Ayudantía 13

Redes Neuronales Convolucionales

Por Martín Lagies y Willy Pugh

24 de junio 2024

Recordemos los MLP's...

Modelos basados en el uso de perceptrones, que al tener varias capas:

- Pueden resolver problemas complejos
- Son difíciles de entrenar y de interpretar
- Sufren serios problemas de sobreajuste (son tan poderosos que memorizan hasta el ruido)

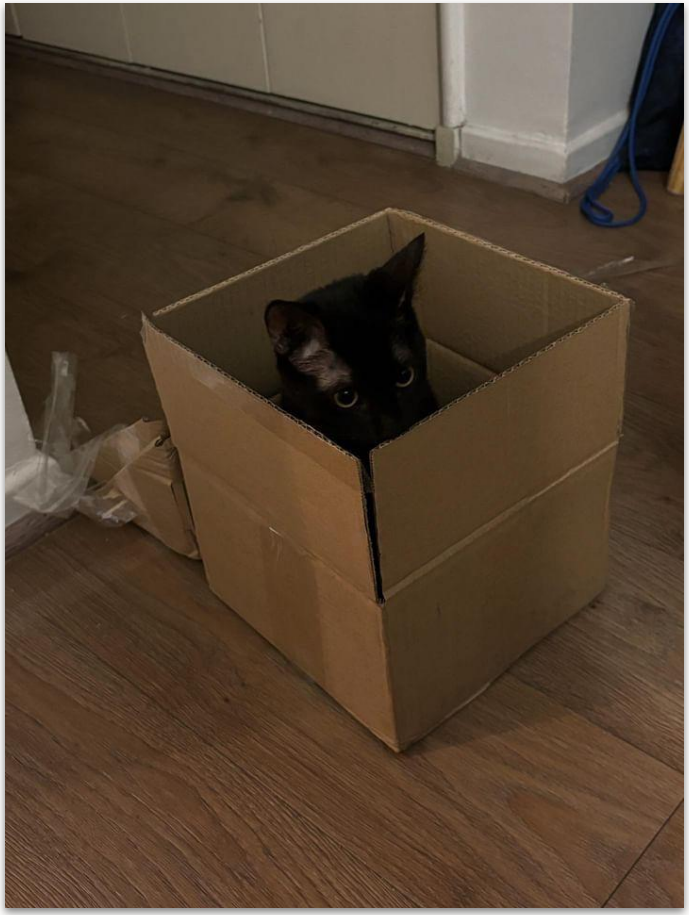
Los MLP's tiene un problema mayor...

Fracasan catastróficamente en conjuntos de datos visuales

Problemas de los MLP's

Cuando utilizamos un MLP para analizar una imagen, estamos utilizando capas densas, en las que cada neurona analiza toda la imagen.

Esto genera una cantidad absurda de parámetros.

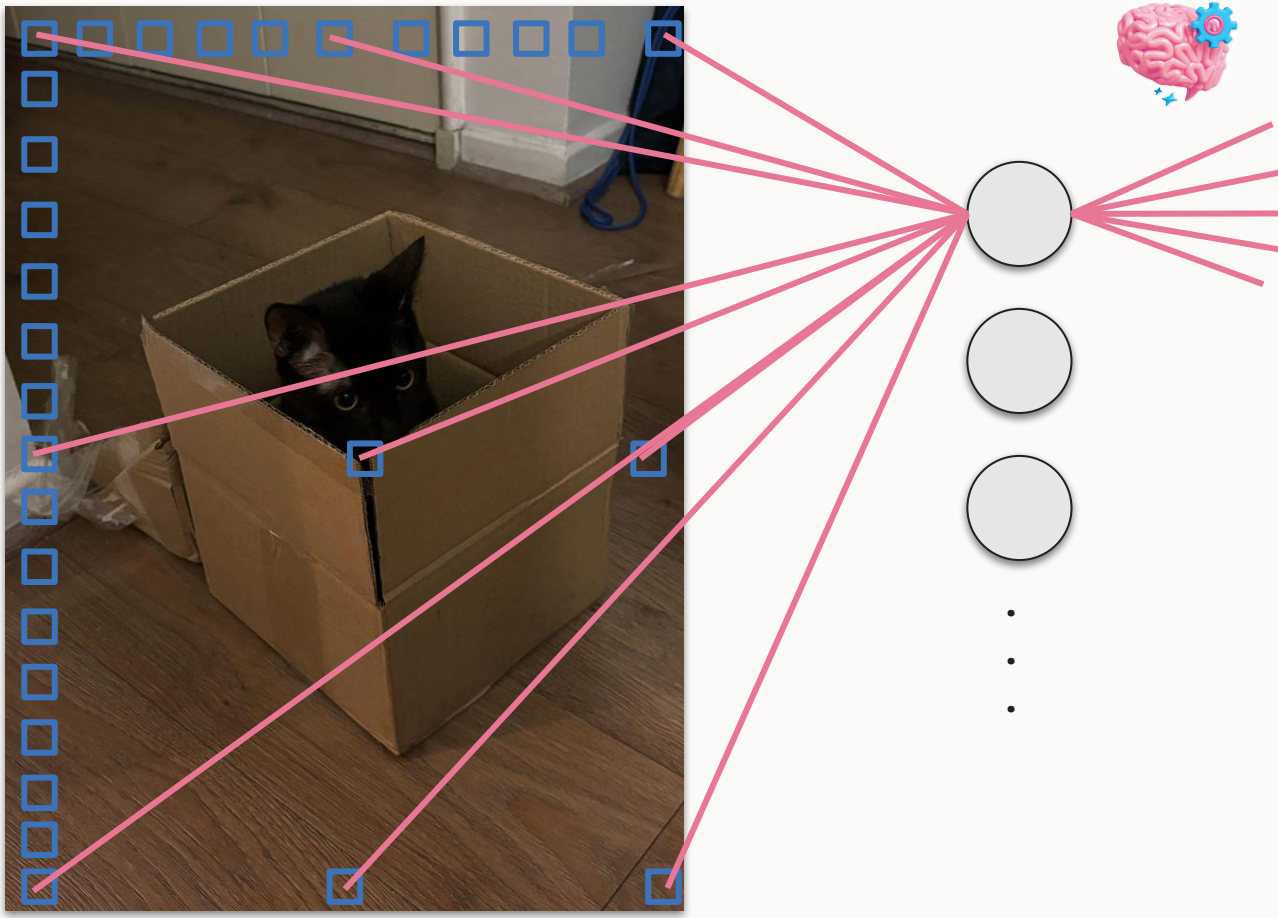


Supongamos que queremos analizar esta imagen...

Problemas de los MLP's

Cuando utilizamos un MLP para analizar una imagen, estamos utilizando capas densas, en las que cada neurona analiza toda la imagen.

Esto genera una cantidad absurda de parámetros.



Utilicemos otro modelo

Podemos basarnos nuevamente en lo que hace nuestro cerebro: utilizar neuronas específicas, que reconozcan *features* o patrones específicos de forma jerárquica, detectando desde patrones locales a globales.

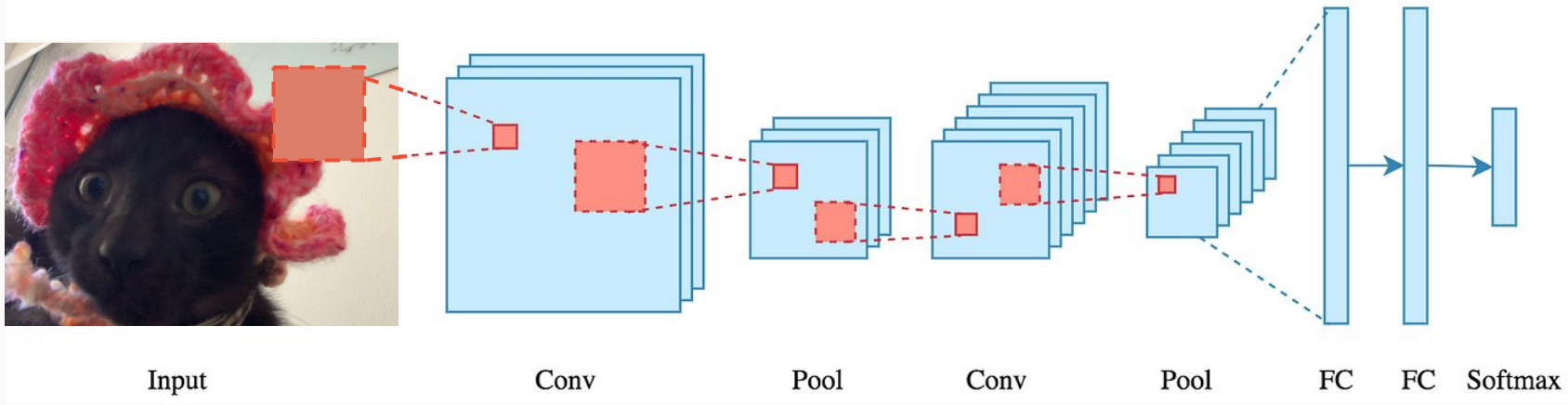
Si modelamos esto con neuronas capaces de barrer una imagen, entonces logramos disminuir considerablemente el número de parámetros.

Utilicemos otro modelo

Si modelamos esto con neuronas capaces de barrer una imagen, entonces logramos disminuir considerablemente el número de parámetros.

¿Qué son las Redes Neuronales Convolucionales (CNN's)?

Tipo de arquitectura de red neuronal especialmente diseñada para el procesamiento de datos estructurados en cuadrícula, como imágenes o audio



CNN's vs MLP's

CNN's

- Más utilizadas con datos de imágenes
- Se utilizan para grandes bases de datos con mucha información, por sus reducidos parámetros
- Es intrínsecamente más complejo debido a sus variadas capas

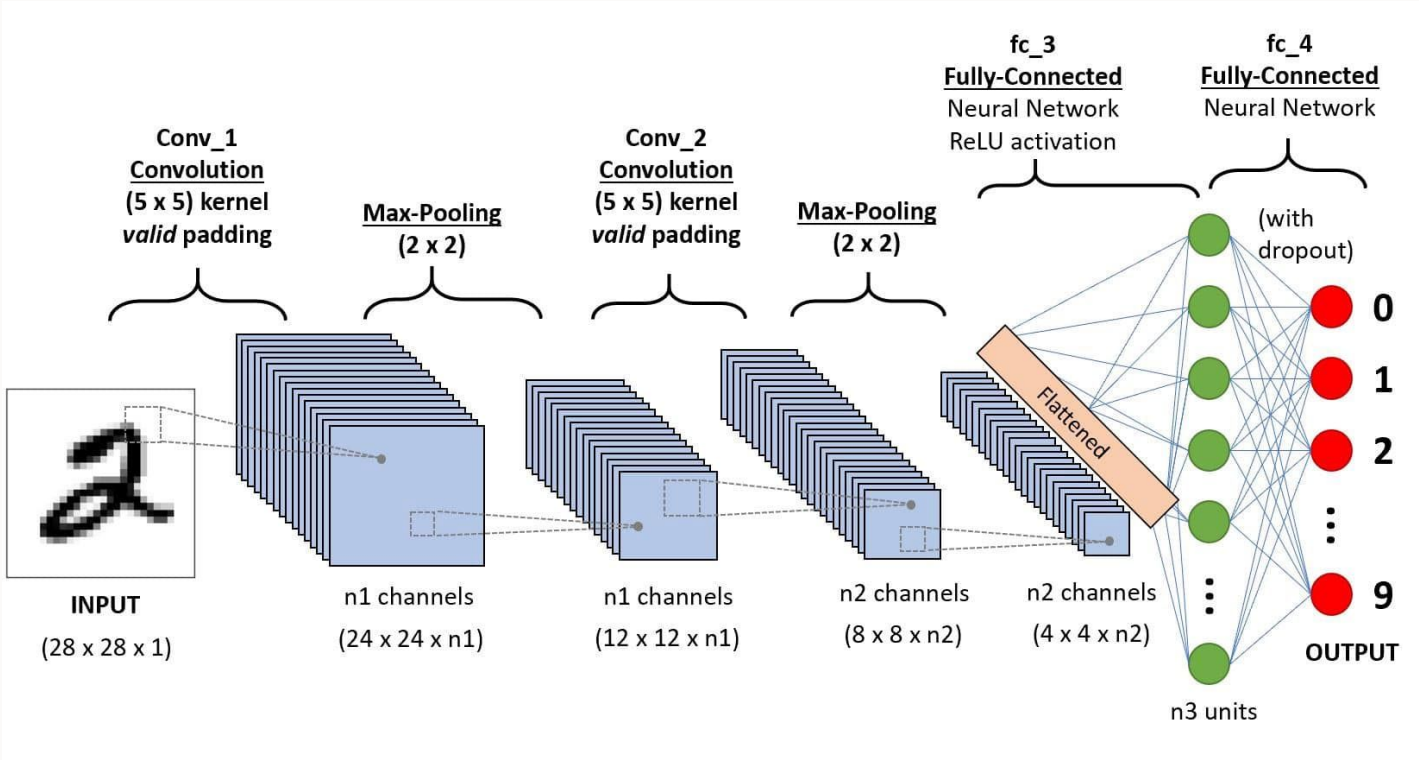
- Son las redes profundas más eficientes en el análisis de imágenes

MLP's

- Más utilizadas con datos tabulados
jas
- Su número de parámetros crece con los datos, por lo que no siempre son convenientes
- "Solo" es una red de neuronas
jsdjsdjsjd

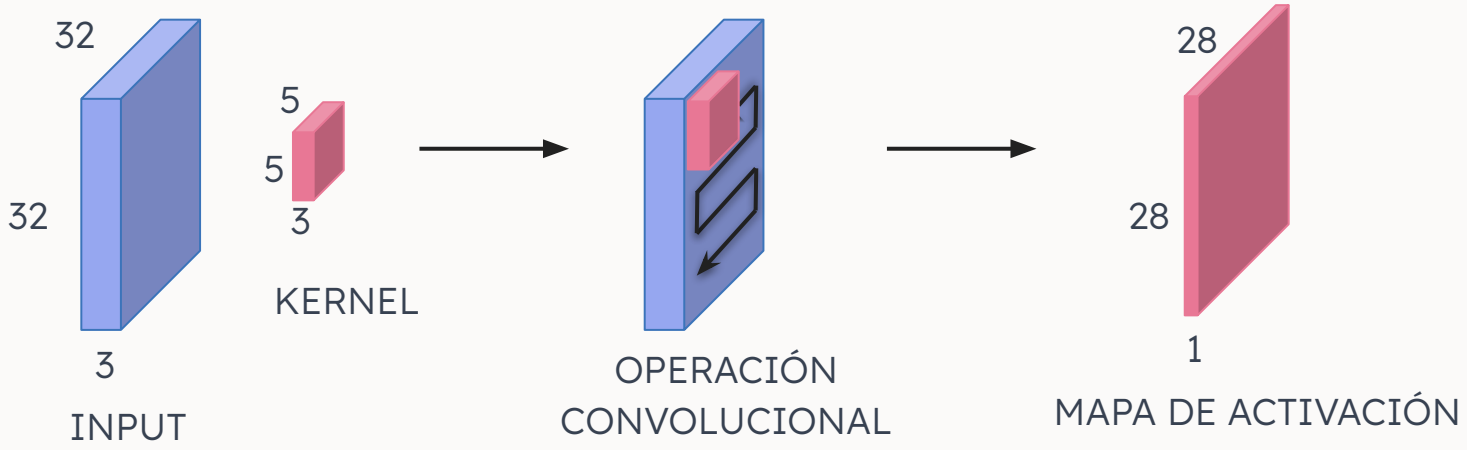
- Son ineficientes en el análisis de imágenes, y no son invariantes bajo traslaciones

Capas en las CNN's



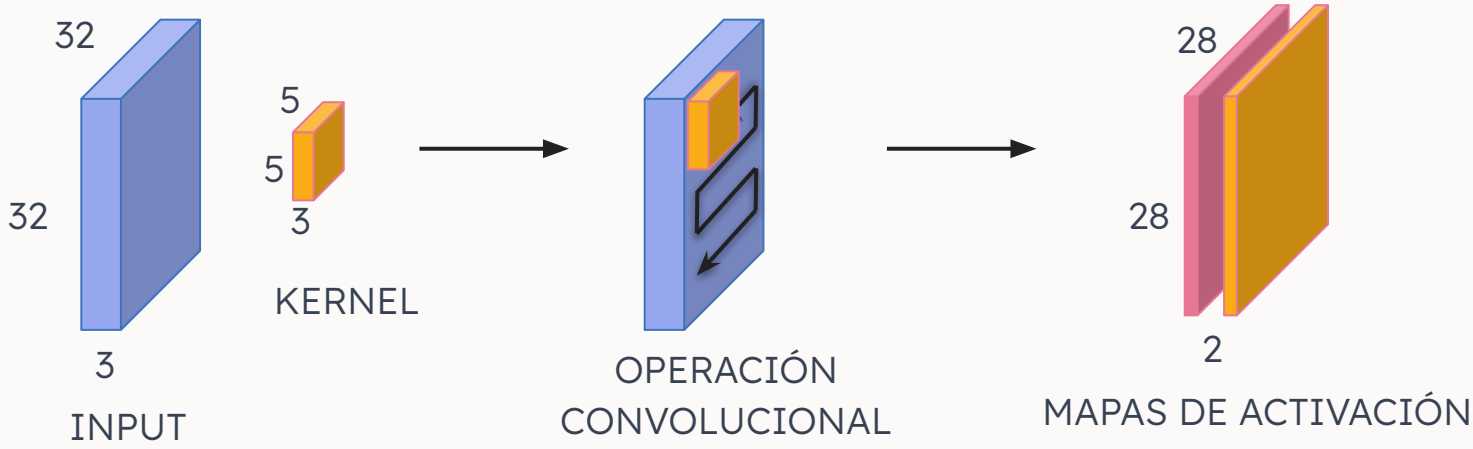
Capas Convolucionales

Pueden ser pensadas como los “ojos” de una CNN, en el sentido de que buscan atributos específicos en los datos de entrada. Esto se logra a partir de una operación convolucional, que a su vez utiliza un kernel.



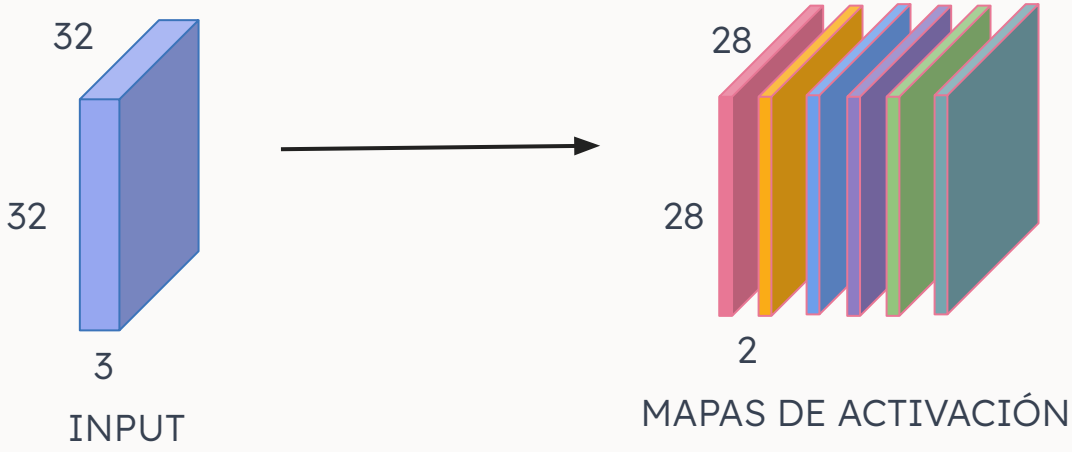
Capas Convolucionales

Si ahora utilizamos un segundo filtro para el mismo input, generamos otro mapa de activación:



Capas Convolucionales

Podemos hacer esto con una serie de filtro:

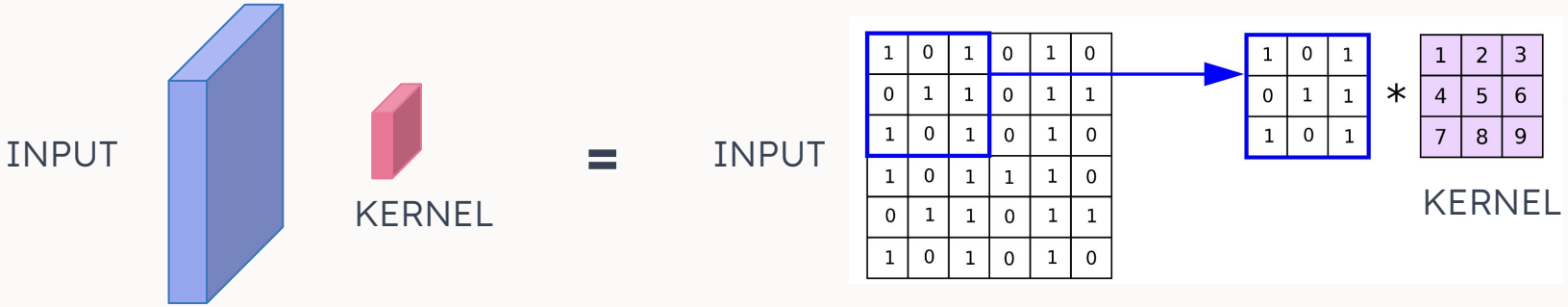


Notamos que si seguimos haciendo esto, con más capas, iremos creando más mapas de menores dimensiones.

Siendo más formales...

Kernel

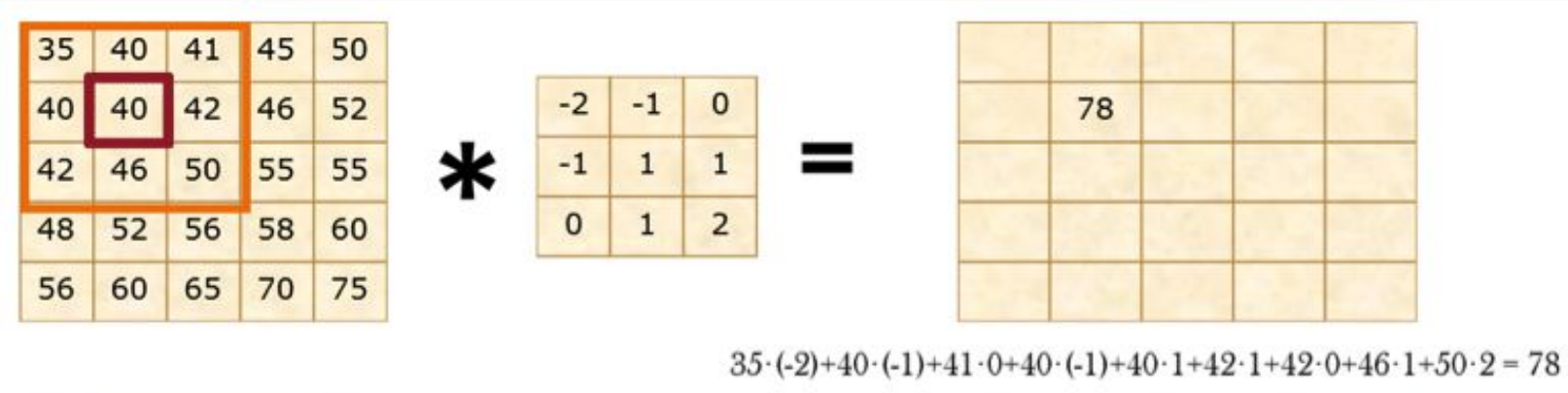
- Todos los elementos que componen un filtro son llamados colectivamente “kernel”.
- Matrices de pesos, en las que cada elemento contribuye al cálculo de la capa de salida.
- Estos elementos son justamente los pesos que aprende la red neuronal durante el entrenamiento.
- En análisis de imágenes existen distintos kernels que identifican distintos patrones (por ejemplo, líneas verticales, horizontales o diagonales).

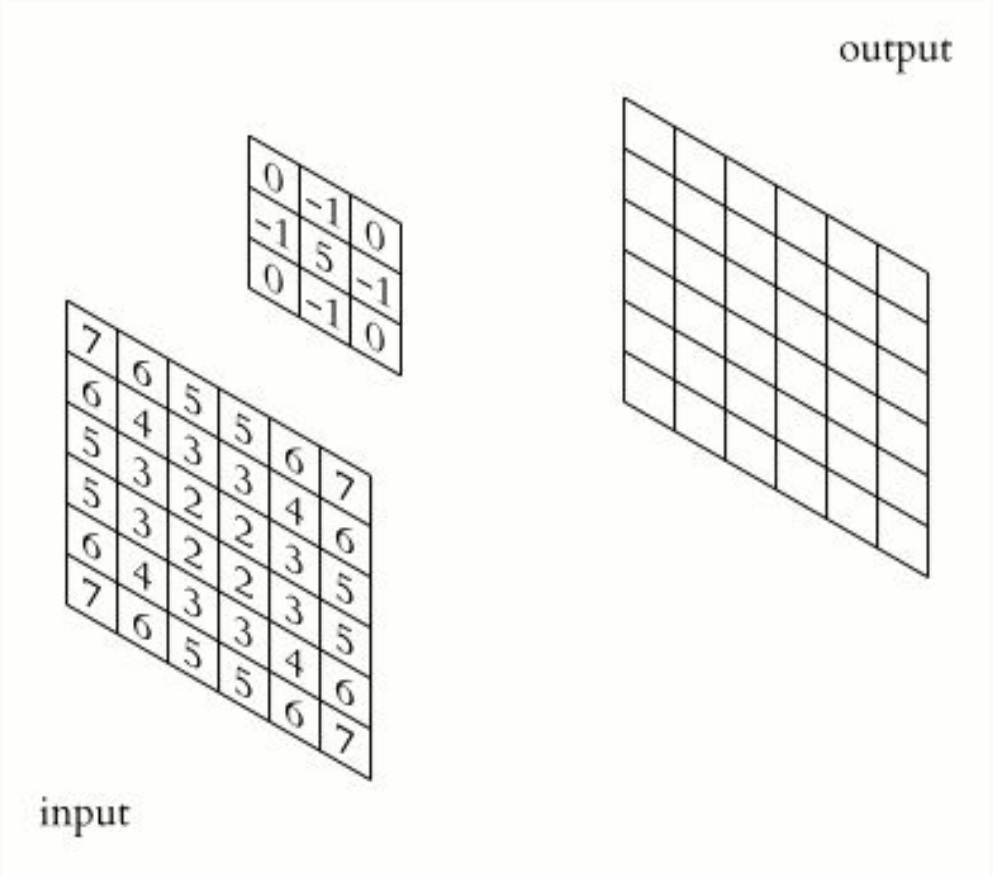


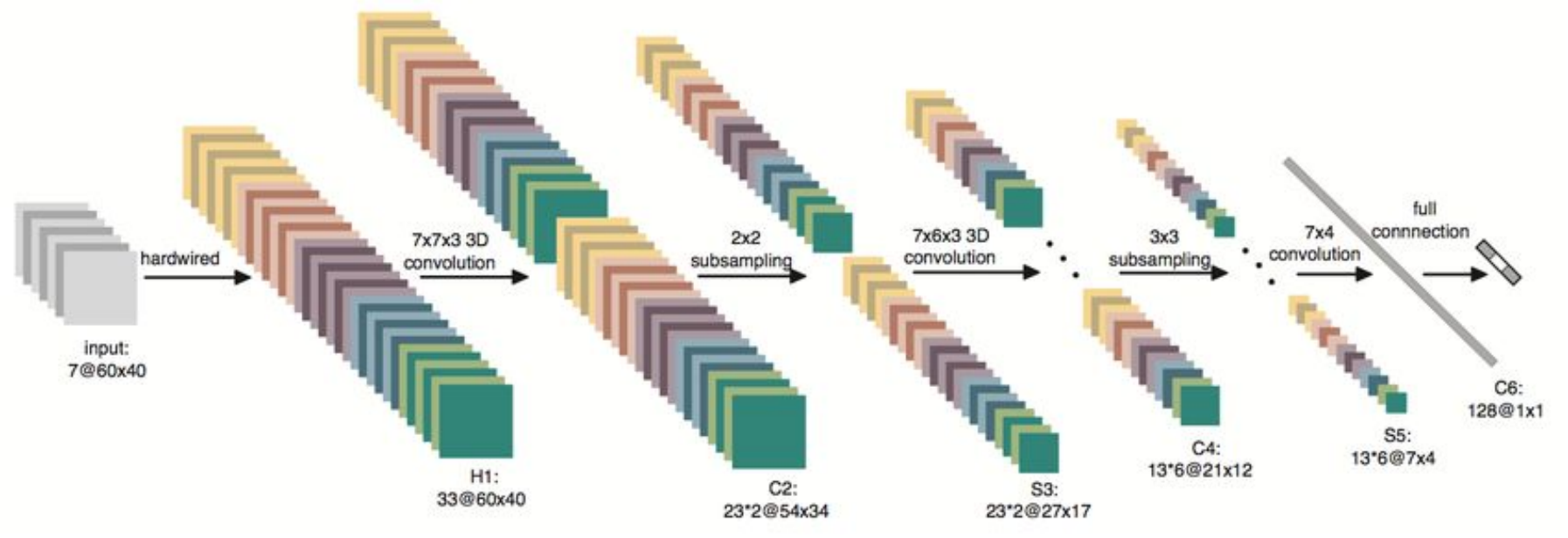
Siendo más formales...

Operación Convolucional

- Consiste en posicionar el kernel sobre una porción del input, y multiplicar los elementos de ambos.
- Puede interpretarse como un producto punto, y su resultado es un único número que representa la transformación para una parte específica del input.
- Con los resultados se construyen los mapas de activación.




output
0 -1 0
-1 5 -1
0 -1 0
7 6 5 5 6 7
6 4 3 3 4 6
5 3 2 2 3 5
5 3 2 2 3 5
6 4 3 3 4 6
7 6 5 5 6 7
input

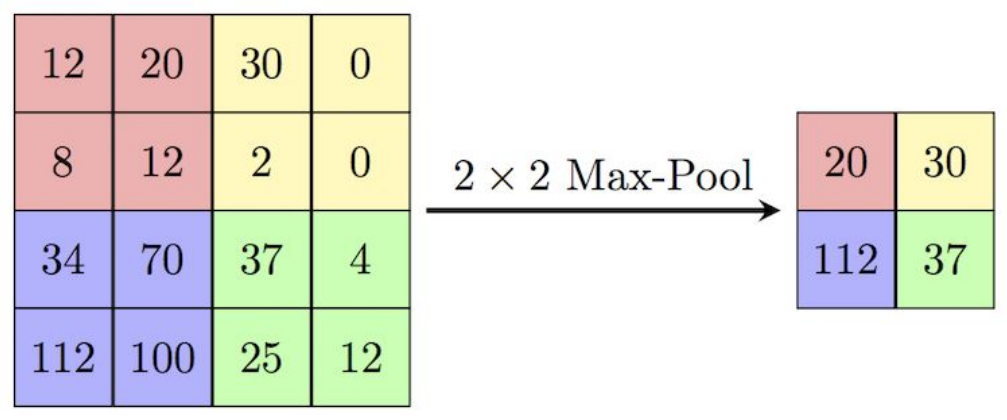


Esto ya lo entendemos

Nos falta esto

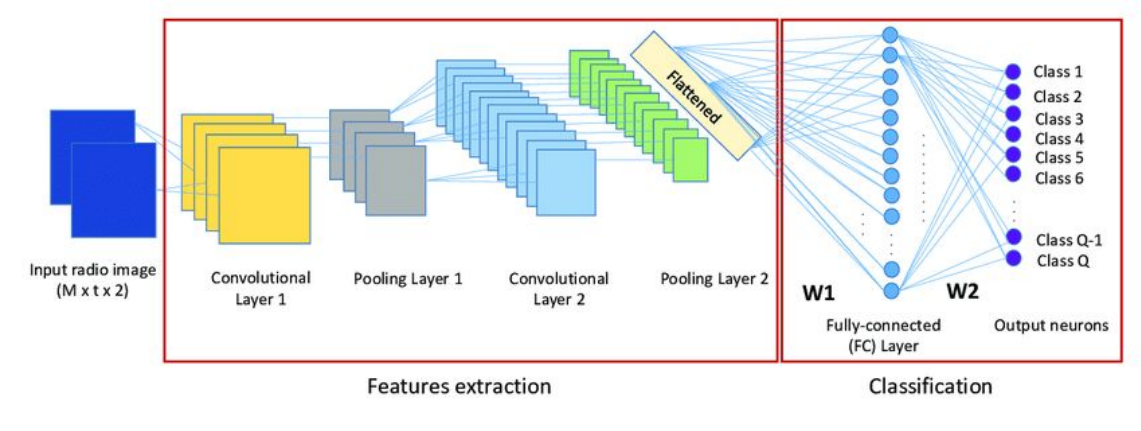
Pooling

- Resumen la información de atributos, eliminando parámetros de aprendizaje en las capas siguientes.
- Útil para disminuir los costos computacionales y el sobreajuste.
- Max pooling es la estrategia más utilizada, consiste en seleccionar el máximo valor de la sección sobre la cual pasa el filtro, llevándolo a una nueva capa de dimensión reducida respecto al activation map.



Fully Connected Layer

- Los atributos filtrados son transformados en un vector unidimensional a partir de una *flatten layer*.
- Este vector es entregada a una capa de redes neuronales completamente conectadas (MLP).
- Tiene el fin de interpretar los outputs como probabilidades de clase, e introducir funciones no lineales en la red.



Para una visualización interesante de este modelo (y más información):

https://poloclub.github.io/cnn-explainer/

Entrenamiento

Layer activation functions

Usage of activations

Activations can either be used through an `Activation` layer, or through the `activation` argument supported by all forward layers:

```
model.add(layers.Dense(64, activation=activations.relu))
```

This is equivalent to:

```
from keras import layers
from keras import activations

model.add(layers.Dense(64))
model.add(layers.Activation(activations.relu))
```

All built-in activations may also be passed via their string identifier:

```
model.add(layers.Dense(64, activation='relu'))
```

Available activations

`relu` function [source]

```
keras.activations.relu(x, negative_slope=0.0, max_value=None, threshold=0.0)
```