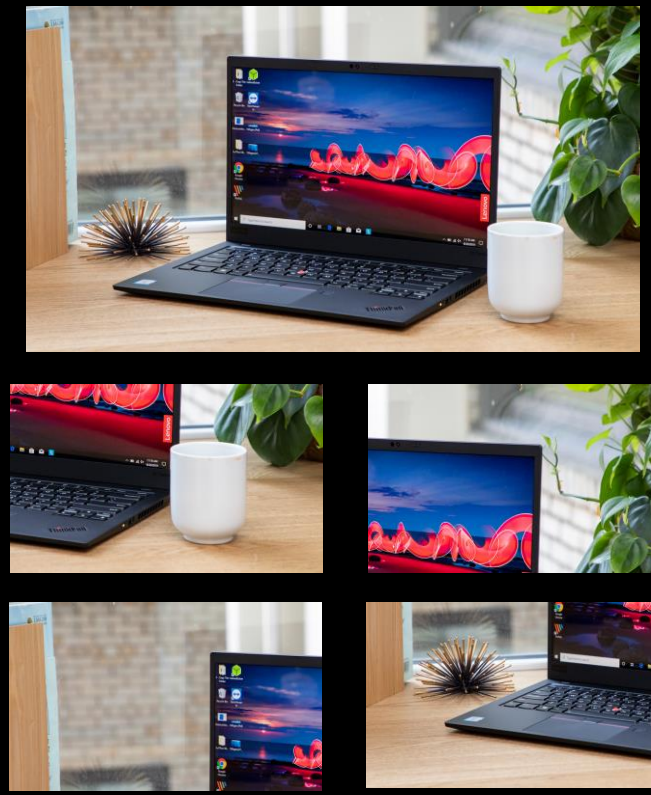
Neural Networks - Convolutional

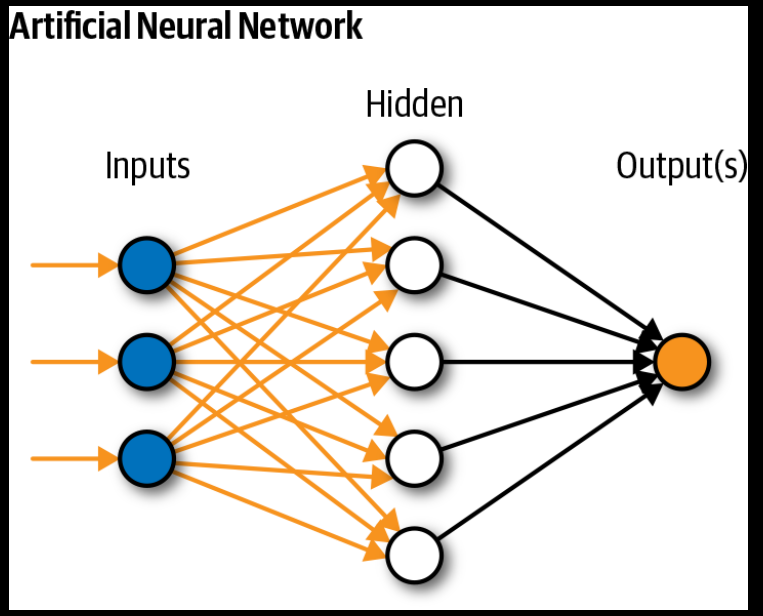

Lenovo
ThinkPad

Dos dimensiones

Los pixels entran como un vector, en una red neuronal normal. Independiza los pixels, y perderemos la relación 2D, formas, estructuras y patrones.

Las redes convolucionales se crean para sacarle partido a la estructura espacial 2D de una imagen.



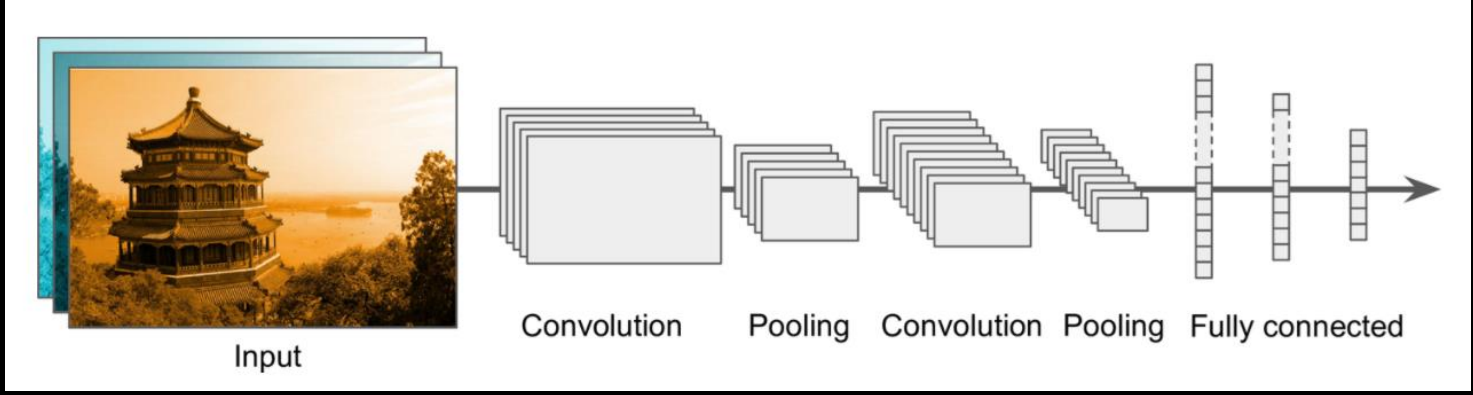


Imágenes

Para realizar predicciones sobre datos espaciales lo mejor es acudir a RRNN, ya que el espacio dimensional es muy grande (todos los pixels), y con técnicas como las convoluciones y el pooling, podremos trabajar con estos datos.

¿Cómo interpretamos el problema?

1. Cada pixel es una entrada del algoritmo. Una neurona de entrada. Ahora manejamos el problema en 2D.
2. Después de la entrada habrá una serie de hidden layers mediante las que transformaremos y reduciremos la dimensionalidad de la entrada. En esta etapa se aplica lo que se denomina convoluciones en la imagen, que no son más que transformaciones.
3. Después habrá una serie de capas de neuronas con las que entrenaremos el output de las convoluciones.
4. Output



Input

Tendremos que poner los datos en el formato de entrada de la red, que será un tensor, un array de numpy compuesto por la altura x anchura de las imágenes x los canales. Los canales podrían ser R,G,B o escala de grises.

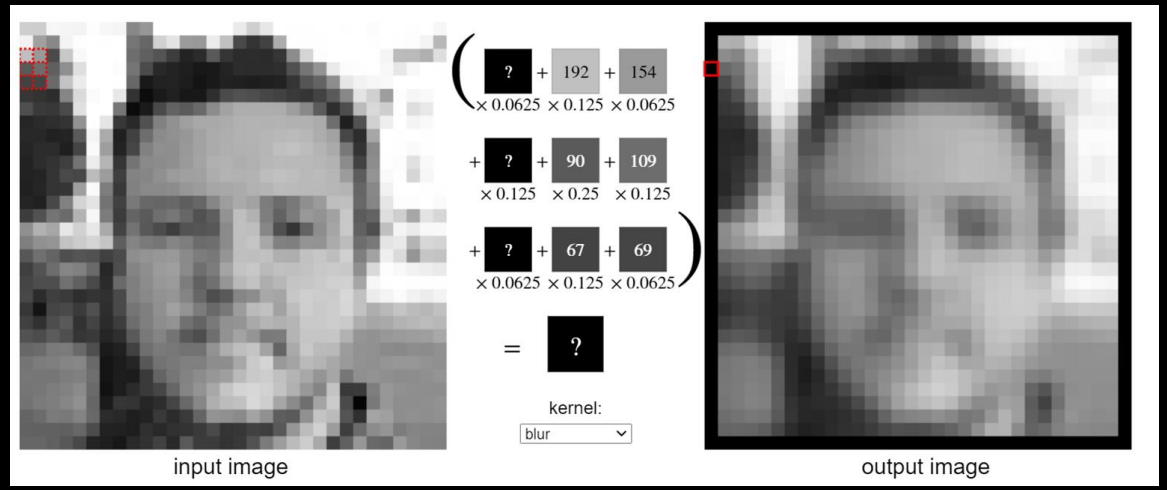
Normalizar
Otro punto importante es normalizar los datos para mejorar los tiempos de entrenamiento y el performance del modelo. Dependerá de la codificación de la imagen, pero lo habitual es que los canales de colores vayan de 0 a 255.

Convolutional layer

¿Qué es una convolución? preprocesado o transformación que se aplica a la imagen para destacar las características predictivas de la misma. Se trata de aplicarle un filtro a la imagen.

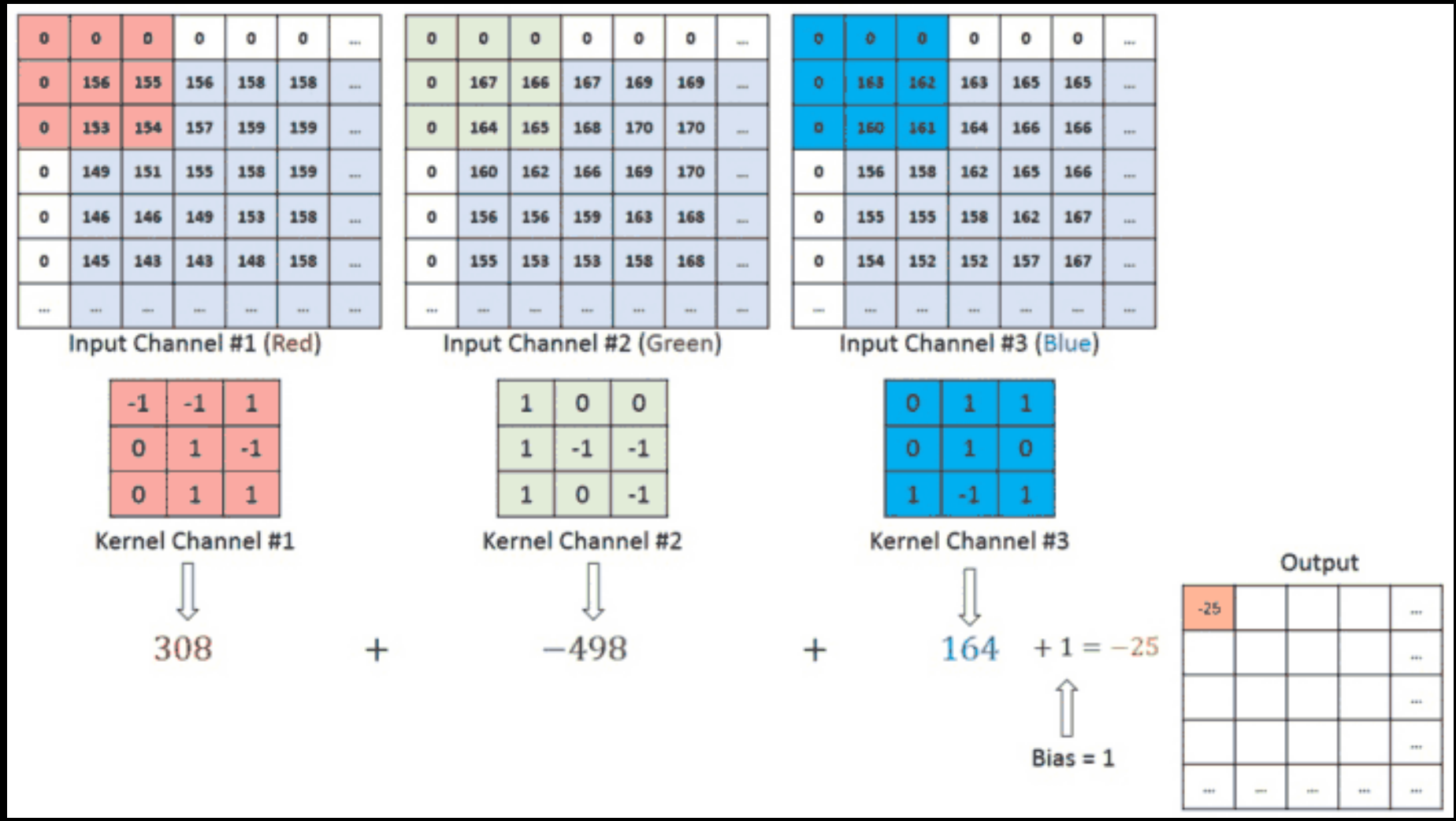
Una convolución se define por:

1. **Matriz o kernel**. Tendrá que ser de dimensiones menor a la imagen
2. **Padding**: márgenes por fuera de la imagen. Se usa para que el centro del kernel coincida con el pixel en cuestión.
3. **Stride**: saltos de la matriz entre un pixel y otro.



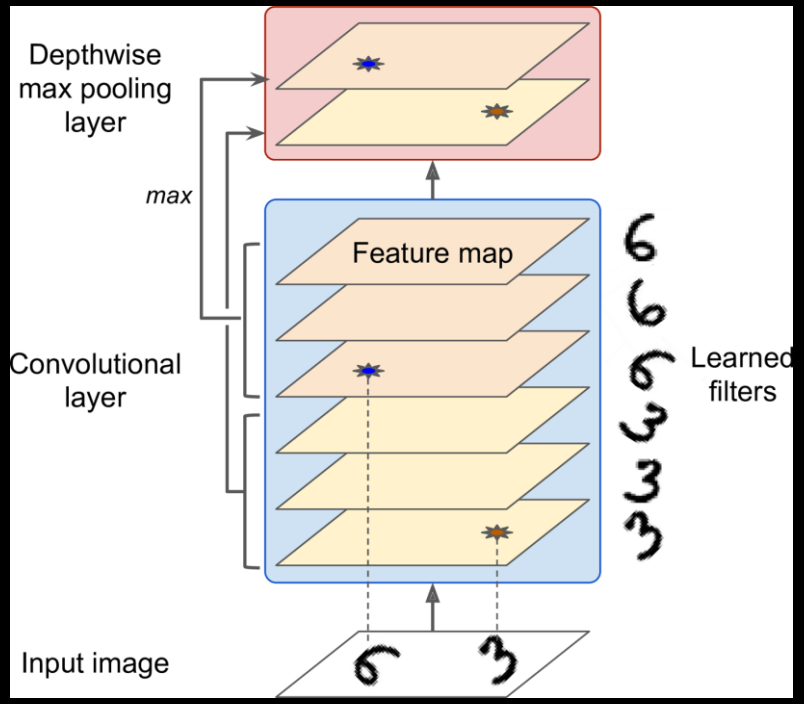
En [esta página] podrás jugar con diferentes convoluciones

Convolutional layer



¿Qué filtro defino?

No lo decicimos nosotros. Probamos varios filtros aleatorios. La red los elige aleatoriamente, y crea features nuevas. Se va entrenando la red con esas features, ajustando el tipo de filtro que mejor prediga.

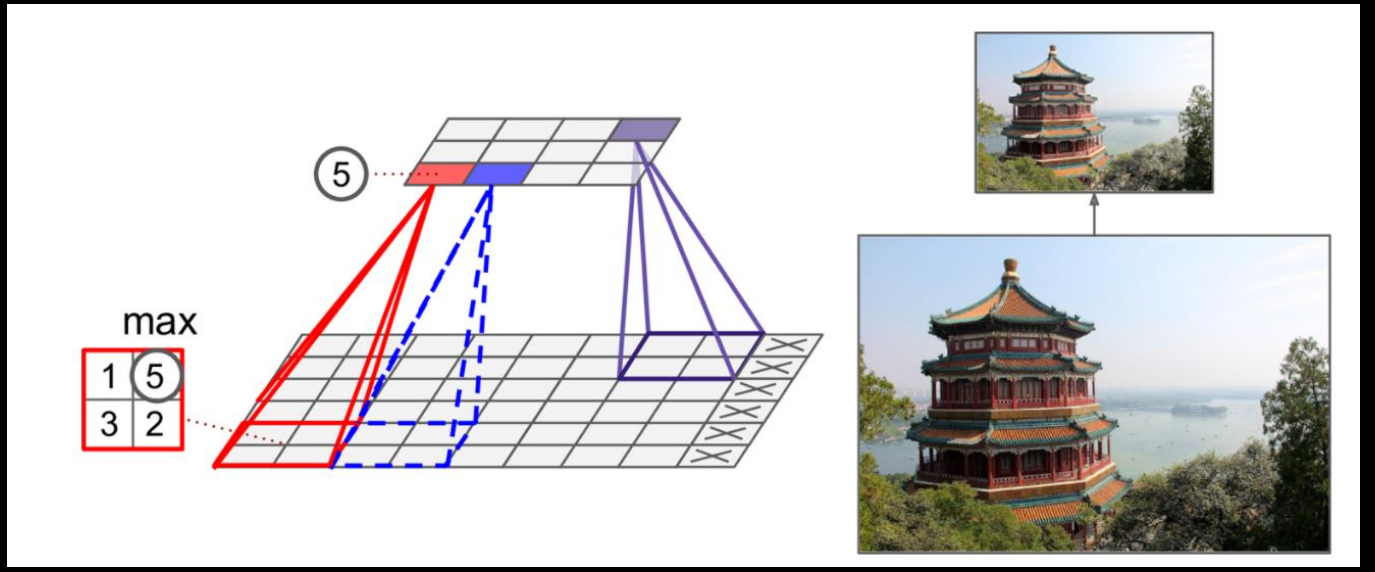


Operation	Filter	Convolved Image
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur (approximation)	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

Pooling layers

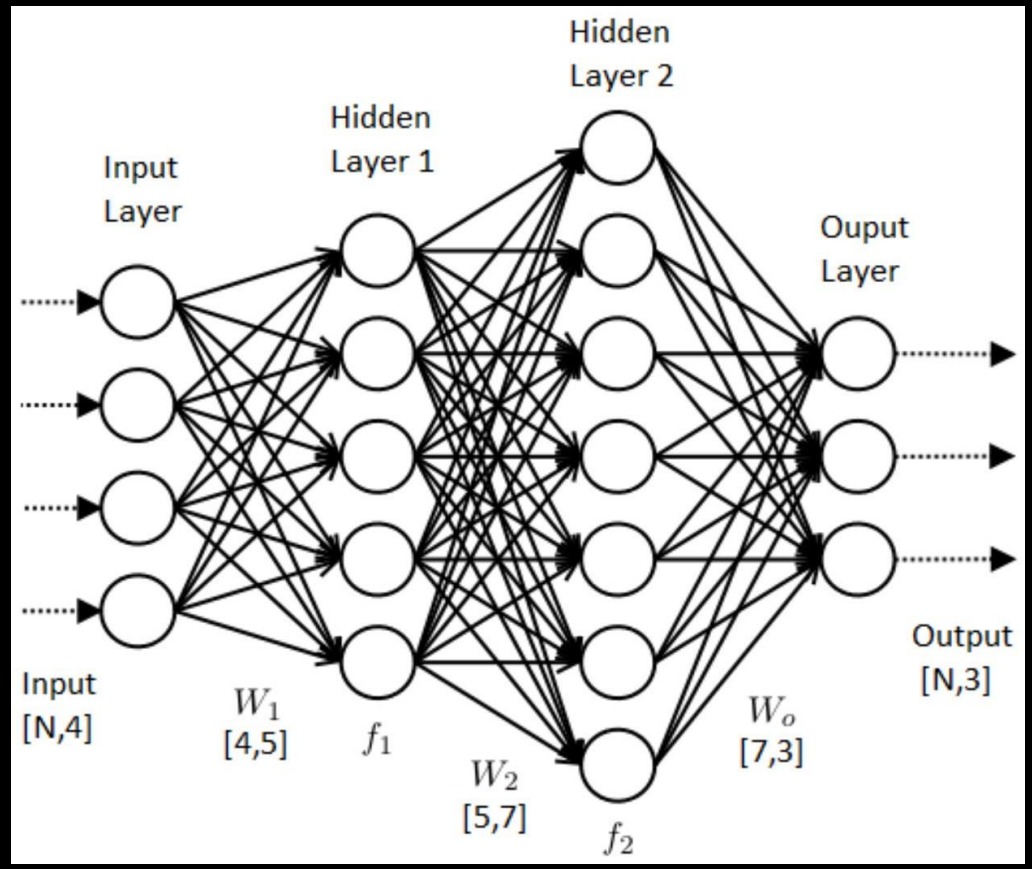
Las pooling layers se encargan de realizar un submuestreo de sus inputs, de cara a reducir la dimensionalidad, el uso de memoria y el número de parámetros.

Estas capas funcionan como agregadoras, normalmente con el máximo de una matriz. En el siguiente ejemplo calcula el máximo dentro de una matriz de 2x2.



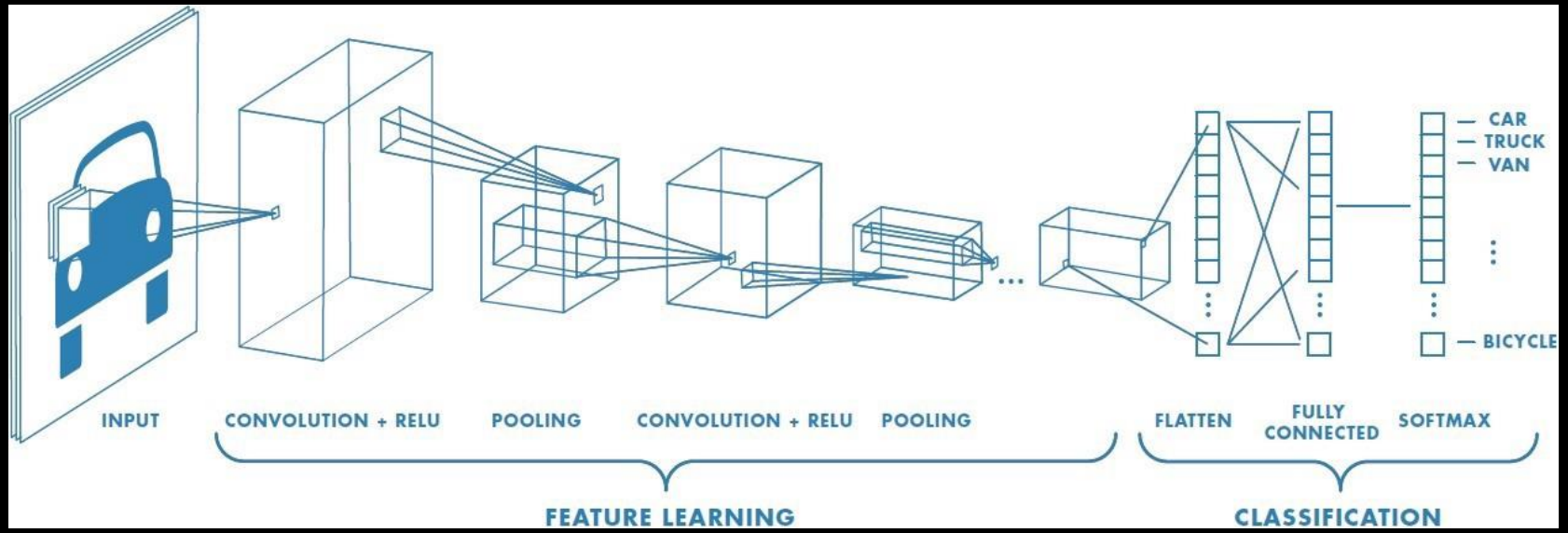
Fully connected layers

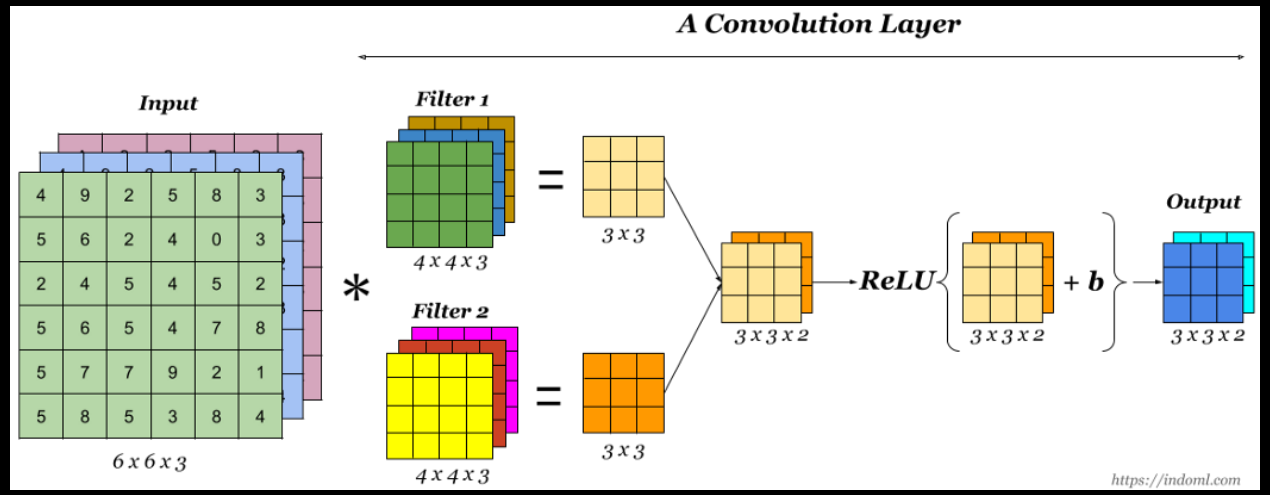
Tras una serie de capas convolucionales, las features se aplanan y entran como un vector 1D a una fully connected layers, que son las capas de neuronas que ya conocemos.

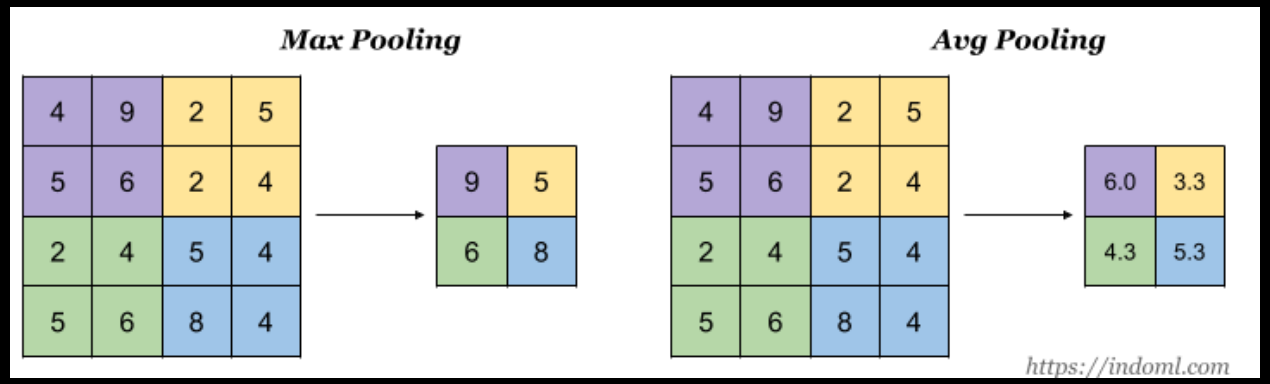


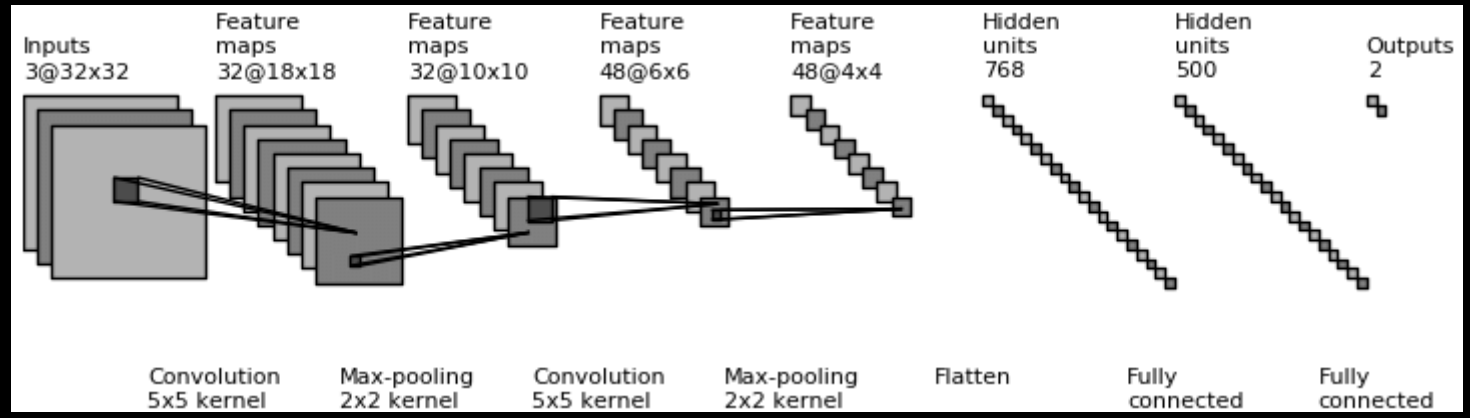
Convolutional arquitecture

convolutional + max pooling
La capa convolucional detecta features locales interesantes, mientras que la max pooling focaliza la atención en esas features, reduciendo la dimensionalidad




A Convolution Layer
Input
Filter 1
Filter 2
6 x 6 x 3
4 x 4 x 3
4 x 4 x 3
3 x 3
3 x 3
3 x 3 x 2
ReLU
3 x 3 x 2
+ b
Output
3 x 3 x 2
https://indoml.com


Max Pooling
Avg Pooling
https://indoml.com


Inputs
3@32x32
Feature
maps
32@18x18
Feature
maps
32@10x10
Feature
maps
48@6x6
Feature
maps
48@4x4
Hidden
units
768
Hidden
units
500
Outputs
2
Convolution
5x5 kernel
Max-pooling
2x2 kernel
Convolution
5x5 kernel
Max-pooling
2x2 kernel
Flatten
Fully
connected
Fully
connected

Preguntas