

# Aprendizaje Profundo

Facultad de Ingeniería  
Universidad de Buenos Aires



Profesores:

Marcos Maillot  
Antonio Zarauz  
Gerardo Vilcamiza

# Redes Neuronales Convolucionales

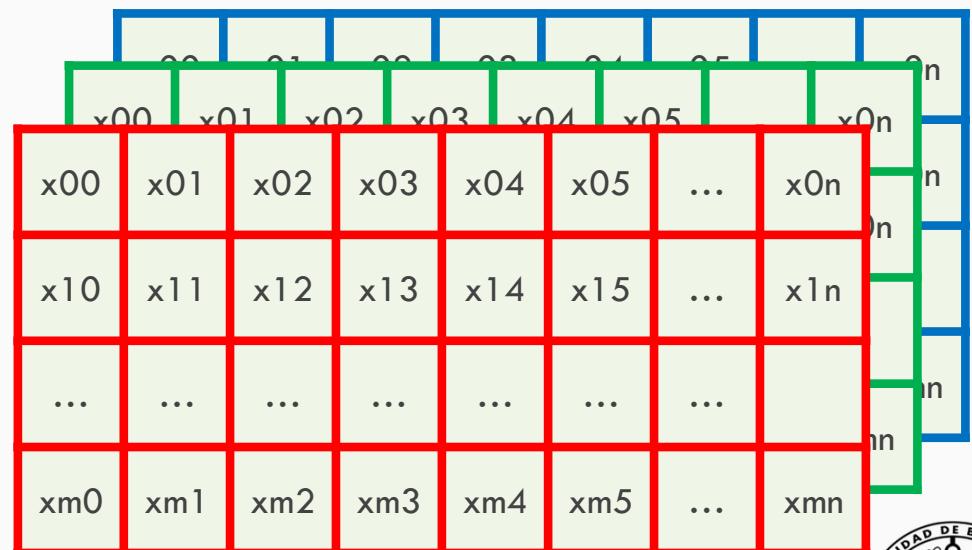
## Convolutional Neural Network (CNN)

- . Introducción
- . Kernel, límites y padding
- . Arquitectura base de ejemplo
- . ¿Ventajas?
  - Sparse interaction
  - Parameter sharing
- . Capa Pooling
- . Resumen de capas CNN.
- . Práctica con capas CNN
- . Back propagation en CNN
- . Implementación y práctica
- . ¿Qué hay en los kernels?



# Redes Neuronales Convolucionales - Introducción

Arquitectura de red neuronal **favorita** para el trabajo con imágenes (o datos que tengan **relación espacial**)



Array de 2d [ $m+1 \times n+1$ ]

Ejemplo: imagen de  $m+1 \times n+1$  pixeles  
con 3 canales (RGB)

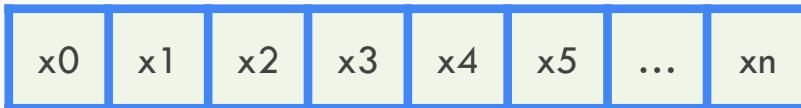


# Redes Neuronales Convolucionales - Introducción

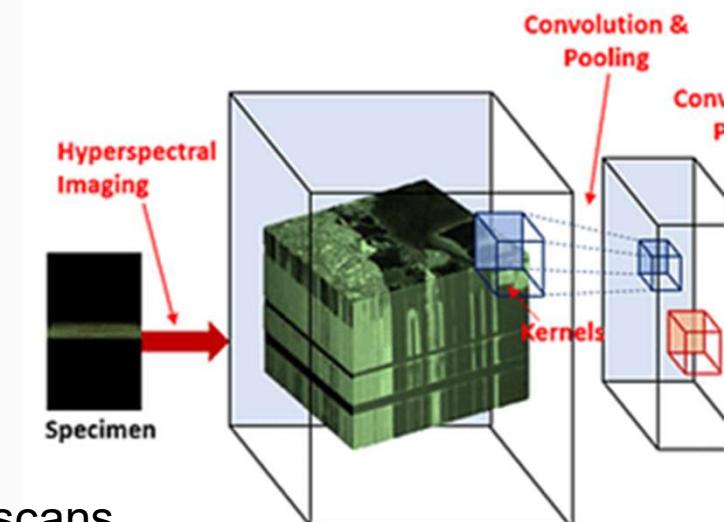
... o datos que tengan **relación espacial**...

Array de 1d [1 x n+1]

Ejemplo: señal temporal muestreada



<https://medium.com/@saba99/3d-cnn-4ccfab119cc2>

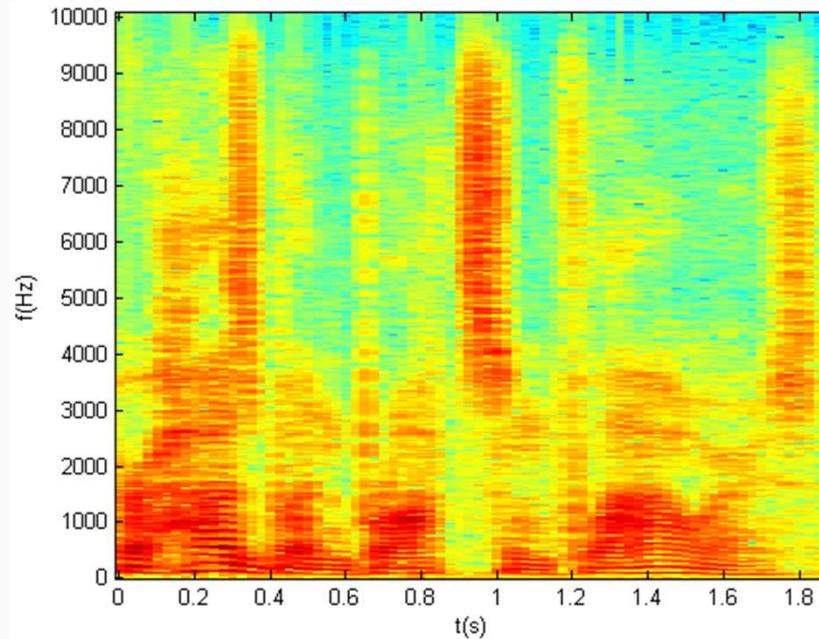


“...medical volumetric images (e.g., CT scans, MRI scans) or video sequences [...] to capture spatial and temporal dependencies”



# Redes Neuronales Convolucionales - Introducción

Otros ejemplos de array 2d: espectrograma



**DEBE EXISTIR INFORMACIÓN CON RELACIÓN  
ESPACIAL EN LAS DIMENSIONES DEL ARRAY**



# Redes Neuronales Convolucionales - Introducción

Su nombre “CONVOLUCIONALES” provienen de que en su interior realizan operaciones de convolución.

## Operación en 1D

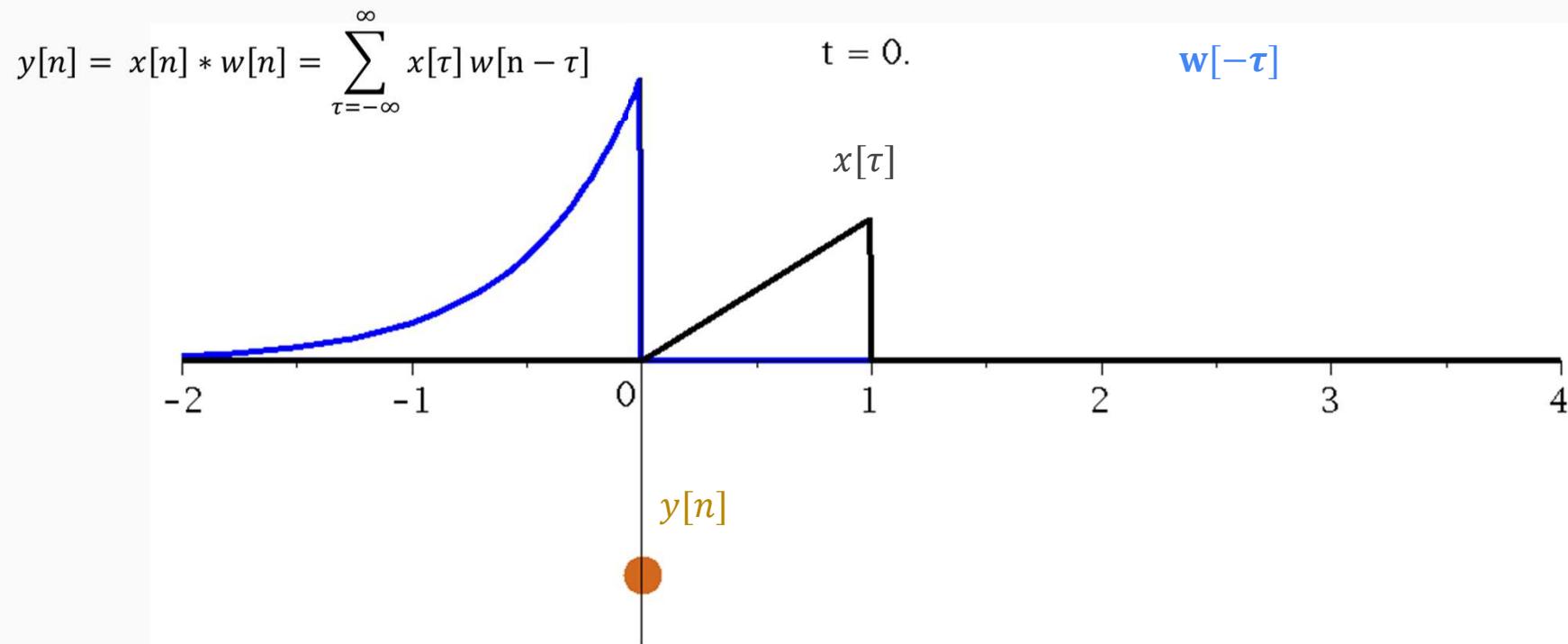
- Conv. continua  $y(t) = x(t) * w(t) = \int_{-\infty}^{\infty} x(\tau)w(t - \tau) d\tau$

- Conv. discreta  $y[n] = x[n] * w[n] = \sum_{\tau=-\infty}^{\infty} x[\tau] w[n - \tau]$



# Redes Neuronales Convolucionales - Introducción

## Convolución (ejemplo gráfico 1d)



<http://amber.feld.cvut.cz/vyu/eo2/english/lectures.htm>

**Conv. está ampliamente usado en el análisis de señales y sistemas**



# Redes Neuronales Convolucionales - Introducción

## CONVOLUCION 2d – INPUT – KERNEL – OUTPUT - LIMITES

$$S[i, j] = I[i, j] * K[i, j] = \sum_m \sum_n I[m, n]K[i - m, j - n]$$

PROP. CONMUTATIVA

$$S[i, j] = K[i, j] * I[i, j] = \sum_m \sum_n K[m, n]I[i - m, j - n]$$

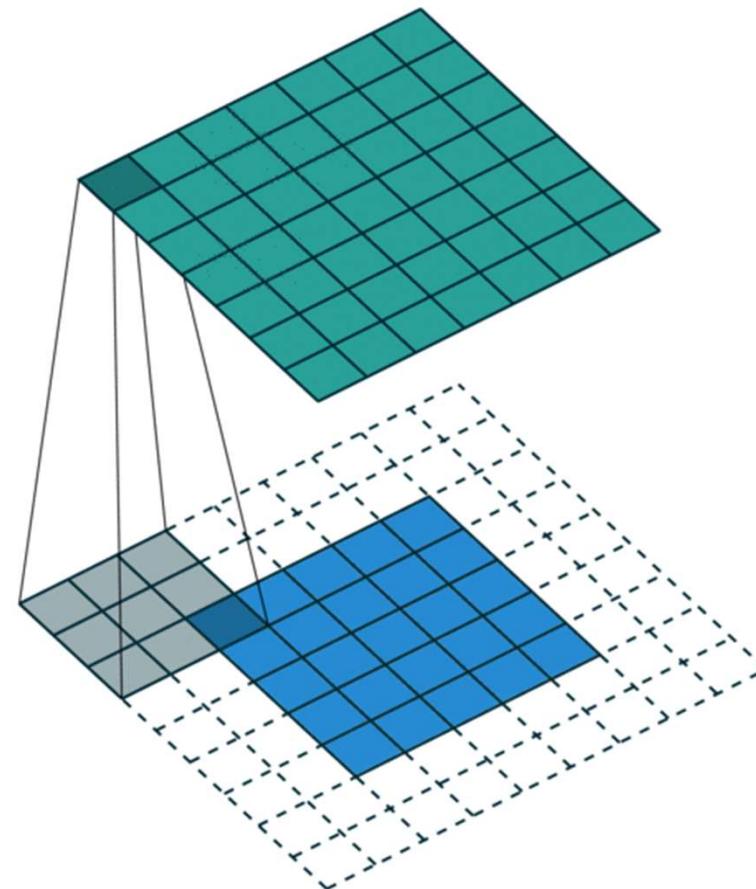
El kernel es entrenado por la red!  
(son sus parámetros o pesos sinápticos)



# Redes Neuronales Convolucionales - Introducción

Convolución (ejemplo gráfico 2d)

El kernel suele tener dimensión impar para definir bien su centro



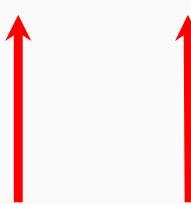
A guide to convolution arithmetic for deep learning

<https://arxiv.org/abs/1603.07285>



# Redes Neuronales Convolucionales - Introducción

FLIPPED CONVOLUTION or CROSS-CORRELATION

$$S[i, j] = K[i, j] \otimes I[i, j] = \sum_m \sum_n I[i + m, j + n] K[m, n]$$




# Redes Neuronales Convolucionales - Introducción

## Convolución y número de canales

Se aplica 1 kernel x canal y se suma todo.

Se suelen emplear varios kernels por capa, para poder “aprender” varias cosas.

Cada kernel podrá “aprender” un feature, que expresará en su salida si lo detecta.

<https://medium.com/apache-mxnet/multi-channel-convolutions-explained-with-ms-excel-9bbf8eb77108>

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
1																									
2	Input					Kernel			Intermediate Output																
3																									
4	1	0	1	0	2																				
5	1	1	3	2	1																				
6	1	1	0	1	1																				
7	2	3	2	1	3																				
8	0	2	0	1	0																				
9																									
10	1	0	0	1	0																				
11	2	0	1	2	0																				
12	3	1	1	3	0																				
13	0	3	0	3	2																				
14	1	0	3	2	1																				
15																									
16	2	0	1	2	1																				
17	3	3	1	3	2																				
18	2	1	1	1	0																				
19	3	1	3	2	0																				
20	1	1	2	1	1																				
21																									

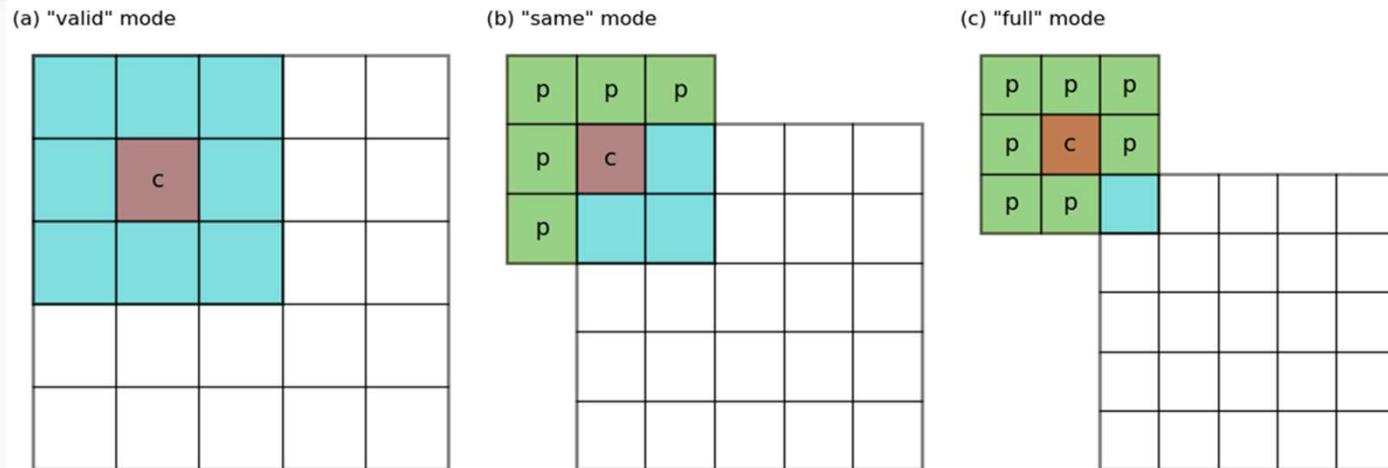
PARAMETROS



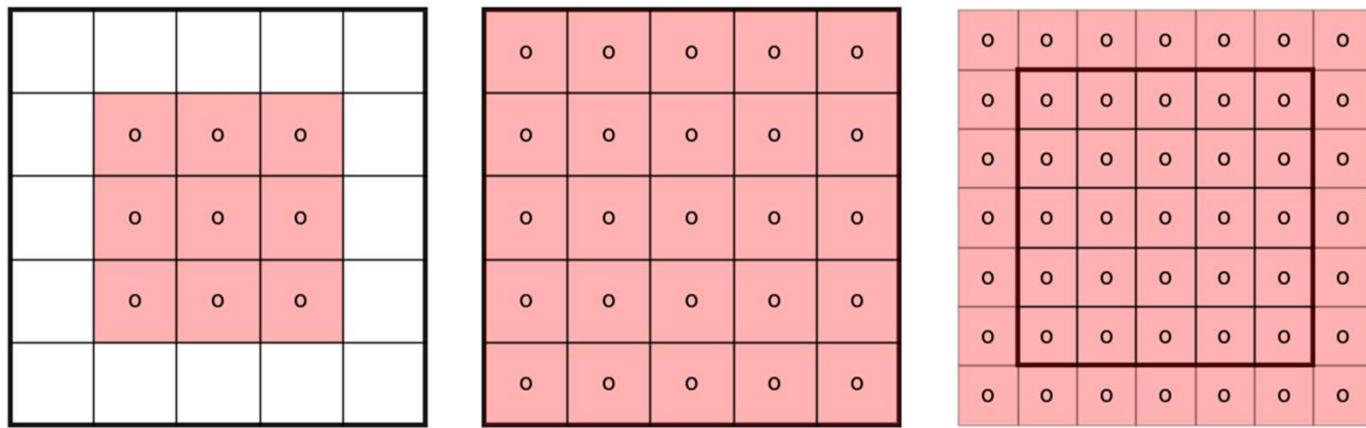
SALIDA

# Redes Neuronales Convolucionales - Bordes (límites) y padding

padding:  
'valid',  
'same' or  
tuple



padding\_mode:  
'zeros',  
'reflect',  
'replicate' or  
'circular'



## Redes Neuronales Convolucionales - Bordes (límites) y padding

**valid** → no se agregan ceros a I, resultado reducido tal cual se vio.

$$S_{\text{valid}}[i - m + 1, j - n + 1]$$

**same** → se agregan ceros a I para que S sea del mismo tamaño que I.

$$S_{\text{same}}[i, j]$$

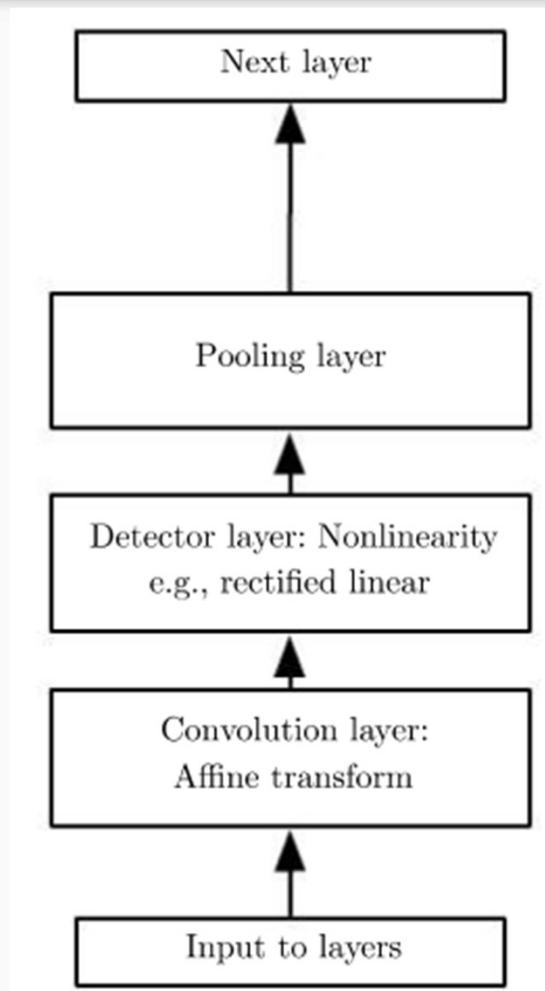
**full** → se agregan ceros a I para que el K entre completo.

$$S_{\text{full}}[i + m - 1, j + n - 1]$$



# Redes Neuronales Convolucionales - Arquitectura base de ejemplo

Arquitectura típica



# Redes Neuronales Convolucionales - ¿Ventajas? Sparse interaction

**Entonces...**

**¿Dónde está la ventaja de usar CNN?**

sparse interaction / sparse connectivity o sparse weight  
sparse → escaso

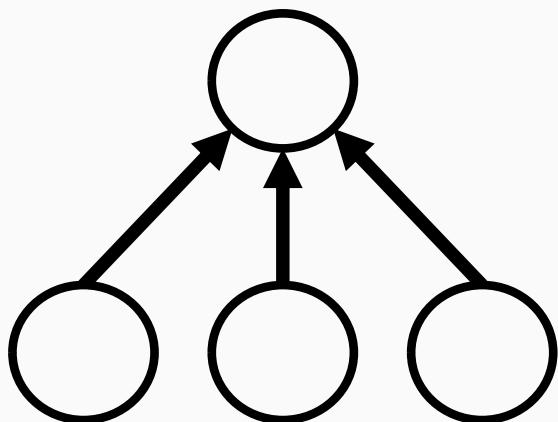
- El KERNEL es de dimensiones muy inferiores al tamaño de la imagen (INPUT) a procesar.
- El uso de un KERNEL permite tener interacciones locales de entradas con sus respectivas salidas.
- Las dimensiones del KERNEL indicarán cuanta “vecindad” analizan en cada posición donde sea evaluado.



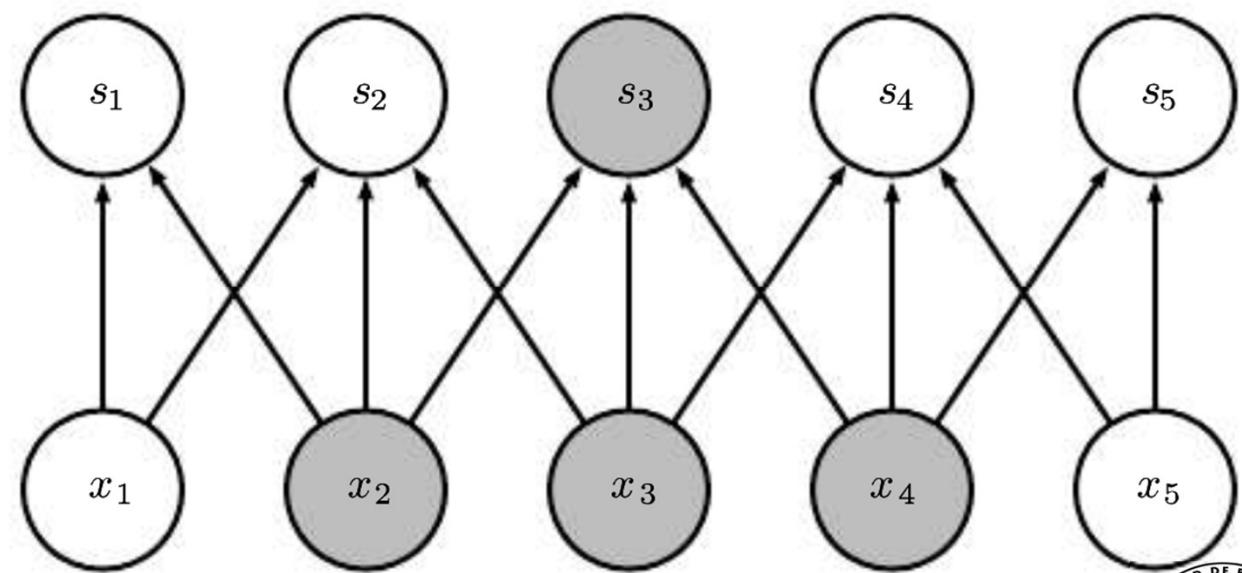
# Redes Neuronales Convolucionales - ¿Ventajas? Sparse interaction

**sparse interaction**

**K[i] - kernel**



**S[i] - output**



**x[i] - input**



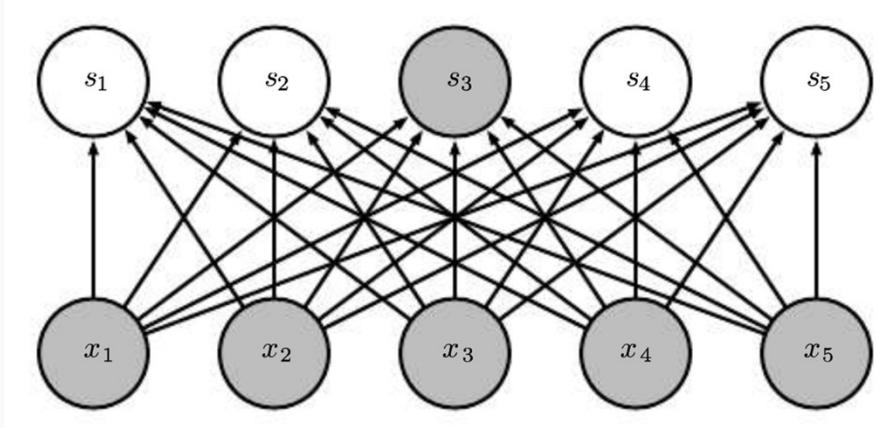
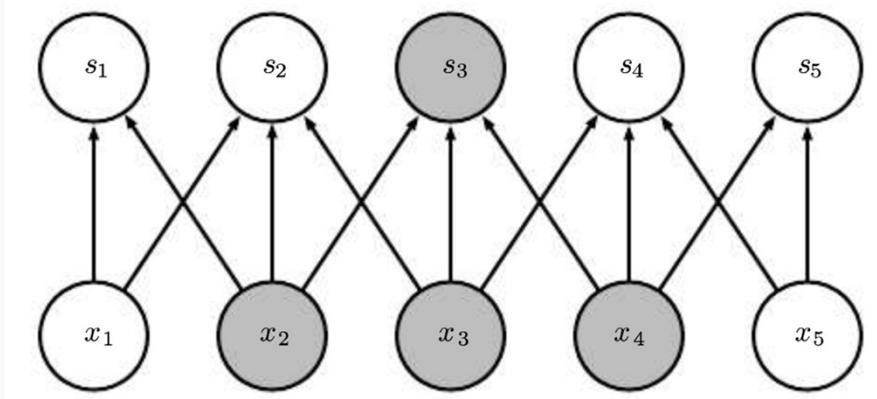
# Redes Neuronales Convolucionales - ¿Ventajas? Sparse interaction

**sparse interaction**

**vs**

**fully connected layer**

**Visto desde la salida**



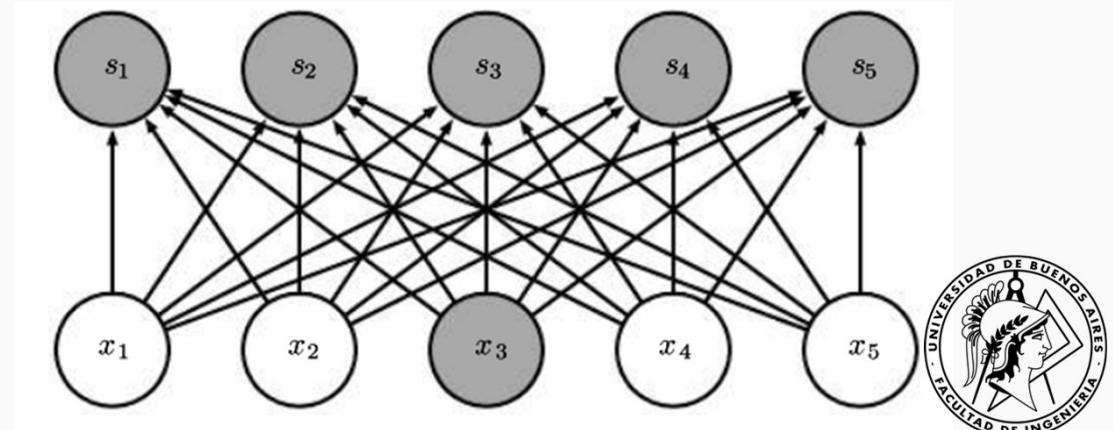
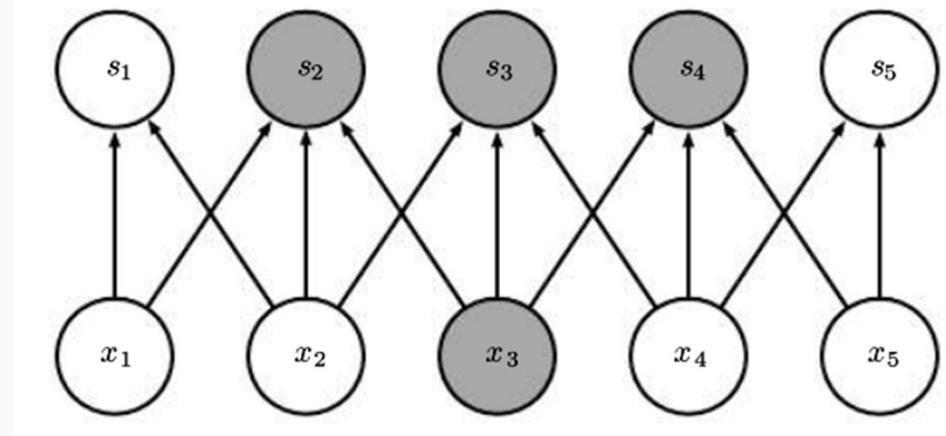
# Redes Neuronales Convolucionales - ¿Ventajas? Sparse interaction

**sparse interaction**

**vs**

**fully connected layer**

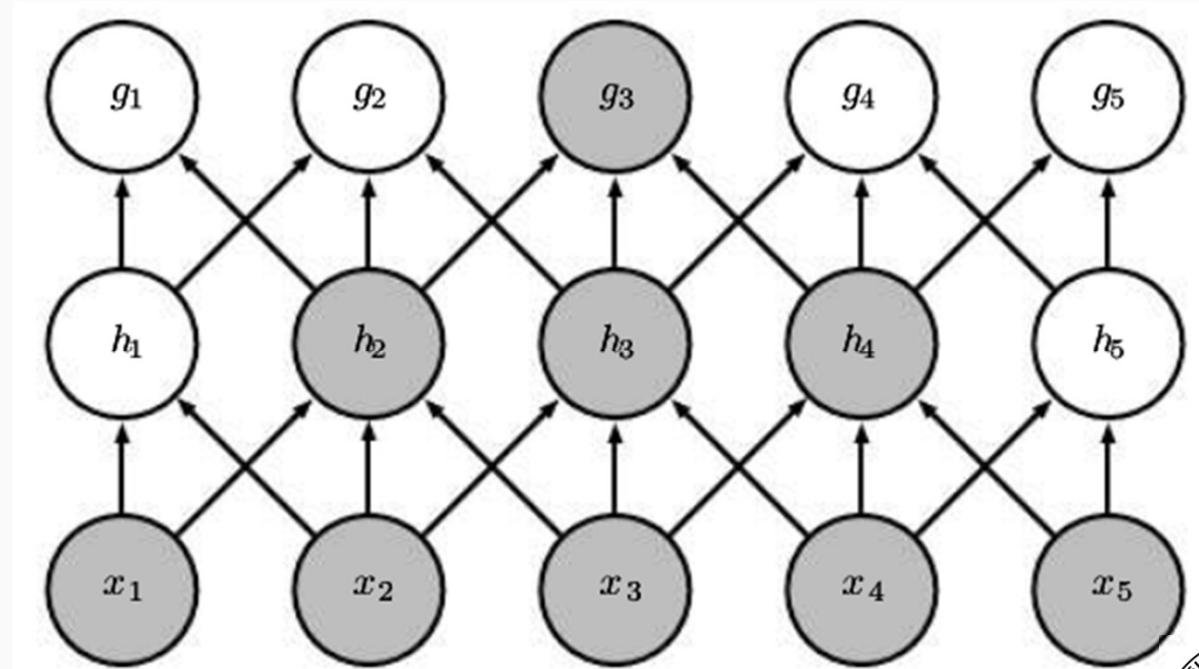
**Visto desde la entrada**



# Redes Neuronales Convolucionales - ¿Ventajas? Sparse interaction

sparse interaction

in deep CNN



# Redes Neuronales Convolucionales - ¿Ventajas? Parameter sharing

## parameter sharing

- El uso de un mismo KERNEL para una capa dada, implica que los parámetros (pesos sinápticos  $W_{ij}$ ) se comparten.
- Esto permite optimizar dicho KERNEL para que sea capaz de detectar **algo específico** (un borde por ejemplo).
- De esta manera, el KERNEL entrenado, será capaz de detectar **algo específico** en cualquier lugar donde se aplique el KERNEL en la foto de entrada (INPUT).

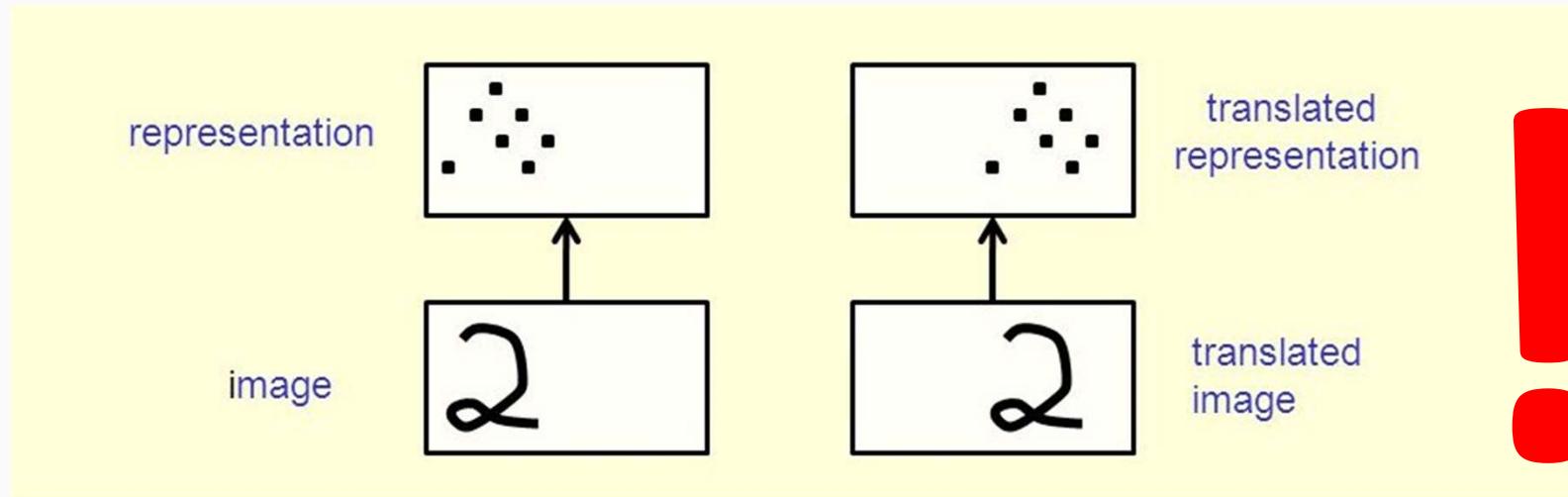


# Redes Neuronales Convolucionales - ¿Ventajas? Parameter sharing

## parameter sharing

*“...detectar **algo específico** en cualquier lugar donde se aplique el KERNEL ...”*

- Esa propiedad se llama **equivariancia a la traslación (translation equivariance)** y es natural de la convolución.



# Redes Neuronales Convolucionales - ¿Ventajas? Parameter sharing

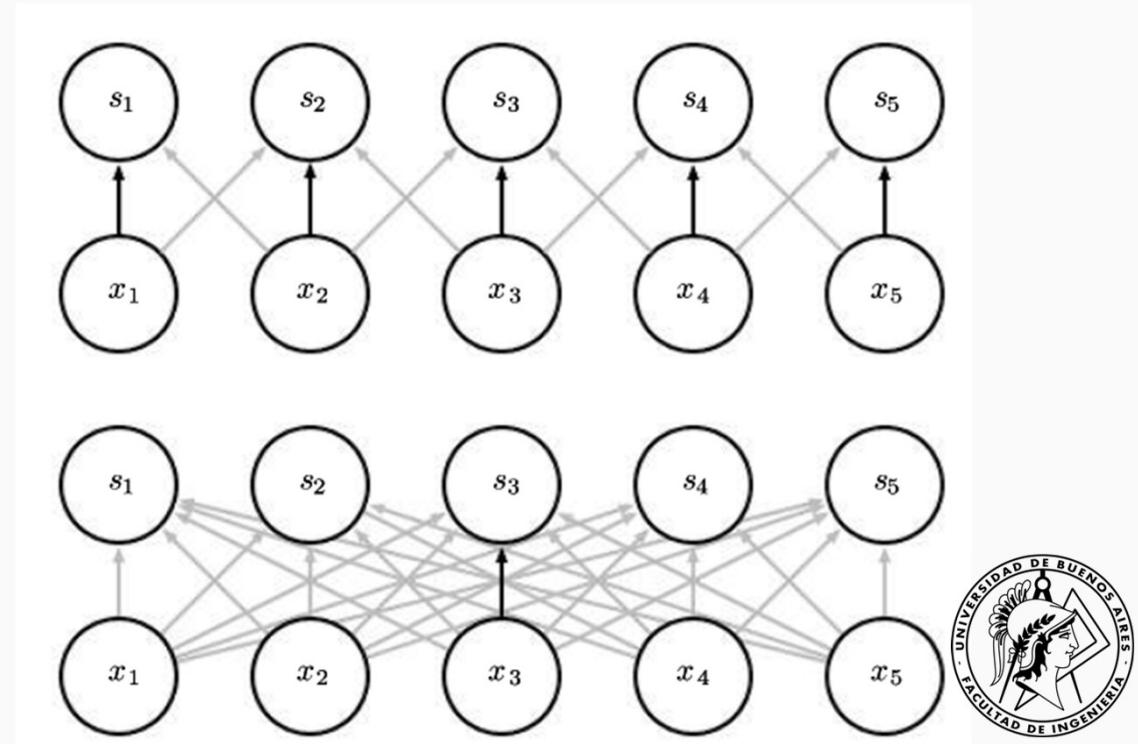
## parameter sharing

Compartir parámetros permite utilizar más de 1 vez un mismo pámetro...

Conv layer

vs

fully connected layer



# Redes Neuronales Convolucionales - ¿Ventajas?

**sparse interaction + parameter sharing**

- Hay una reducción drástica de los parámetros a entrenar y a almacenar.
- Supongamos una capa de  $m$  entradas y  $n$  salidas...

**Conv layer con KERNEL  
de tamaño  $k$**

$$O(k \times n)$$

(cada kernel tendrá 1 salida)

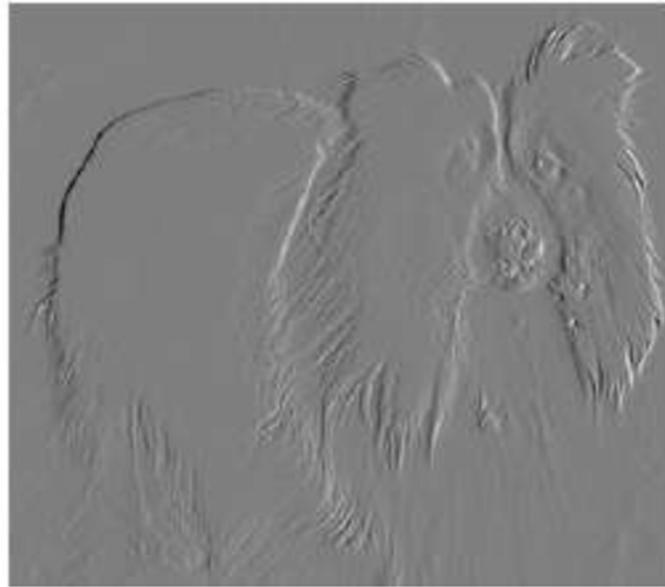
**fully connected layer**

$$O(m \times n)$$



## Redes Neuronales Convolucionales - ¿Ventajas?

**sparse interaction + parameter sharing (edge detecting)**



**Conv layer.**

**Aprox 267960 float op.**

**fully connected layer**

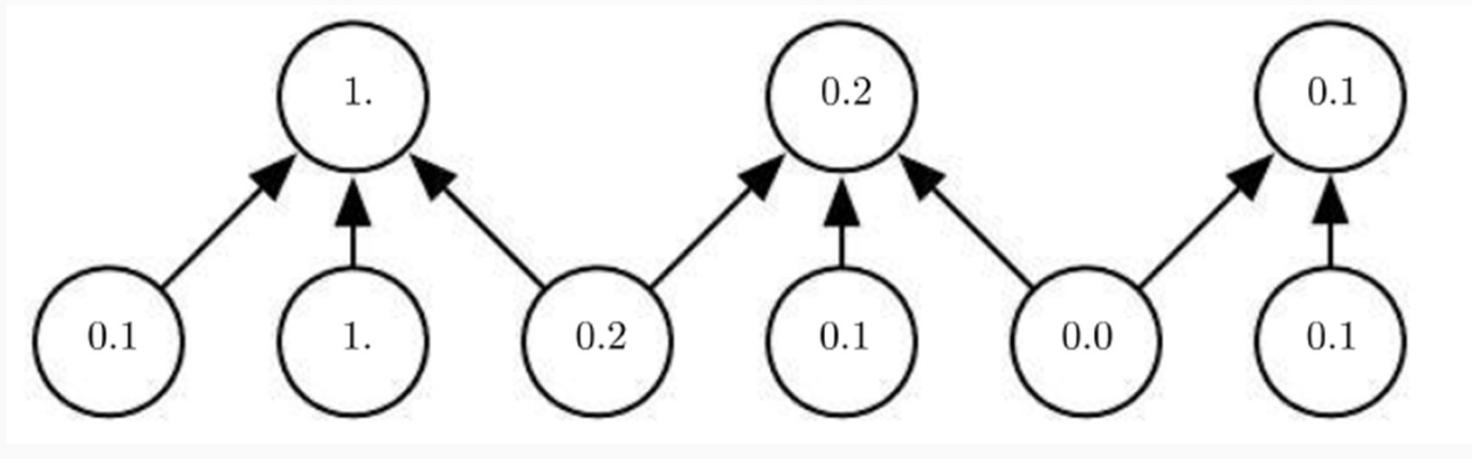
**Aprox 16000000000 float op.**



# Redes Neuronales Convolucionales - Stride

## stride

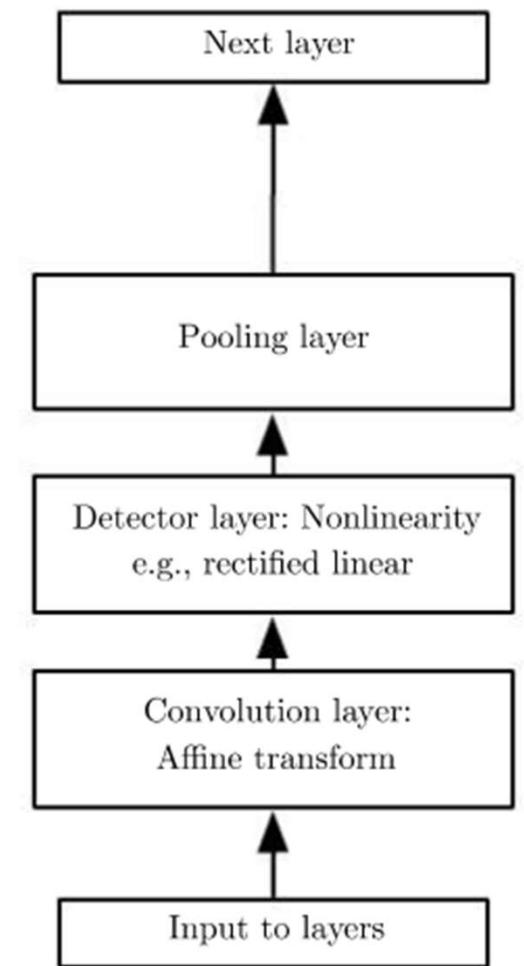
- Regula los pasos en que avanza la convolución.
- Es como hacer un downsampling de la salida.



# Redes Neuronales Convolucionales - Capa de activación

## CAPA DE ACTIVACIÓN/DETECCIÓN

- Es una capa NO LINEAL como las ya conocidas.
- Relación entrada/salida 1/1.
- No se optimizan pesos sinápticos.



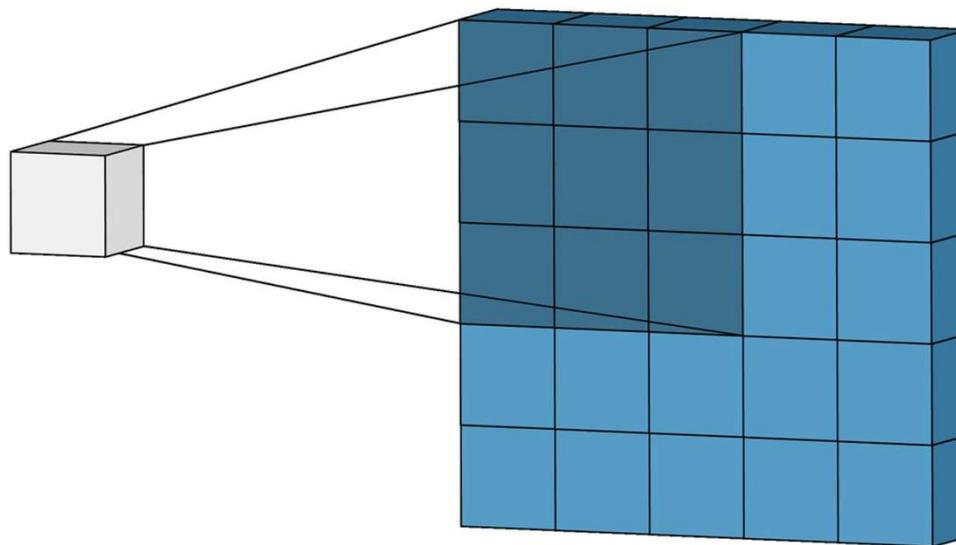
# Redes Neuronales Convolucionales - Capa de pooling

(POOLING = GROUPING OF ASSETS)

- Es similar a la CONV (tiene size y stride), pero aplica un operador matemático específico en lugar de un KERNEL.
- Se busca una reducción de dimensiones de la entrada con operadores que permitan detectar features.
- Como salida, da un valor característico de todos los features de entrada de la entrada.
- **No se optimizan pesos sinápticos** (solo tiene hiper-parámetros)



# Redes Neuronales Convolucionales - Capa de pooling



<https://iq.opengenus.org/pooling-layers/>



# Redes Neuronales Convolucionales - Capa de pooling

*“...operadores que permitan detectar features.”*

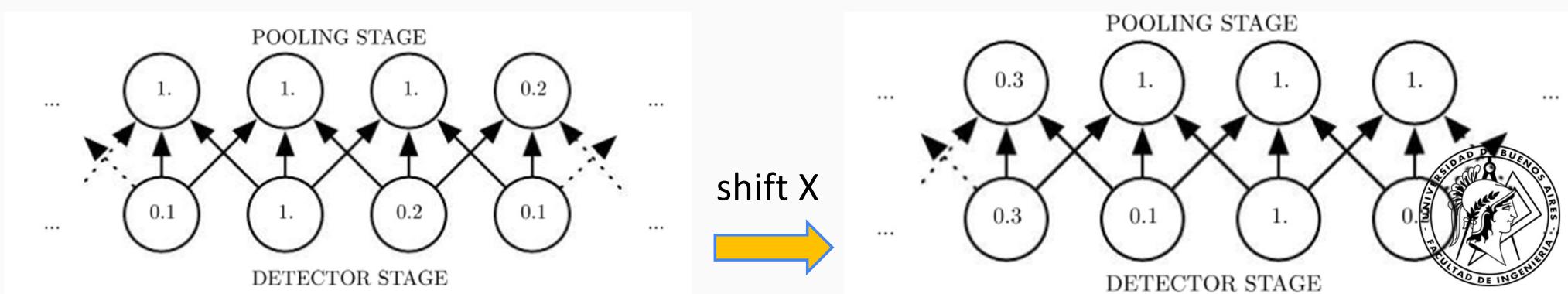
- **Max pooling** → de todos, dame el máximo (¿está el feature acá?)
- **Average pooling** → de todos, dame el promedio (¿qué prob de encontrar el feature tengo?)
- **Power Average pooling** → norma  $p$  de un vector.  $f(X) = \sqrt[p]{\sum x_i^p}$
- **Adaptive pooling**
- **Global, fractional**



# Redes Neuronales Convolucionales - Capa de pooling

## ¿Qué ventaja gano haciendo pooling?

- Reducción de dimensión (resumen estadístico de la entrada)
- Translation invariant → invariancia al desplazamiento.
- Permite lograr una detección que no depende de la posición (es invariante a la posición de dicho feature)



# Redes Neuronales Convolucionales - Capa de pooling

## Translational invariance

### Translation Invariance



<https://kamathhrishi.github.io/MyWebsite/jekyll/update/2022/06/10/modelvsdataml.html>



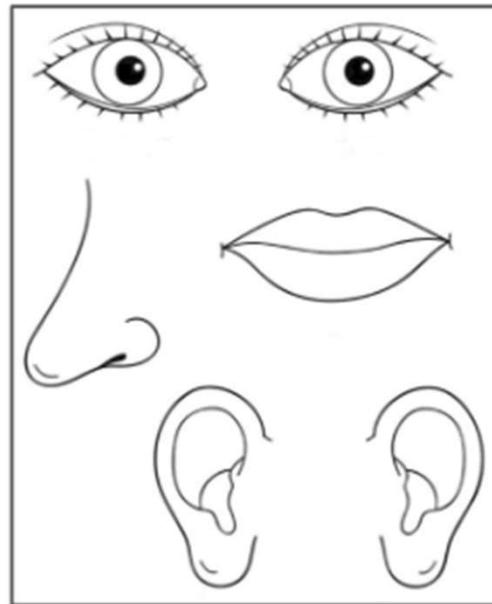
# Redes Neuronales Convolucionales - Capa de pooling

## Translational invariance.... ¿?

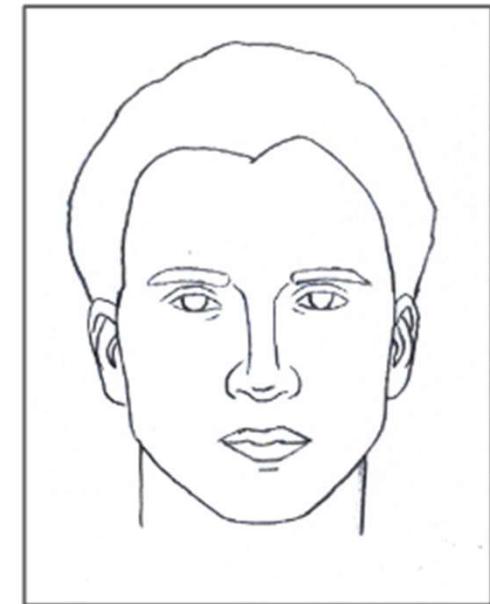
Una cara tiene:

- 2 ojos
- 1 nariz
- 1 boca
- 2 orejas

... ¿correcto?



Not Face



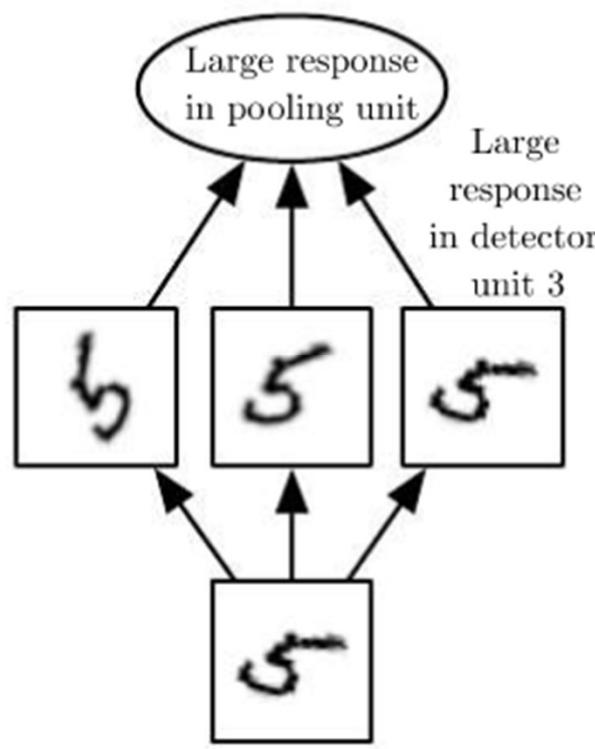
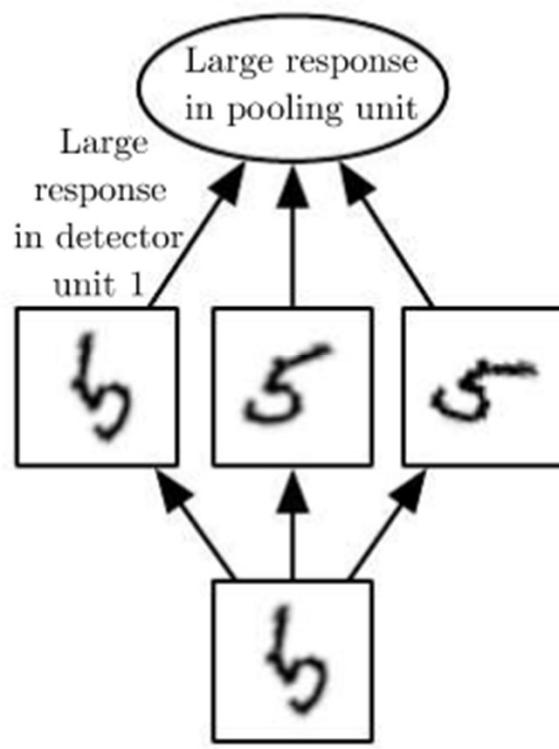
Face

<https://divsoni2012.medium.com/translation-invariance-in-convolutional-neural-networks-61d9b6fa03df>



# Redes Neuronales Convolucionales - Capa de pooling

**rotational invariance**



**Polling over channels!**

**Libreria -> einops**

<https://stackoverflow.com/questions/46562612/pytorch-maxpooling-over-channels-dimension>



## Redes Neuronales Convolucionales - Capa de pooling

**rotational/translational/scale INvariance or EQUIvariance**

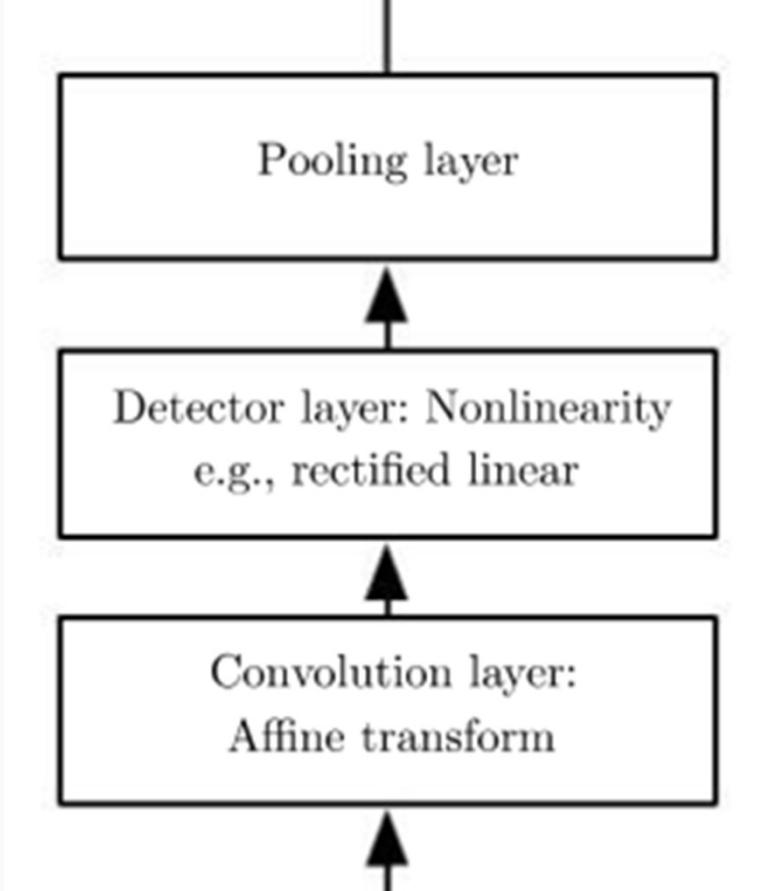
**LE ENSEÑÉ A MI TÍO A USAR EL PHOTOSHOP  
Y UNA SEMANA DESPUÉS ME MANDÓ ESTO...**



Lo que no Sabías



# Redes Neuronales Convolucionales - Resumen de capas CNN



Hiper parámetros (se eligen)	Parametros (se entrenan)
Pooling func Kernel size stride	ninguno
Function Function param	ninguno
Nro kernel (CH out) Kernel size Padding Stride	$\text{Nro kernel} * (\text{kernel size}^2 * \text{CH input} + 1)$



# Redes Neuronales Convolucionales - Práctica con capas CNN

**Unos minutos de descanso...**

**coffee / mate / pizza break**



**Luego... parte práctica 1**



# Redes Neuronales Convolucionales - Back-propagation en CNN

**Y ahora....  
...back-propagation...**



# Redes Neuronales Convolucionales - Implementación

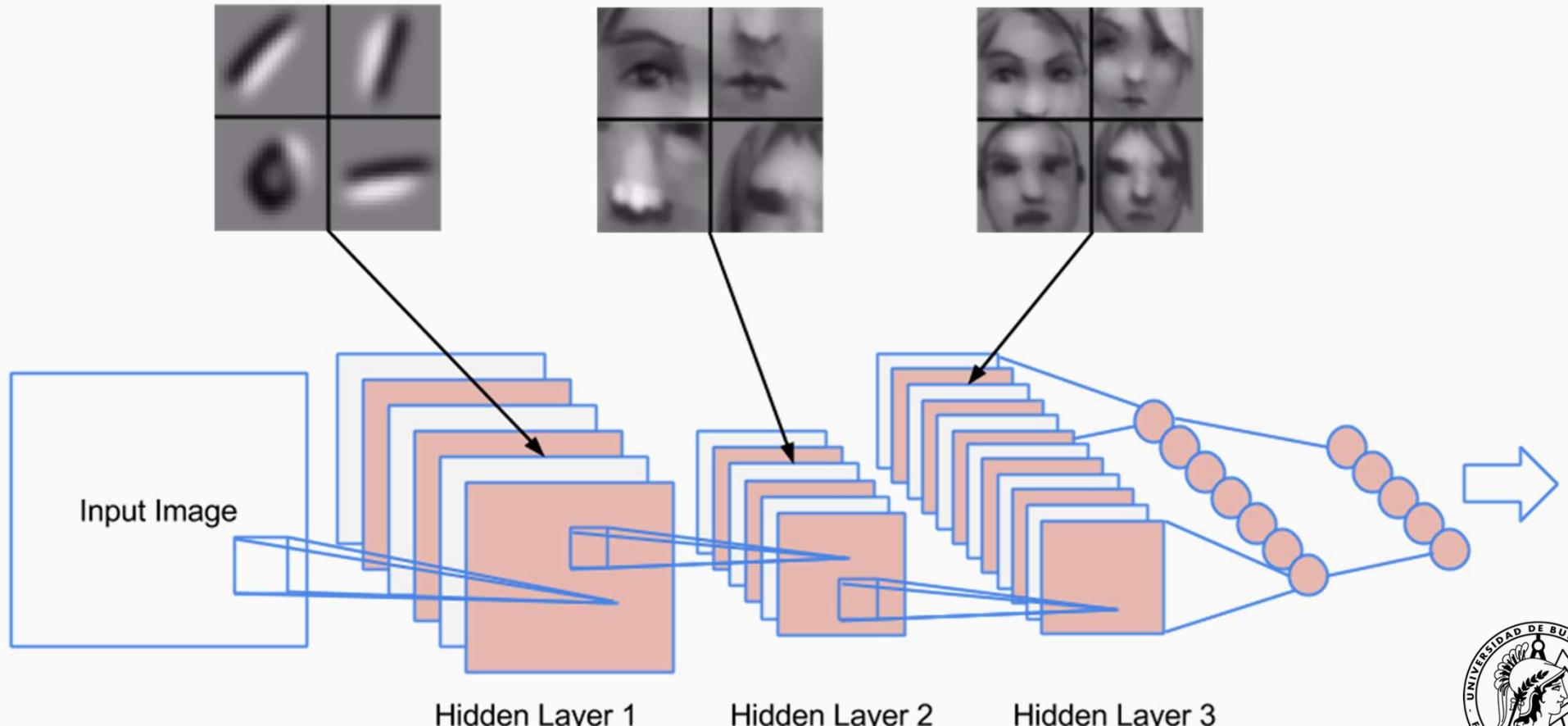
## Parte práctica 2

**Unos minutos de descanso...**

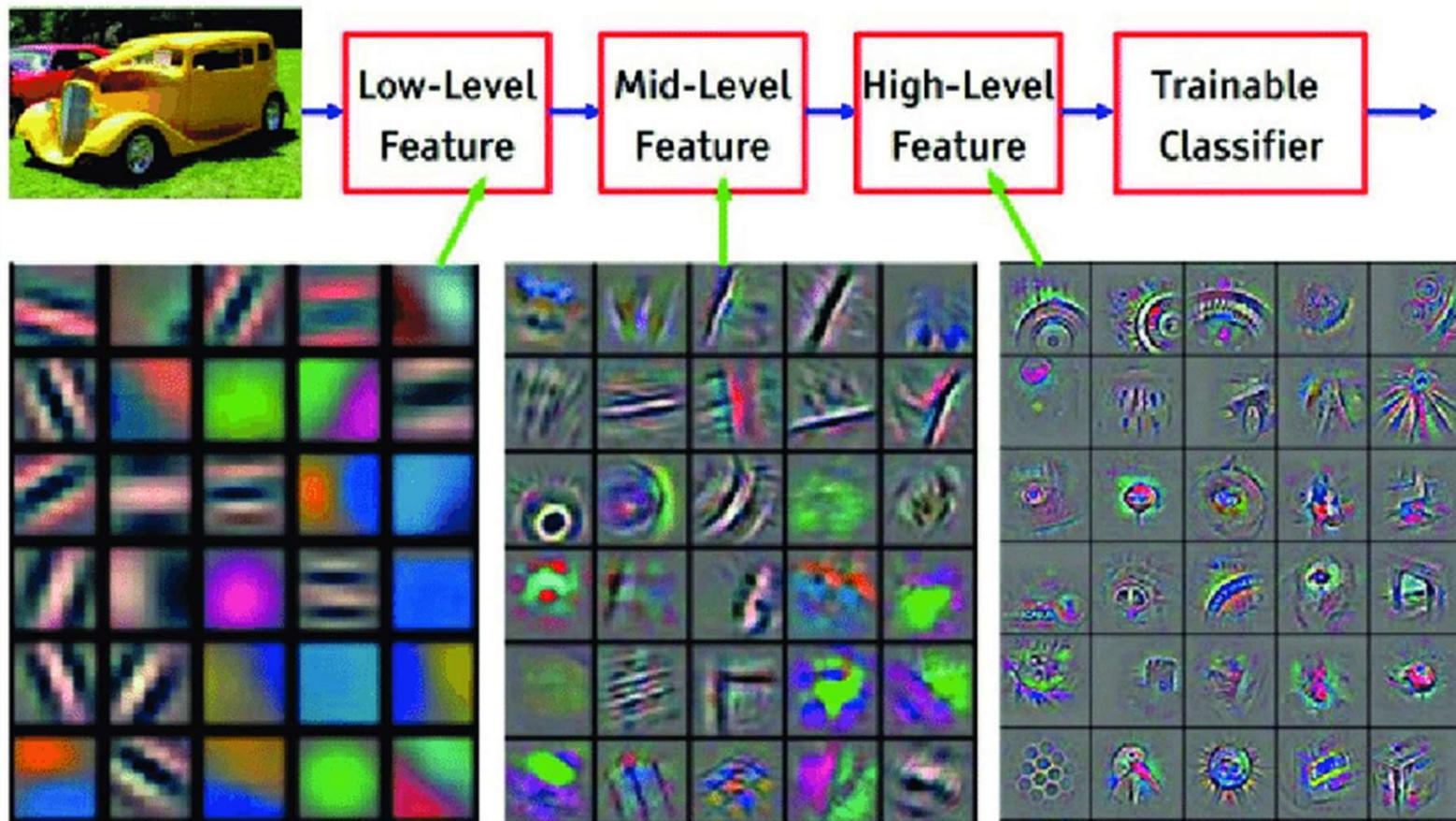
**coffee / mate / pizza break**



# Redes Neuronales Convolucionales - ¿que hay en los kernels?



# Redes Neuronales Convolucionales - ¿que hay en los kernels?



Example of features that the filters in a convolution layer look for at different levels in a network. The deeper into the network (higher level), the more complex the features are. Source: (F.-F. Li & Karpathy, 2015).

