

Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación



IIC2613 - Inteligencia Artificial

Redes Neuronales Convolucionales (CNN)

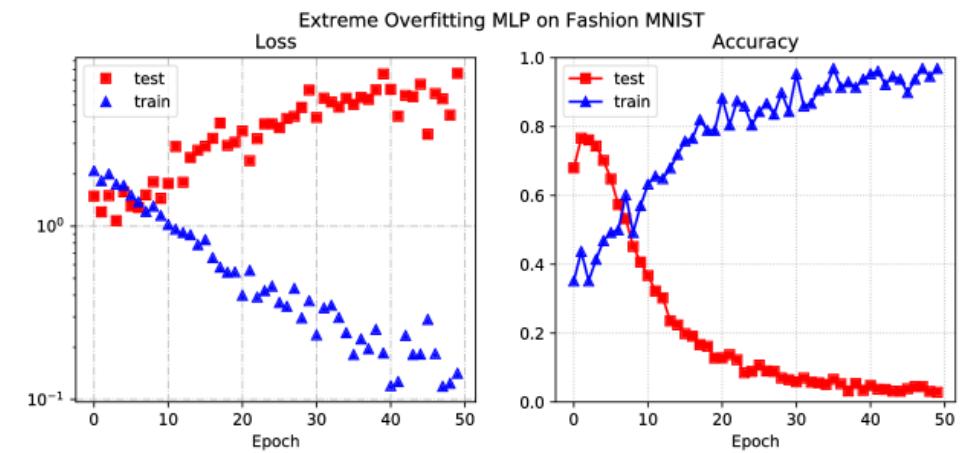
Hans Löbel
Dpto. Ingeniería de Transporte y Logística
Dpto. Ciencia de la Computación

CNNs buscan responder la siguiente pregunta: ¿necesitamos redes distintas a un MLP para datos visuales?

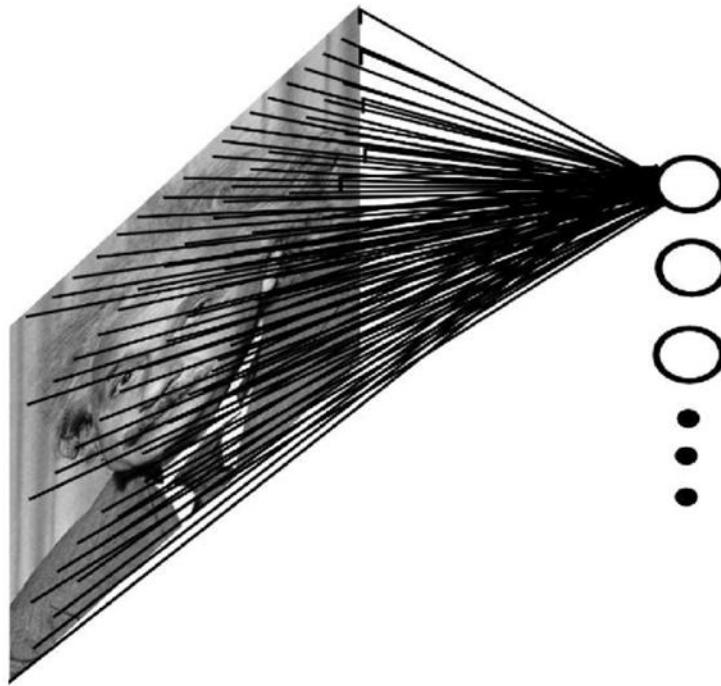
Tal como vimos anteriormente, los MLP fracasan catastróficamente en conjuntos de datos visuales más difíciles que MNIST.

¿Por qué pasa esto?

Label	Description	Examples
0	T-Shirt/Top	
1	Trouser	
2	Pullover	
3	Dress	
4	Coat	
5	Sandals	
6	Shirt	
7	Sneaker	
8	Bag	
9	Ankle boots	



Un primero culpable es la excesiva (ridícula) cantidad de parámetros



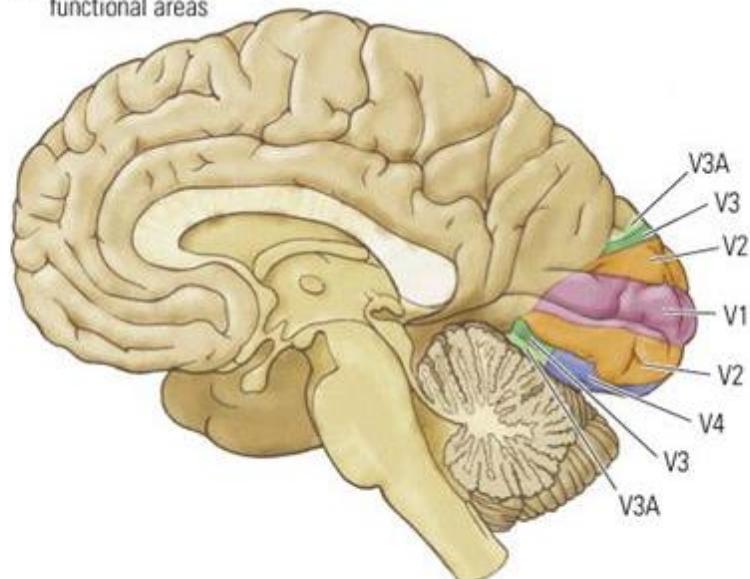
- Imagen de 256x256
- Capa *fully connected* de 256 neuronas
- 16.777.216 parámetros

Cada neurona ve todo simultáneamente, ¿es esto adecuado para las imágenes?

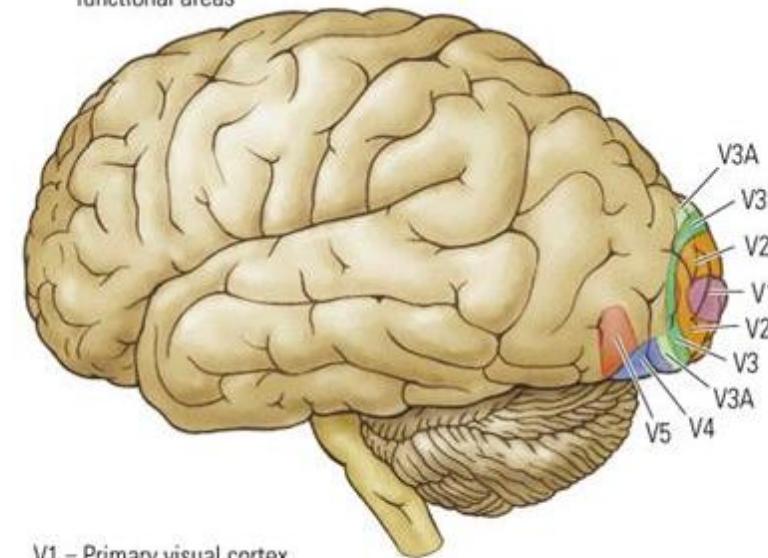
Veamos un poco qué pasa en la corteza visual de los mamíferos

Areas V1 – V5

(A) Medial view of functional areas

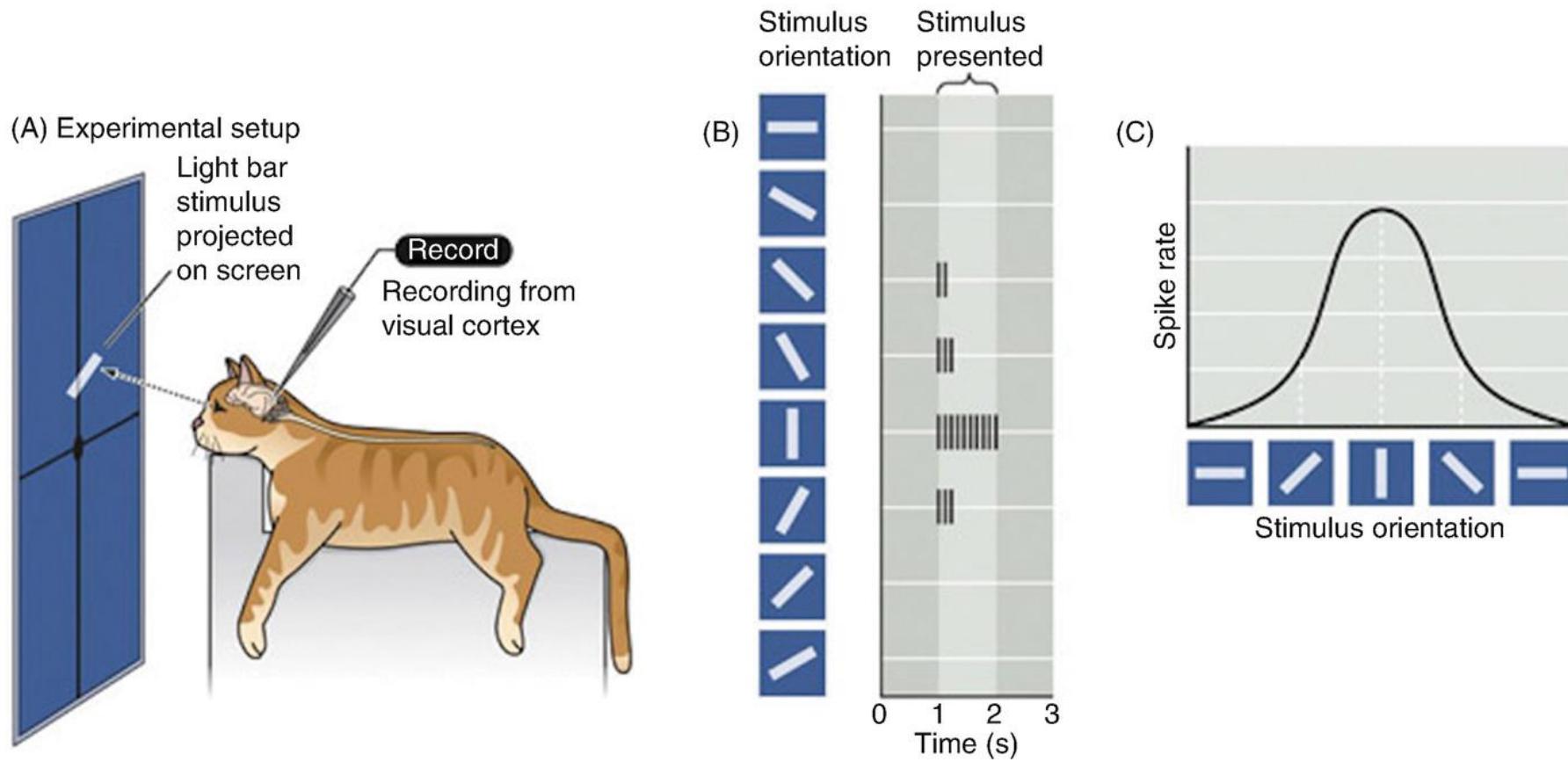


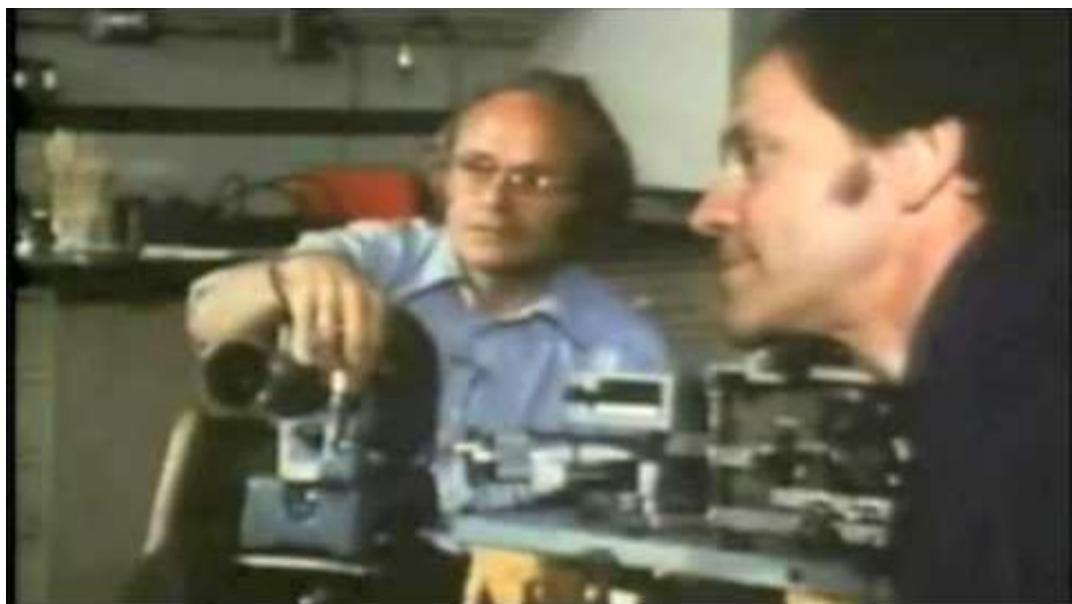
(B) Lateral view of functional areas



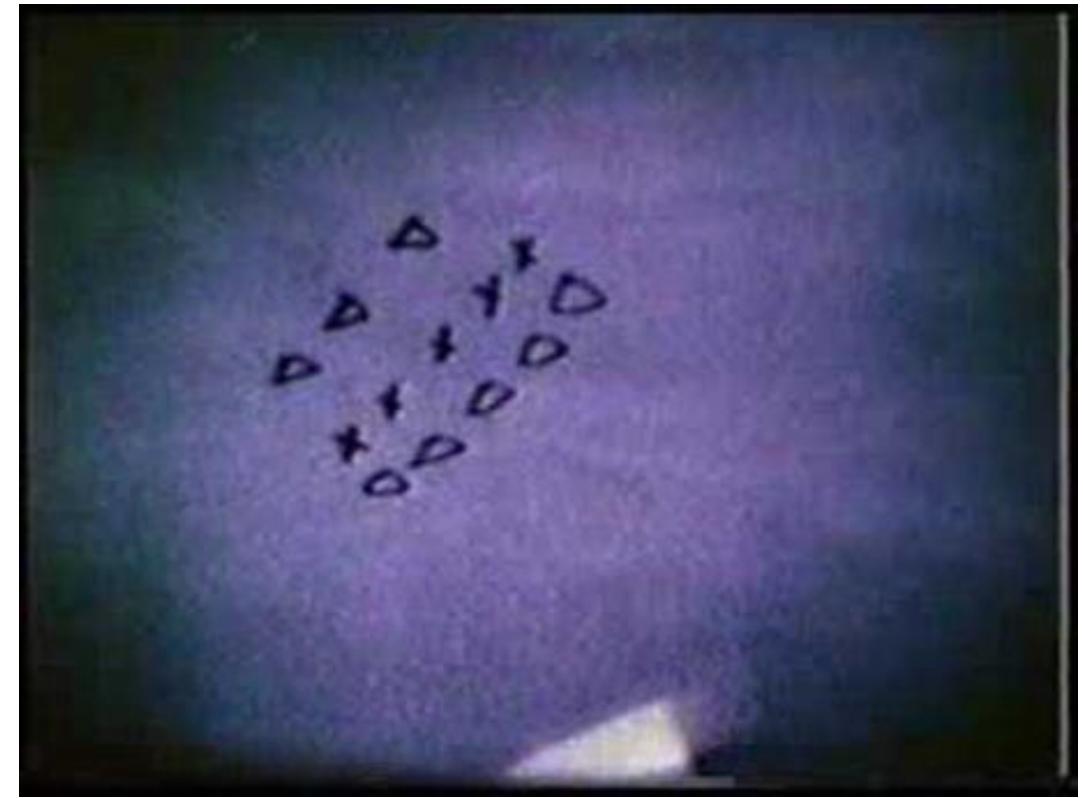
V1 = Primary visual cortex
V2–V5 = Extrastriate cortex

Patrones de estímulo son detectados por células (neuronas) específicas, altamente localizadas (Hubel y Wiesel, \approx 60s)



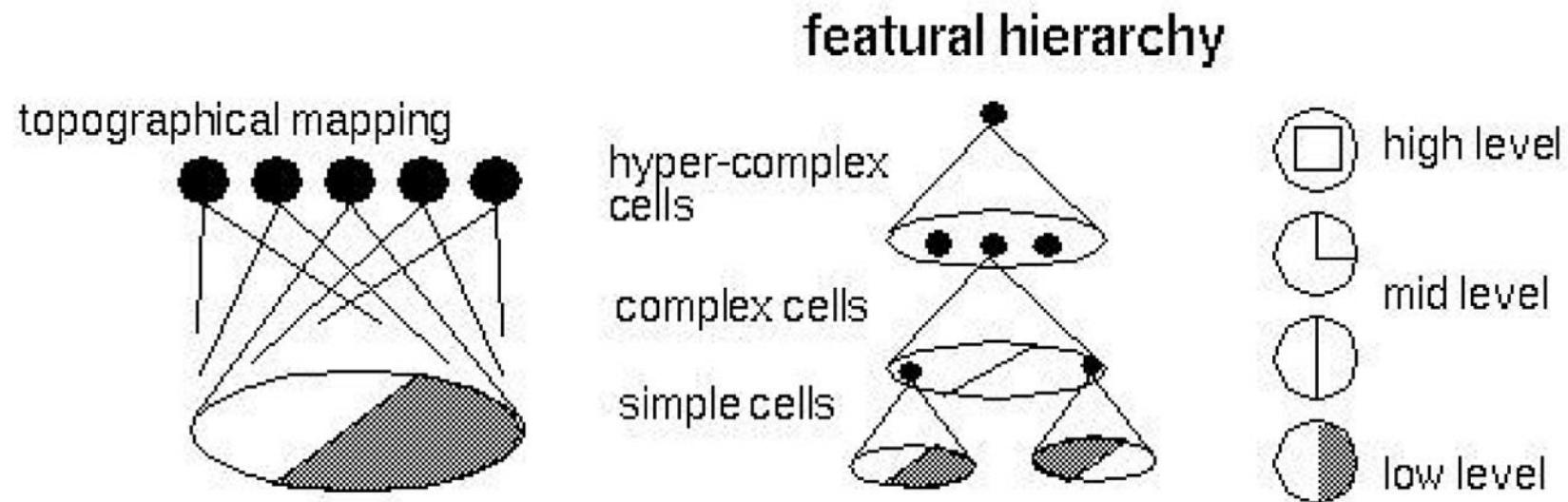


youtu.be/I0Hayh06LJ4

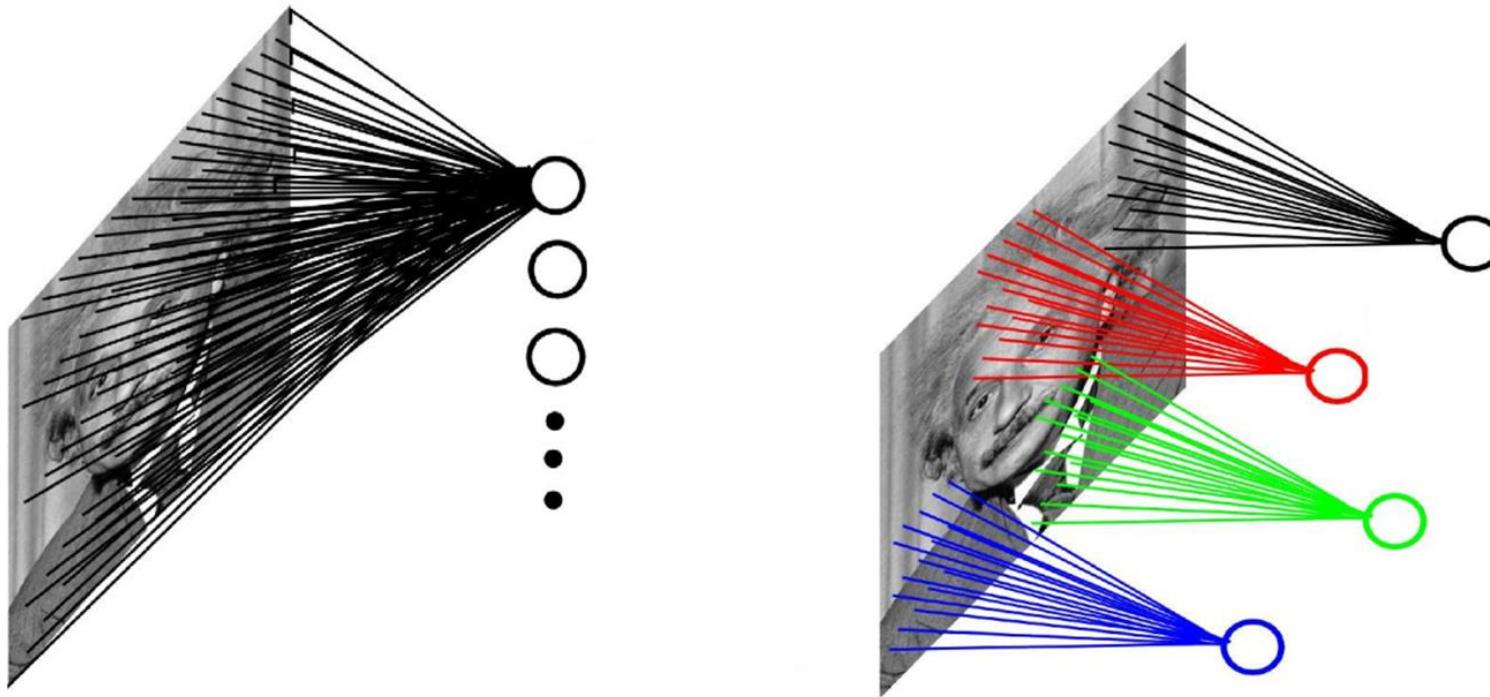


youtu.be/Cw5PKV9Rj3o

Neuronas se organizan de manera **jerárquica**, aumentando el nivel de **abstracción** de los patrones **locales** detectados



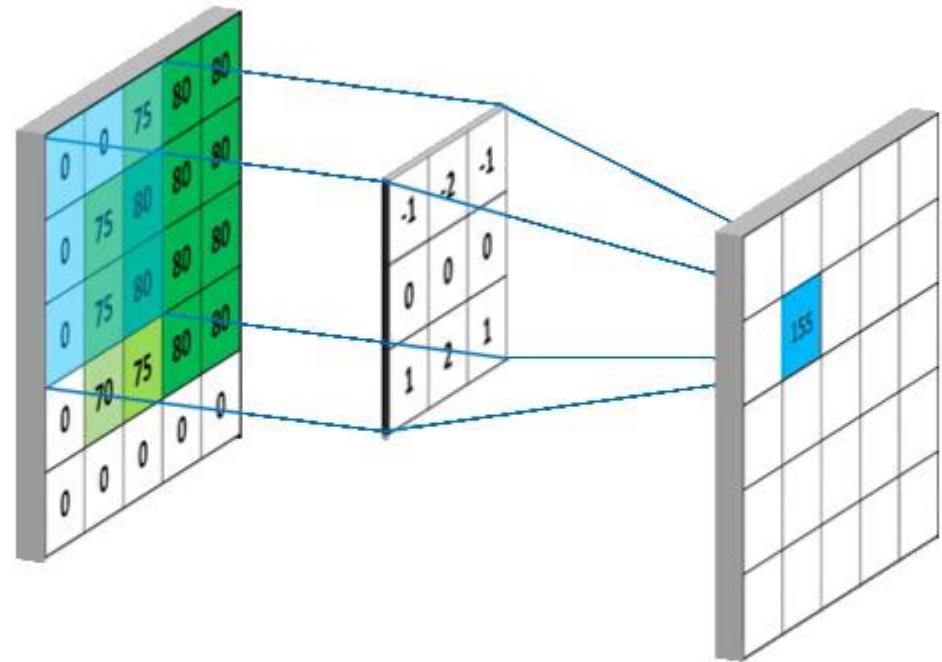
Podemos modelar este comportamiento con “neuronas deslizantes”



- Neuronas capaces de barrer la entrada proveen una reducción significativa en el número de parámetros (no depende de la conectividad)
- Además, dado su tamaño, permiten capturar patrones locales, pudiendo activarse en cualquier posición de la imagen de entrada.

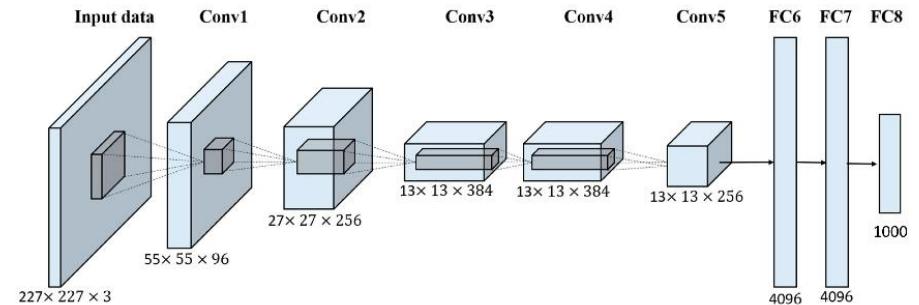
Una posible opción para modelar esto nos lo entregan las convoluciones

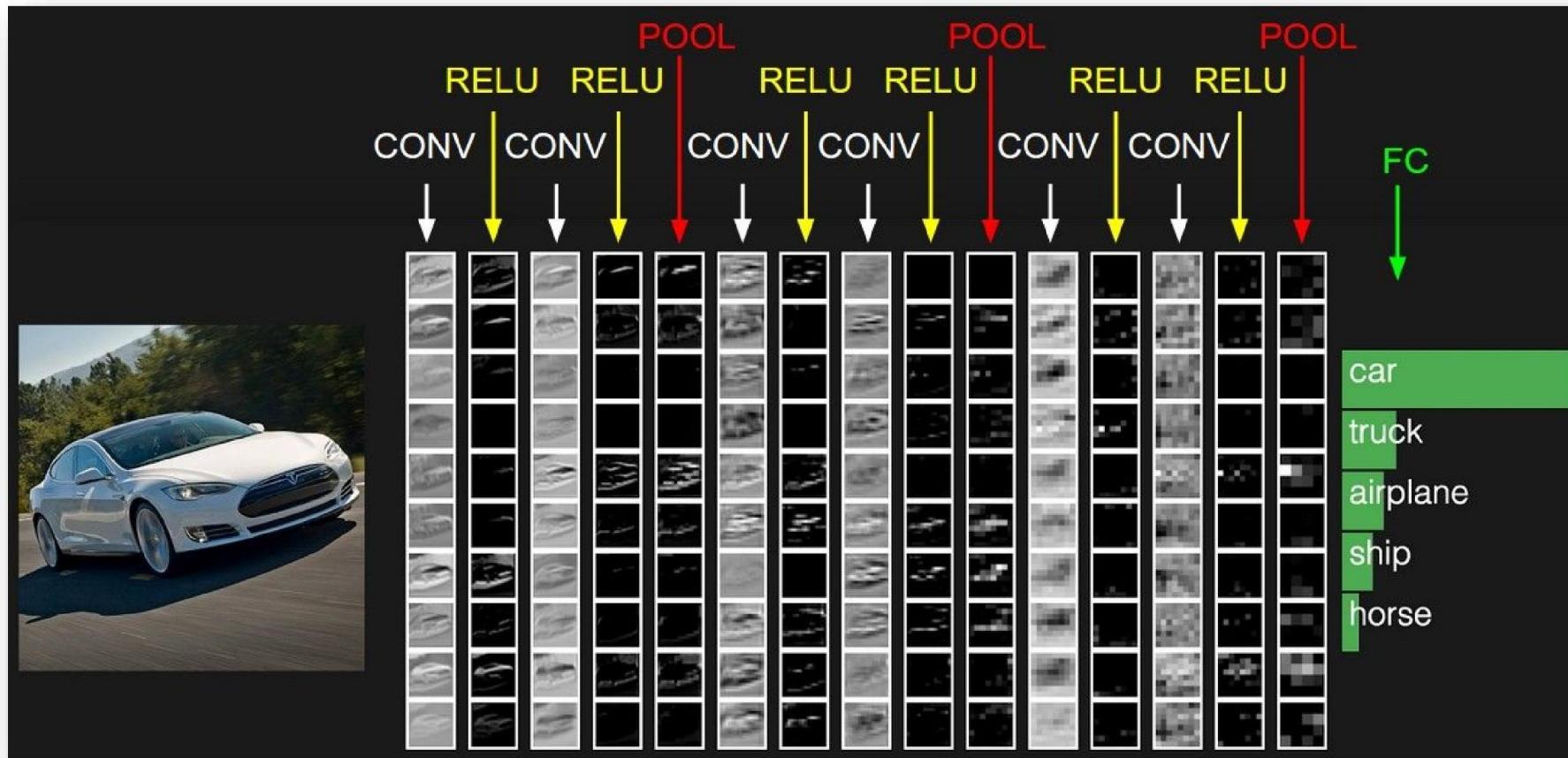
- Un patrón local puede ocurrir en cualquier parte de la imagen, pero las neuronas tradicionales no permiten capturar eso.
- La convolución entrega una solución a este problema, al modelar filtros (neuronas) locales que son aplicadas en toda la imagen.
- Los coeficientes de estos filtros son el análogo a los pesos de las neuronas en un MLP.
- Salida de una neurona ahora entrega un mapa (antes era un número), que indica dónde se podría encontrar la *feature* capturada por la neurona.



Todo esto da paso a las **redes convolucionales**, las redes profundas más exitosas/reconocidas/comunes

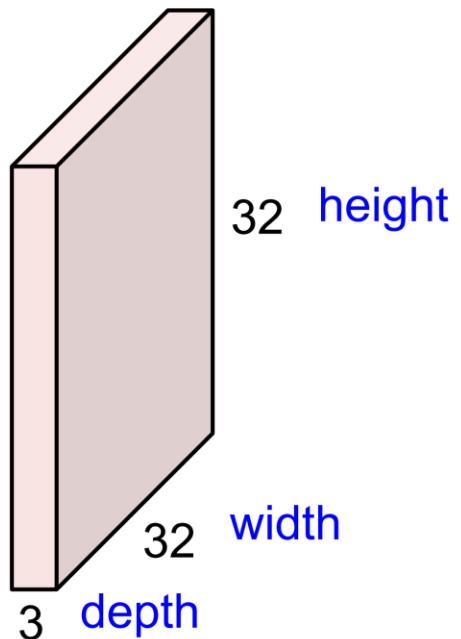
- Están formadas (generalmente) por capas convolucionales, de *pooling* y fully connected (las mismas de un MLP).
- Sustituyen la mayoría de las capas *fully connected* de los MLP, por capas convolucionales.
- Disminuyen drásticamente el número de parámetros, en comparación con los MLP.
- Sustentadas (ahora sí) en bases biológicas.





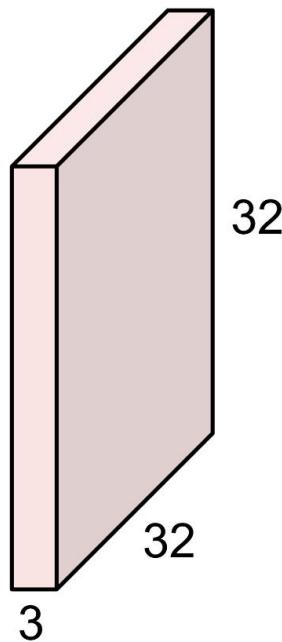
Convolution Layer

32x32x3 image

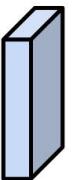


Convolution Layer

32x32x3 image



5x5x3 filter



Convolve the filter with the image
i.e. “slide over the image spatially,
computing dot products”

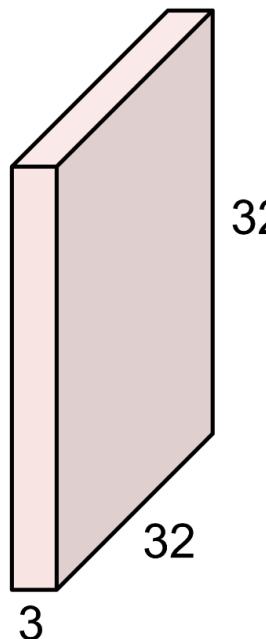
$$f[x,y] * g[x,y] = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} f[n_1, n_2] \cdot g[x - n_1, y - n_2]$$



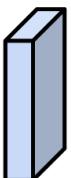
elementwise multiplication and sum of
a filter and the signal (image)

Convolution Layer

32x32x3 image



5x5x3 filter



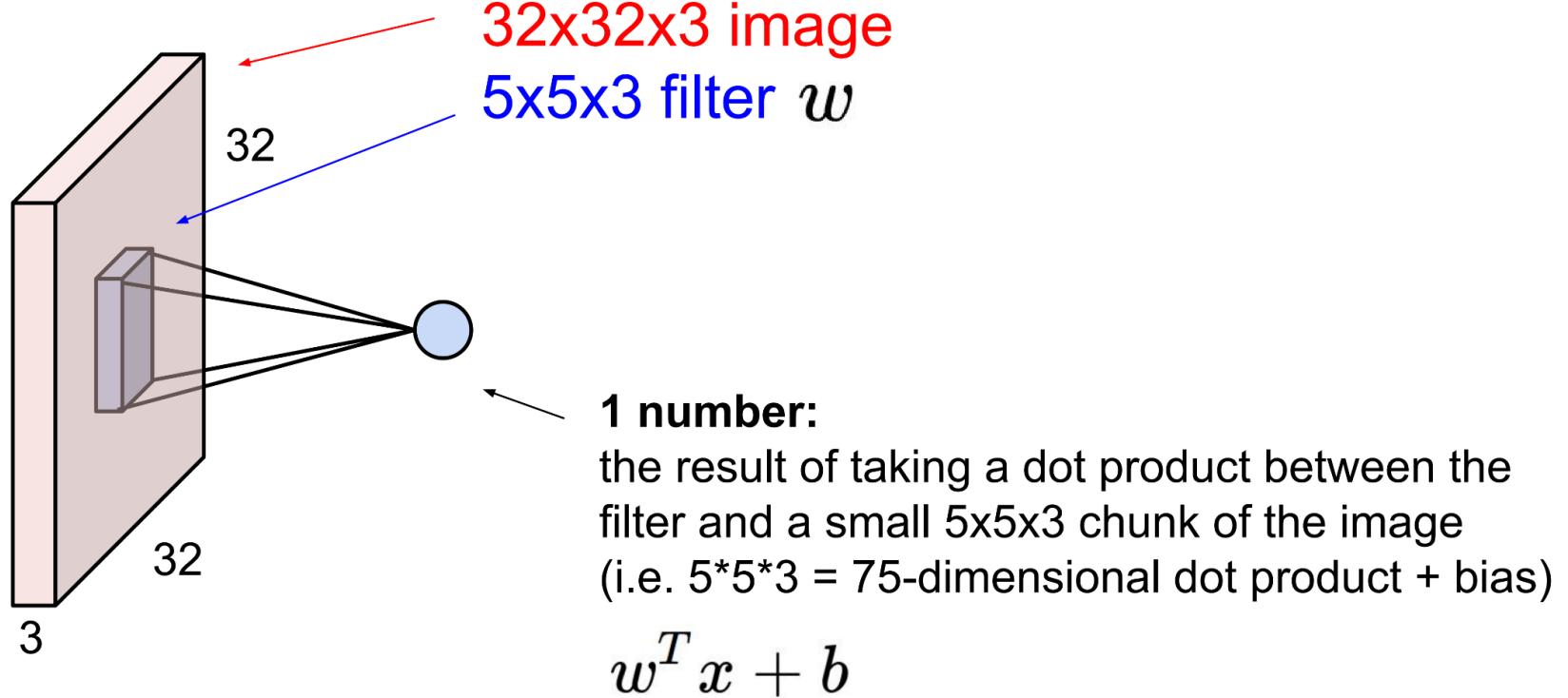
Filters always extend the full depth of the input volume

Convolve the filter with the image
i.e. “slide over the image spatially,
computing dot products”

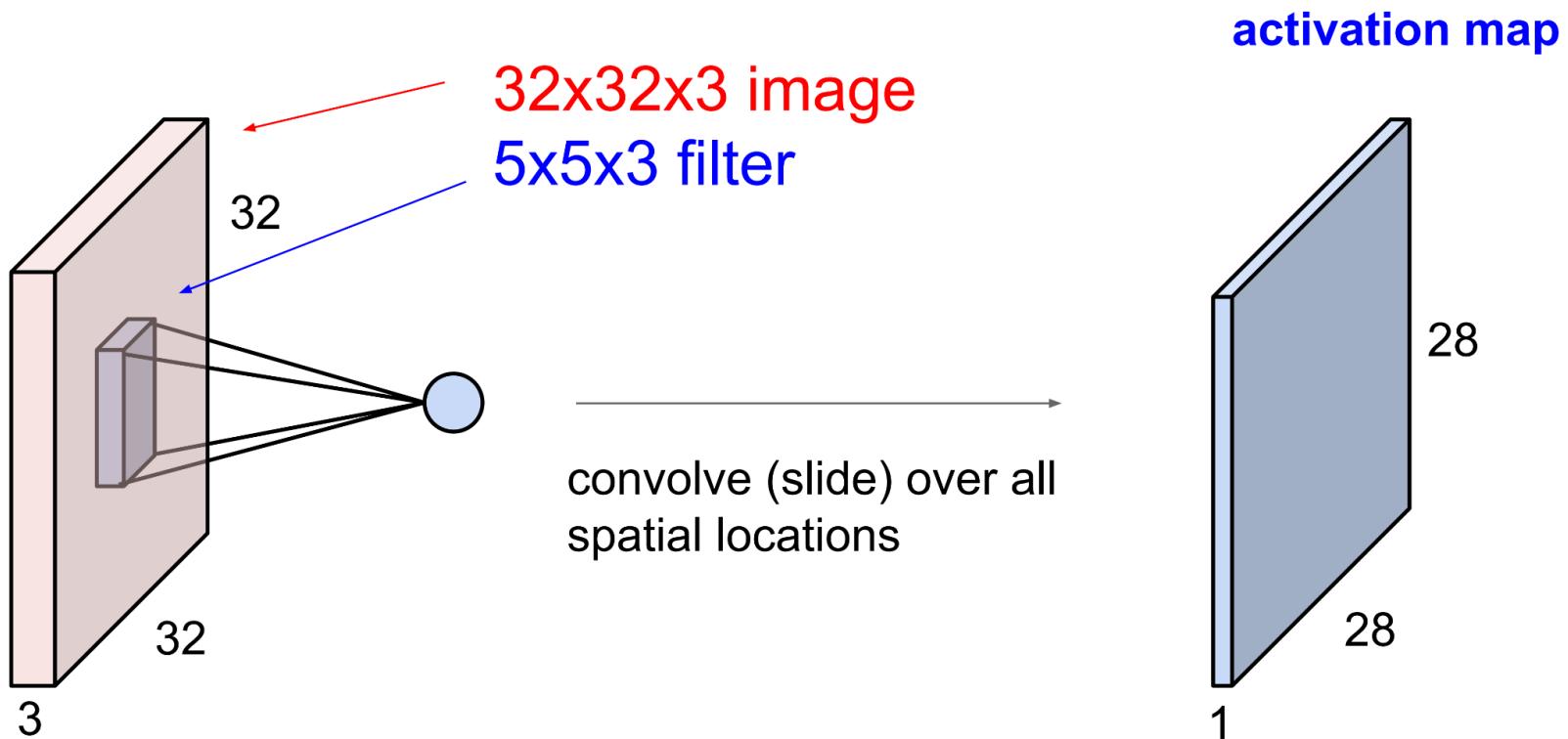
$$f[x,y] * g[x,y] = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} f[n_1, n_2] \cdot g[x - n_1, y - n_2]$$

elementwise multiplication and sum of
a filter and the signal (image)

Convolution Layer

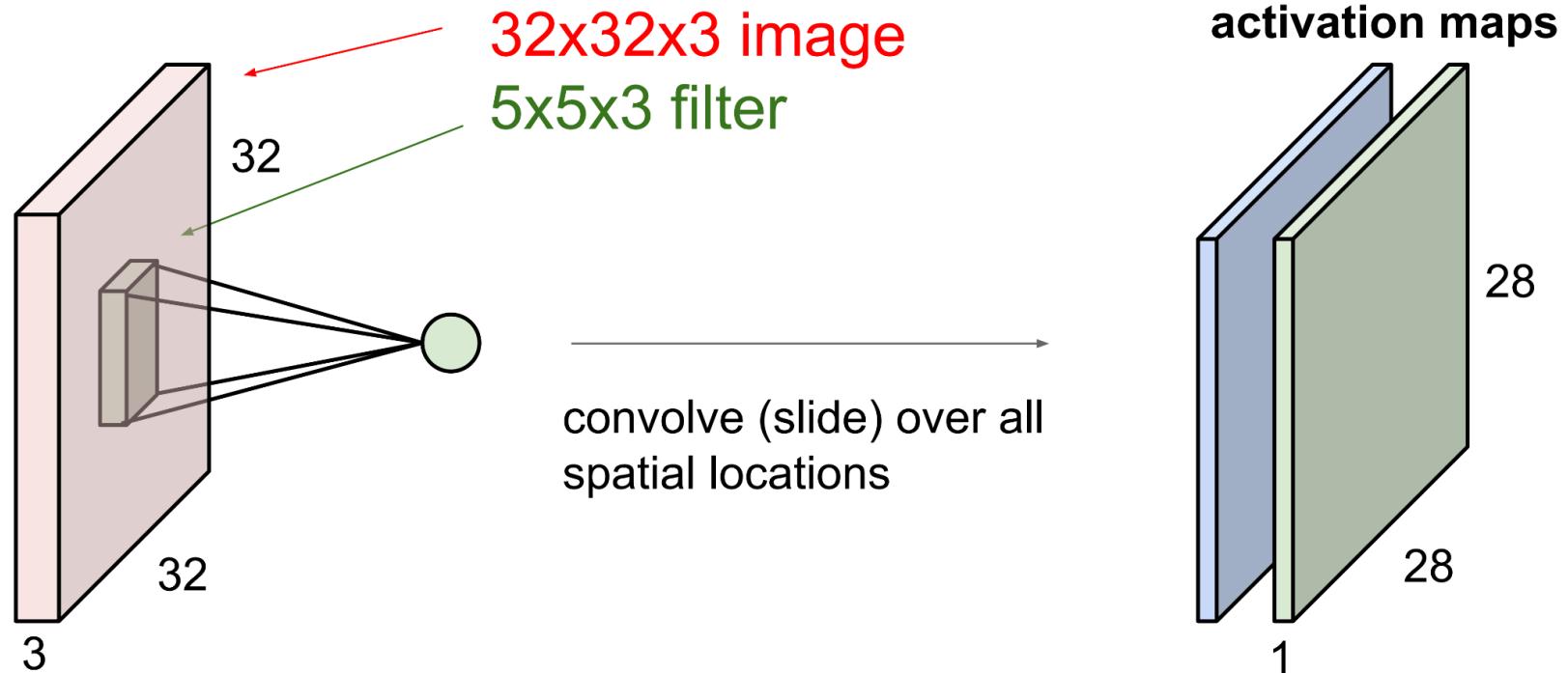


Convolution Layer

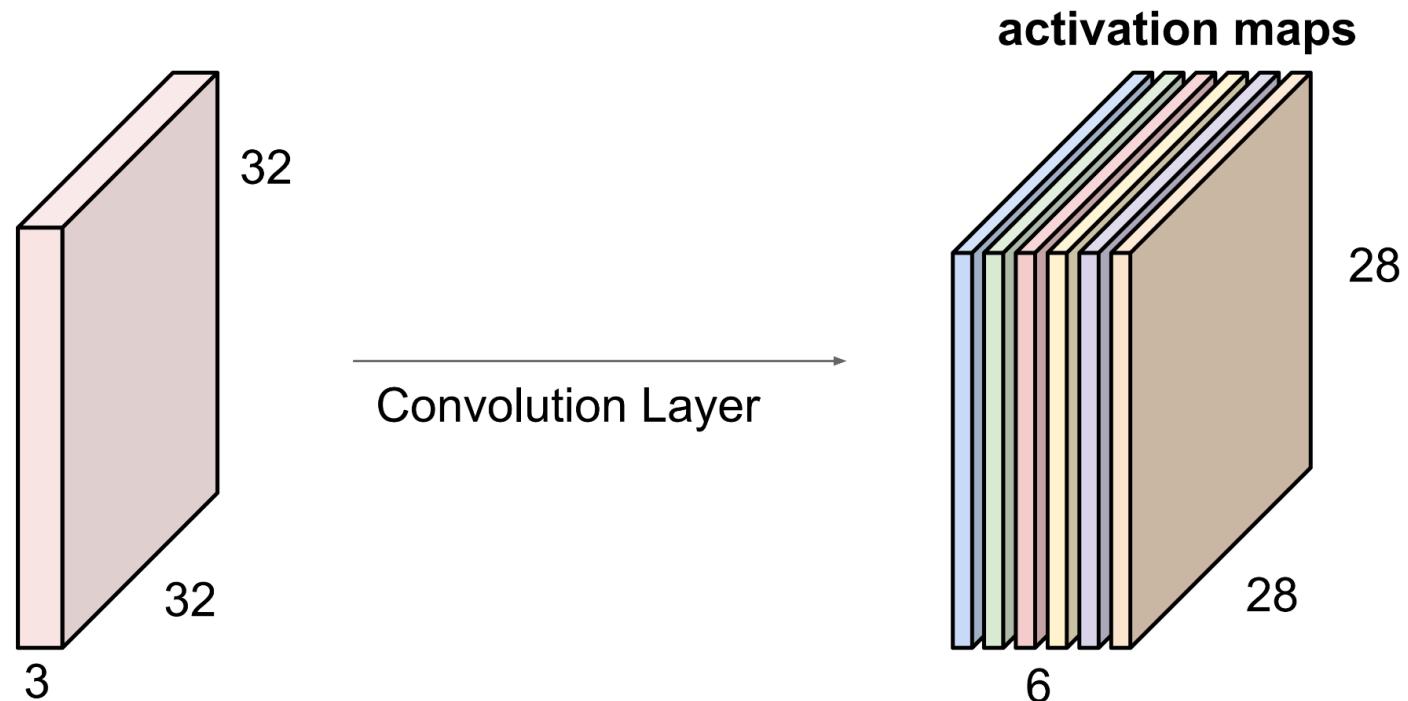


Convolution Layer

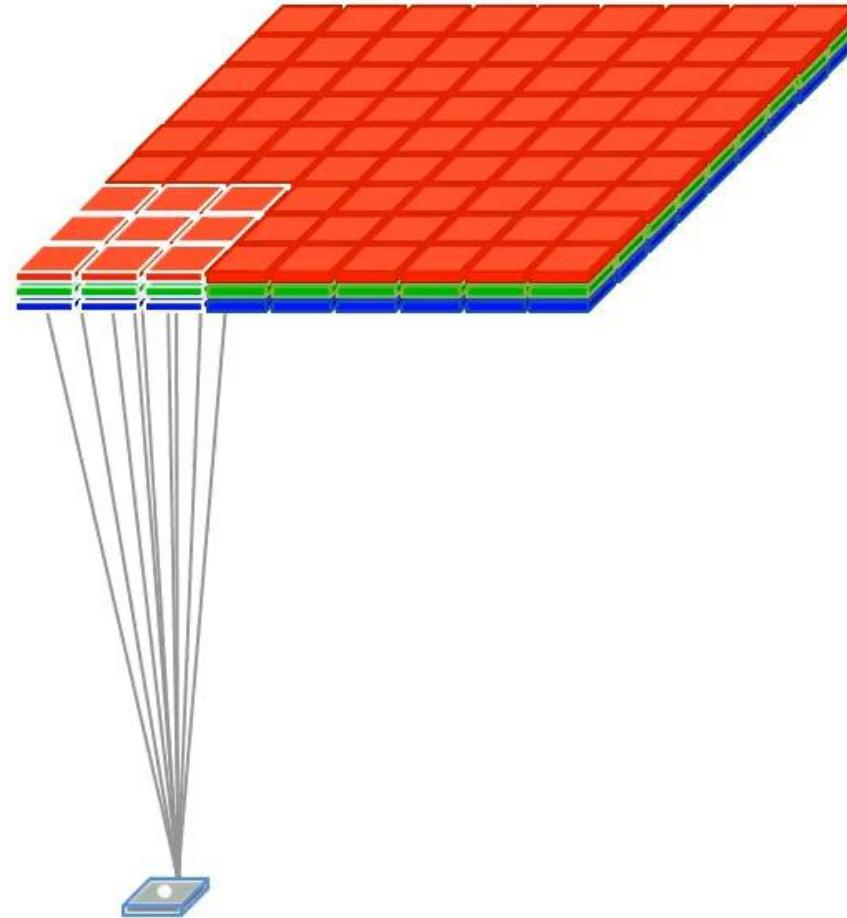
consider a second, green filter

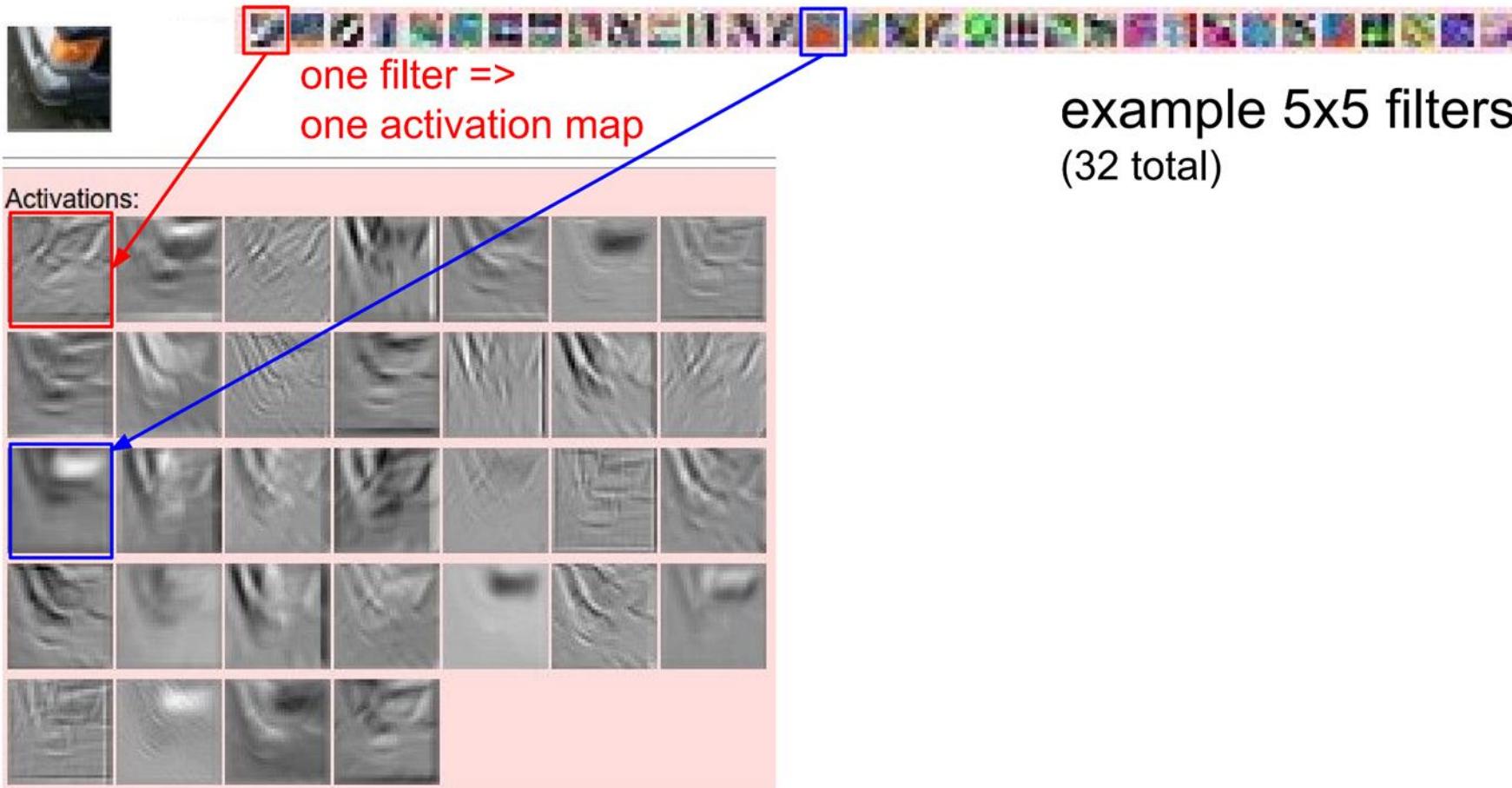


For example, if we had 6 5x5 filters, we'll get 6 separate activation maps:



We stack these up to get a “new image” of size 28x28x6!

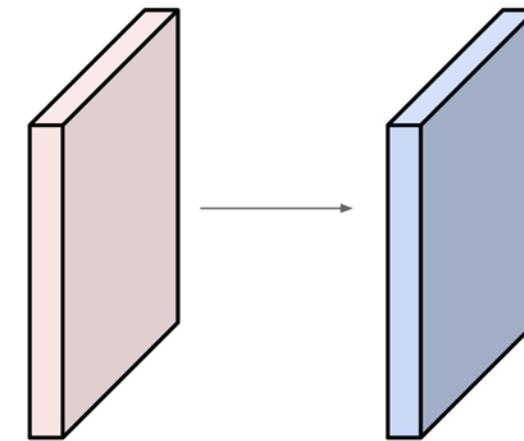




example 5x5 filters
(32 total)

Examples time:

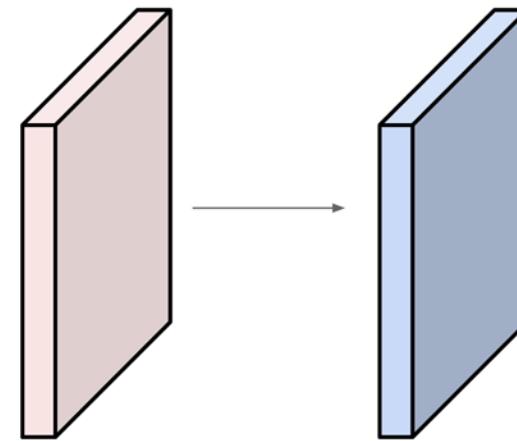
Input volume: **32x32x3**
10 5x5 filters



Number of parameters in this layer?

Examples time:

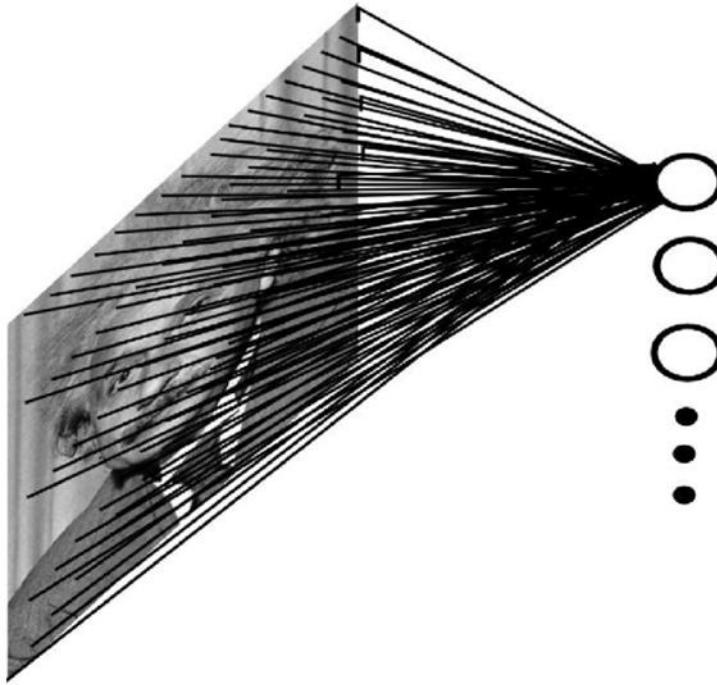
Input volume: **32x32x3**
10 5x5 filters



Number of parameters in this layer?

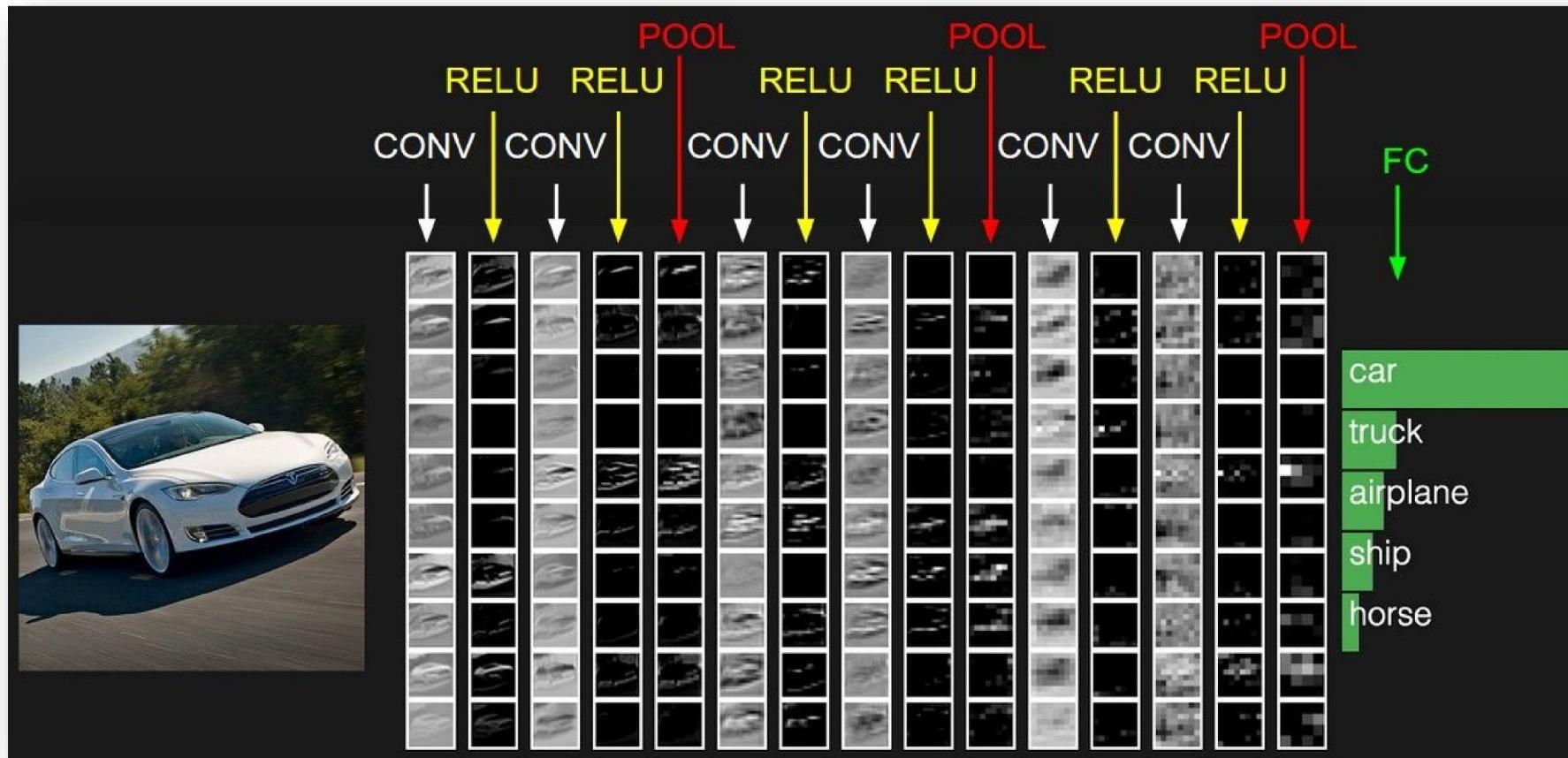
each filter has $5*5*3 + 1 = 76$ params (+1 for bias)
=> $76*10 = 760$

Comparemos el número anterior, con el número de parámetros de una capa *fully connected* para un caso similar

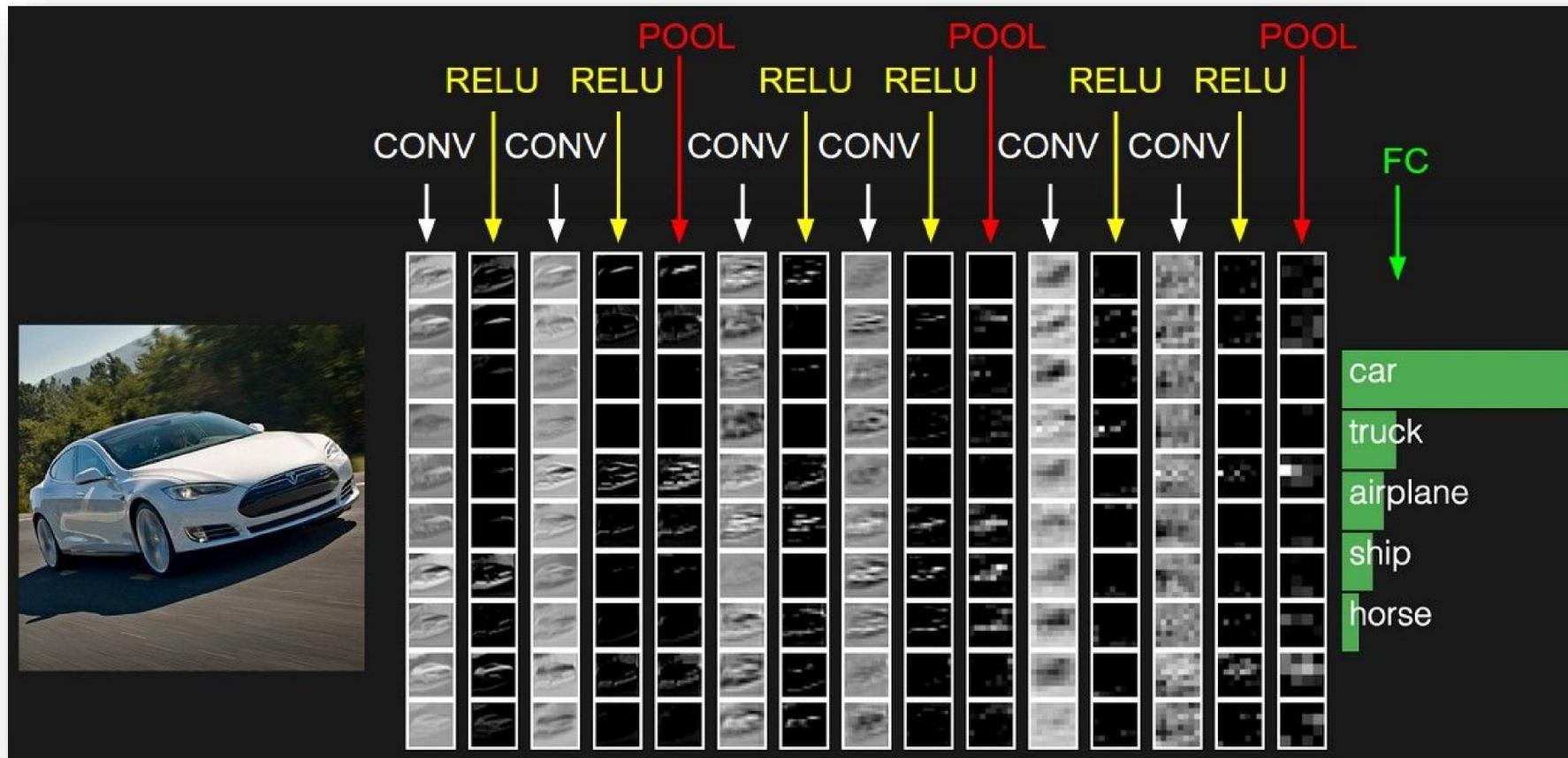


- Imagen de 32x32
- Capa *fully connected* de 10 neuronas con bias
- 10.250 parámetros

La situación es en realidad peor aún para los MLP, ya que incluso si usamos una imagen más grande como entrada (p. ej. 256x256), el número de parámetros de una red convolucional no cambiaría, pero el de un MLP se elevaría por sobre los el medio millón.

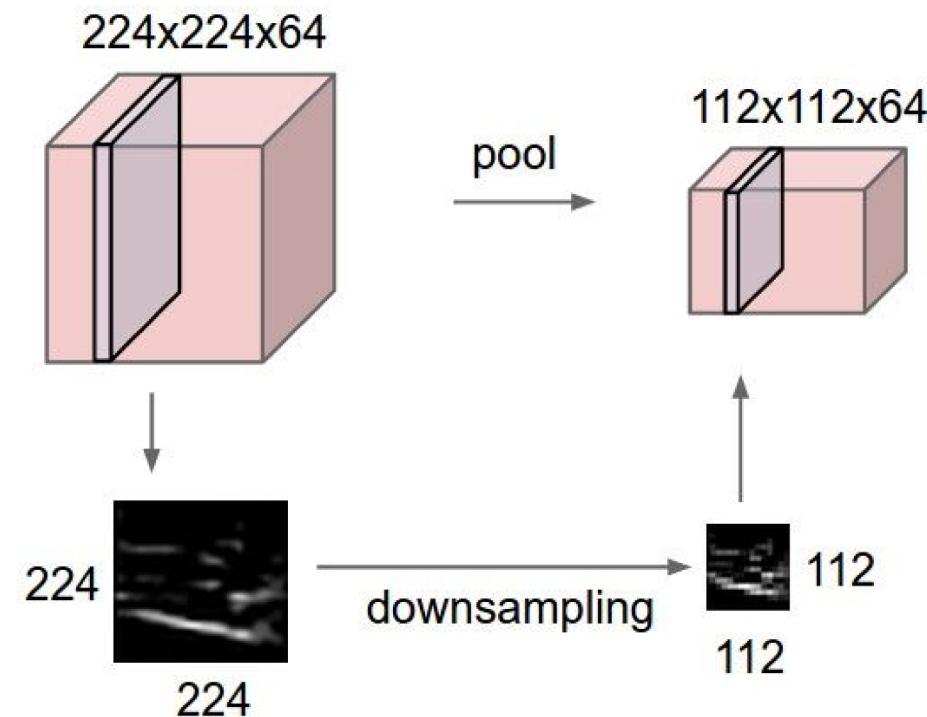


Name	Plot	Equation
Identity		$f(x) = x$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$
ArcTan		$f(x) = \tan^{-1}(x)$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$

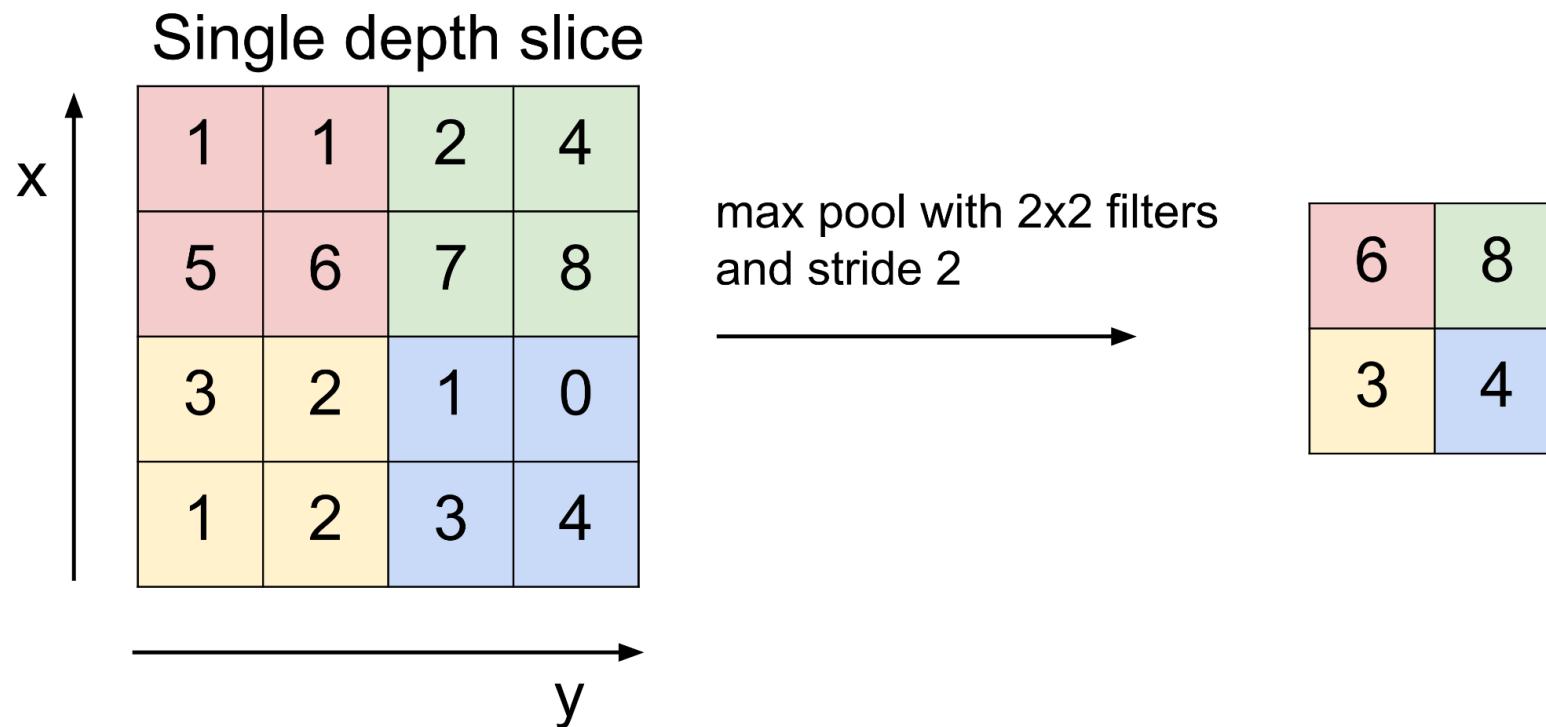


Pooling layer

- makes the representations smaller and more manageable
- operates over each activation map independently:

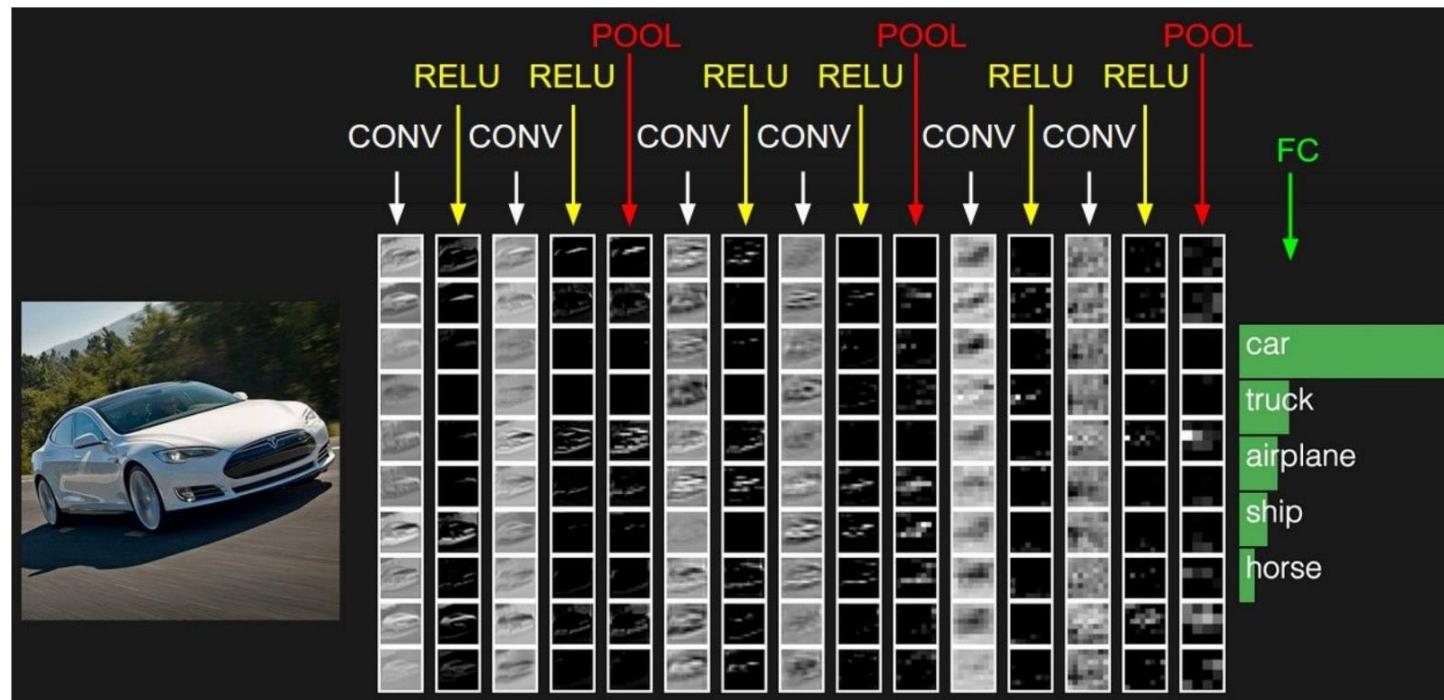


MAX POOLING

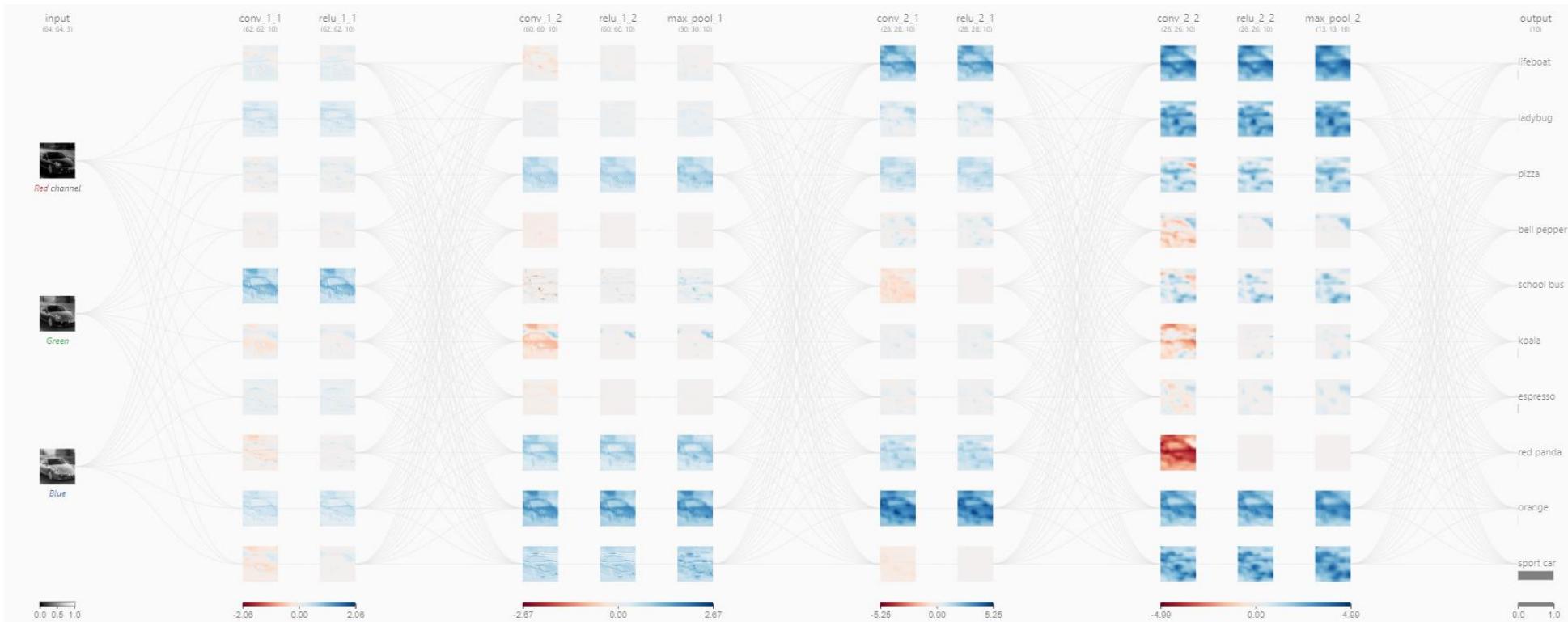


Fully Connected Layer (FC layer)

- Contains neurons that connect to the entire input volume, as in ordinary Neural Networks

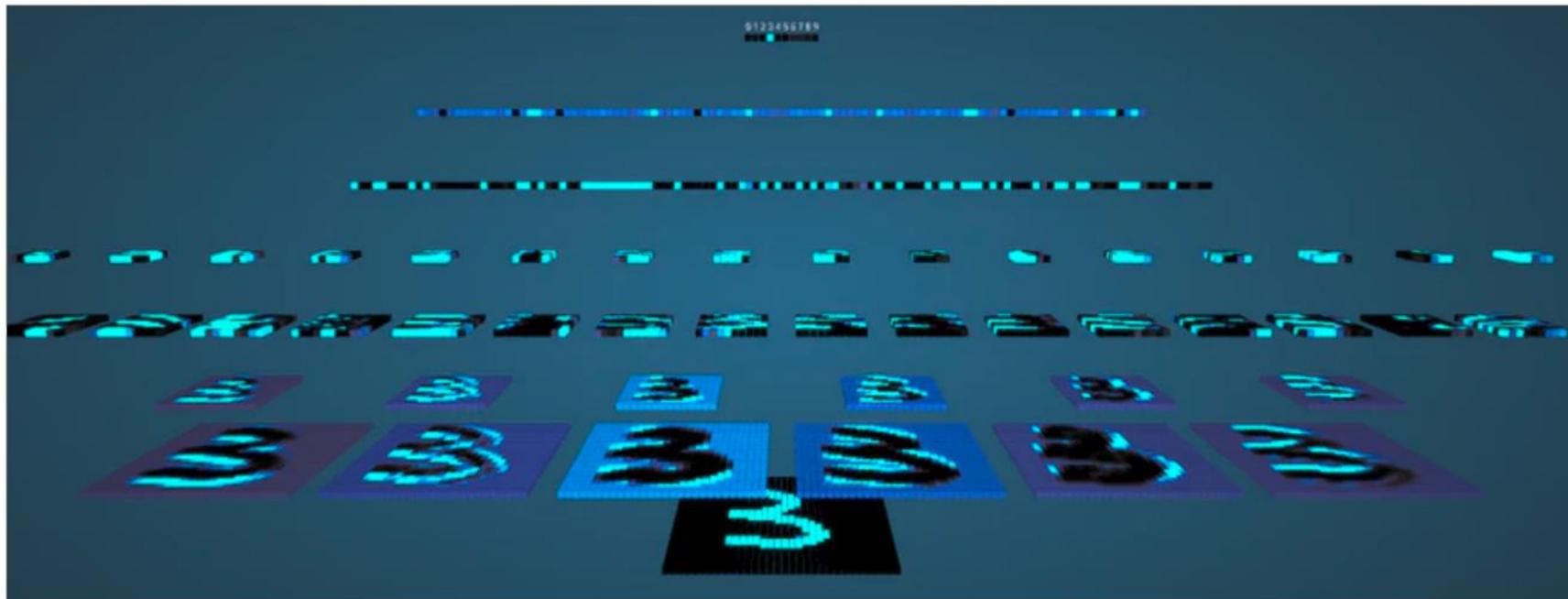


Revisemos como se **ve** todo esto en la práctica



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

Revisemos como se **ve** todo esto en la práctica



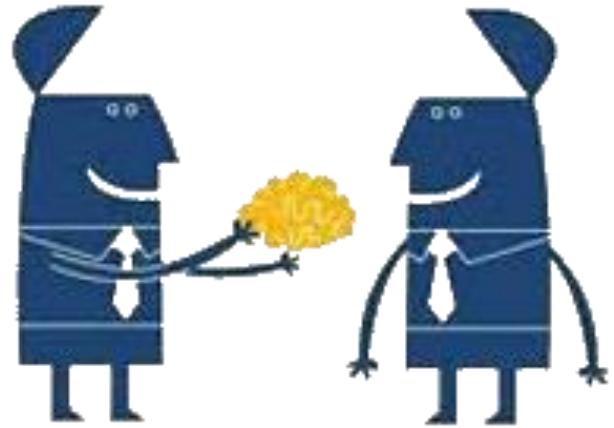
https://adamharley.com/nn_vis/

Reutilización es la máxima evidencia de generalización

- Resultados de DL con imágenes son claramente impresionantes, pero presentan un problema muy puntual, que no necesariamente se alinea con muchos casos de uso reales.
- Afortunadamente, y a diferencia de los métodos que hemos estudiado antes, las redes neuronales profundas ya entrenadas pueden ser usadas/adaptadas con poco esfuerzo en otras tareas.
- Esta propiedad es una consecuencia de la habilidad de las redes para aprender *features/embeddings* generales.

Este tipo de ideas son parte de un área llamada *Transfer Learning*

- En resumen, la idea es aprender inicialmente modelos poderosos utilizando sets de datos abundantes y de buena calidad.
- Luego, se transfiere de alguna manera el aprendizaje a una nueva tarea, generalmente con un conjunto de datos más limitado, ya sea en cuanto a cantidad, calidad o seguridad.
- Cómo y cuándo son temas de profunda investigación y con gran relevancia en muchos problemas, como por ejemplo el desarrollo de vehículos autónomos.



Este tipo de ideas son parte de un área llamada *Transfer Learning*

Source Domain

- Lots of labeled data

$$P_S = (X_S, Y_S)$$



Target Domain

- Limited labels

$$P_T = (X_T, Y_T)$$



Hope is that:

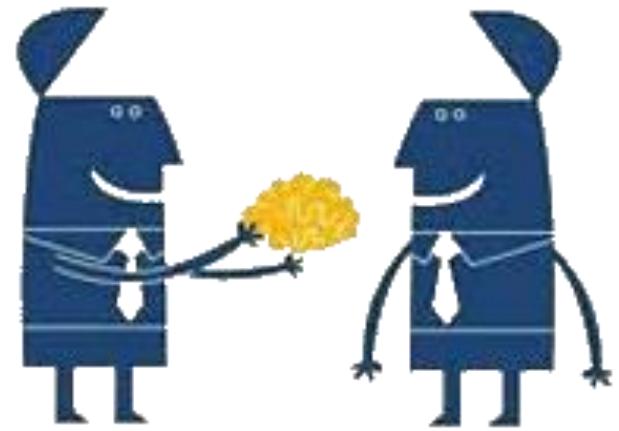
$$P_T \approx P_S$$

Or at least

$$P_T(X_T) \approx P_S(X_S)$$

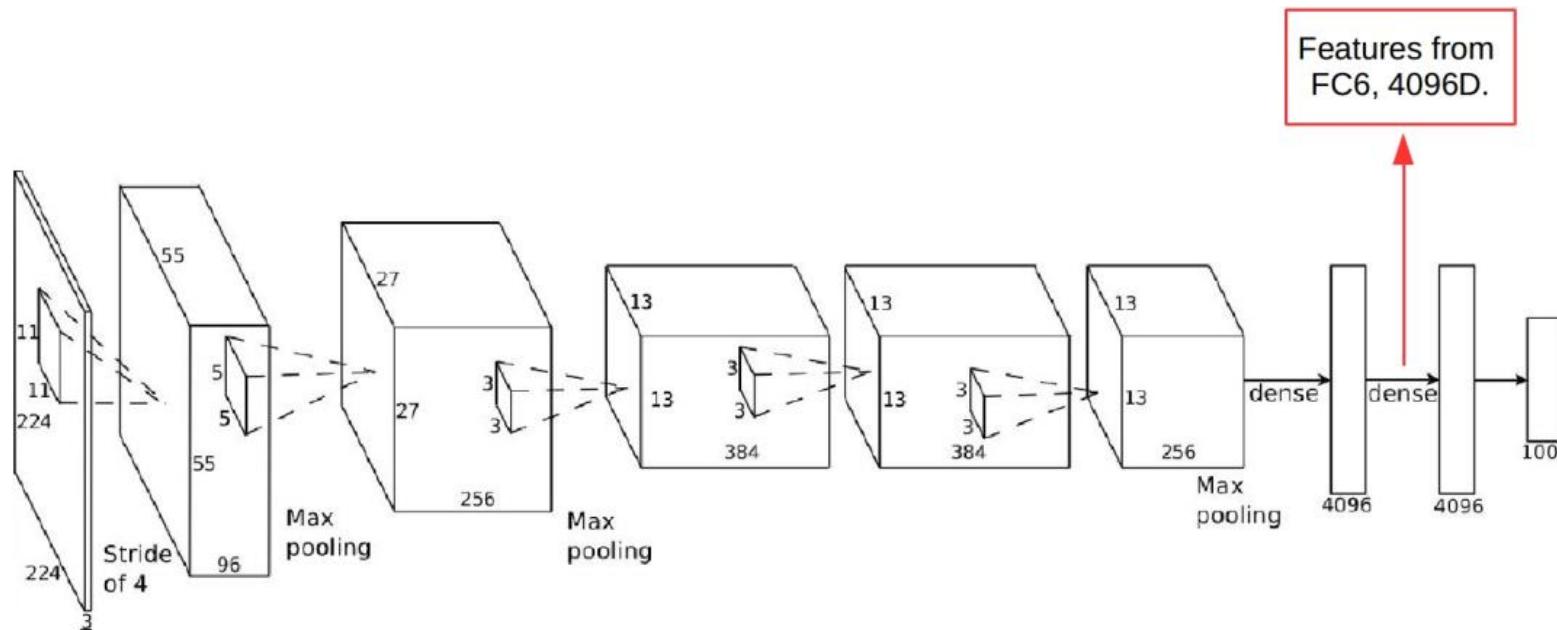
A nivel básico, existen dos mecanismos típicos para hacer esta adaptación

- **Transferencia directa:** *features/embeddings* aprendidos en un dominio se utilizan directamente en otro, sin noción de la existencia de una red neuronal subyacente.
- **Fine-tuning:** *features/embeddings* aprendidos son reentrenados “suavemente” en el nuevo dominio, cambiando el clasificador/regresor de la última capa.

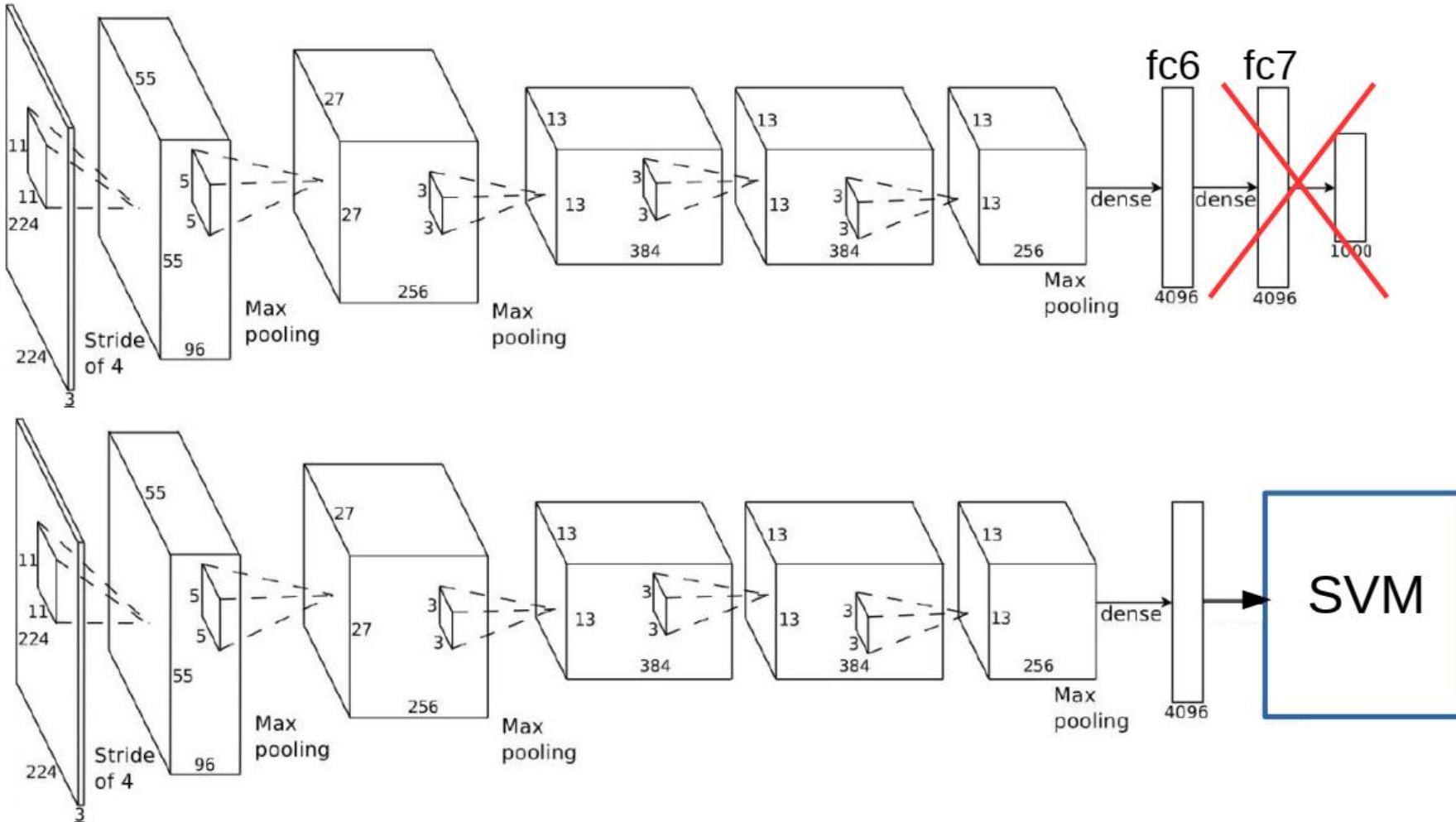


Transferencia directa de features/embeddings

Si le “cortamos la cabeza” a una CNN previamente entrenada y la utilizamos como extractor de características, se obtienen excelentes resultados en una gran cantidad de tareas visuales: reconocimiento de escenas, caras, acciones, personas, etc.

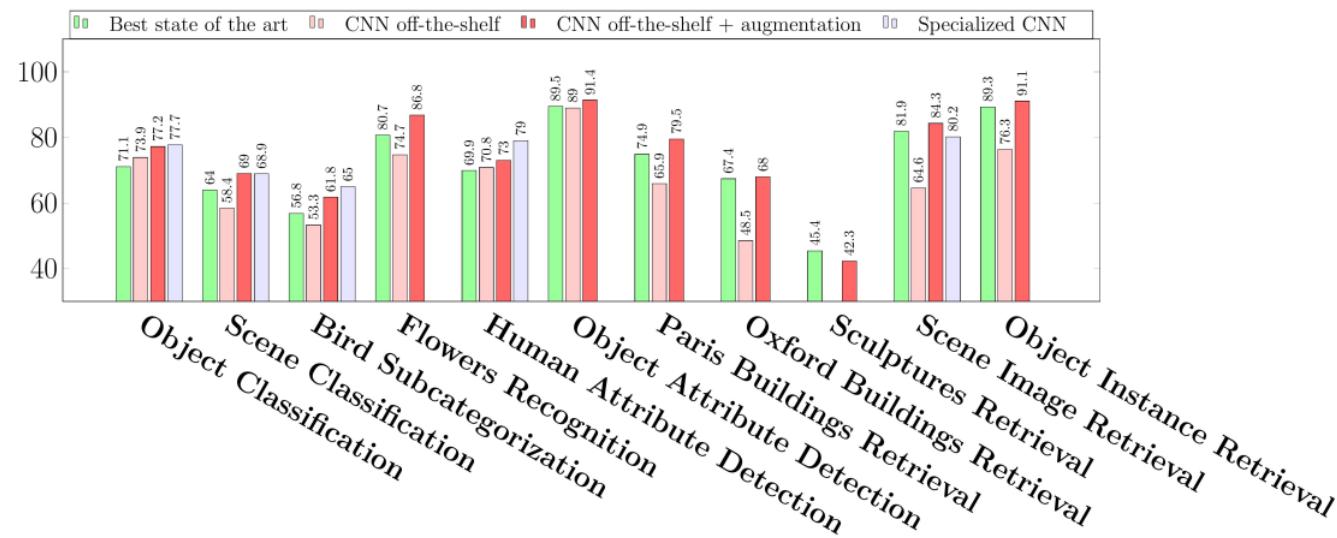
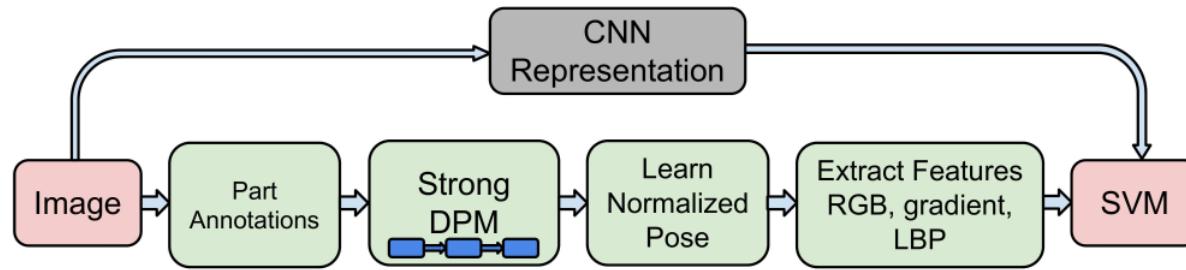


Transferencia directa de features/embeddings

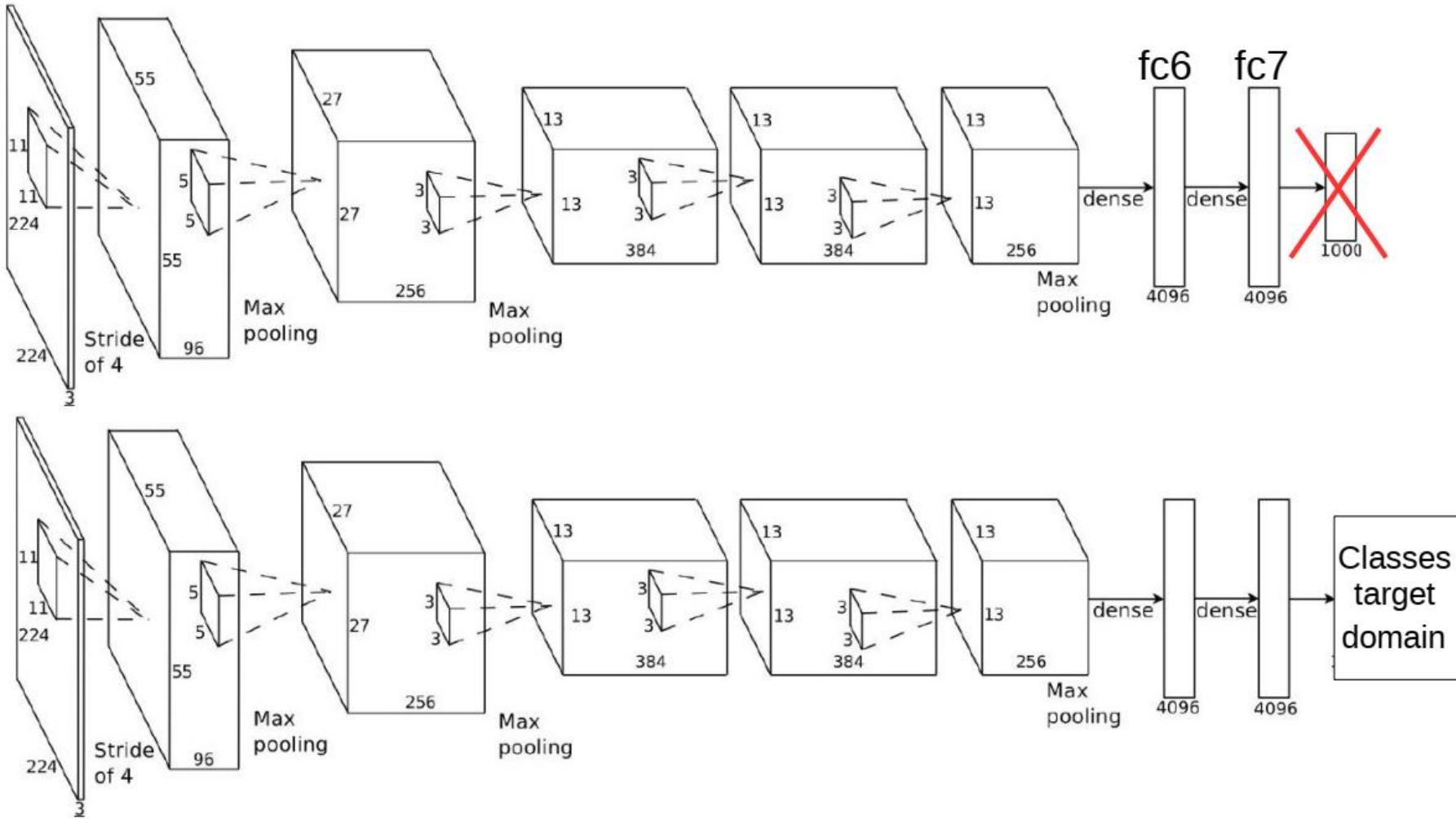


Transferencia directa de features/embeddings

Ex. A. Razavian et al., “*CNN Features off-the-shelf: an Astounding Baseline for Recognition*”, 2014.

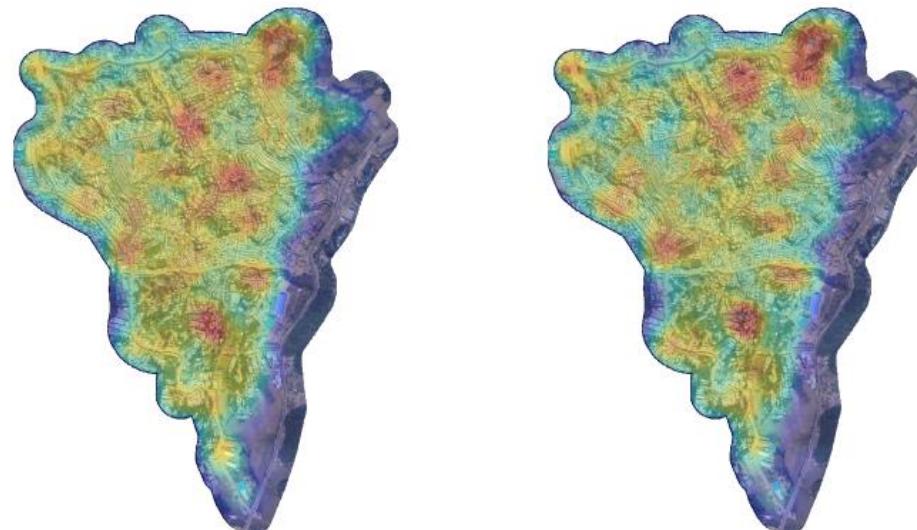
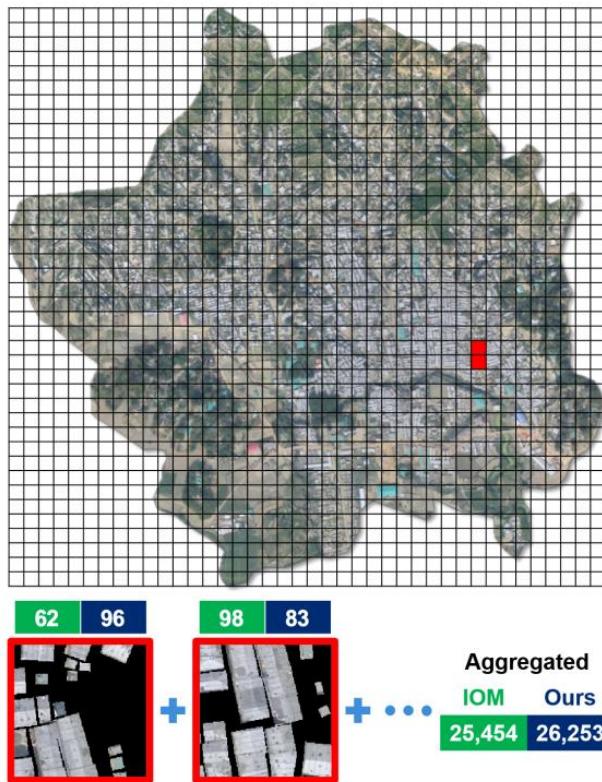


Fine-tuning de *features/embeddings*



Fine-tuning permite adaptar eficientemente las redes a dominios distintos

Estimating Displaced Populations From Overhead: Usando una CNN previamente entrenada con imágenes “tradicionales”, se cambia clasificador final por regresor y se “finetunea” usando imágenes satelitales.

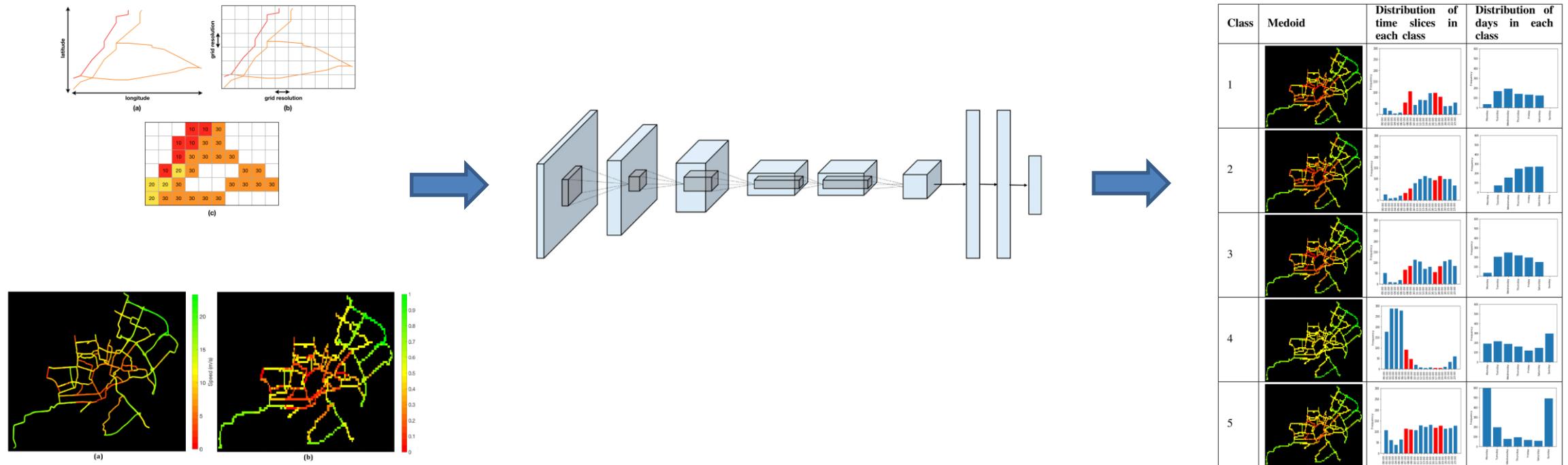


(a) Our Prediction

(b) IOM Estimate

Podemos ir más lejos aún, ya que las imágenes tampoco tienen que ser “reales”

Understanding Network Traffic States using Transfer Learning: se usa como base una CNN previamente entrenada en imágenes “tradicionales”, para extraer features de estados de tráfico codificados como imágenes y luego agruparlos con un algoritmo de clustering cualquiera.



Podemos ir más lejos aún, ya que las imágenes tampoco tienen que ser “reales”

Understanding Network Traffic States using Transfer Learning: se usa como base una CNN previamente entrenada en imágenes “tradicionales”, para extraer features de estados de tráfico codificados como imágenes y luego agruparlos con un algoritmo de clustering cualquiera.



Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación



IIC2613 - Inteligencia Artificial

Redes Neuronales Convolucionales (CNN)

Hans Löbel
Dpto. Ingeniería de Transporte y Logística
Dpto. Ciencia de la Computación