

Procesamiento Digital de Imágenes



OpenCV (Open Source Computer Vision Library) es una biblioteca de código abierto especializada en procesamiento de imágenes y visión por computadora. Permite realizar tareas como detección de rostros, reconocimiento de objetos, seguimiento de movimiento y filtrado de imágenes. Es ampliamente utilizada en aplicaciones de inteligencia artificial, robótica y análisis visual en tiempo real.

OPEN CV



CONVOLUCIÓN 1D

En matemáticas, la convolución es una operación entre dos funciones $f(t)$ y $g(t)$, definida como:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau) \cdot g(t - \tau) d\tau$$

Combina ambas funciones para producir una tercera que representa cómo la forma de una se modifica por la otra.

$$(f * g)[n] = \sum_{k=-\infty}^{\infty} f[k] \cdot g[n - k]$$

En **tiempo discreto** también se usa la convolución, especialmente en procesamiento digital de señales e imágenes. Su definición es:

Donde $f[k]$ es la señal de entrada y $g[k]$ es la respuesta al impulso (kernel o filtro). Se calcula sumando productos desplazados, igual que en imágenes digitales.

CONVOLUCIÓN 2D

- Es un filtro de propósito general para las imágenes
- Es una matriz aplicada a una imagen
- Determina el valor de un píxel agregando los valores ponderados de los píxeles en la vecindad

En 2D, la convolución discreta se define como:

$$(f * h)[i, j] = \sum_m \sum_n f[m, n] \cdot h[i - m, j - n]$$

Donde $f[m, n]$ es la imagen y h es el filtro (kernel); se aplica desplazando el filtro sobre la imagen para obtener una nueva matriz transformada.

CONVOLUCIÓN 2D

- La matriz representa el filtro a aplicar
- Aplicamos el filtro a cada uno de los píxeles de la imagen
- Se estila tener la imagen en escala de grises

2	0	1	1
0	1	0	0
0	0	1	0
0	3	0	0

*An input image
(no padding)*

1	0	1
0	0	0
0	1	0

*A filter
(3x3)*

*Output image
(after convolving with stride 1)*

2	0	1	1
0	1	0	0
0	0	1	0
0	3	0	0

×

1	0	1
0	0	0
0	1	0

=

3	

2	0	1	1
0	1	0	0
0	0	1	0
0	3	0	0

×

1	0	1
0	0	0
0	1	0

=

3	2

2	0	1	1
0	1	0	0
0	0	1	0
0	3	0	0

×

1	0	1
0	0	0
0	1	0

=

3	2
3	

2	0	1	1
0	1	0	0
0	0	1	0
0	3	0	0

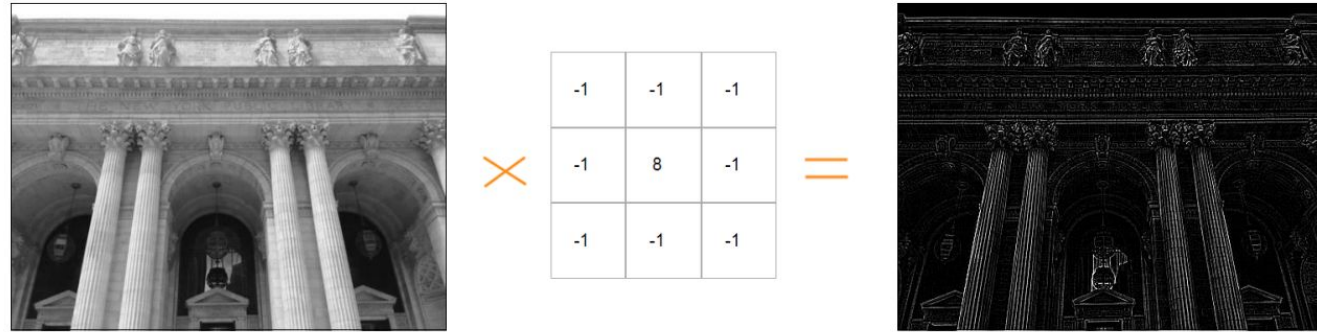
×

1	0	1
0	0	0
0	1	0

=

3	2
3	1

CONVOLUCIÓN 2D



filter2D : Aplica una convolución a una imagen haciendo uso de un kernel

Sintaxis: `cv2.filter2D(imagen, profundidad, kernel)`

Parametros:

imagen : Imagen a la cual aplicar la convolución

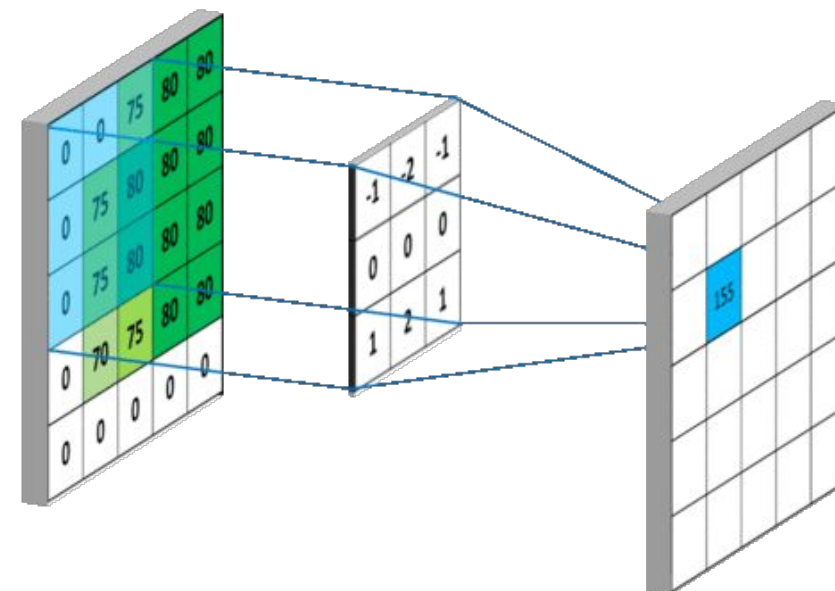
profundidad: Es la profundidad deseada para la imagen resultante

kernel: Filtro a aplicar en la imagen. En el caso que la imagen tenga varios canales, se puede dividir la imagen en sus distintos canales para aplicar diferentes filtros en cada canal. Los valores deben estar como punto flotante

Nota La profundidad es la precisión que tiene cada píxel. Define la cantidad de bits a usar y si se usa «unsigned, int, float o double». Si es igual a -1, entonces la imagen resultante tiene la misma precisión que la imagen original

APLICACIONES

- Detección de bordes (filtros Sobel, Prewitt, Canny)
- Suavizado o desenfoque (filtro promedio o gaussiano)
- Aumento de contraste o realce de detalles
- Eliminación de ruido (filtro de mediana o gaussiano)
- Enfoque (sharpening) de imágenes
- Detección de esquinas y formas
- **Extracción de características para machine learning o redes neuronales**
- Reconocimiento de patrones u objetos
- **Filtros de convolución en redes neuronales convolucionales (CNNs)**
- Transformaciones morfológicas (dilatación, erosión con kernels estructurantes)
- Segmentación de imágenes
- Mejoramiento en imágenes médicas y satelitales

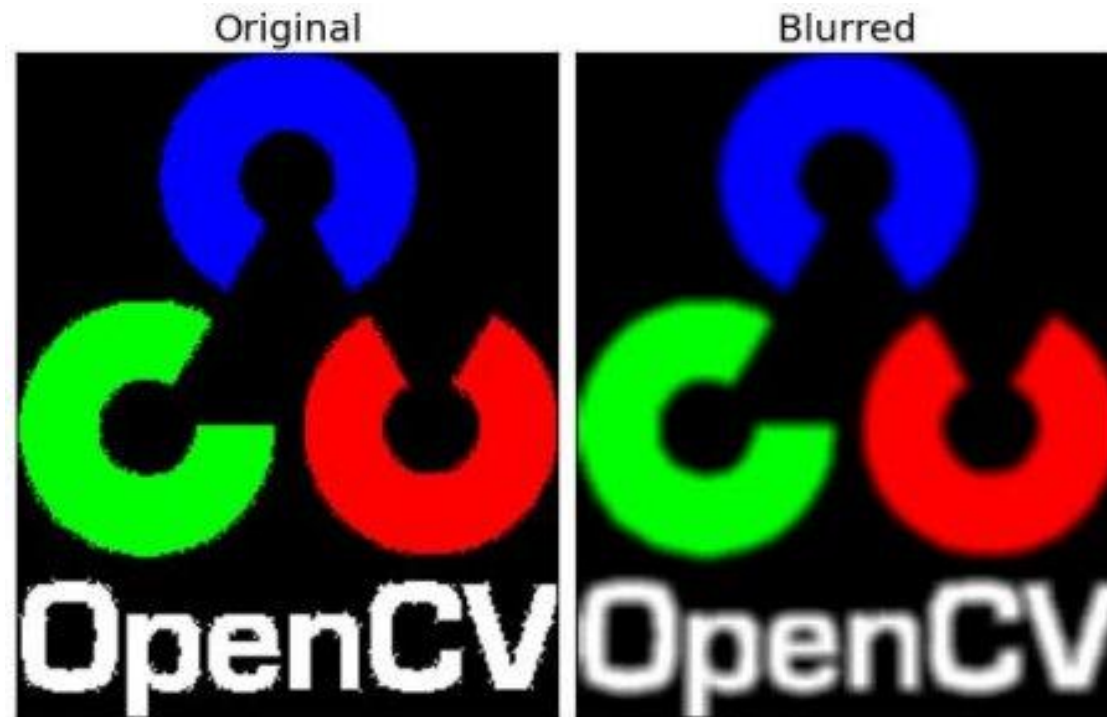


FILTROS PASA BAJO

`blur`

`medianBlur`

`gaussianBlur`



FILTROS PASA BAJO

blur

blur : Desenfoca/Difumina una imagen mediante el filtro de cuadro normalizado

Sintaxis: `cv2.blur(imagen, tamaño)`

Parametros:

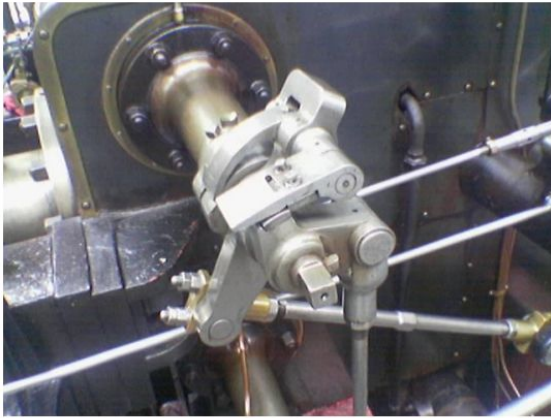
imagen : Imagen a la cual aplicar la convolución

tamaño : Tamaño del kernel como una tupla (ancho, alto)

Nota El filtro usado es:

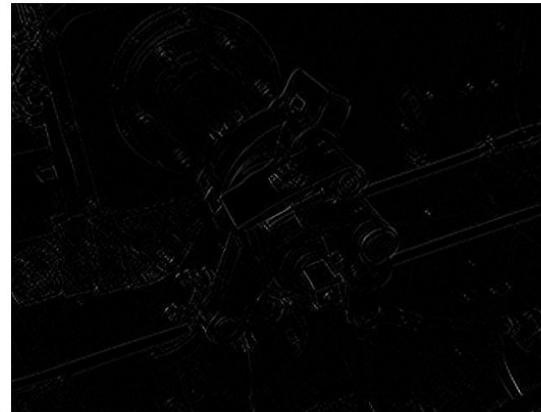
$$\frac{1}{\text{ancho} \times \text{alto}} * \begin{bmatrix} 1 & \dots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \dots & 1 \end{bmatrix}$$

FILTROS PASA ALTO

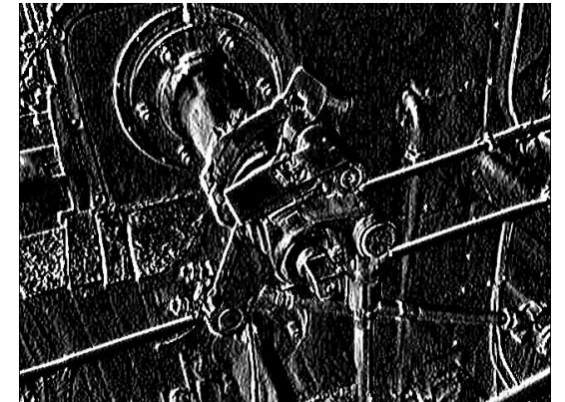


Original

Sobel



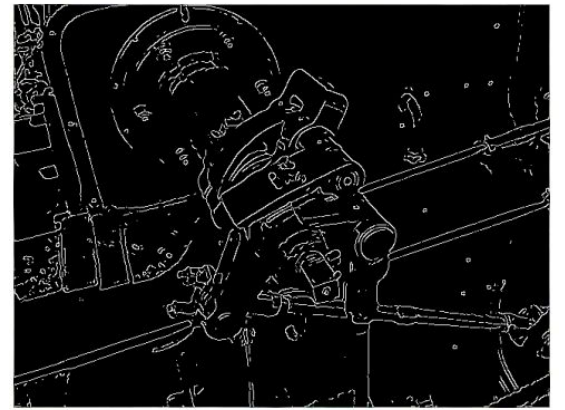
Scharr



Laplacian



Canny



Derivada de Sobel

- Es una operación conjunta de suavizado gaussiano más la diferenciación, lo que le permite ser más resistente al ruido
- Debido que el operador puede ser aplicado sobre el eje X e Y, se usa 2 kernels para aplicar una convolución a la imagen, con el fin de obtener las derivadas
- Un kernel se usa para los cambios horizontales y otro para los verticales

A: Imagen original

B: Imagen modificada

K: Kernel

$$B_x = K_x * A$$

$$B_y = K_y * A$$

$$B_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * A$$

$$B_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} * A$$

$$B = \sqrt{B_x^2 + B_y^2} \quad \text{ó} \quad B = |B_x| + |B_y|$$

Derivada de Sobel

Sobel : Aplica el operador Sobel a una imagen

Sintaxis: `cv2.Sobel(imagen, profundidad, dx, dy, kernel)`

Parametros:

imagen : Imagen a la cual aplicar la operación

profundidad: Es la profundidad deseada para la imagen resultante

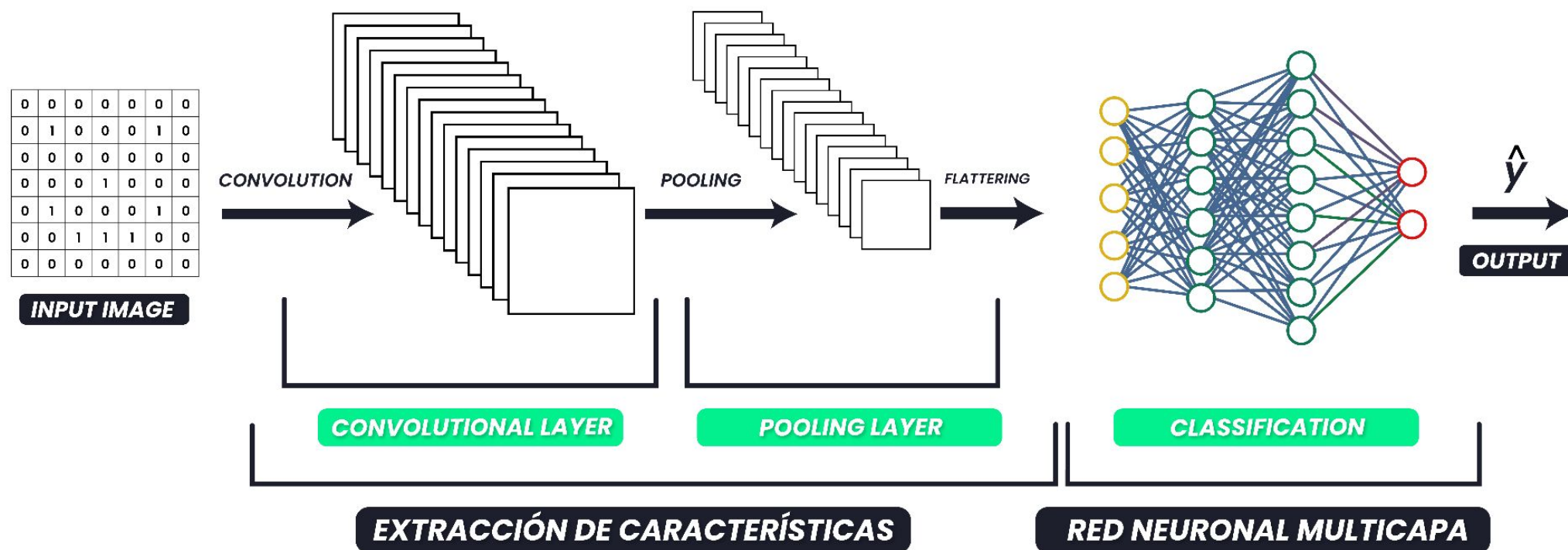
dx : Orden de la derivada en el eje X

dy : Orden de la derivada en el eje Y

kernel: Tamaño del kernel a aplicar en la imagen

Nota Para imágenes de 8 bits, el resultado serán derivadas truncadas. El tamaño del kernel debe ser números enteros positivos e impares.

Redes Neuronales Convolucionales

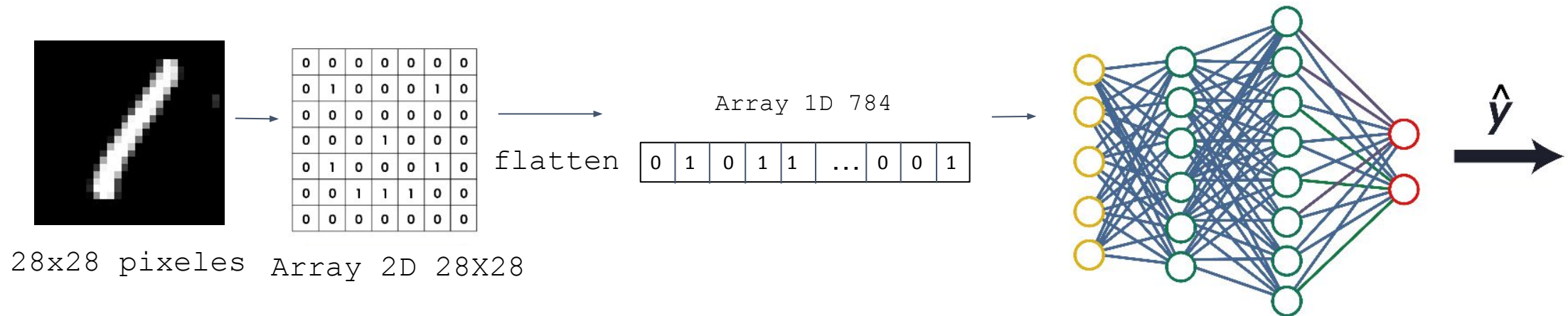


Redes Neuronales Convolucionales

Las redes neuronales convolucionales (CNN) son una arquitectura específica de red neuronal muy efectiva para trabajar con imágenes.

Las CNN constan principalmente de capas convolucionales que aplican filtros sobre las imágenes.

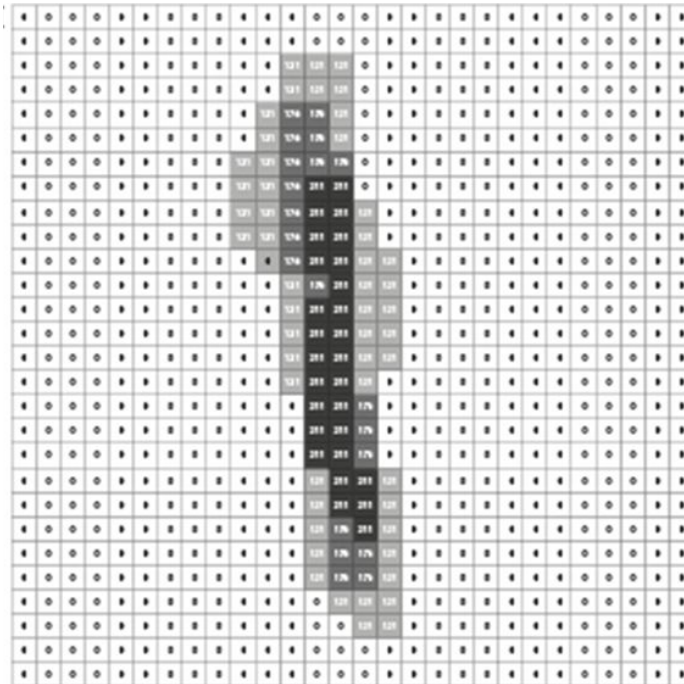
¿Por qué no podemos utilizar redes neuronales artificiales (ANN)?



El "aplanamiento" hace que perdamos información 2D como por ejemplo la relación de cada píxel con sus píxeles vecinos, por eso la arquitectura de ANN no es útil en imágenes.

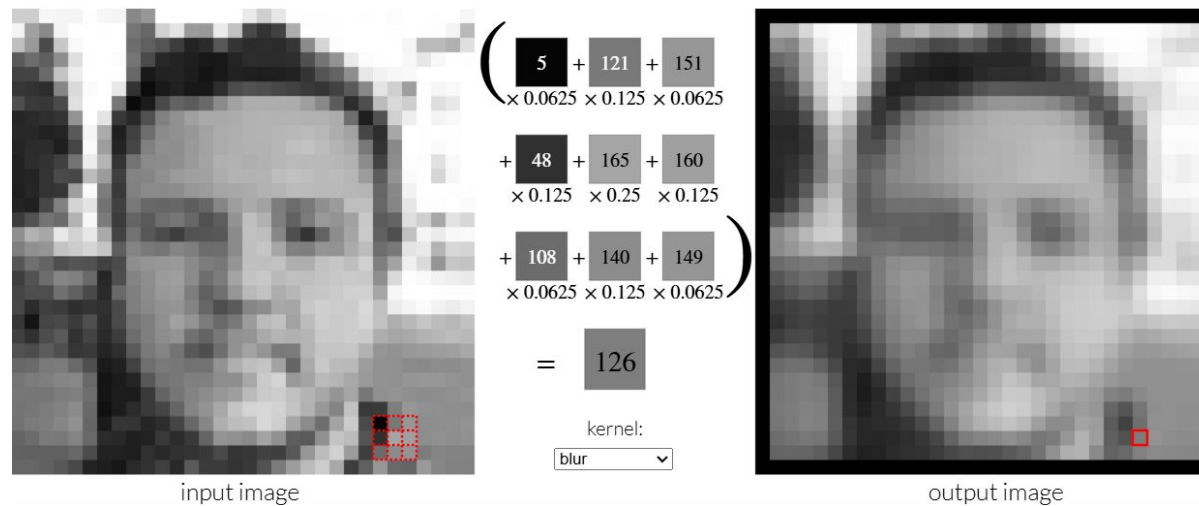
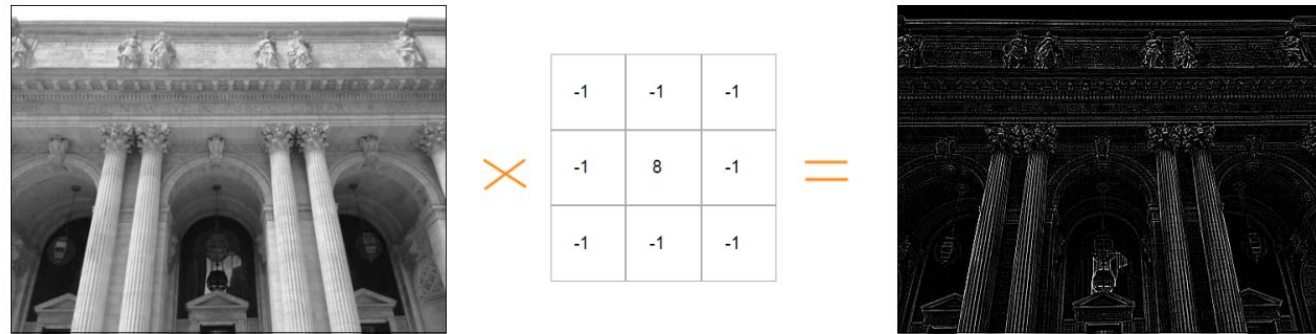
Redes Neuronales Convolucionales

El "aplanamiento" hace que perdamos información 2D como por ejemplo la relación de cada píxel con sus píxeles vecinos, por eso la arquitectura de ANN no es útil en imágenes.



¿Qué son los filtros de Imagen y los kernels?

Los filtros de imagen nos permiten transformar imágenes mediante la aplicación de una pequeña matriz (filtro) a esa imagen.



CONVOLUCIÓN 2D

- La matriz representa el filtro a aplicar
- Aplicamos el filtro a cada uno de los píxeles de la imagen
- Se estila tener la imagen en escala de grises

2	0	1	1
0	1	0	0
0	0	1	0
0	3	0	0

*An input image
(no padding)*

1	0	1
0	0	0
0	1	0

*A filter
(3x3)*

*Output image
(after convolving with stride 1)*

2	0	1	1
0	1	0	0
0	0	1	0
0	3	0	0

×

1	0	1
0	0	0
0	1	0

=

3	

2	0	1	1
0	1	0	0
0	0	1	0
0	3	0	0

×

1	0	1
0	0	0
0	1	0

=

3	2

2	0	1	1
0	1	0	0
0	0	1	0
0	3	0	0

×

1	0	1
0	0	0
0	1	0

=

3	2
3	

2	0	1	1
0	1	0	0
0	0	1	0
0	3	0	0

×

1	0	1
0	0	0
0	1	0

=

3	2
3	1