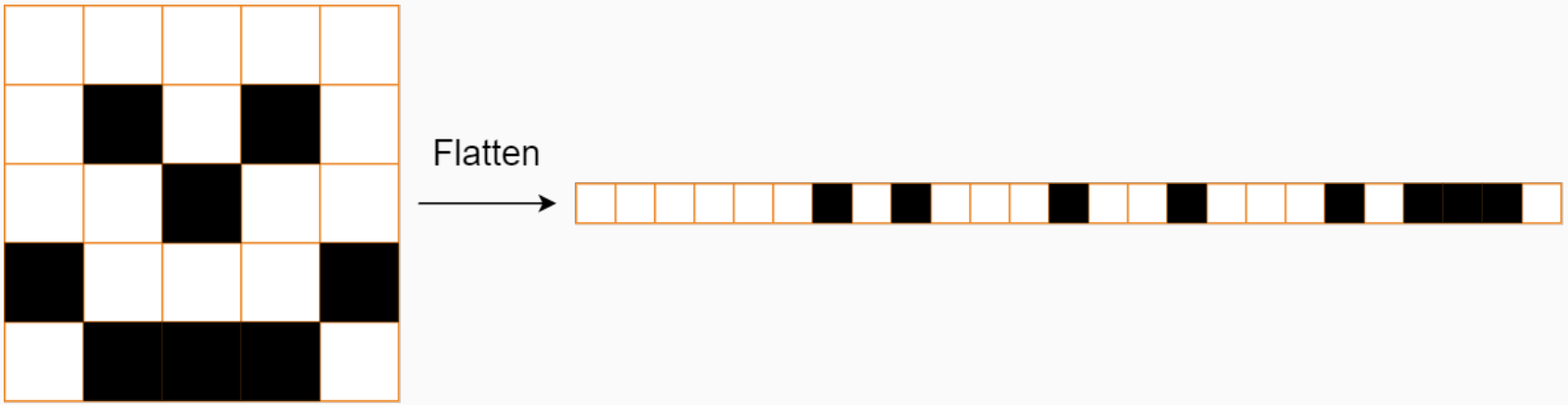
6 de noviembre de 2024

U-tad | Centro Universitario de Tecnología y Arte Digital

Introducción

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. Para ello utilizan capas de convolución con el fin de la detección de patrones en la información de entrada.

Por lo tanto son las principales redes para tareas de clasificación en visión por computador.

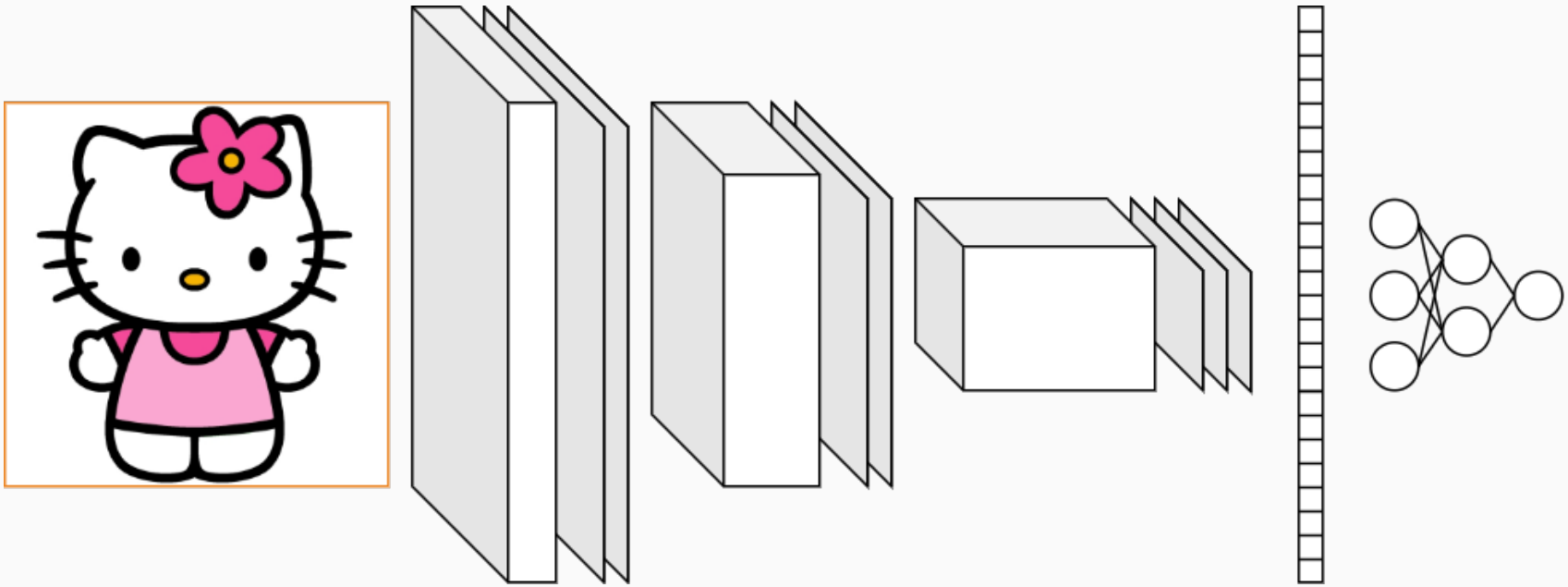
Los principales beneficios de su uso son los siguientes:

- Aprovechamiento de la información espacial.
- Reducción del número de parámetros.
- Invarianza aprendida de los datos.
- Eficiencia computacional

Arquitectura de las redes neuronales convolucionales

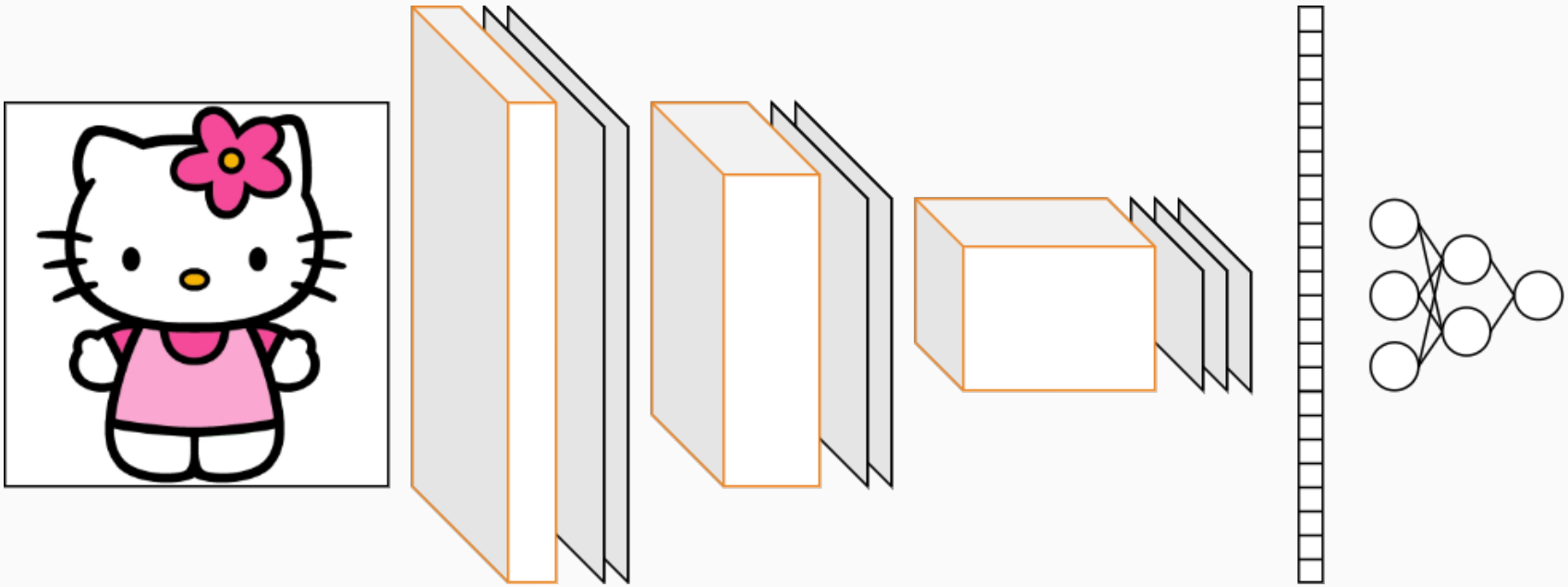
Esquema de construcción de redes convolucionales

- Input
- Conv2D
- Activation
- Pooling
- Flatten
- Dense
- Output



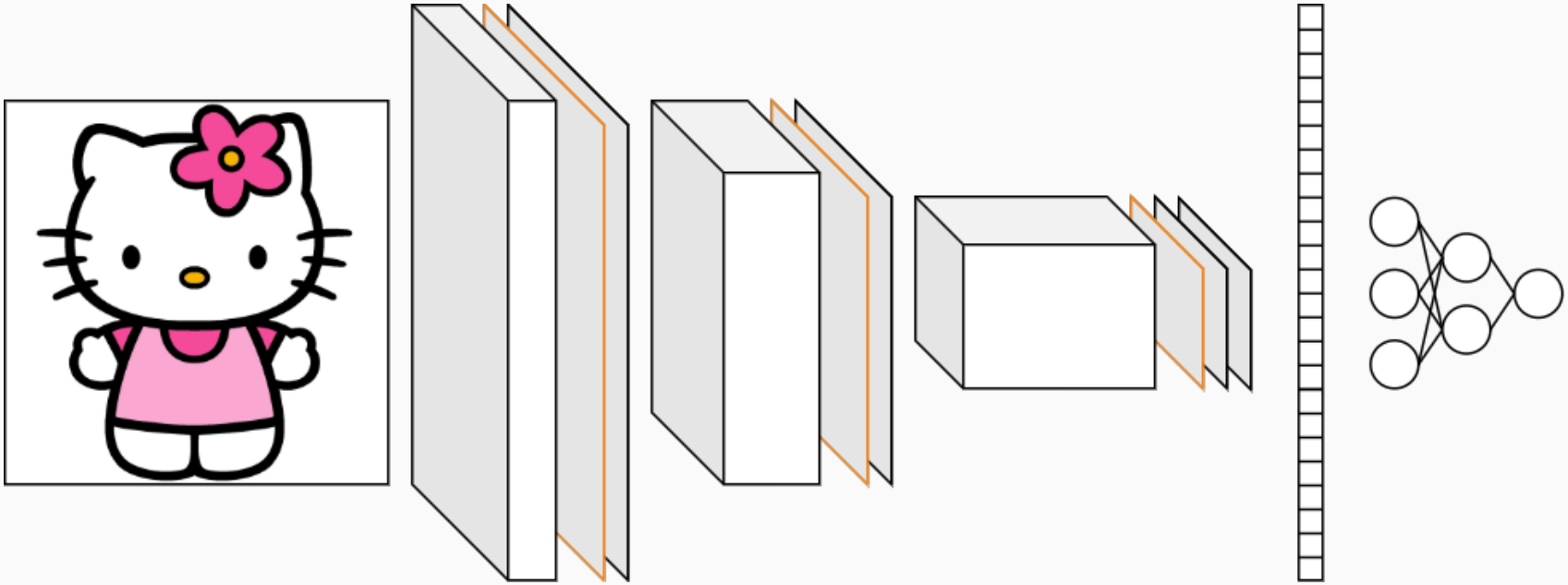
Esquema de construcción de redes convolucionales

- Input
- Conv2D
- Activation
- Pooling
- Flatten
- Dense
- Output



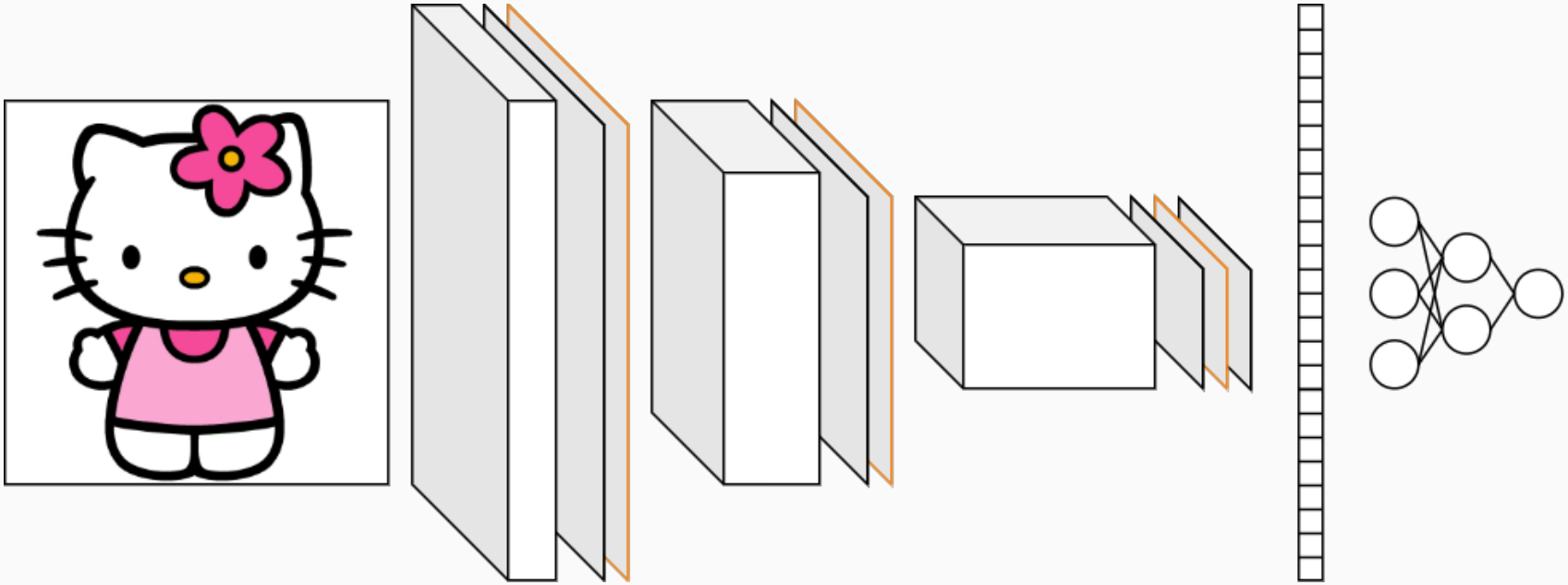
Esquema de construcción de redes convolucionales

- Input
- Conv2D
- Activation
- Pooling
- Flatten
- Dense
- Output



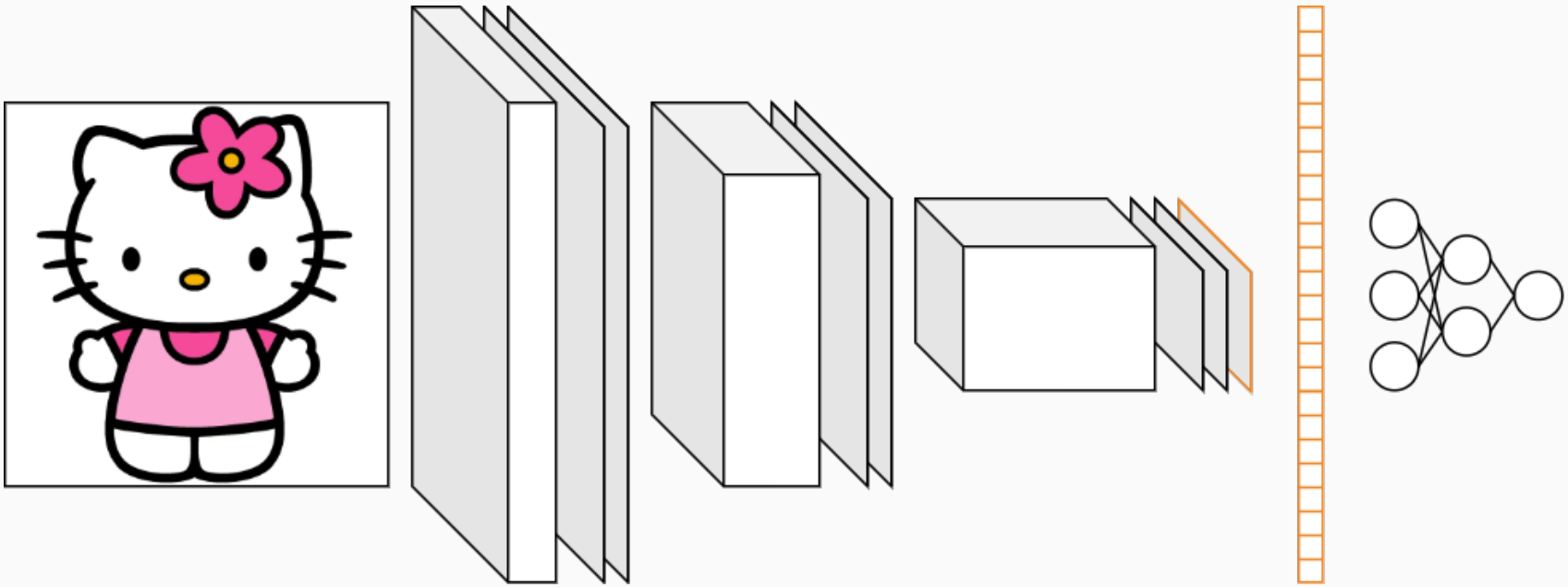
Esquema de construcción de redes convolucionales

- Input
- Conv2D
- Activation
- Pooling
- Flatten
- Dense
- Output



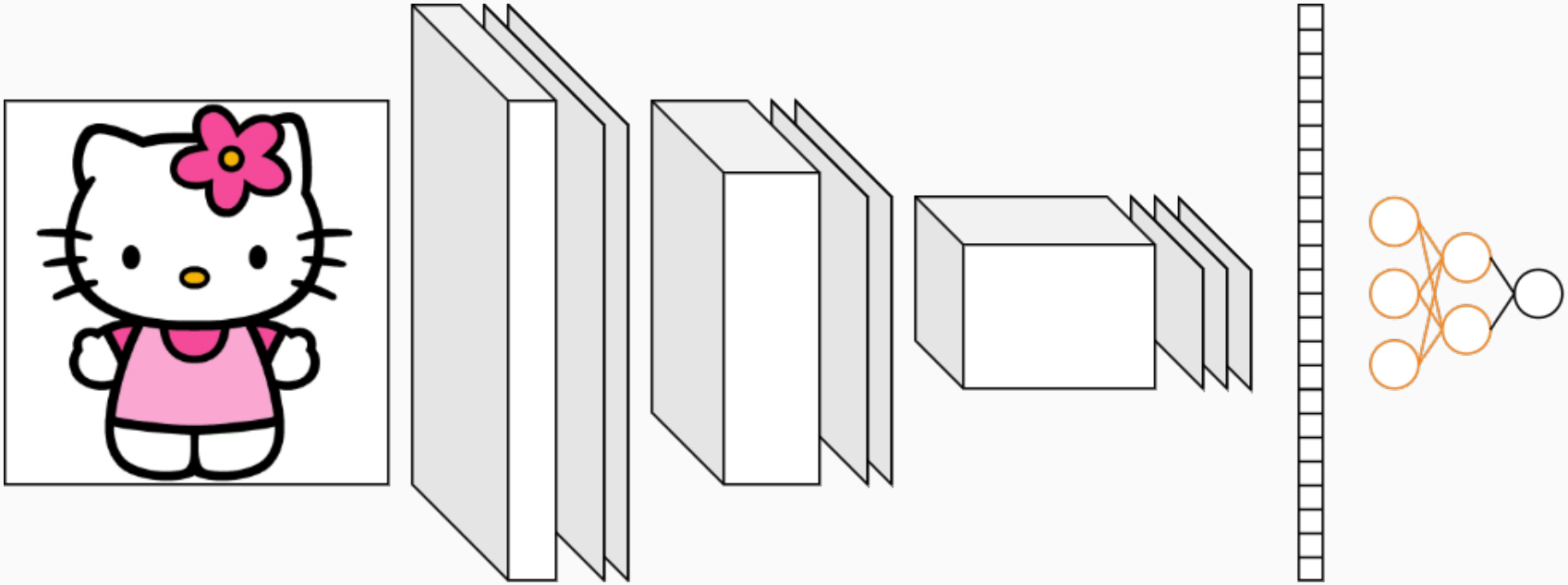
Esquema de construcción de redes convolucionales

- Input
- Conv2D
- Activation
- Pooling
- Flatten
- Dense
- Output



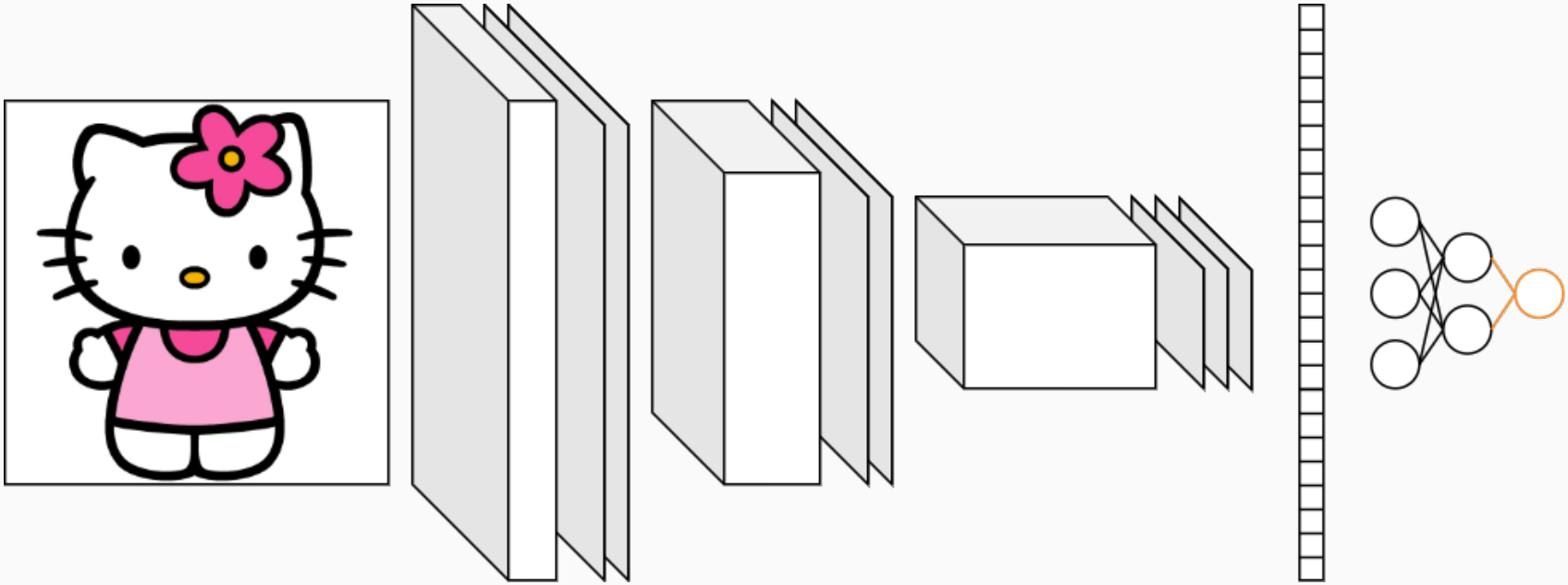
Esquema de construcción de redes convolucionales

- Input
- Conv2D
- Activation
- Pooling
- Flatten
- Dense
- Output



Esquema de construcción de redes convolucionales

- Input
- Conv2D
- Activation
- Pooling
- Flatten
- Dense
- Output



Capas convolucionales

Función

Las capas convolucionales, como su nombre indica, utilizan la operación de convolución y son un componente fundamental de las CNN.

Durante la convolución, los filtros se deslizan por diferentes regiones de la matriz correspondiente a la imagen, con el objetivo de detectar características relevantes, como bordes, texturas o patrones. El resultado de esta detección se conoce como mapas de características, que codifican la presencia y la ubicación de estas características en la imagen.

Lo que hace que las capas convolucionales sean especialmente poderosas es su capacidad de extracción de características jerárquica. A medida que se apilan múltiples capas convolucionales en la CNN, se pueden identificar características a diferentes niveles de abstracción (de más simple a más complejo).

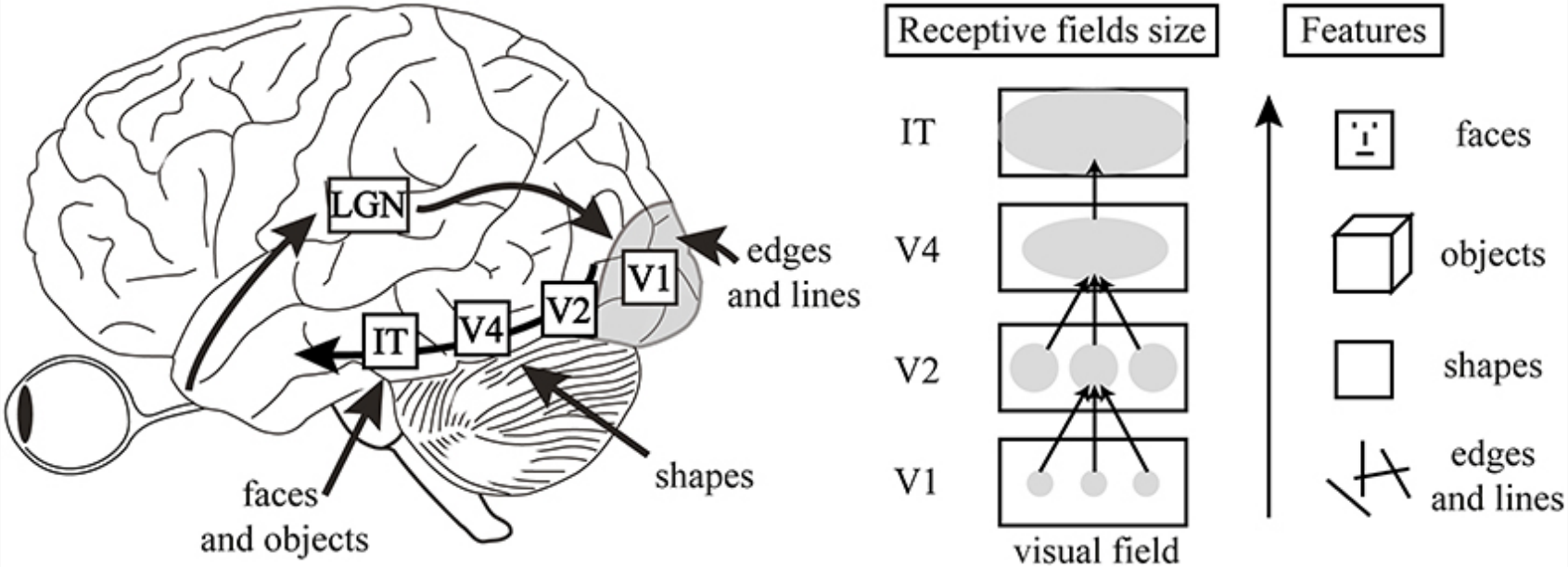
Aprendizaje de una red convolucional

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.

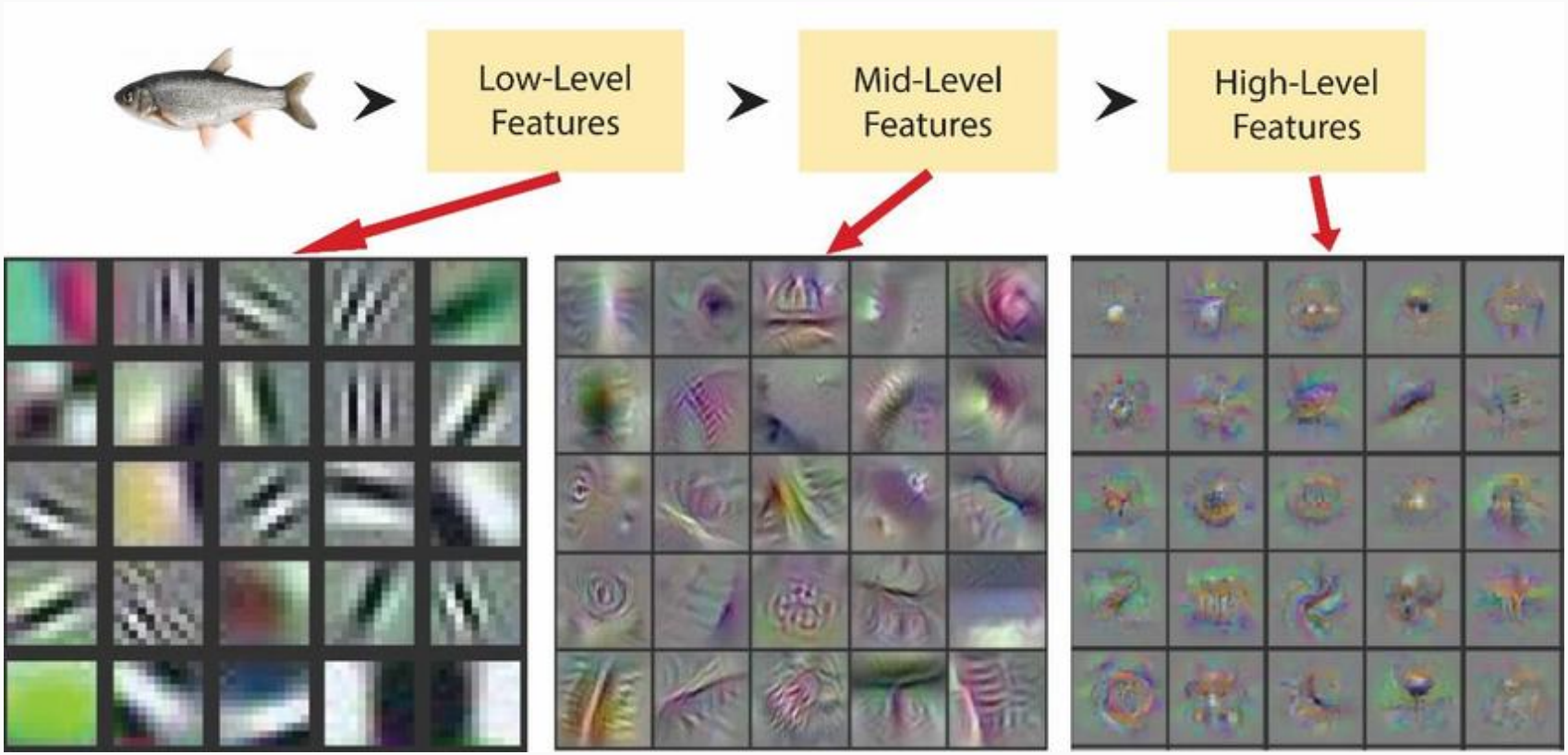
Esto se traduce en que, la primeras capas de la red se encargan de procesar información de bajo nivel, como líneas o curvas; mientras que las últimas capas se encargan de información de alto nivel, como una cara o la silueta de un animal.

Aprendizaje de una red convolucional



[1]

Aprendizaje de una red convolucional



[2]

Operación de convolución

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

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

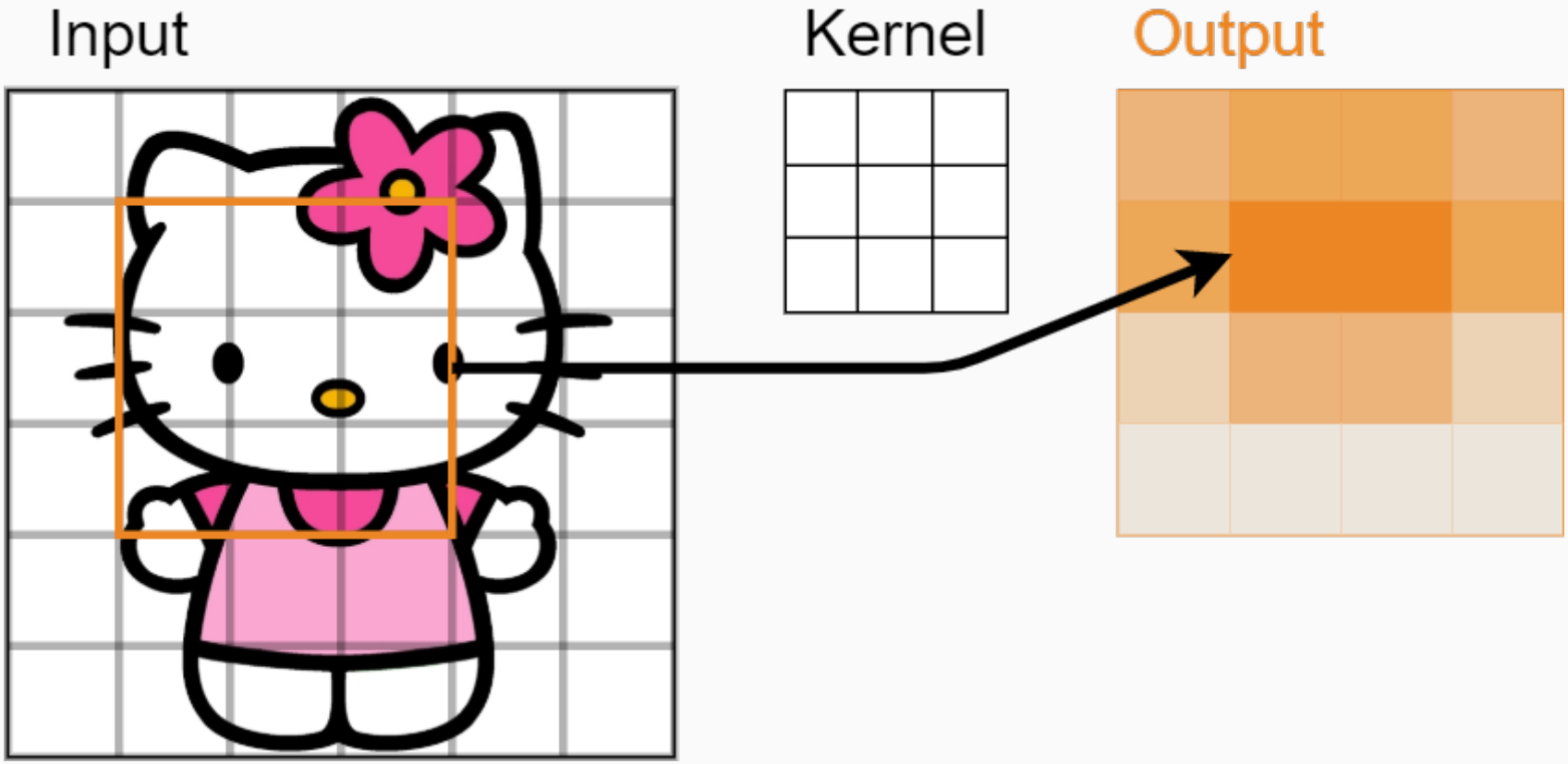
Output

4	8
5	5

Campo receptivo

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

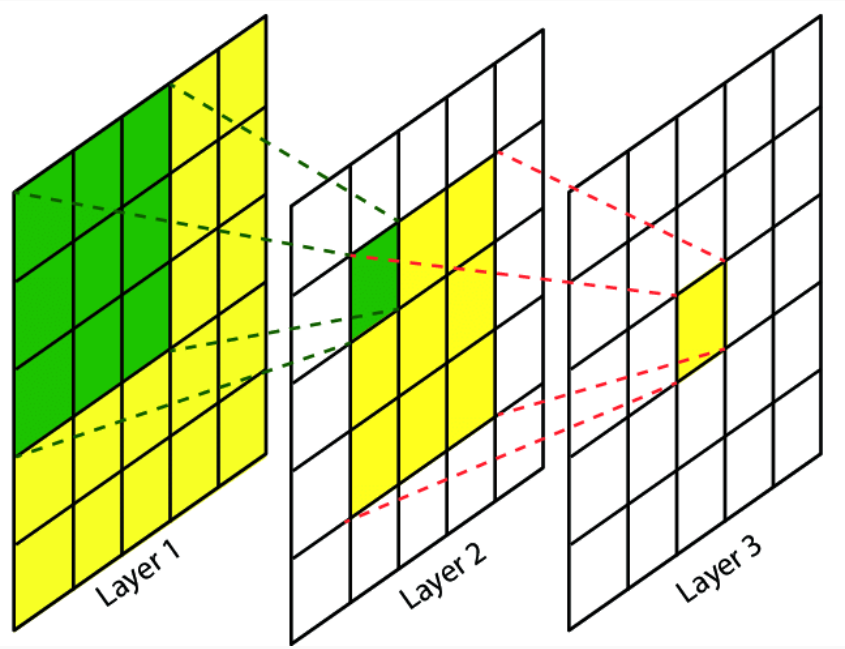
El campo receptivo de cada celda de la salida se activa cuando detecta una estructura de interés.



Campo receptivo

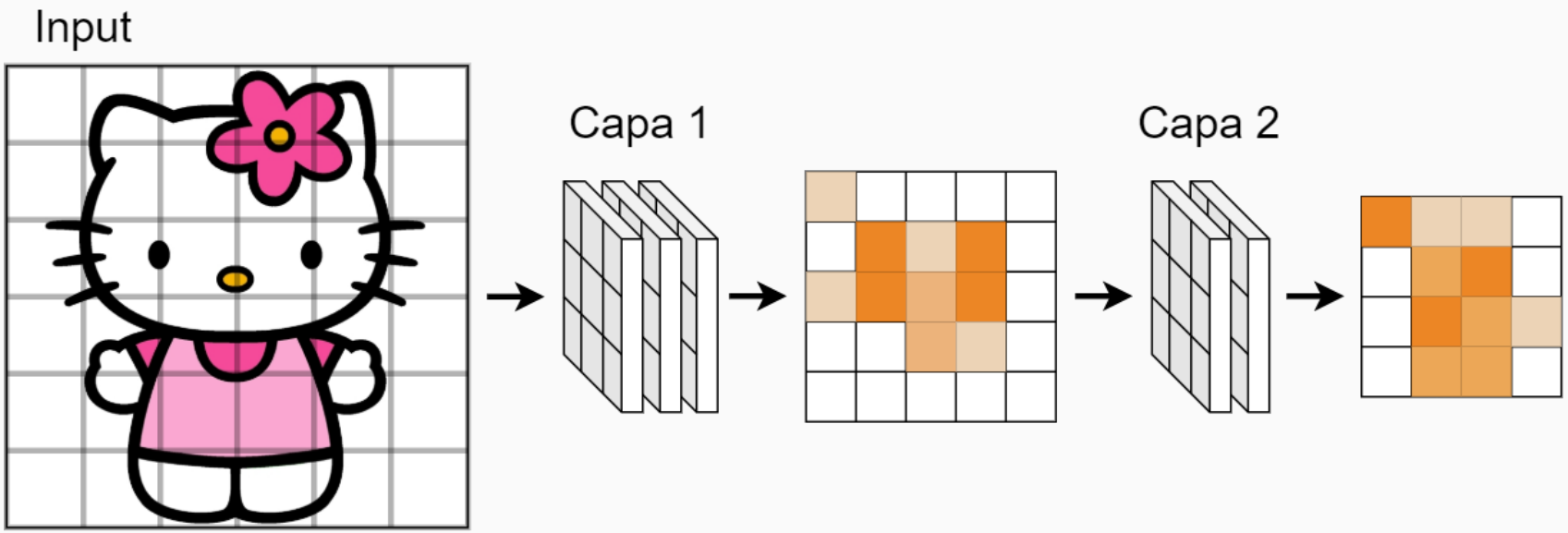
A medida que avanzamos en la red, estas ventanas se hacen más grandes, permitiendo a las neuronas captar características de nivel superior.

El tamaño de campo receptivo de cada neurona está marcado por el tamaño de kernel y stride, a mayores más amplio.



Filtros de la capa convolucional

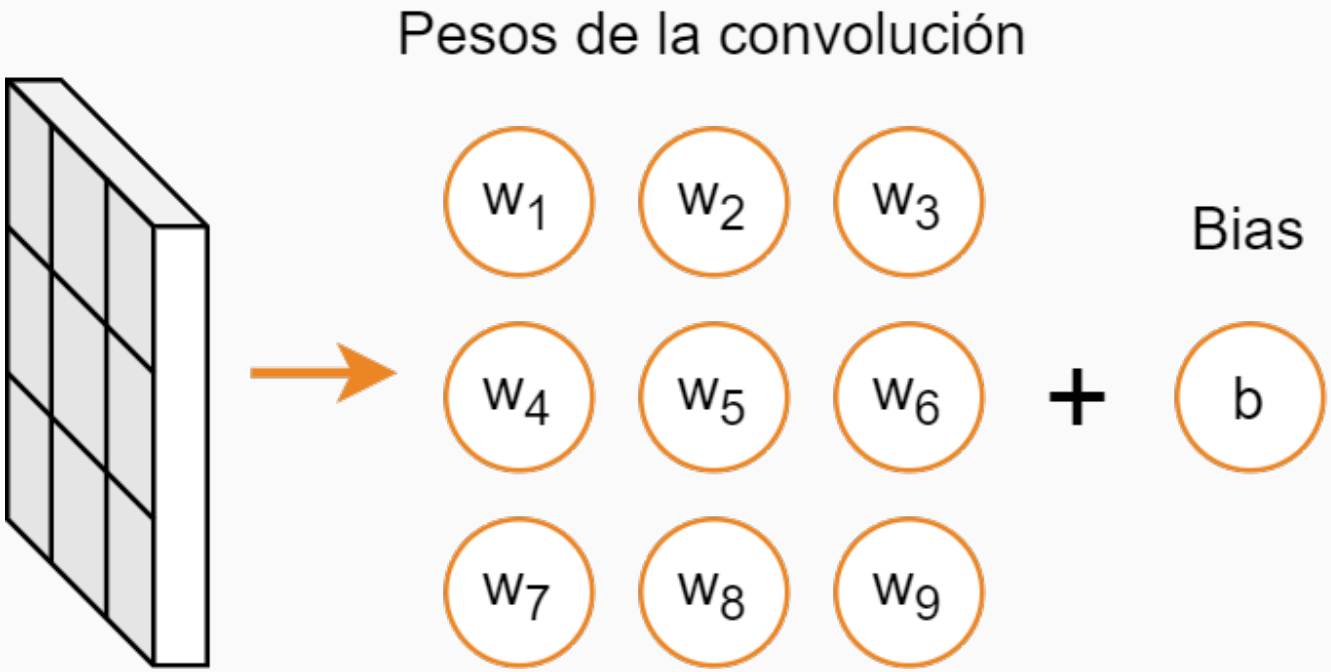
Cada capa convolucional está compuesta por una serie de filtros de igual tamaño. Estos filtros se encargan las respectivas operaciones de convolución para obtener los mapas de características.



Filtros de la capa convolucional

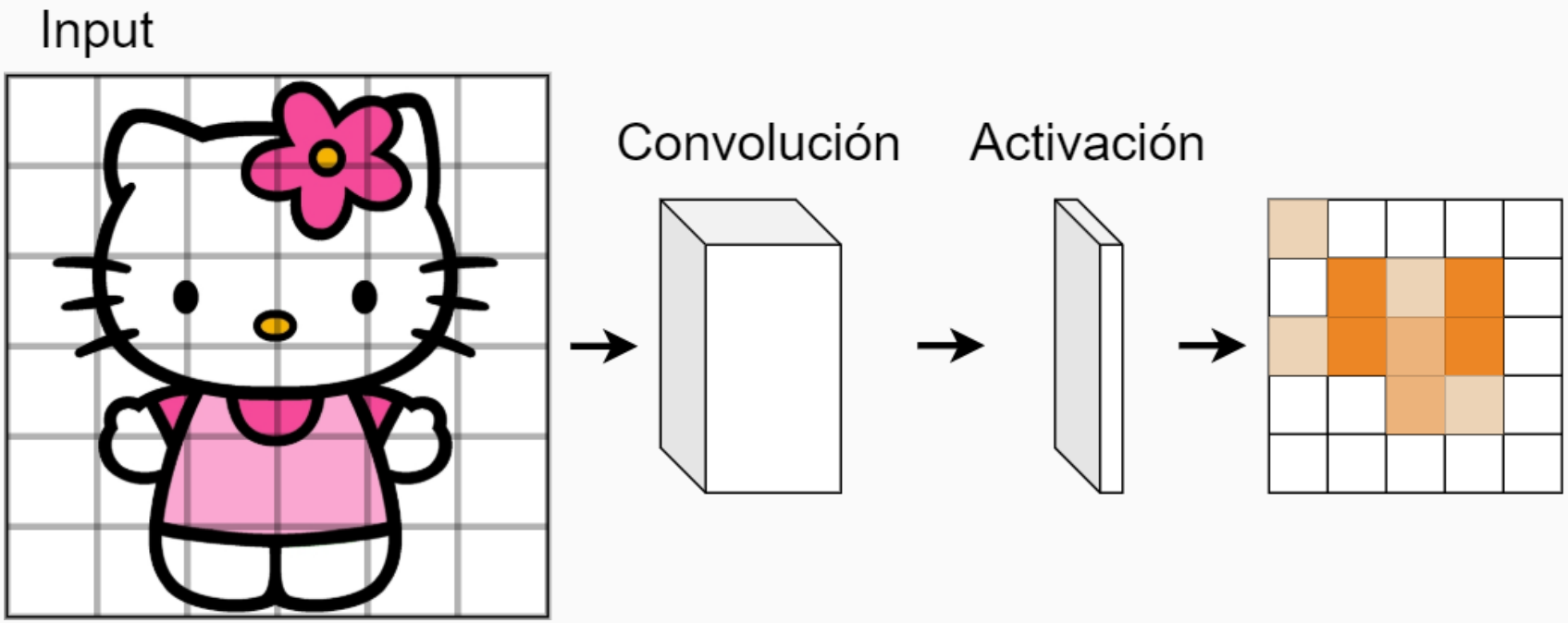
Cada filtro de la red está compuesto por una serie de neuronas. Estas, igual que con las redes tradicionales tienen un peso asociado. Este peso es el que regula cómo se realiza la convolución.

Estos pesos son inicializados aleatoriamente al inicio y ajustados durante la retropropagación de forma que los filtros aprendan a detectar características en los datos de entrada.



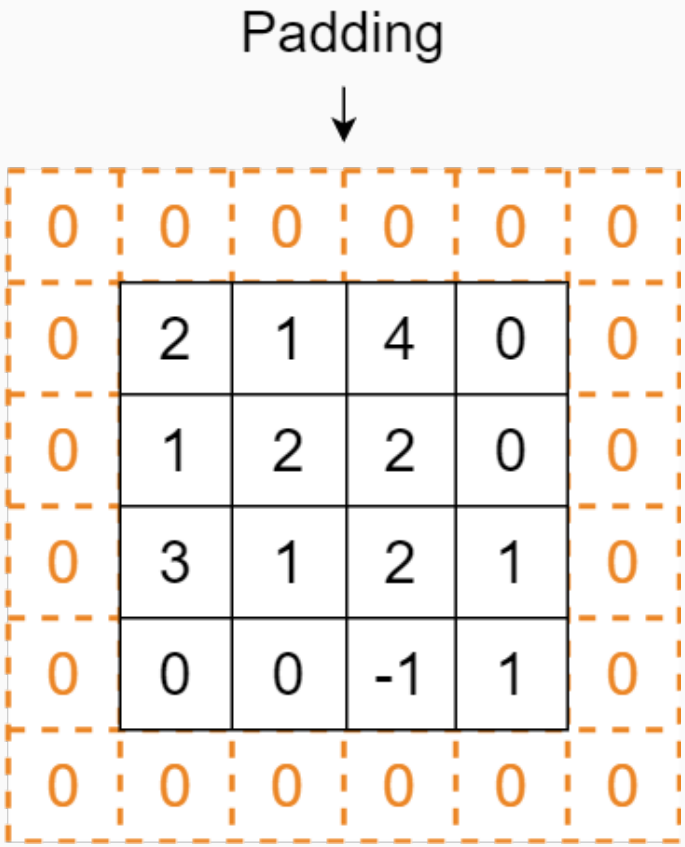
De neuronas a convoluciones

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.



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

Capas convolucionales en Keras

La capa librería keras tiene una serie de funciones que permiten la utilización de distintos tipos de capa convolucional:

- Conv1D: se utiliza para datos unidimensionales (texto o series temporales de una sola variable)
- Conv2D: aplicable a datos bidimensionales (imágenes/matrices).
- Conv3D: se emplean en datos tridimensionales (datos volumétricos o video).

A continuación se explican algunos de los parámetros.

Capas convolucionales en Keras

- filters

Corresponden al número de filtros que se le aplican a los datos de entrada. A más filtros mayor capacidad de aprender representaciones complejas y mayor complejidad del modelo.

Se define con un integer.

- kernel_size

Determina el tamaño de los filtros que constituyen la capa. Un kernel pequeño es más apropiado para detalles pequeños y uno mayor para características que ocupan regiones más amplias.

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

Capas convolucionales en Keras

- ## strides

Define el paso de la convolución a lo largo de los ejes. A mayor stride menor dimensión de salida.

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".

Capas convolucionales en Keras

Existen dos configuraciones predominantes para la elección de padding en la implementación:

- **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.

- activation

Define la activación aplicada tras la convolución para introducir no linealidad en la red.

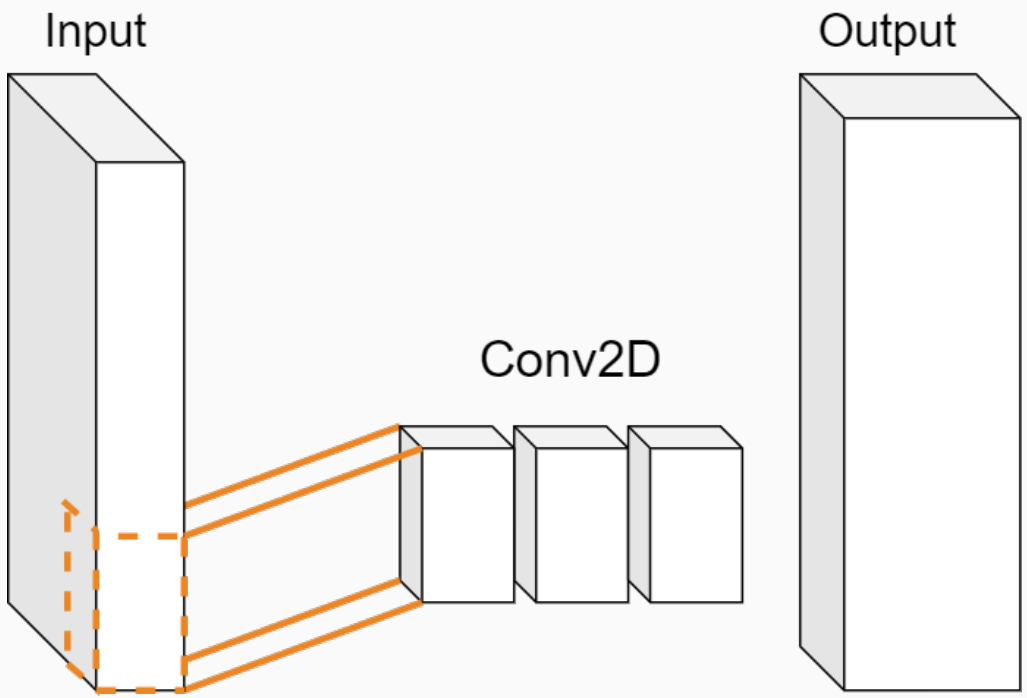
Dentro de las posibles activaciones se encuentran: relu, sigmoid, softmax, softplus, softsign, tanh, selu, elu, exponential.

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 su número de filtros y la profundidad de la información de la capa anterior:

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

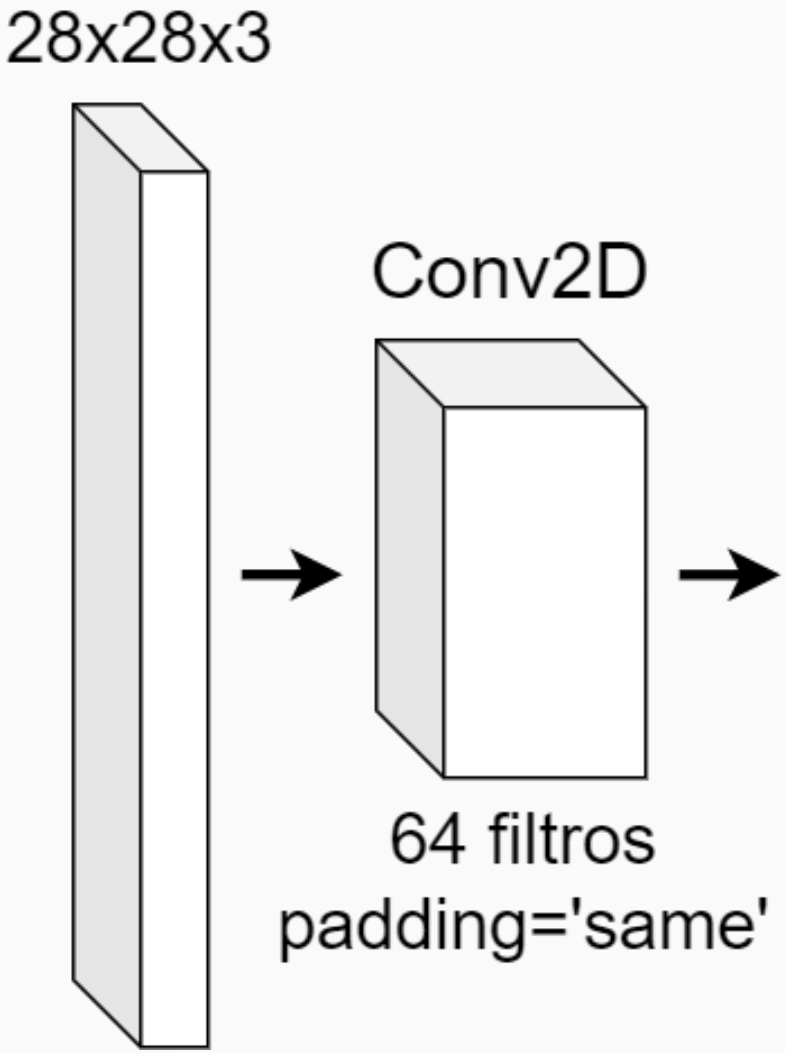


Resultado de una capa convolucional

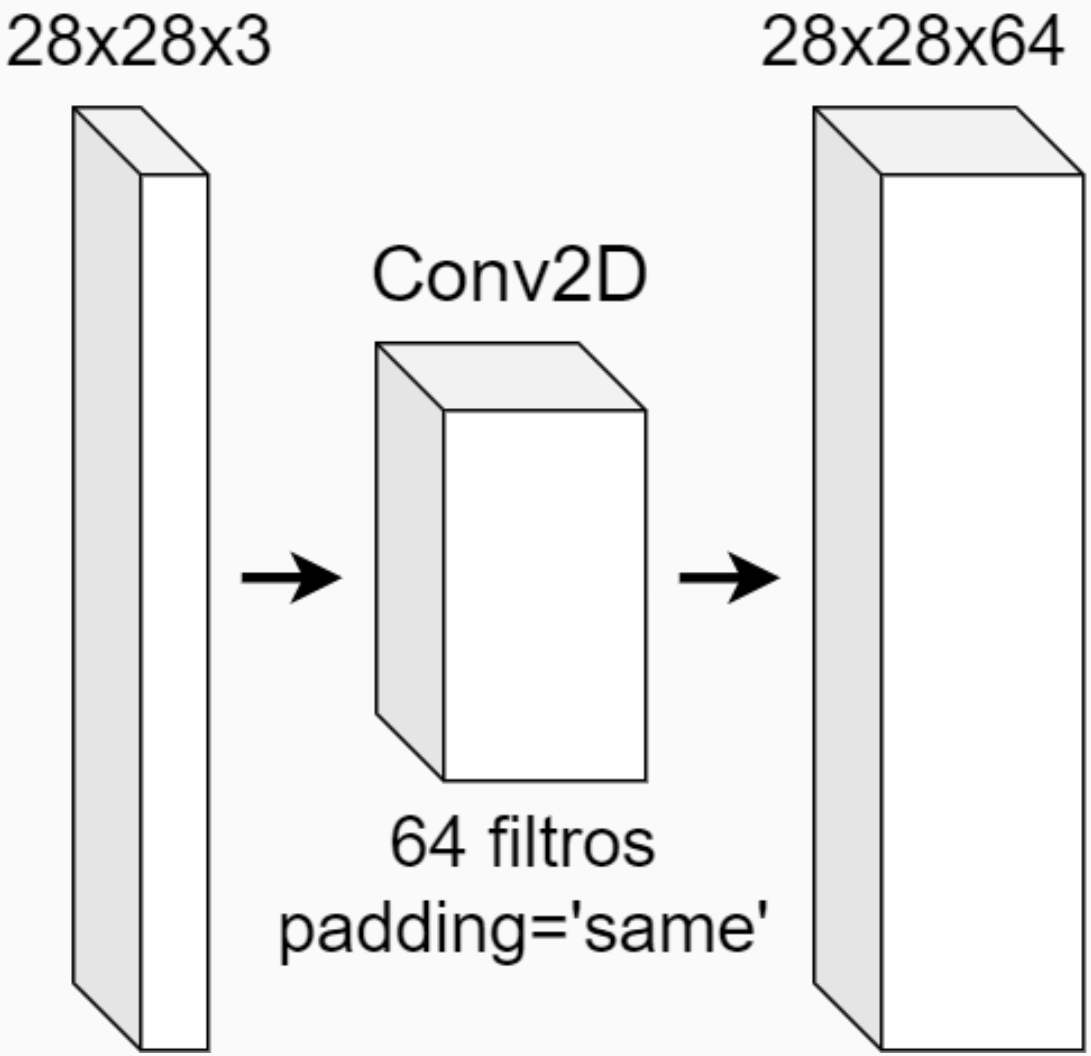
Cada capa convolucional está formada por una serie de convoluciones. Las dimensiones de salida de cada capa vienen dadas por:

- Alto y ancho: Dependen de las dimensiones de los datos recibidos y el padding utilizado.
- Profundidad: Corresponde con el número de filtros aplicados a los datos.

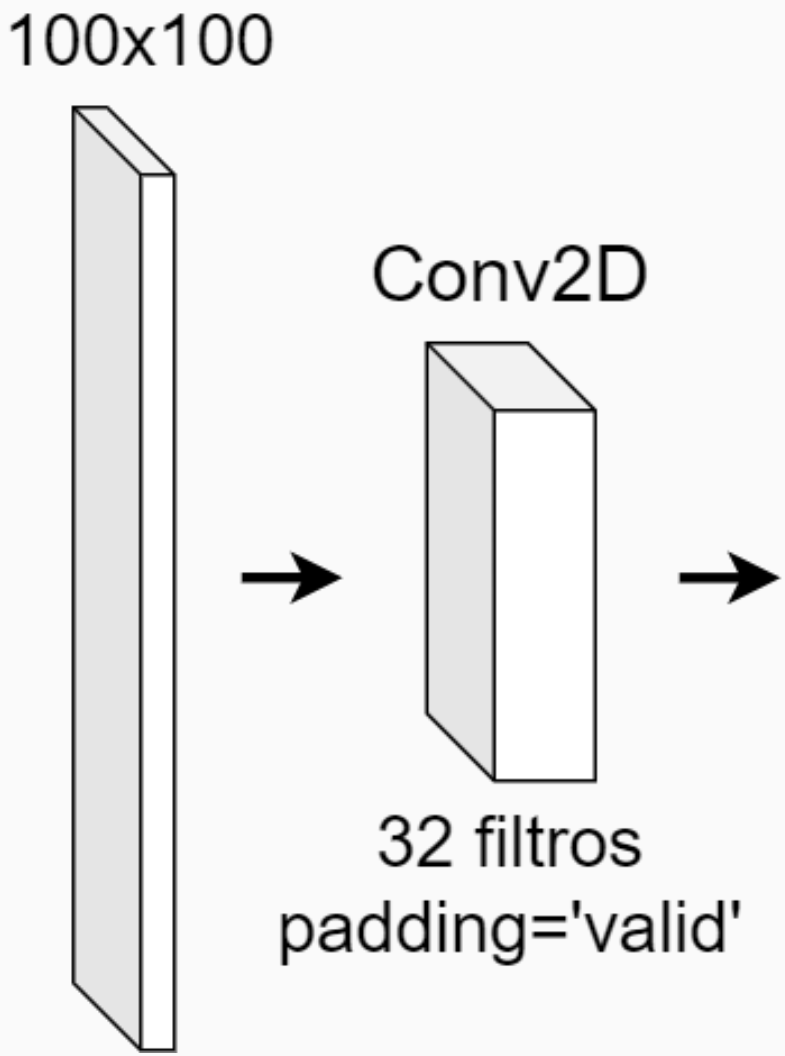
Resultado de una capa convolucional



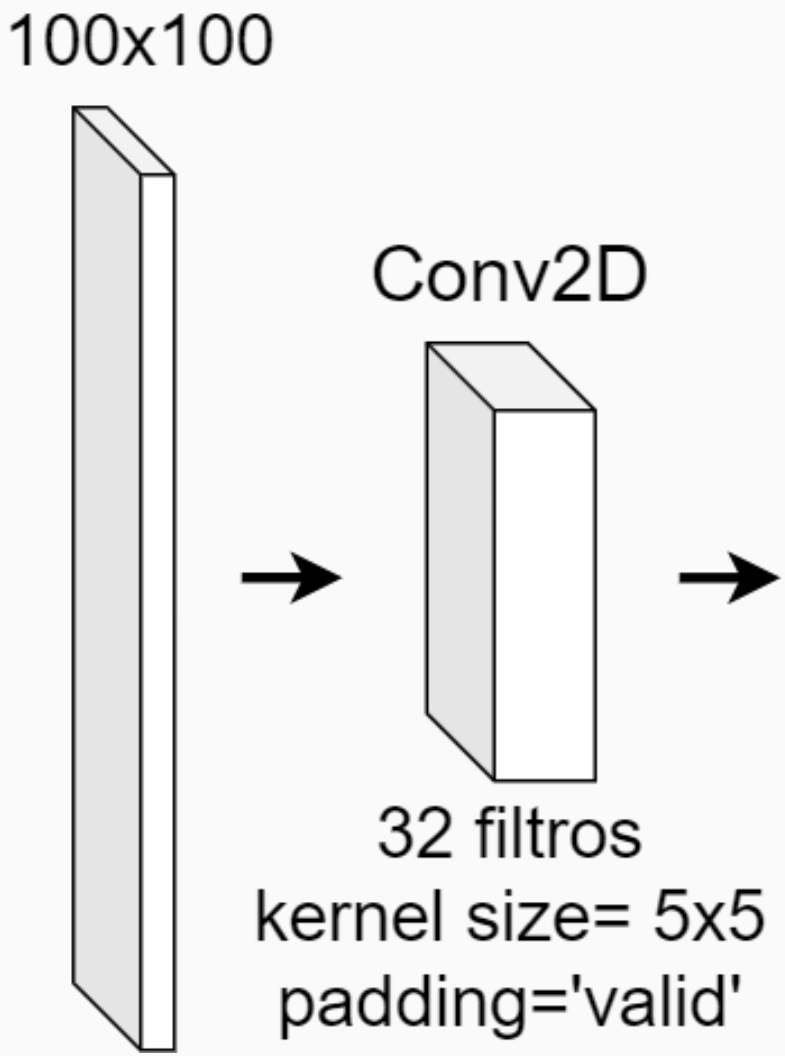
Resultado de una capa convolucional



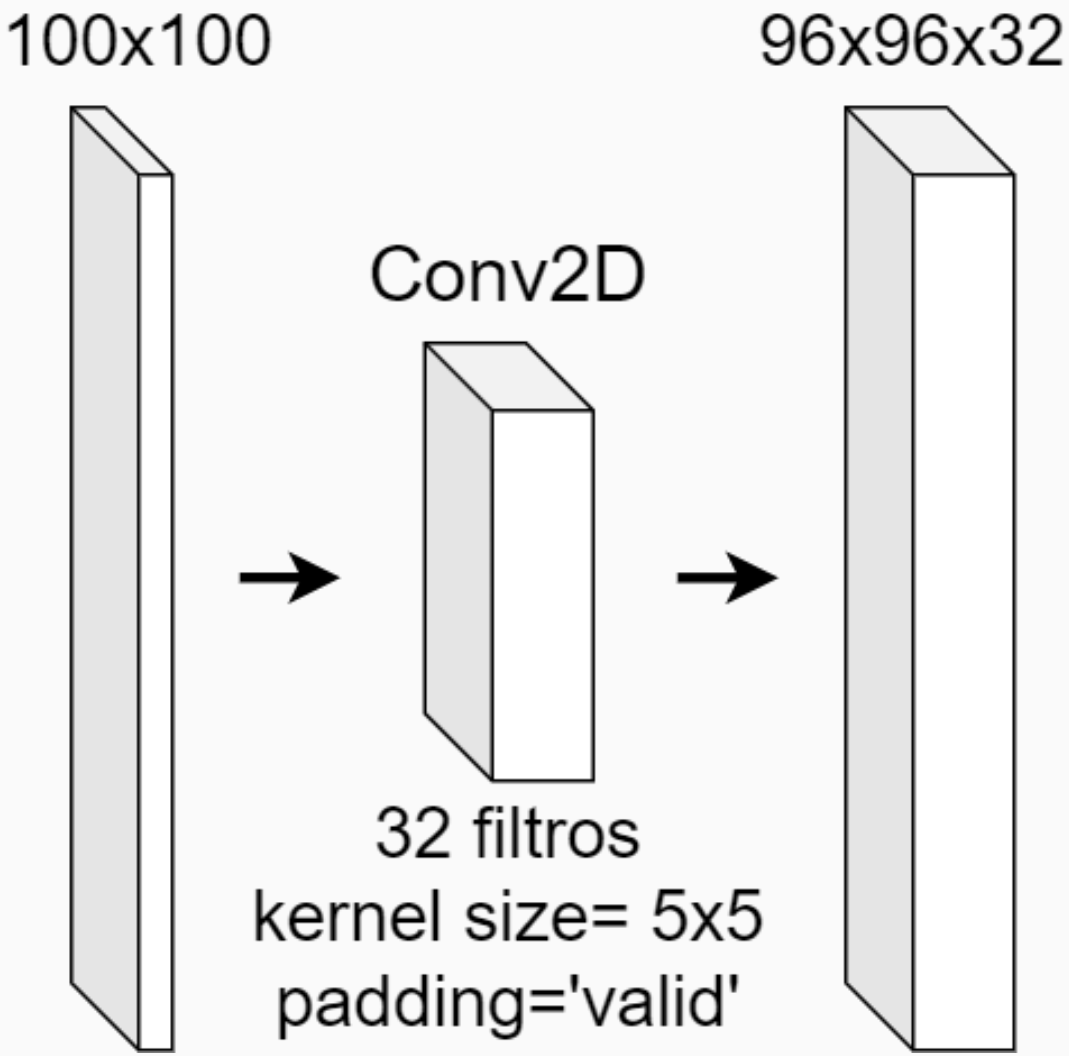
Resultado de una capa convolucional



Resultado de una capa convolucional



Resultado de una capa convolucional



Notebook de ejemplo, resultado de convolución

El siguiente notebook contiene un breve código para explorar el resultado de una capa convolucional.

- 03.02-DimensionesConv2D.ipynb

Referencias i

[1] Michael H. Herzog and Aaron M. Clarke (frontiers).
Cortex image.
[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.