

Indexación, búsqueda y análisis en repositorios multimedia: imagen-video

Juan Carlos San Miguel

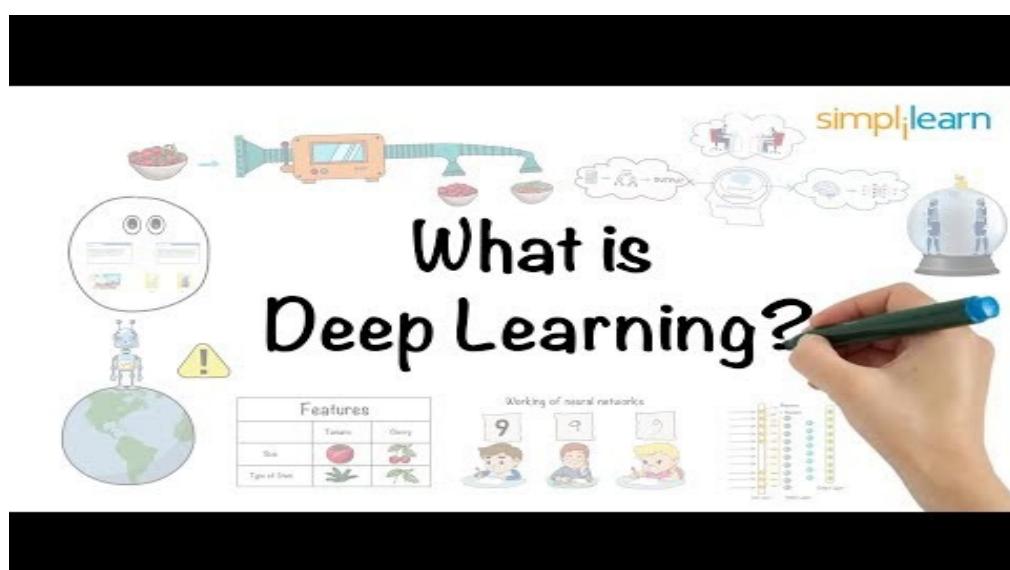
Agenda: Multimedia (imagen, video)

- **Redes convolucionales: fundamentos**
 - Introducción
 - Redes neuronales (repaso)
 - Redes Neuronales Convolucionales

Agenda: Multimedia (imagen, video)

- **Redes convolucionales: fundamentos**
 - Introducción
 - Redes neuronales (repaso)
 - Redes Neuronales Convolucionales

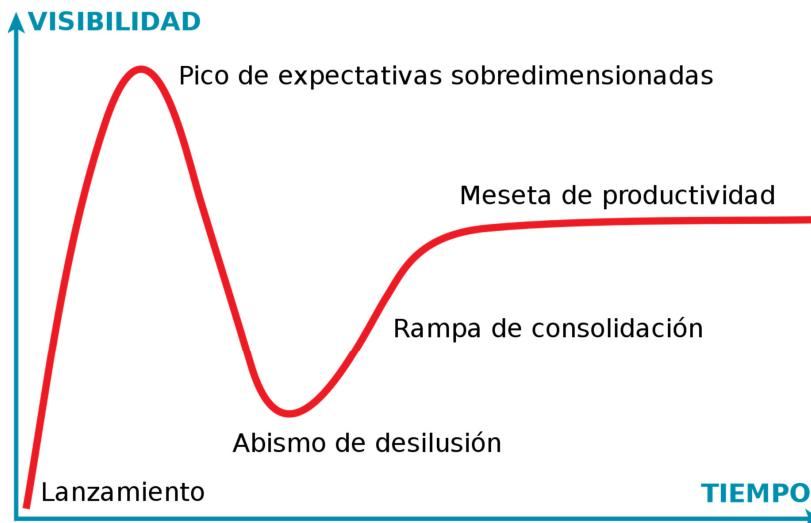
INTRODUCCION



<https://youtu.be/6M5VXKlf4D4>

INTRODUCCION

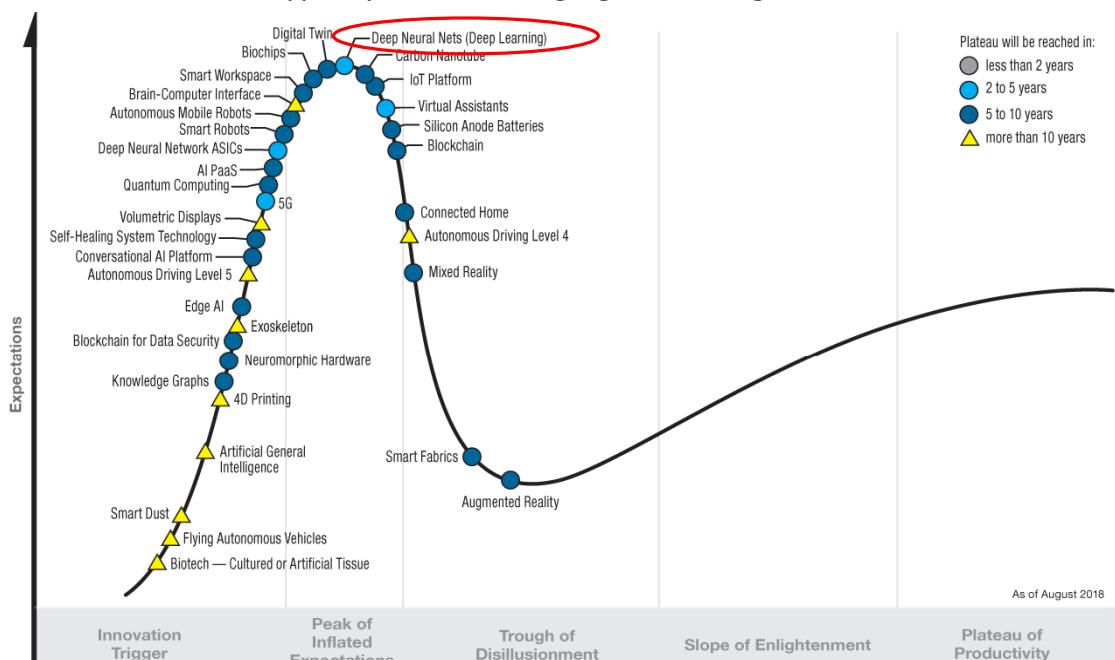
Ciclo de sobre-expectación de Gartner (*hype cycle*)



Fuente: https://es.wikipedia.org/wiki/Ciclo_de_sobreexpectaci%C3%B3n

INTRODUCCION

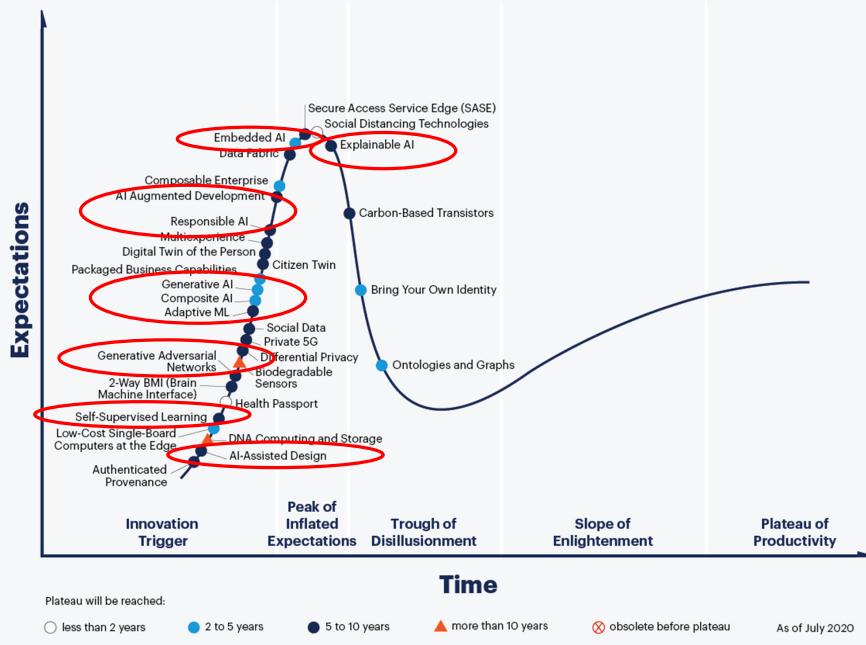
Hype Cycle for Emerging Technologies, 2018



INTRODUCCION

Hype Cycle for Emerging Technologies, 2020

Fuente:
<https://www.gartner.com/>



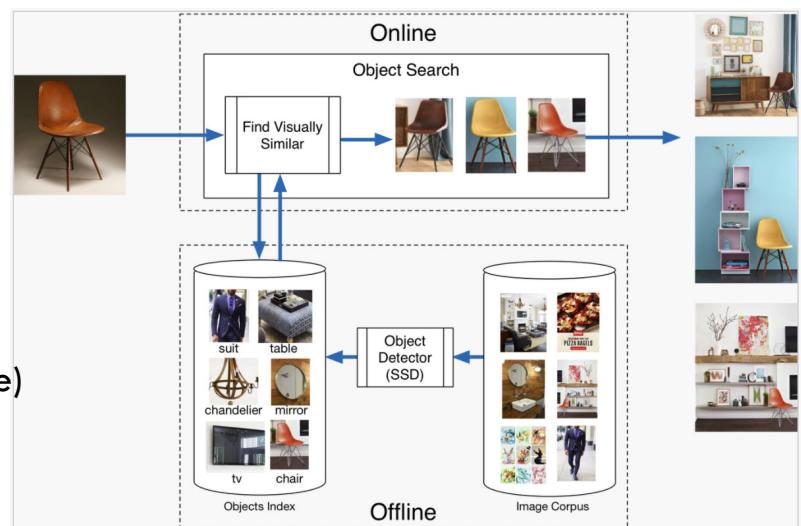
Máster

y análisis
imedia

6

INTRODUCCION: indexación basada en Deep Learning

- Visual Search* @ Pinterest
 - Apache Hadoop
 - Apache Hbase para almacenamiento
 - PinLater for scheduling
 - Extracción características
 - Localidad/saliente color
 - Deep features (CNNs + Caffe)
 - 5 GPUs para subidas datos en cada día
 - Ejecutado en Amazon EC2



*A fecha de 2015

<https://engineering.pinterest.com/tags/deep-learning>

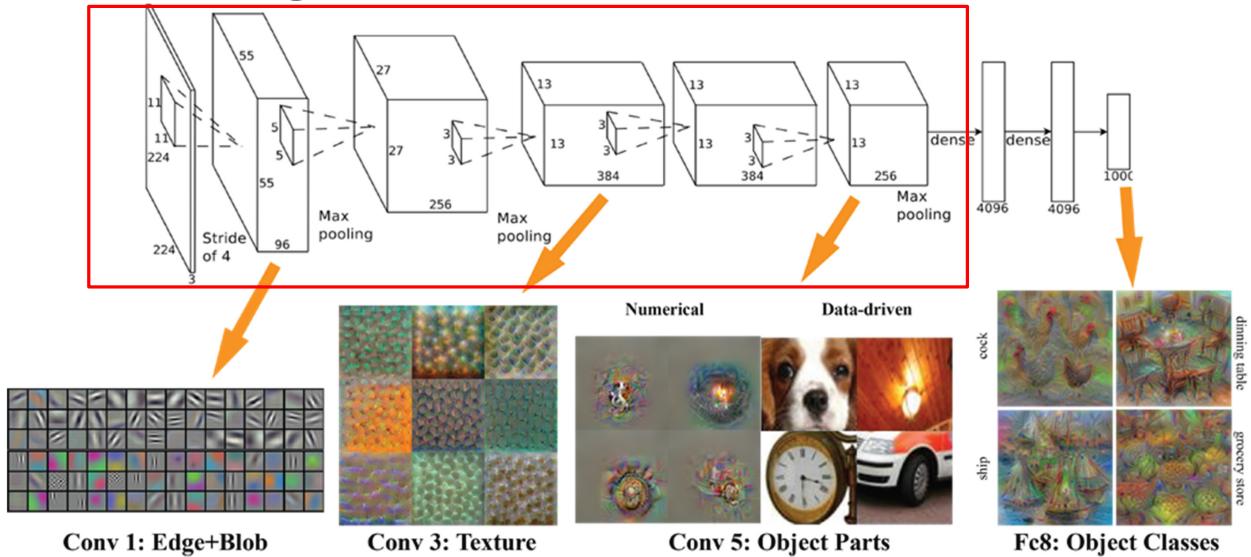
Máster en Big Data y Data Science

Indexación, búsqueda y análisis
en repositorios multimedia

7

INTRODUCCION: indexación basada en Deep Learning

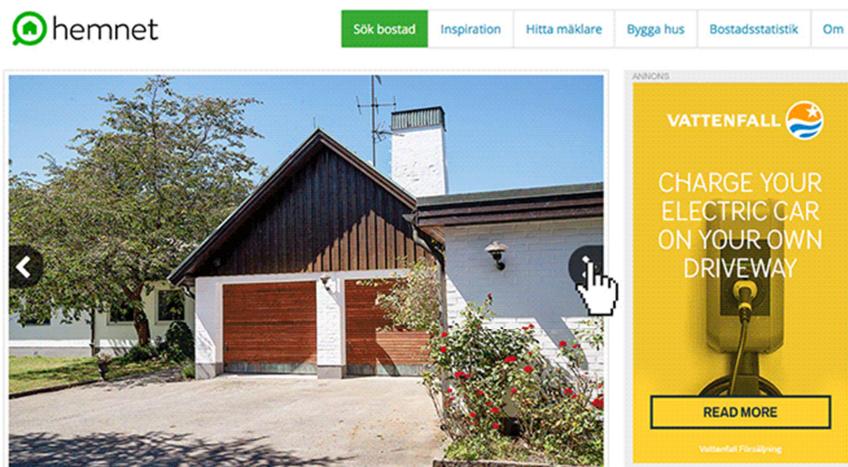
- Visual Search @ Pinterest – Deep features con CNNs



<https://www.cc.gatech.edu/~hays/compvision/>

INTRODUCCION: indexación basada en Deep Learning

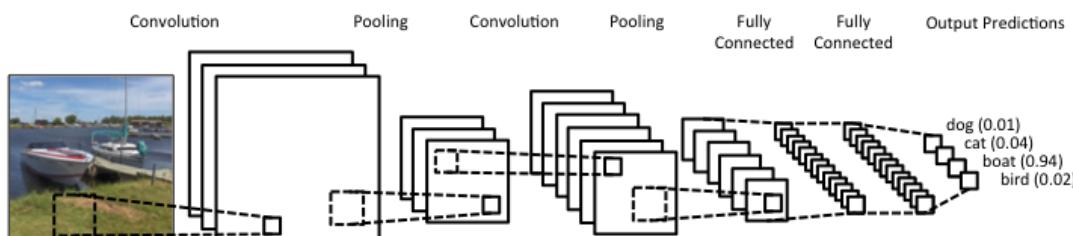
- Anuncios inteligentes en el sector inmobiliario @ Clarifai
- Indexación en tiempo real de imágenes subidas por el usuario



<https://vimeo.com/165334874>

INTRODUCCION: indexación basada en Deep Learning

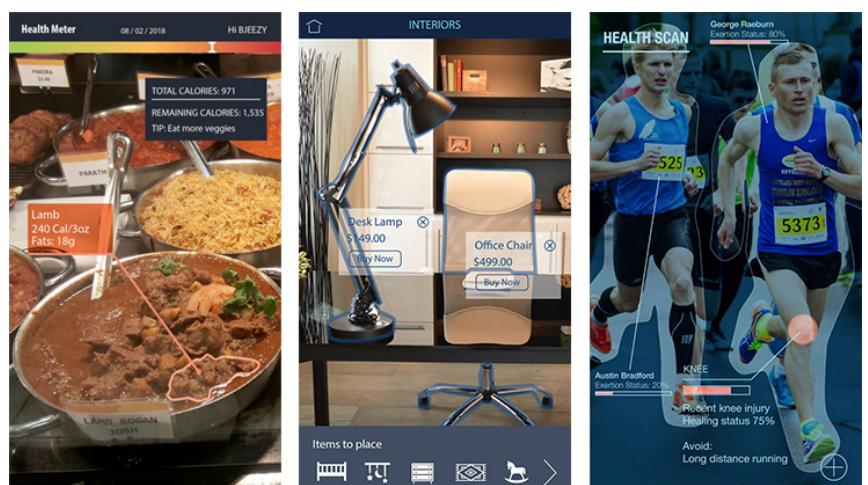
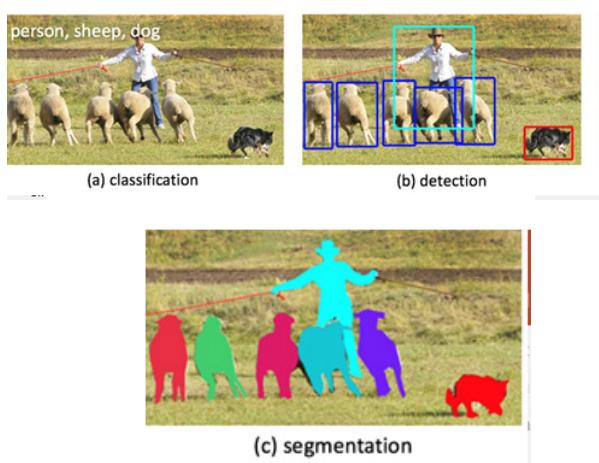
- Anuncios inteligentes en el sector inmobiliario @ Clarifai
 - Una red convolucional entrenada para clasificar 1K tipos de objetos
 - La arquitectura tiene 65M parámetros, entrenada durante 10 días en una GPU Nvidia (2013)
 - Tecnología validada en la competición ImageNet (1.4M imágenes, 1K clases)



<https://www.clarifai.com/technology>

INTRODUCCION: indexación basada en Deep Learning

- Reconocimiento de objetos @ Facebook



<https://www.facebook.com/aidemos/>

Images © Facebook

INTRODUCCION: indexación basada en Deep Learning

➤ Reconocimiento de objetos @ Facebook

<https://arxiv.org/pdf/1506.06204.pdf>

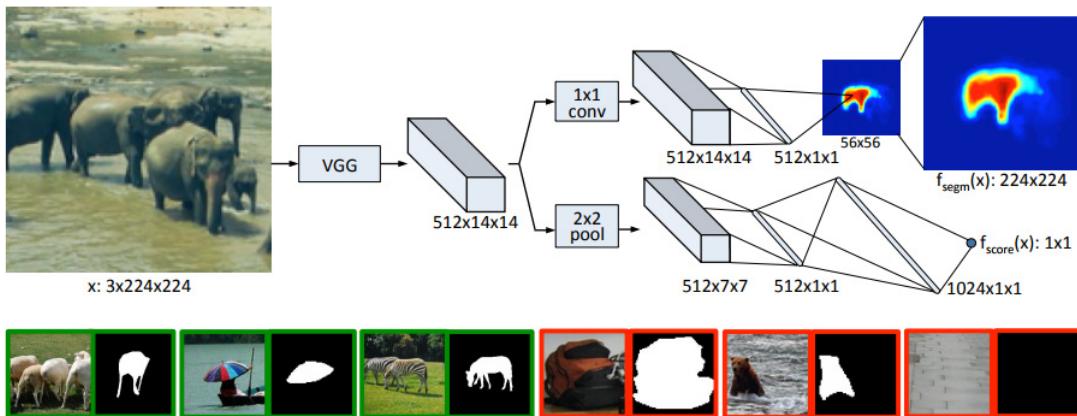


Figure 1: (**Top**) Model architecture: the network is split into two branches after the shared feature extraction layers. The top branch predicts a segmentation mask for the the object located at the center while the bottom branch predicts an object score for the input patch. (**Bottom**) Examples of training triplets: input patch x , mask m and label y . Green patches contain objects that satisfy the specified constraints and therefore are assigned the label $y = 1$. Note that masks for negative examples (shown in red) are not used and are shown for illustrative purposes only.

Más

12

INTRODUCCION: indexación basada en Deep Learning

➤ Entrenamiento de sistemas requiere grandes cantidades de datos



¿Ayuda de usuarios?



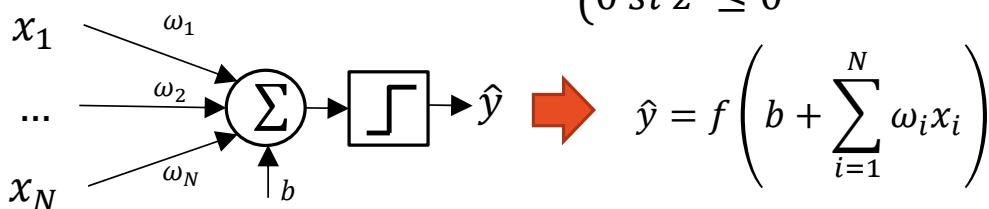
Agenda: Multimedia (imagen, video)

- **Redes convolucionales: fundamentos**
 - Introducción
 - Redes neuronales (repaso)
 - Perceptrón
 - Perceptrón multicapa
 - Redes Neuronales Convolucionales

REDES NEURONALES

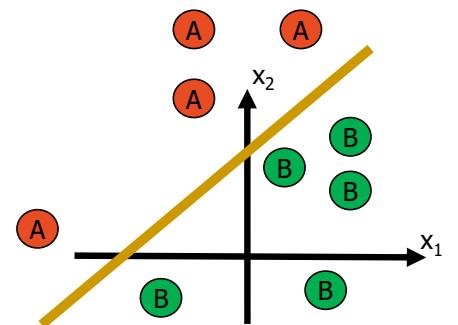
- Perceptrón: red neuronal básica [Frank Rosenblatt, ~1957]
 - Capa de entrada $x_1 \dots x_N$
 - Unidad de salida \hat{y}
 - Pesos ω_i y sesgo b

- Función de activación escalón $f(z) = \begin{cases} 1 & \text{si } z > 0 \\ 0 & \text{si } z \leq 0 \end{cases}$



REDES NEURONALES

- Perceptrón: red neuronal básica
 - Adecuada para problemas lineales
 - Capa de entrada $x_1 \dots x_N$
 - Unidad de salida \hat{y}
 - Pesos ω_i y sesgo b
 - Función de activación escalón f



Límite de decisión (lineal)

$$w_1x_1 + w_2x_2 + b_0 = 0$$

Regla de clasificación:

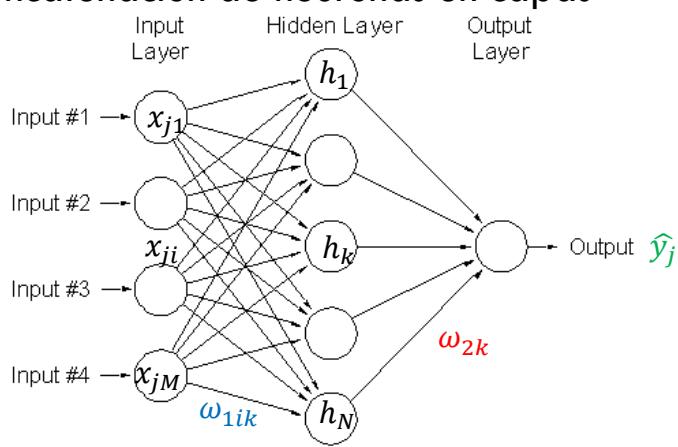
$$b_0 + w_1x_1 + w_2x_2 > 0 \rightarrow \text{clase A}$$

$$b_0 + w_1x_1 + w_2x_2 \leq 0 \rightarrow \text{clase B}$$

REDES NEURONALES

- Perceptrón multicapa: problemas no lineales

- Concatenación de neuronas en capas



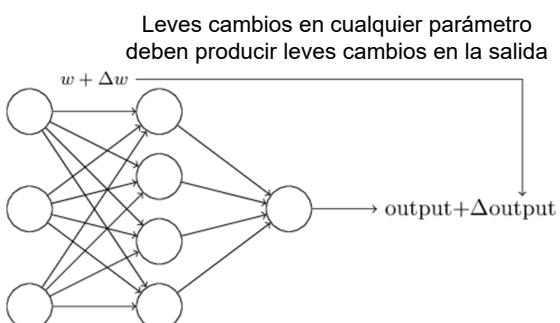
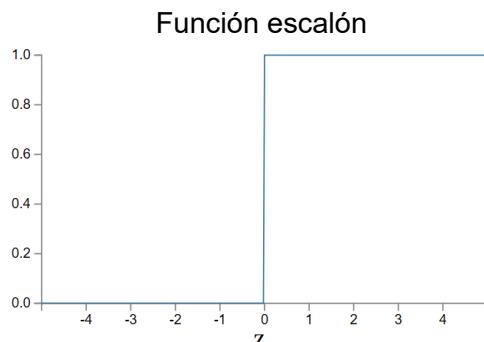
$$\hat{y}_j = \phi_2 \left(b_2 + \sum_{k=1}^N \omega_{2k} \cdot \phi_{1k} \left(b_{1j} + \sum_{i=1}^M \omega_{1ik} \cdot x_{ji} \right) \right)$$

Input x_j con unidades x_{ji}
Salida - output \hat{y}_j
unidades ocultas h_k
Funciones activación ϕ_{1k} & ϕ_2
Parámetros: ω_{1ik} , ω_{2k} & b 's

Feedforward pass:
calcular salida \hat{y}_j
dado el dato x_j

REDES NEURONALES

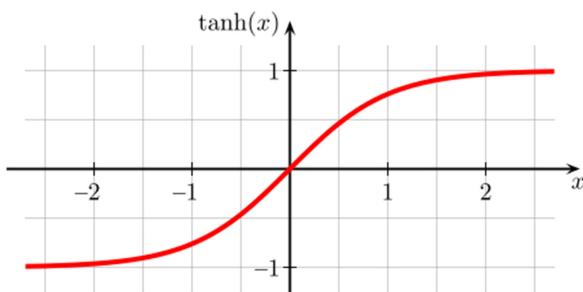
- Perceptrón multicapa: problemas no lineales
 - Limitaciones de funciones de activación previamente utilizadas
 - Composición de funciones lineales se reduce a una función lineal $g(f(h(z)))$
 - Pequeño cambio en los datos de entrada puede implicar un gran cambio en la salida del clasificador



REDES NEURONALES

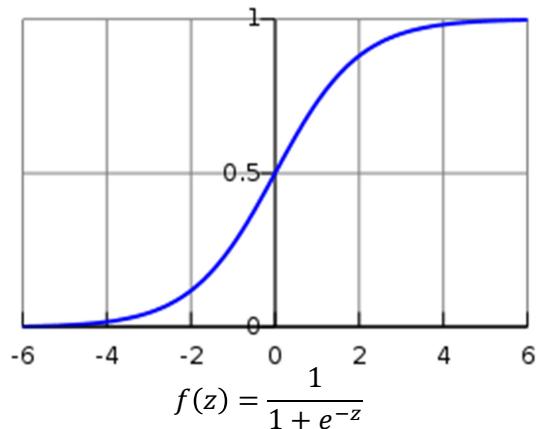
- Perceptrón multicapa: problemas no lineales
 - Funciones no lineales $f(z) = f(\omega \cdot x + b)$

Función tangente hiperbólica



$$f(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

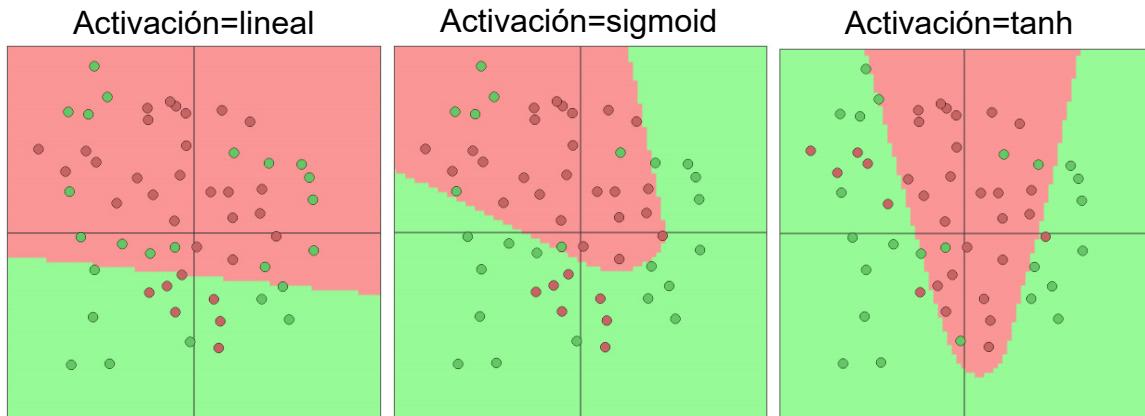
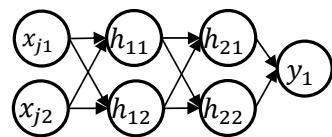
Función sigmoide



Más funciones disponibles en https://en.wikipedia.org/wiki/Activation_function

REDES NEURONALES

➤ Perceptrón multicapa



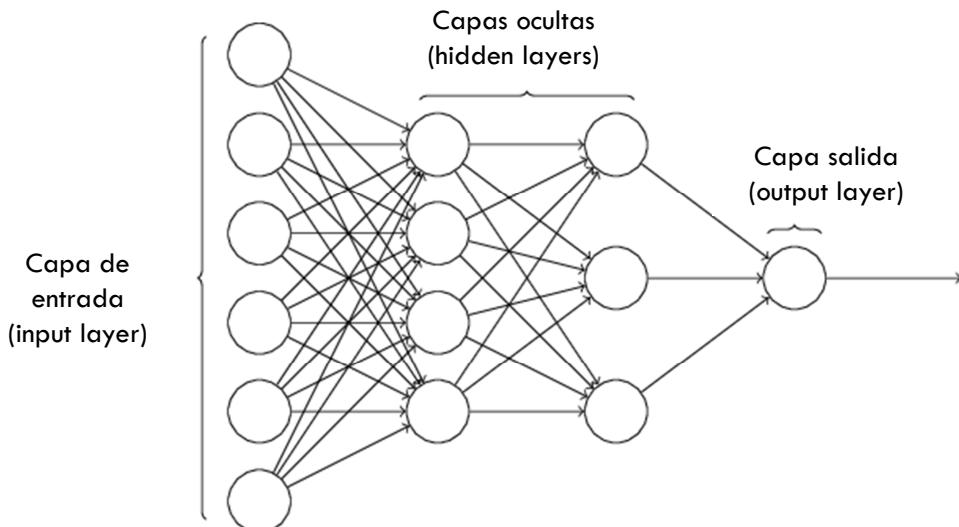
Ejemplos obtenidos con la herramienta

<https://cs.stanford.edu/people/karpathy/convnetjs/demo/classify2d.html>

REDES NEURONALES

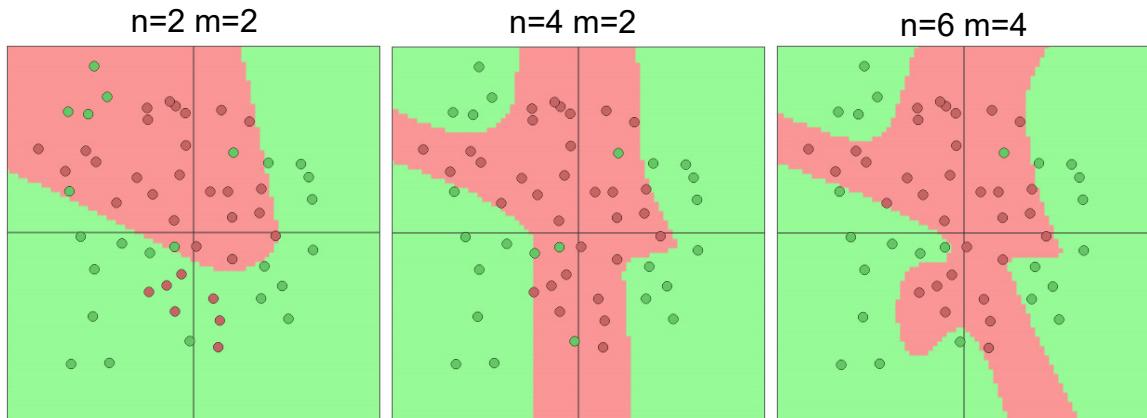
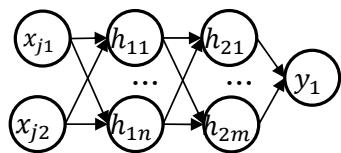
➤ Perceptrón multicapa: problemas no lineales

- Complejidad: añadir capas ocultas, modificar neuronas en cada capa



REDES NEURONALES

➤ Perceptrón multicapa

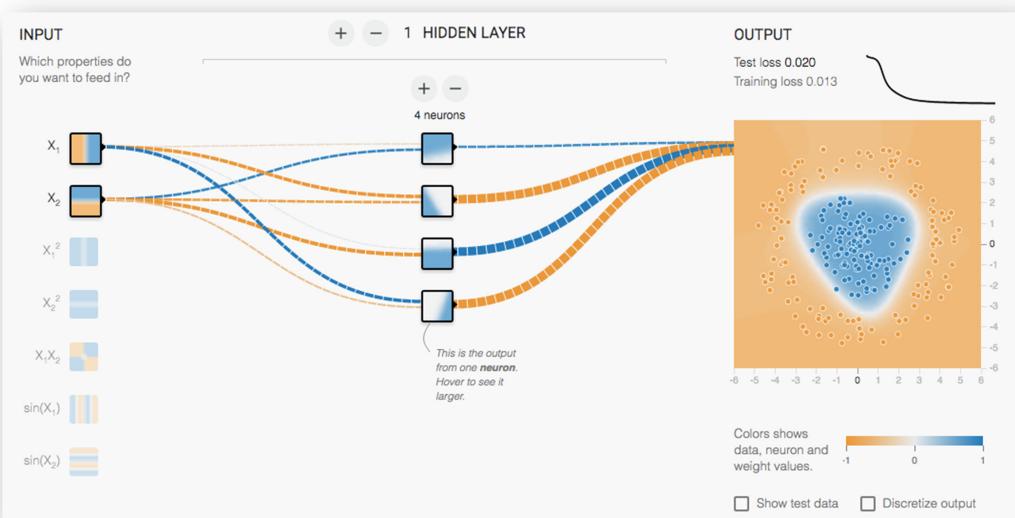


Ejemplos obtenidos con la herramienta

<https://cs.stanford.edu/people/karpathy/convnetjs/demo/classify2d.html>

REDES NEURONALES

➤ Perceptrón multicapa: ¿qué aprende cada neurona?



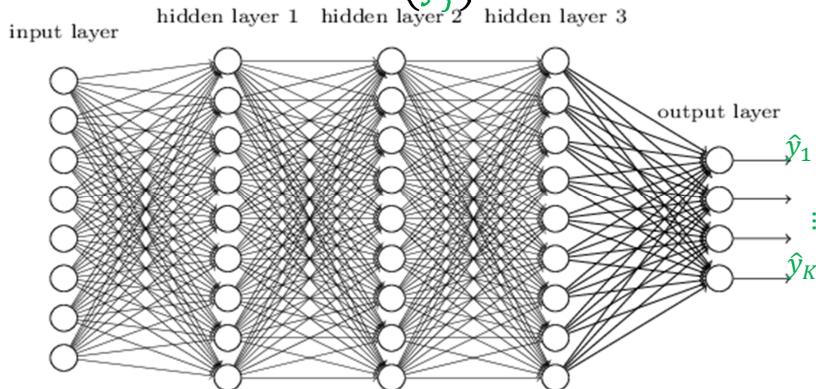
<http://playground.tensorflow.org>

1. ¿Qué entendemos por la *complejidad* de una red neuronal?
 - a) El número de entradas de la red
 - b) El número de salidas de la red
 - c) El número de patrones que puede analizar la red

2. En un perceptrón multicapa con una capa oculta, el número de entradas es 10 y el número de unidades en la capa oculta es 5. ¿Cuál es el número máximo de conexiones desde la capa entrada a la oculta?
 - a) 50
 - b) Menos de 50
 - c) Más de 50

REDES NEURONALES

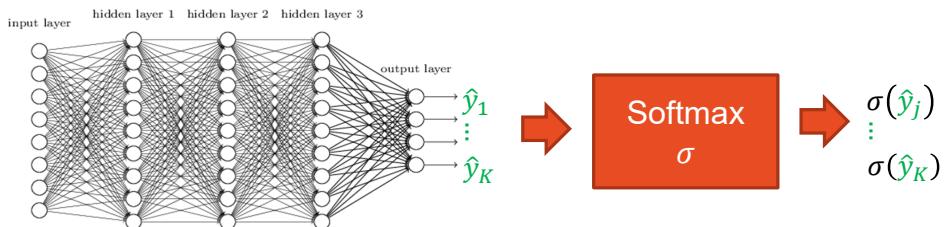
- Perceptrón multicapa: múltiples clases
 - Problema de clasificación con K clases
 - Se añaden K múltiple salidas \hat{y}_j de la red
 - Resultado clasificación con $\max(\hat{y}_j)$



REDES NEURONALES

➤ Perceptrón multicapa: múltiples clases

- ¿Cómo interpretar las salidas \hat{y}_j de un perceptrón multicapa en un problema de clasificación con K clases?



$$\sigma: \mathbb{R}^K \rightarrow [0,1]^K$$

$$\sigma(\hat{y}_j) = \frac{e^{\hat{y}_j}}{\sum_{j=1}^K e^{\hat{y}_j}}$$

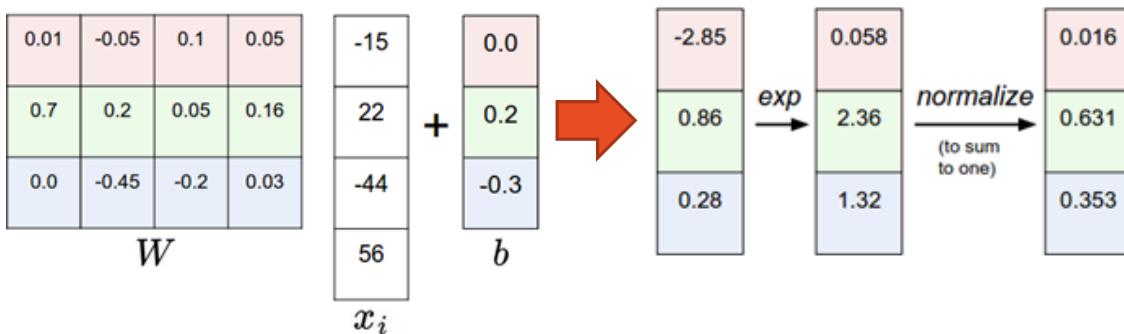
Selección de clase
elegir máximo valor $\sigma(\hat{y}_j)$
(índice correspondiente es la clase seleccionada)

REDES NEURONALES

➤ Perceptrón multicapa: múltiples clases

➤ Ejemplo clasificación con Softmax

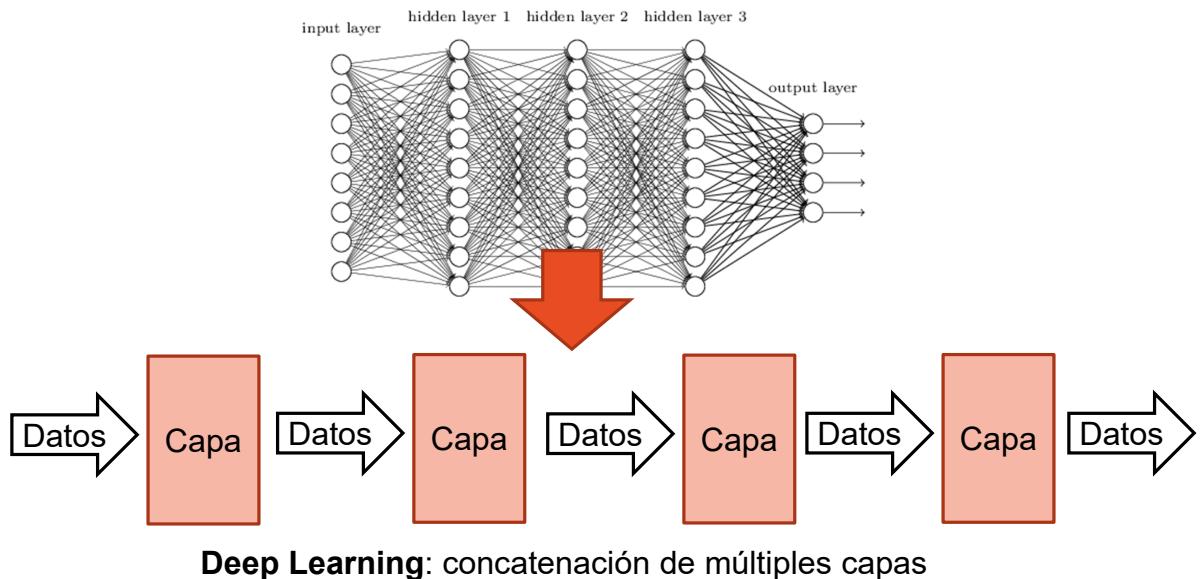
- 1 input de 4 dimensiones x_{ji} , 3 outputs \hat{y}_j para 3 clases
- Sin capa oculta
- Activación lineal (identidad)



Fuente: <http://cs231n.github.io/linear-classify/>

REDES NEURONALES

➤ Perceptrón multicapa: múltiples clases



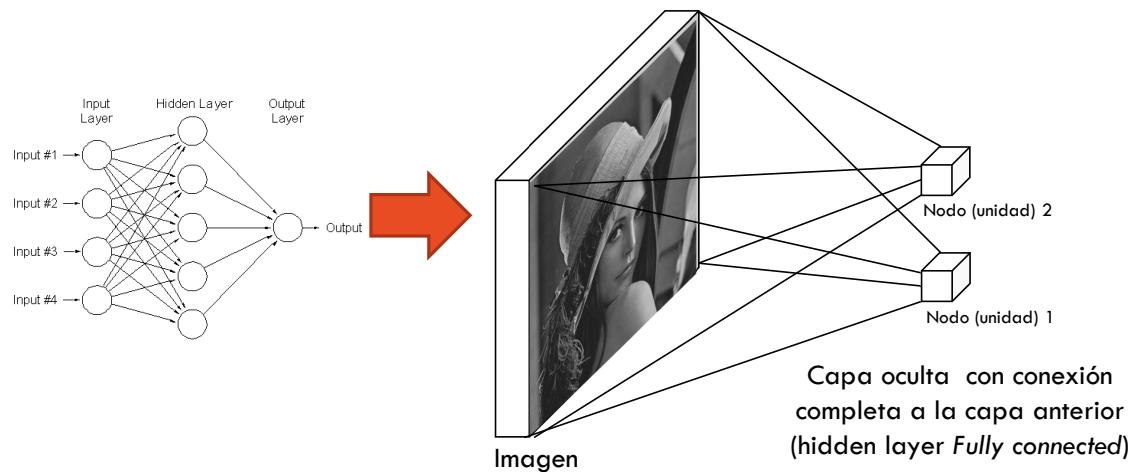
Agenda: Multimedia (imagen, video)

➤ Redes convolucionales: fundamentos

- Introducción
- Redes neuronales (repaso)
- Redes Neuronales Convolucionales
 - Idea
 - Arquitectura (global): red
 - Arquitectura (local): capas

CNNs: IDEAS

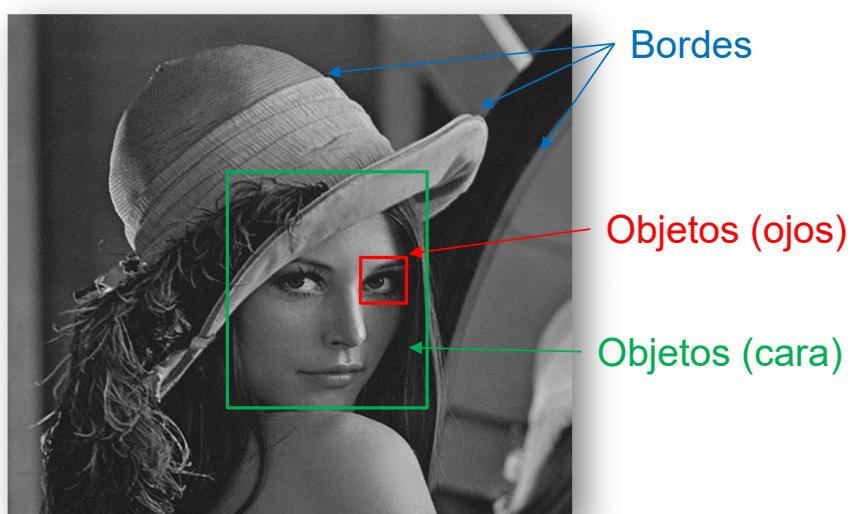
- Perceptrón multicapa aplicado a imágenes



Para analizar una imagen 200x200 con 40K unidades/nodos/neuronas en la capa oculta → se requieren $\sim 1.6 \cdot 10^9$ parámetros!

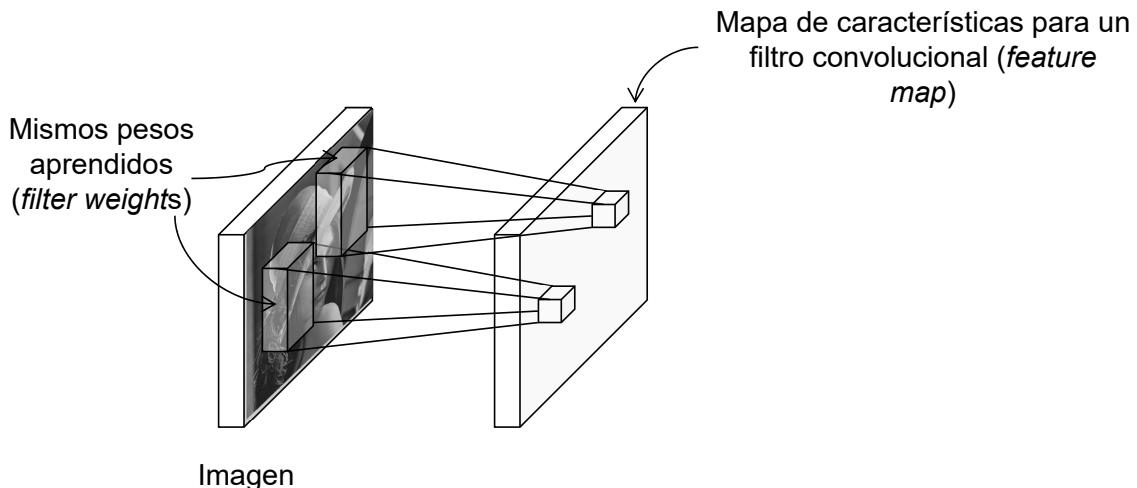
CNNs: IDEAS

- Pero las imágenes tienen cierta estructura local...



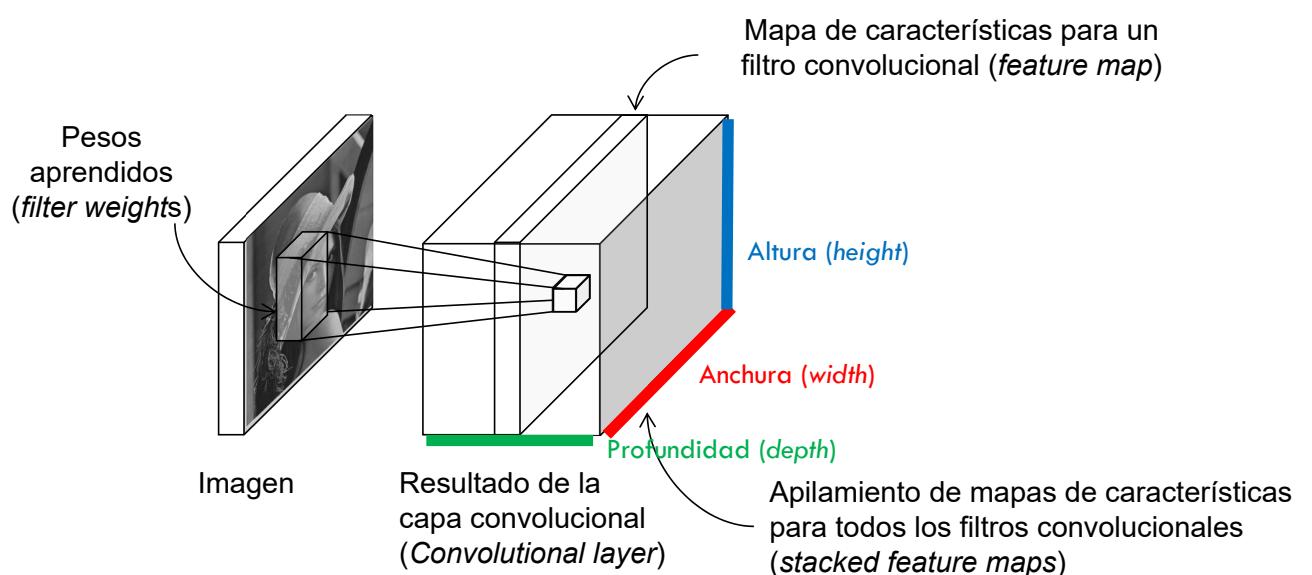
CNNs: IDEAS

➤ Convoluciones como extractores locales de características



CNNs: IDEAS

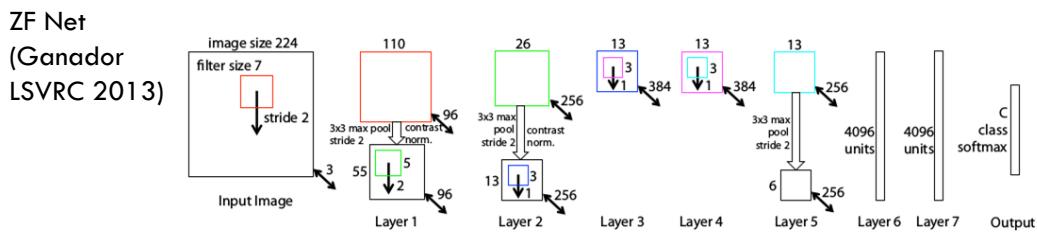
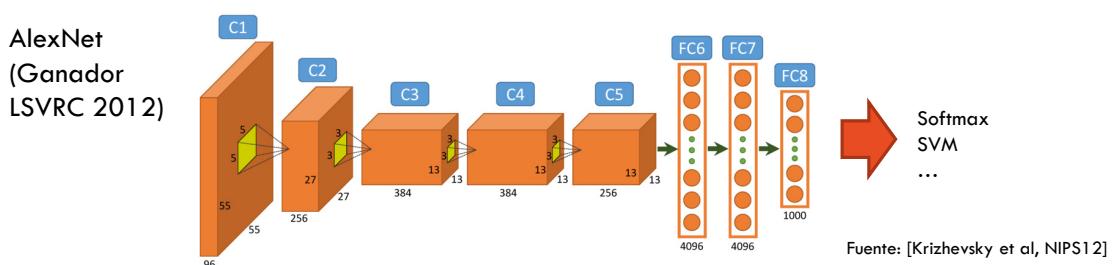
➤ Convoluciones como extractores locales de características



- Suponga que dispone de una base datos con imágenes RGB de dimensiones 300x300 pixeles.
- Para analizar cada una de estas imágenes se utiliza un perceptrón multicapa, cuya primera capa oculta tiene 100 unidades. ¿Cuántos parámetros tiene esta capa?
 - a) 9000001
 - b) 9000100
 - c) 27000001
 - d) 27000100

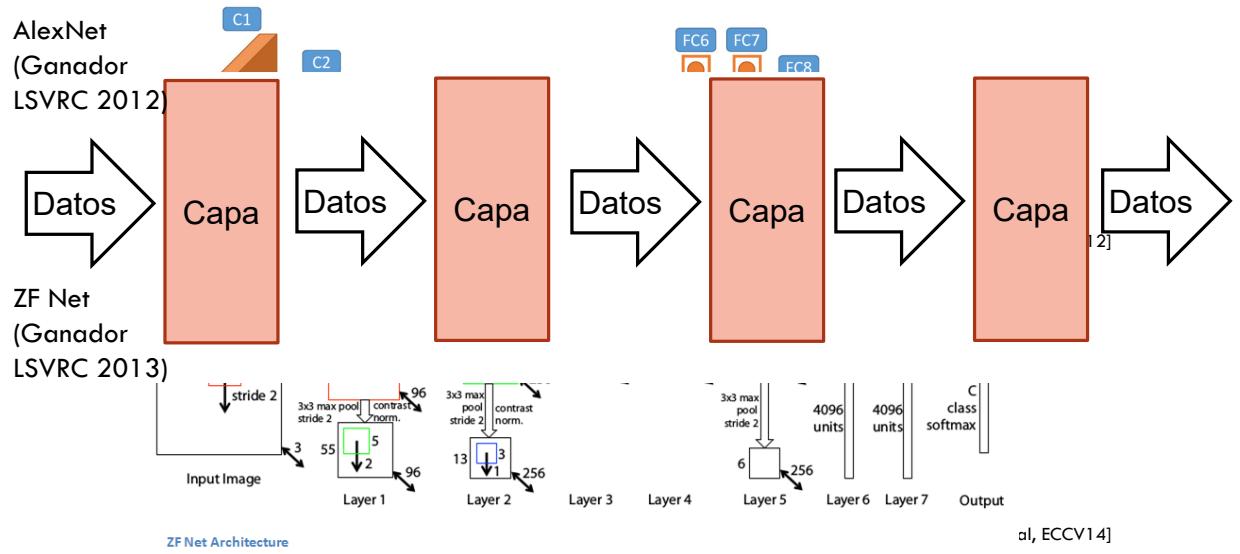
CNNs: ARQUITECTURA

- Red: secuenciación de capas en cascada



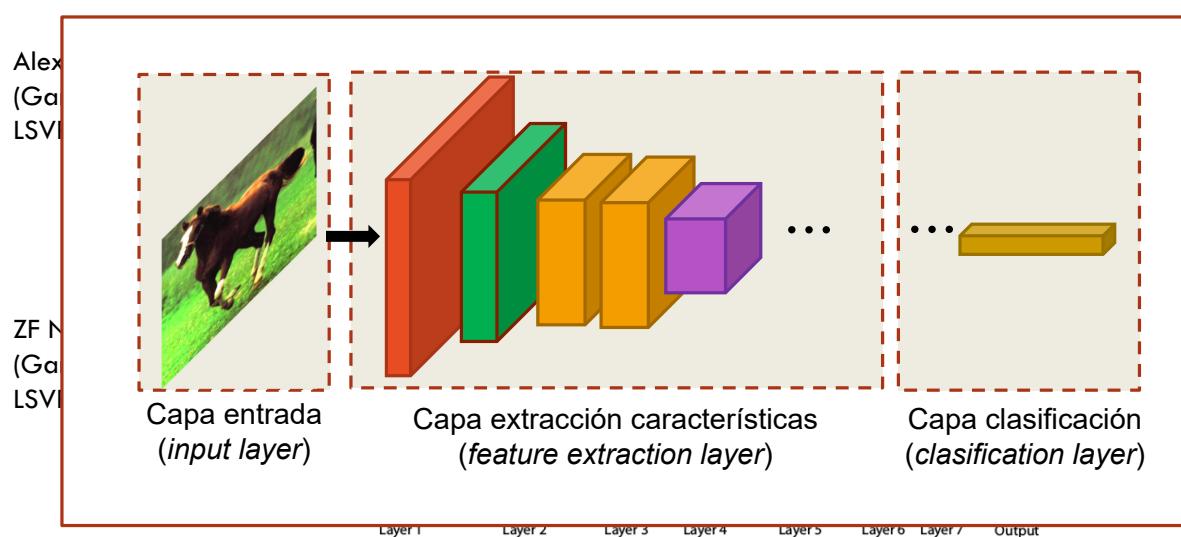
CNNs: ARQUITECTURA

- Red: secuenciación de capas en cascada



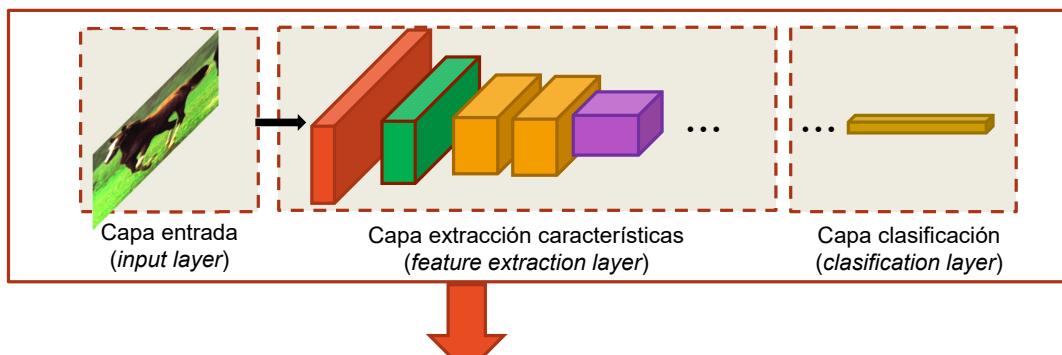
CNNs: ARQUITECTURA

- Red: secuenciación de capas en cascada



CNNs: ARQUITECTURA

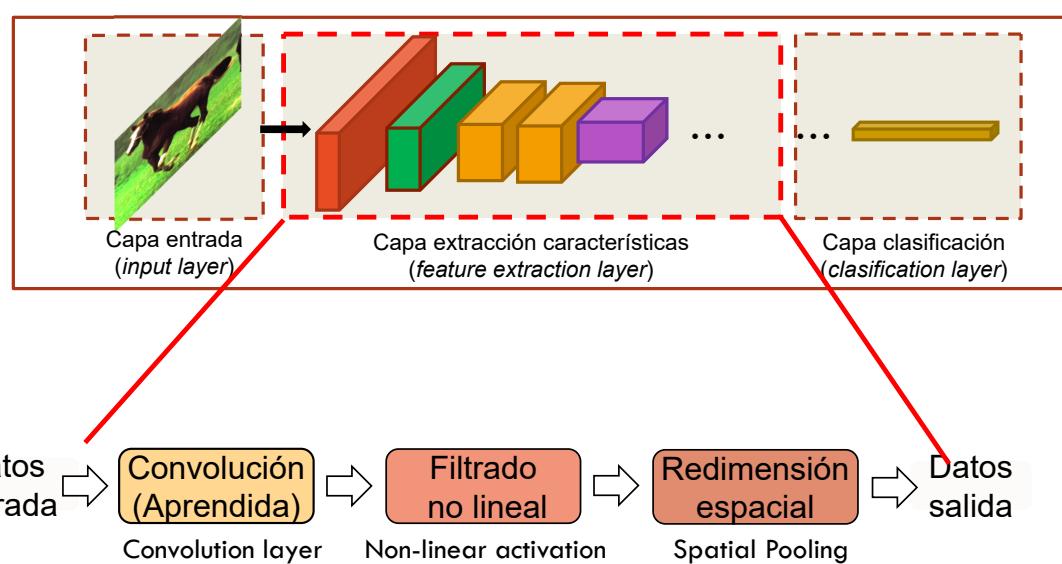
- Similitudes con la aproximación clásica...



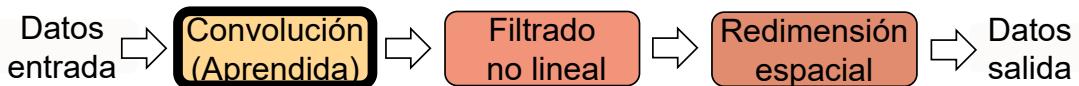
Extracción características			Clasificación	Referencia
1 ^a etapa	2 ^a etapa	3 ^a etapa		
SIFT	K-means	Pyramid Pooling	SVM	[Lazebnik et al, CVPR06]
SIFT	Fisher Vect.	Pooling	SVM	[Sanchez et al, IJCV13]

CNNs: CAPAS

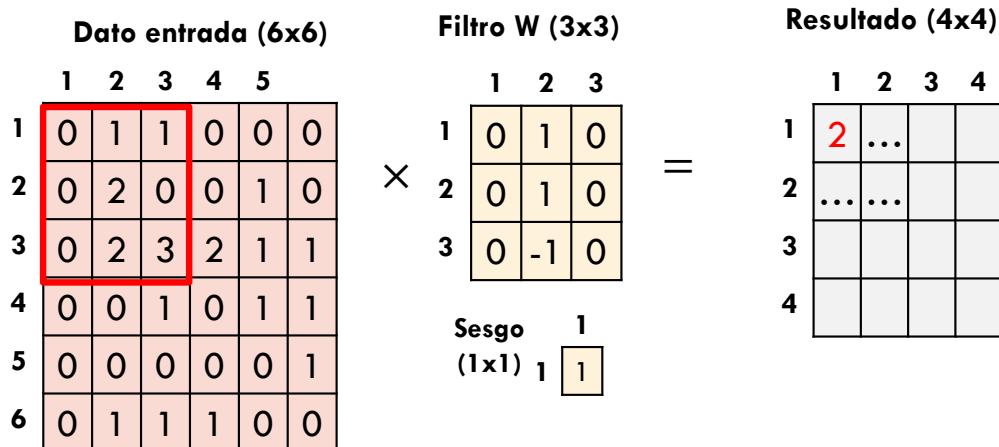
- Operaciones básicas en cada capa



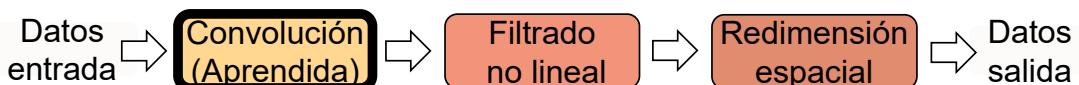
CNNs: CAPAS



➤ Capa convolucional

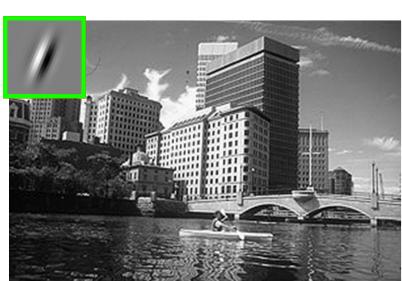


CNNs: CAPAS

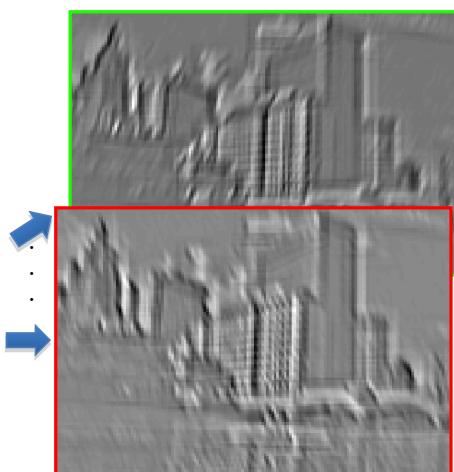


➤ Capa convolucional

- Imagen monocroma



Dato de entrada
(Input data)

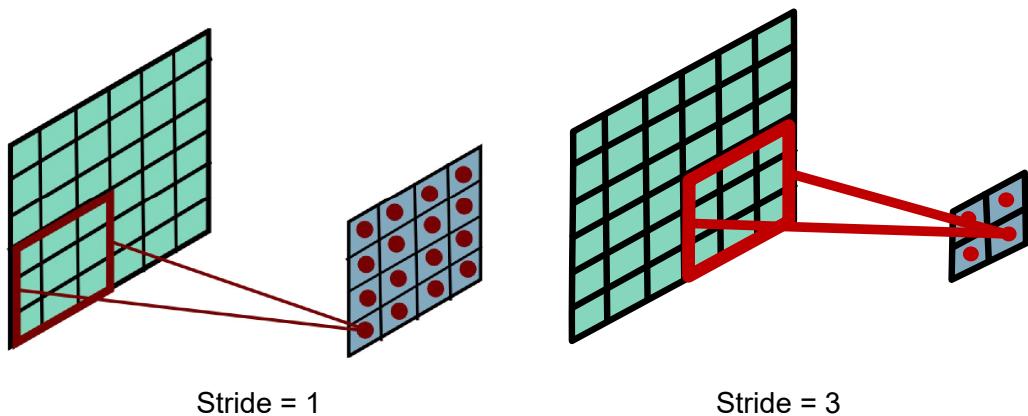


Mapas de características
(Feature Maps)

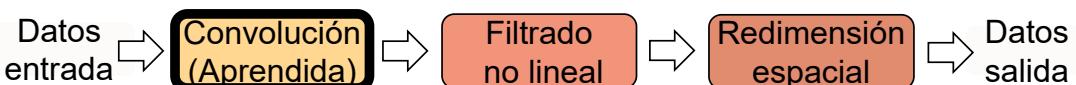
CNNs: CAPAS



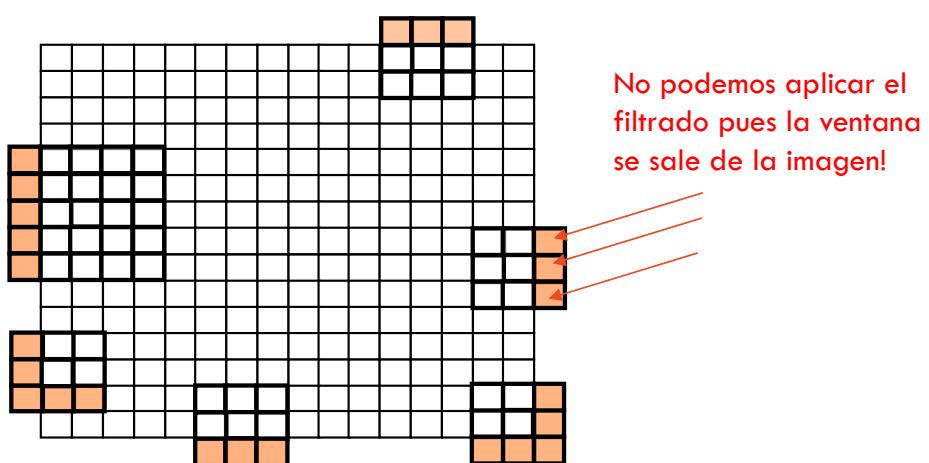
➤ Capa convolucional: paso (stride)



CNNs: CAPAS



➤ Capa convolucional: relleno (padding)

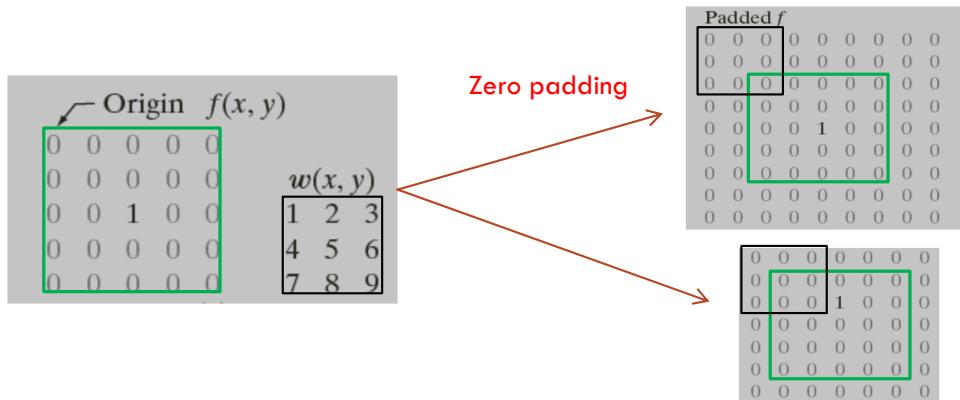


CNNs: CAPAS

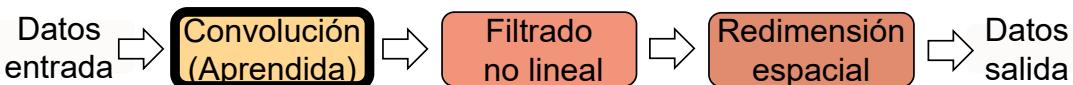


➤ Capa convolucional: relleno (padding)

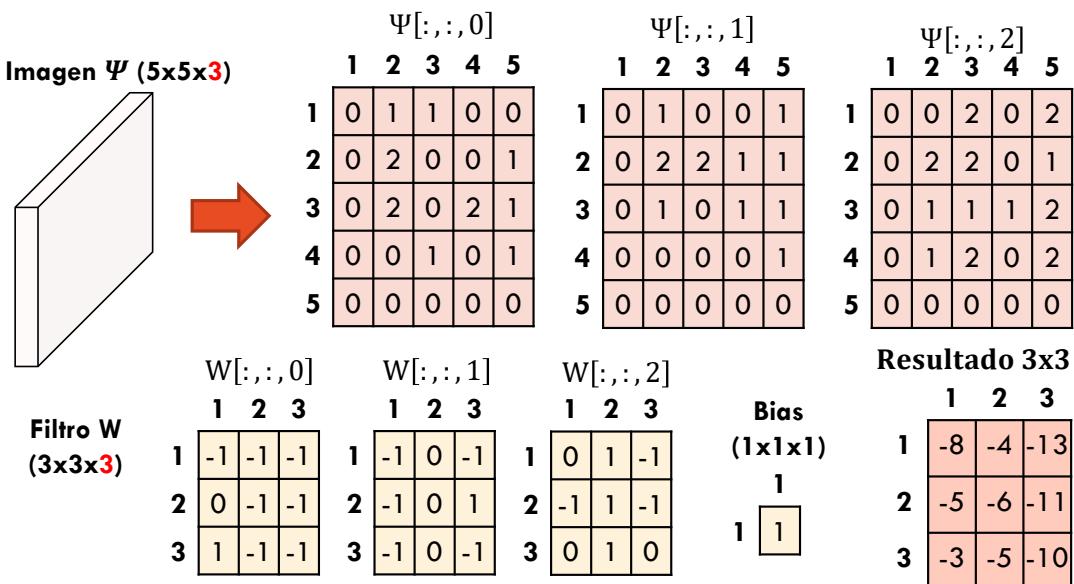
- Solución más utilizada: zero padding
- Alternativas: padding por replicación, circular,...



CNNs: CAPAS



➤ Capa convolucional: volumen de datos (e.g. imagen RBG)

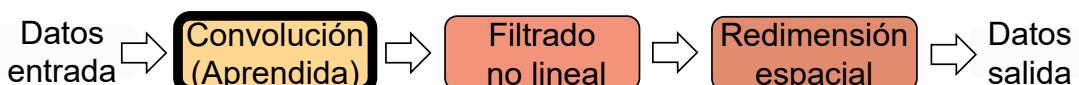


- Suponga que dispone de una base datos con imágenes RGB de dimensiones 300x300 pixeles.
 - Para analizar cada una de estas imágenes se utiliza un red neuronal convolucional, cuya primera capa oculta tiene 100 unidades/filtros de tamaño 5x5.
- ¿Cuántos parámetros tiene esta capa?

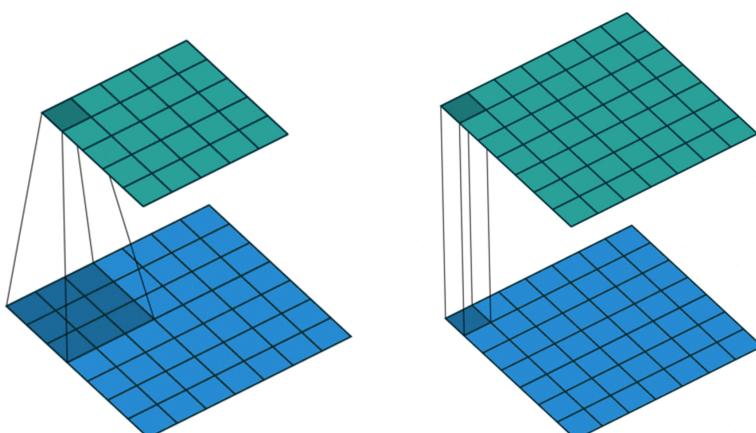
- a) 2501
- b) 2600
- c) 7500
- d) 7600

¿Y si las imágenes ahora son RGB de tamaño 3000x3000?

CNNs: CAPAS



- Capa convolucional: convoluciones 1x1 ¿redundancia?



Filtro 3x3

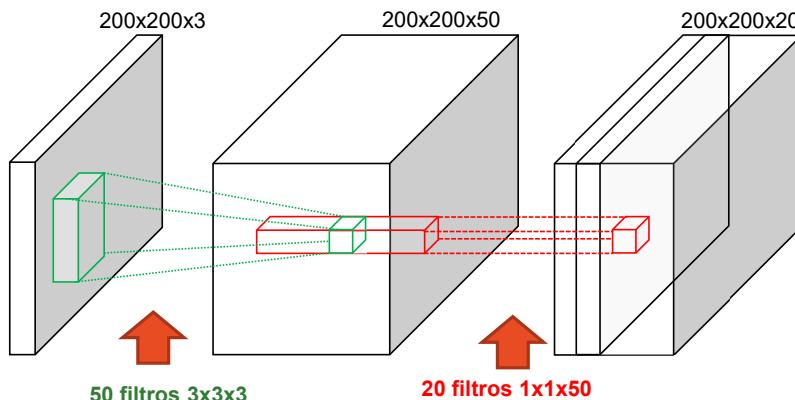
Filtro 1x1

CNNs: CAPAS

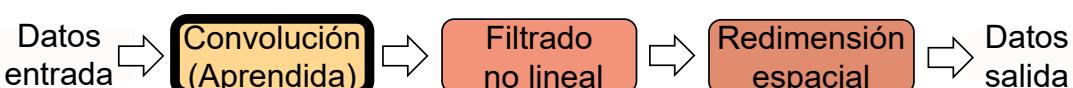


➤ Capa convolucional: convoluciones 1x1 ¿redundancia?

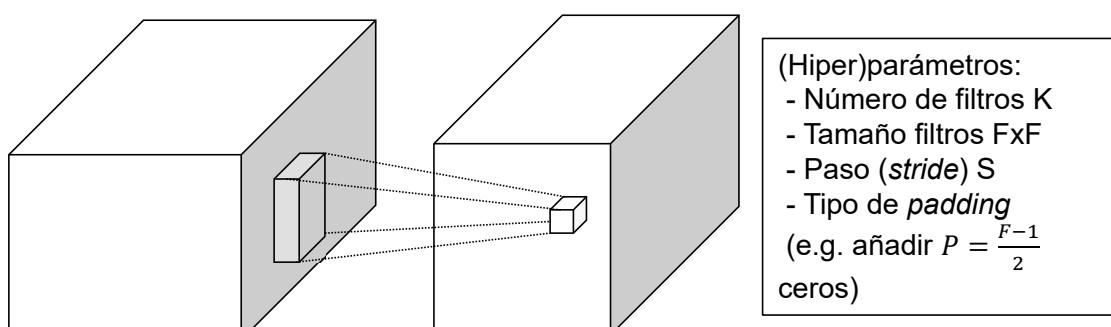
- Utilidad como método para reducir dimensionalidad



CNNs: CAPAS



➤ Capa convolucional: caso general



Volumen de datos
de entrada
 $W_1 \times H_1 \times D_1$

Volumen de datos de
salida $W_2 \times H_2 \times D_2$

(Hiper)parámetros:
- Número de filtros K
- Tamaño filtros $F \times F$
- Paso (stride) S
- Tipo de padding
(e.g. añadir $P = \frac{F-1}{2}$
ceros)

$$W_2 = (W_1 - F + 2P)/S + 1$$
$$H_2 = (H_1 - F + 2P)/S + 1$$
$$D_2 = K$$

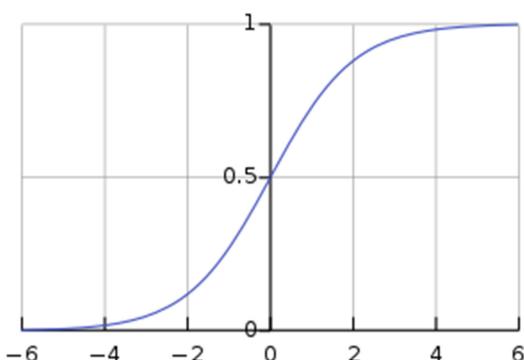
CNNs: CAPAS



➤ Permite resolver problemas no-lineales

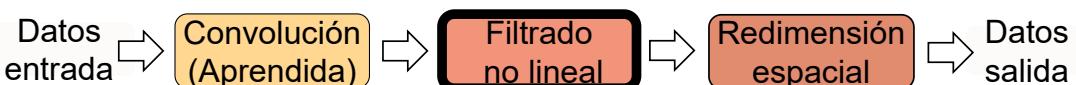
Sigmoid

$$f(z) = \frac{1}{1 + e^{-z}}$$



- Rango de salida limitado
- Opción preferente para aprender funciones “lógicas” con entradas binarias
- No está centrada en cero
- No recomendable su uso para redes de imágenes
- No hay derivada si saturamos

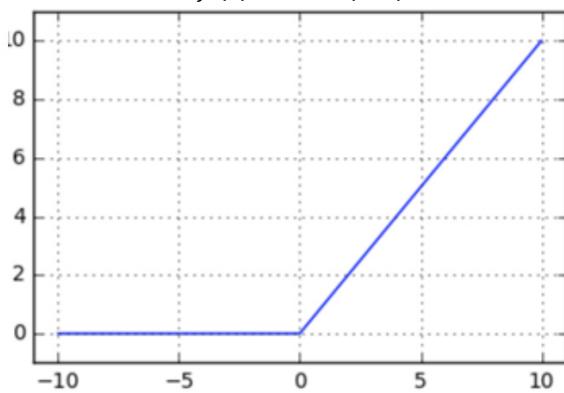
CNNs: CAPAS



➤ Permite resolver problemas no-lineales

ReLU (Rectified Linear Unit)

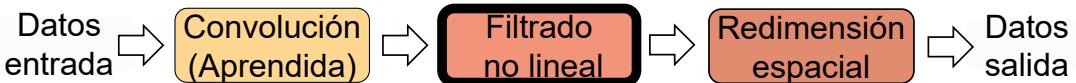
$$f(z) = \max(0, z)$$



- Gradiente más alto si $z > 0$. Entrenamiento más rápido.
- Reducción del número de parámetros (*sparsity*) pues muchos parámetros valen cero
- No útil para variables de control
- No útil para funciones lógicas
- No está centrado en cero

Fuente: [Krizhevsky et al, NIPS12]

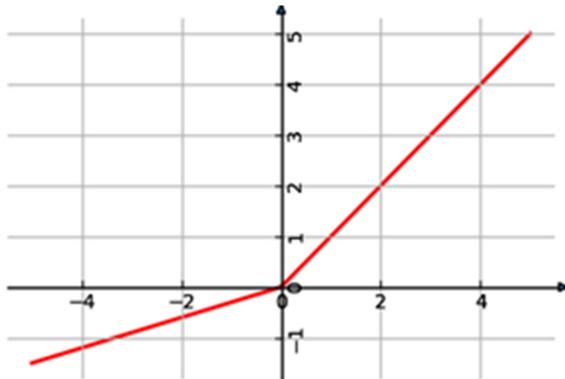
CNNs: CAPAS



- Permite resolver problemas no-lineales

Leaky ReLU ($\alpha = 0.01$)

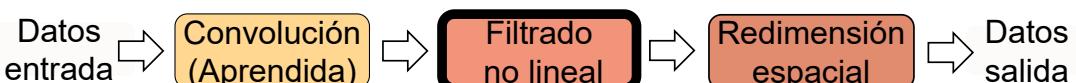
$$f(z) = \max(\alpha \cdot z, z)$$



- No se satura
- Convergencia más rápida que sigmoide/tanh sobre datos de imágenes (~ 6x)
- No tiene gradiente nulo

Fuente: [He et al, Corr15]

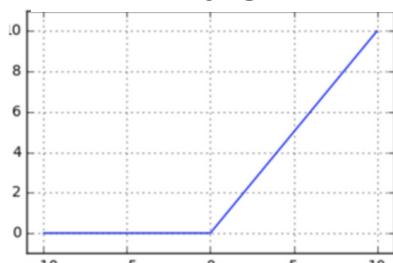
CNNs: CAPAS



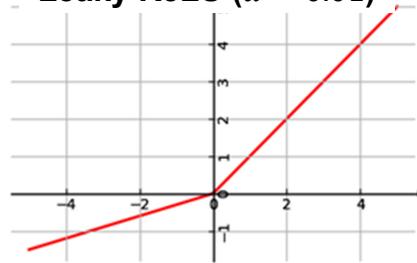
- Permite resolver problemas no-lineales: recomendaciones

- Utilizar **ReLU** en capas próximas a la imagen de entrada
- Utilizar **Leaky ReLU** (o similares) en el resto de capas
- Utilizar sigmoide para funciones “lógicas” (e.g. recomendación, ...)

ReLU

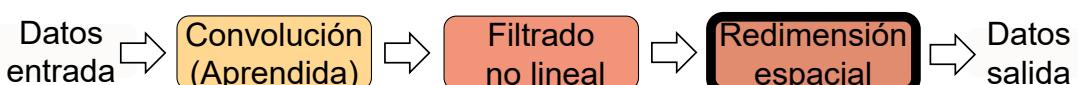


Leaky ReLU ($\alpha = 0.01$)

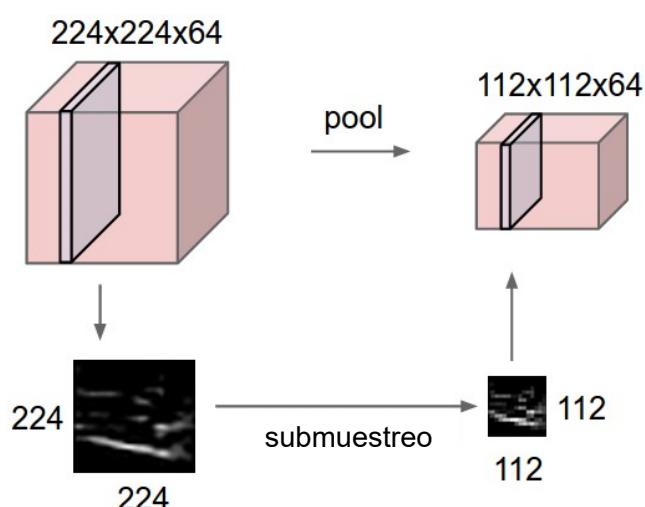


- Se desea diseñar una red neuronal que realice una clasificación binaria para reconocer manzanas ($y=1$) vs. naranjas ($y=0$). ¿Cuál de estas funciones de activación recomendaría usar para la capa de salida?
- (a) ReLU
 - (b) Tanh
 - (c) Leaky ReLU
 - (d) Sigmoid

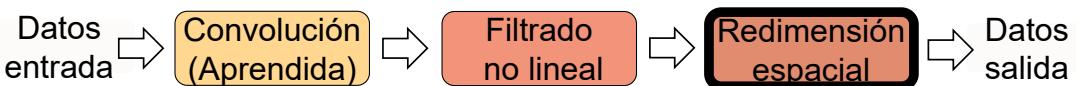
CNNs: CAPAS



- Redimensión espacial (*spatial pooling*)

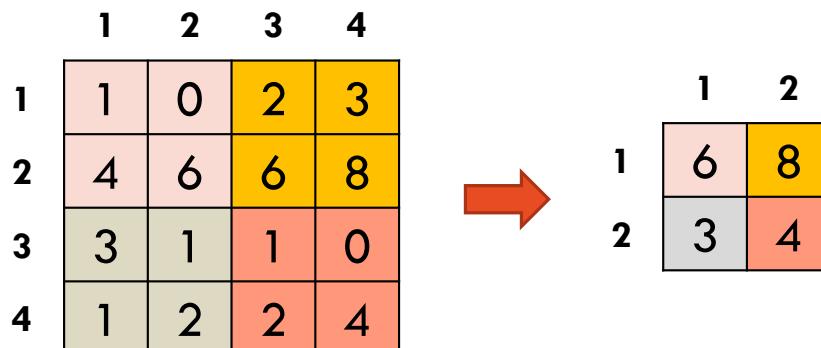


CNNs: CAPAS

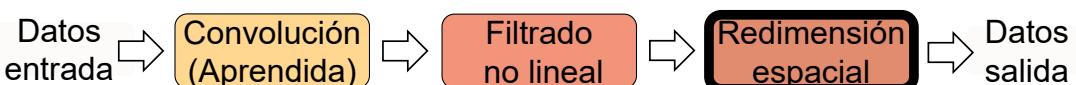


➤ Redimensión espacial (*spatial pooling*)

- Selección del máximo (*max pooling*)

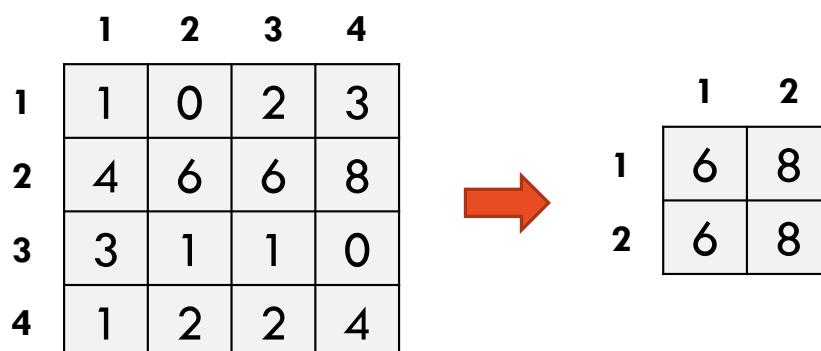


CNNs: CAPAS

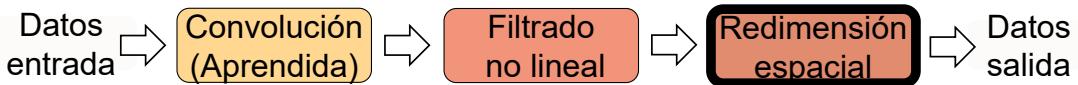


➤ Redimensión espacial (*spatial pooling*)

- Selección del máximo (*max pooling*)

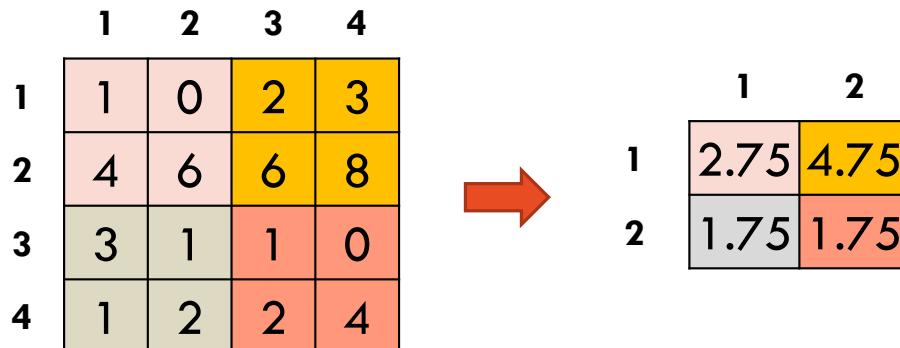


CNNs: CAPAS

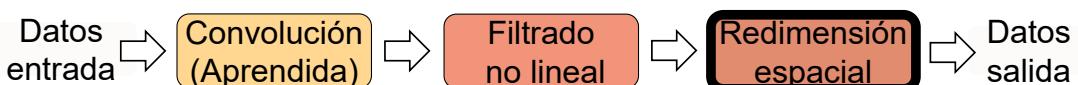


➤ Redimensión espacial (*spatial pooling*)

- Selección de la media (*average pooling*)

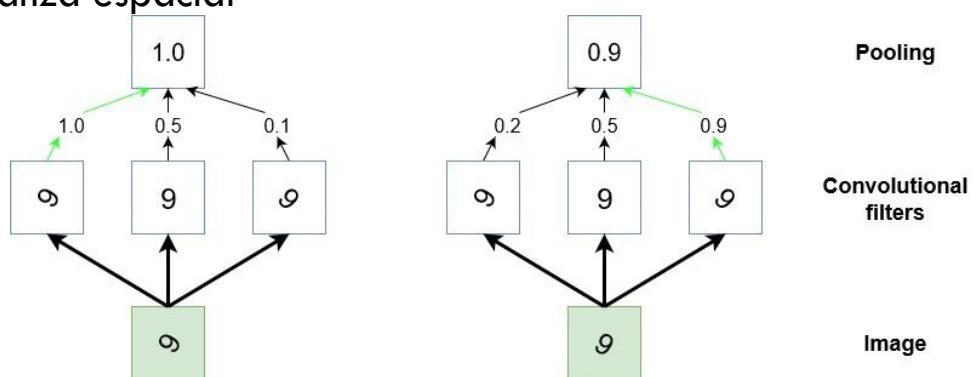


CNNs: CAPAS



➤ Redimensión espacial (*spatial pooling*)

- Reduce la cantidad de parámetros en la red
- Invarianza espacial

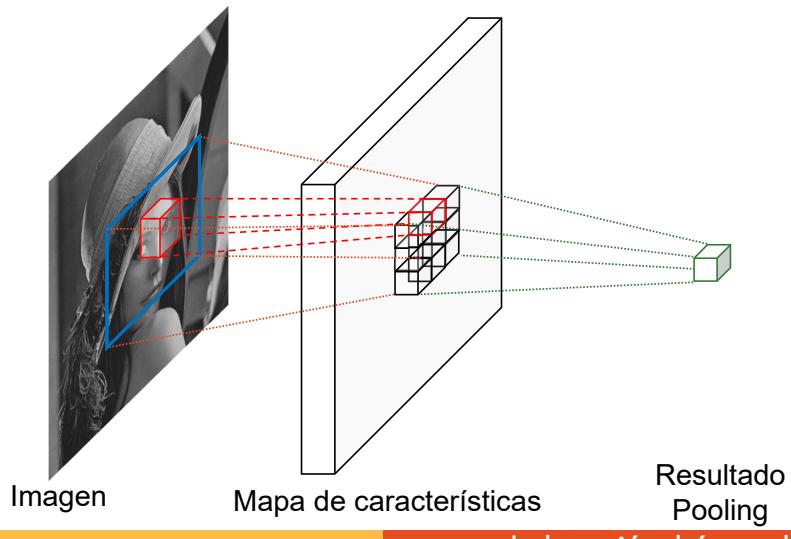


Fuente: <http://adventuresinmachinelearning.com/convolutional-neural-networks-tutorial-tensorflow/>

CNNs: CAPAS

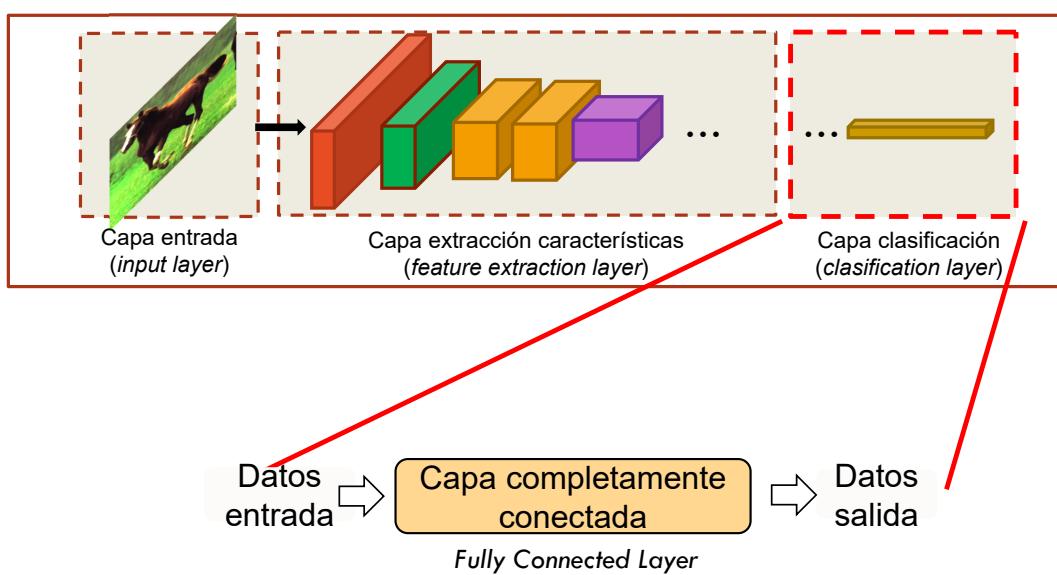


- Redimensión espacial (*spatial pooling*): extensión



CNNs: CAPAS

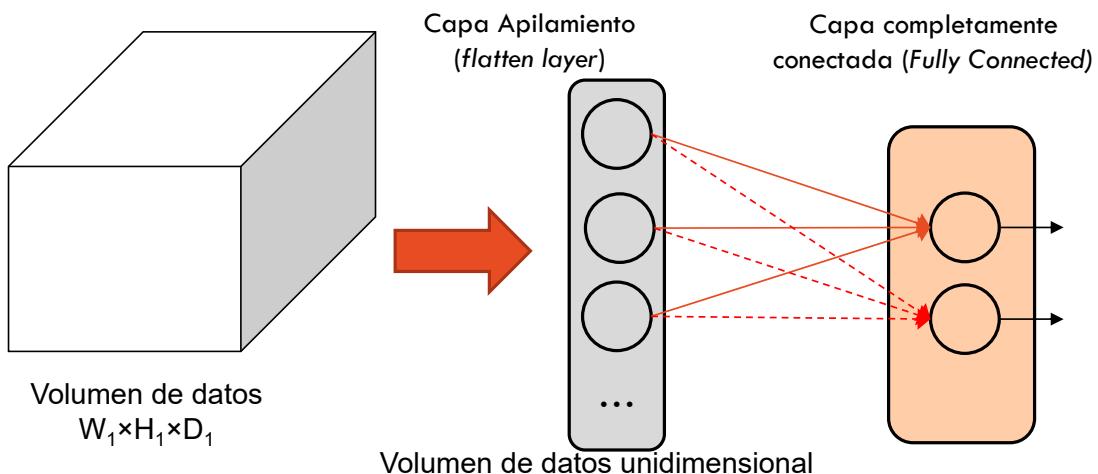
- Operaciones básicas en cada capa



- En una capa de pooling de una red neuronal convolucional, si tenemos un volumen de entrada de $32 \times 32 \times 16$ y le aplicamos *max pooling* con $\text{stride}=2$ y tamaño del filtro 2×2 ¿Cuáles son las dimensiones del volumen de datos a la salida de esta capa?
- $15 \times 15 \times 16$
 - $16 \times 16 \times 16$
 - $32 \times 32 \times 8$
 - $16 \times 16 \times 8$

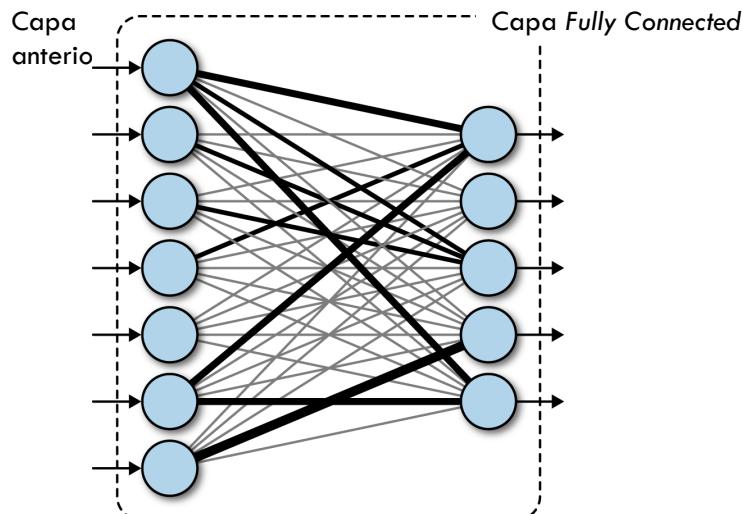
CNNs: CAPAS

- Capa completamente conectada (*Fully Connected layer*)
- El volumen de datos se “convierten” a un vector
 - Después se añaden capas equivalentes al perceptrón multicapa



CNNs: CAPAS

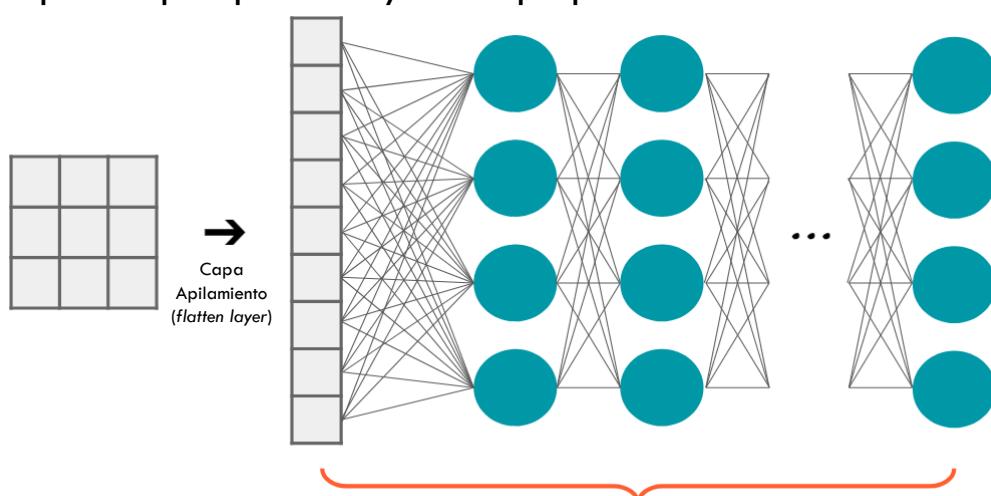
- Capa completamente conectada (*Fully Connected layer*)
 - No todos los pesos tienen valores significativos



Fuente: <https://learning.oreilly.com>

CNNs: CAPAS

- Capa completamente conectada (*Fully Connected layer*)
 - Múltiples capas para mayor complejidad



Capas completamente conectadas (Fully Connected) Fuente: <https://towardsdatascience.com/>

RESUMEN

➤ Redes Neuronales

➤ Ventajas

- Estructura flexible y general para multitud de problemas
- Se puede incrementar la complejidad añadiendo más capas/unidades

➤ Desventajas

- Difícil análisis teórico (sensible a mínimos locales)
- Requiere una gran cantidad de datos de entrenamiento y potencia de cálculo
- Variadas posibilidades de diseño: arquitecturas, parámetros, funciones de coste...

➤ Redes Neuronales Convolucionales

- Explotan jerarquía de la información visual
- Menos parámetros que un perceptrón multicapa
- Tres capas principales: convolución, pooling y *fully connected*

REFERENCIAS

➤ Básico:

- I. Goodfellow, Y. Bengio, and A. Courville, "Deep Learning", MIT Press, 2016 Sec 6 y 9 (disponible gratuitamente en <http://www.deeplearningbook.org/>)

➤ Adicional:

- He et al, Coor15] He, K., Zhang, X., Ren, S., and Sun, J.. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification.arXiv preprint arXiv:1502.01852, 2015
- [Krizhevsky et al, NIPS12] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).
- [Lazebnik et al, CVPR06] Lazebnik, S., Schmid, C., & Ponce, J. (2006, June). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In 2006 IEEE CVPR (Vol. 2, pp. 2169-2178). IEEE.
- [Lee et al., ICML09] Lee, H., Grosse, R., Ranganath, R., & Ng, A. Y. (2009, June). Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In Proceedings of the 26th annual international conference on machine learning (pp. 609-616). ACM.
- [Russakovsky et al, IJCV15] Olga Russakovsky*, Jia Deng*, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg and Li Fei-Fei ImageNet Large Scale Visual Recognition Challenge. IJCV, 2015
- [Sanchez et al, IJCV13] Sánchez, J., Perronnin, F., Mensink, T., & Verbeek, J. (2013). Image classification with the fisher vector: Theory and practice. International journal of computer vision, 105(3), 222-245.
- [Zeiler et al, ECCV14] Zeiler, M. D., & Fergus, R. (2014, September). Visualizing and understanding convolutional networks. In ECCV (pp. 818-833). Springer