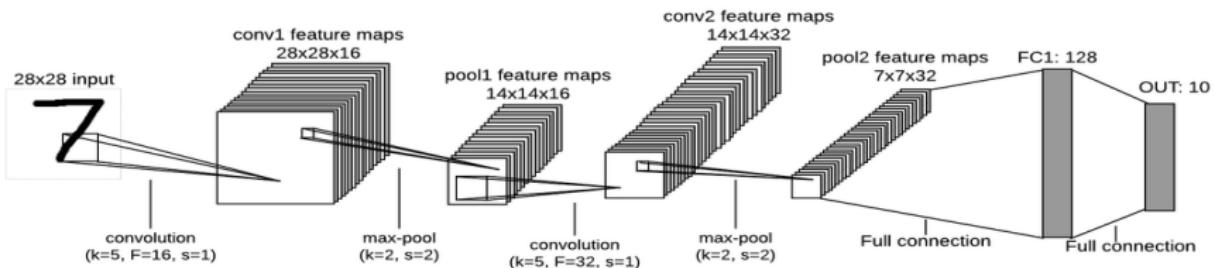


Redes convolucionales y su uso en reconocimiento de imágenes

Julio Waissman

11 de junio de 2019

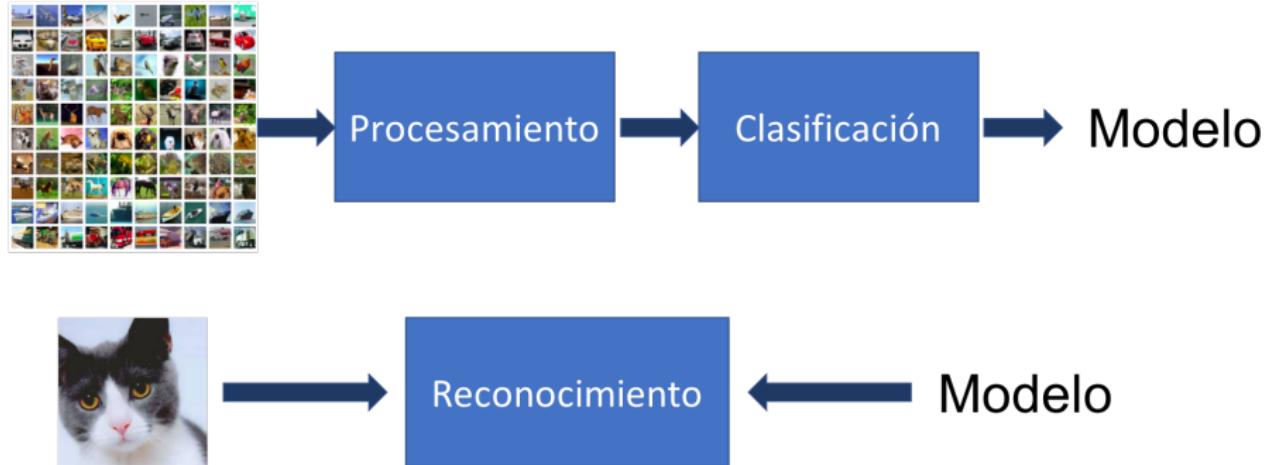




Plan de la presentación

- ① ¿Cómo se ha hecho típicamente el procesamiento de imágenes?
- ② ¿Qué es el aprendizaje profundo?
- ③ ¿Cómo son las redes neuronales convolucionales?
- ④ ¿Cómo funciona esto?
- ⑤ Consideraciones finales

¿Qué es el reconocimiento de imágenes



Imagen

Imagen original

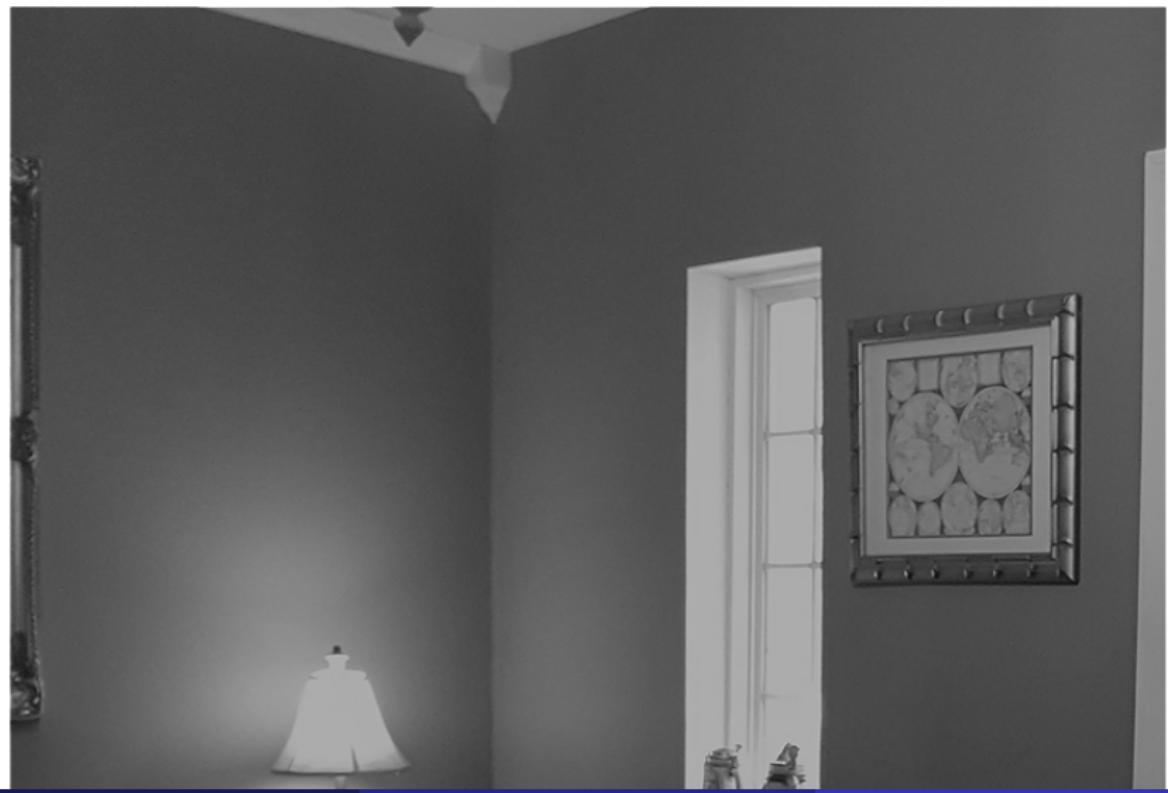


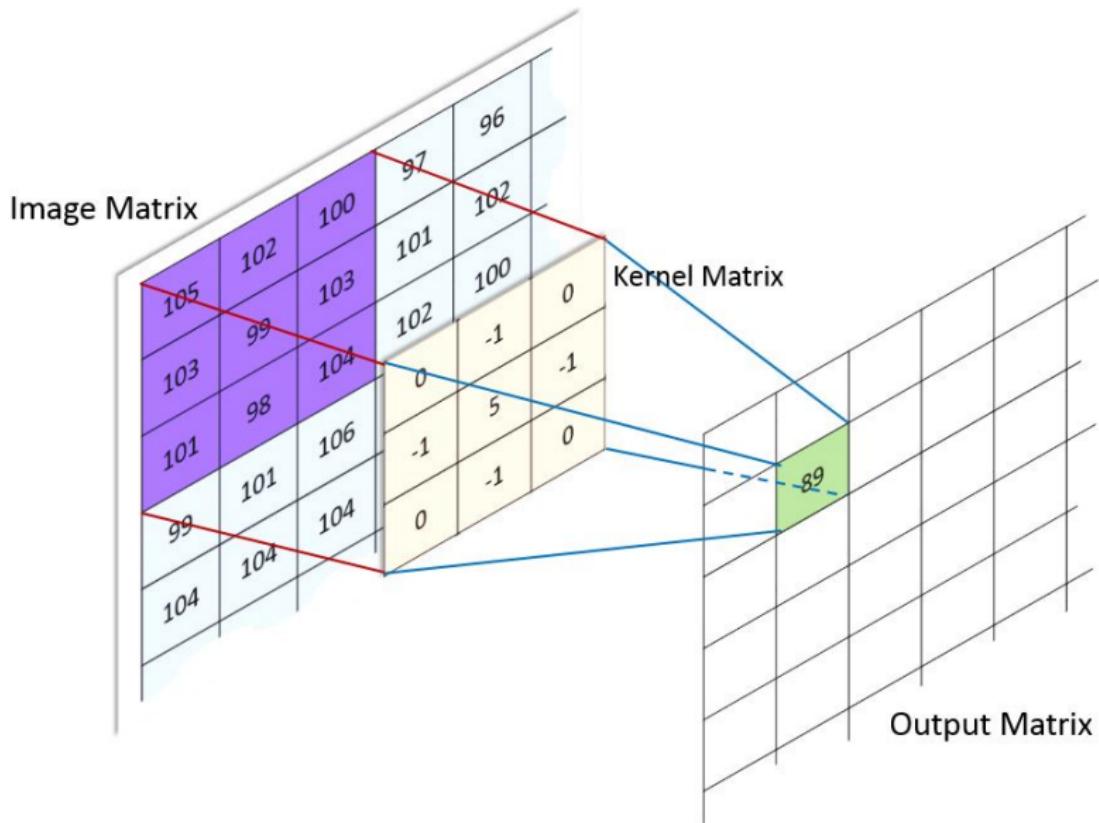
Imagen como matriz de números

Detalle

105	102	100	97	96	
103	99	103	101	102	
101	98	104	102	100	
99	101	106	104	99	
104	104	104	100	98	



Filtro convolucional



Filtro convolucional

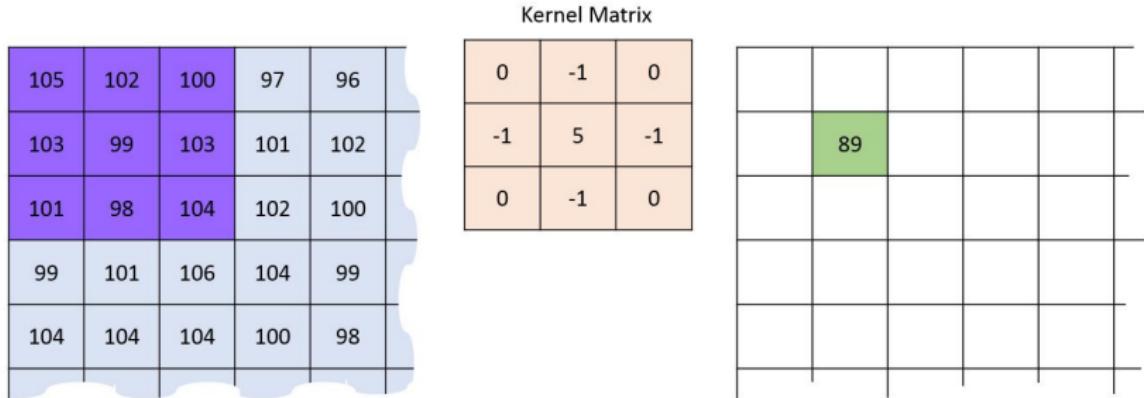


Image Matrix

$$\begin{aligned} & 105 * 0 + 102 * -1 + 100 * 0 \\ & + 103 * -1 + 99 * 5 + 103 * -1 \\ & + 101 * 0 + 98 * -1 + 104 * 0 = 89 \end{aligned}$$

Output Matrix

Filtro convolucional

105	102	100	97	96	
103	99	103	101	102	
101	98	104	102	100	
99	101	106	104	99	
104	104	104	100	98	

Image Matrix

Kernel Matrix

0	-1	0
-1	5	-1
0	-1	0

		89	111	

Output Matrix

$$\begin{aligned} & 102 * 0 + 100 * -1 + 97 * 0 \\ & + 99 * -1 + 103 * 5 + 101 * -1 \\ & + 98 * 0 + 104 * -1 + 102 * 0 = 111 \end{aligned}$$

Filtro convolucional

0	0	0	0	0	0
0	105	102	100	97	96
0	103	99	103	101	102
0	101	98	104	102	100
0	99	101	106	104	99
0	104	104	104	100	98

Image Matrix

Kernel Matrix

0	-1	0
-1	5	-1
0	-1	0

210	89	111		

Output Matrix

$$\begin{aligned} & 0 * 0 + 105 * -1 + 102 * 0 \\ & + 0 * -1 + 103 * 5 + 99 * -1 \\ & + 0 * 0 + 101 * -1 + 98 * 0 = 210 \end{aligned}$$

Filtro convolucional

0	0	0	0	0	0
0	105	102	100	97	96
0	103	99	103	101	102
0	101	98	104	102	100
0	99	101	106	104	99
0	104	104	104	100	98

Image Matrix

Kernel Matrix

0	-1	0
-1	5	-1
0	-1	0

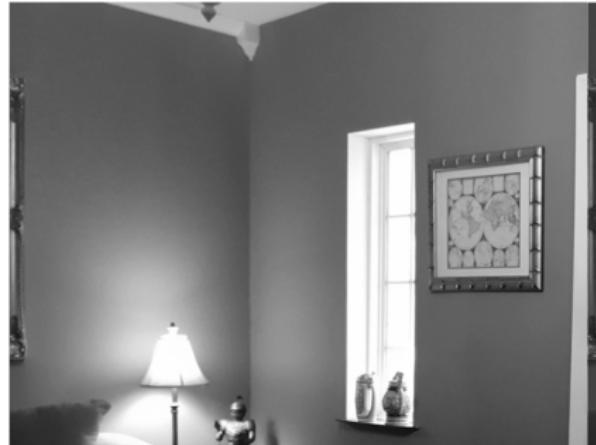
320				
210	89	111		

Output Matrix

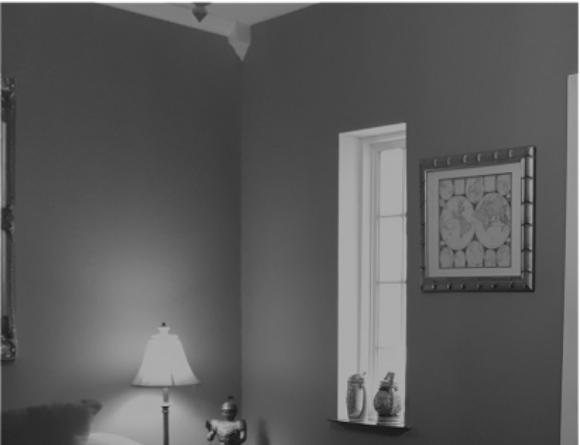
$$\begin{aligned} & 0 * 0 + 0 * -1 + 0 * 0 \\ & + 0 * -1 + 105 * 5 + 102 * -1 \\ & + 0 * 0 + 103 * -1 + 99 * 0 = 320 \end{aligned}$$

Resultado

Original



Filtrada



Los filtros pueden ser extremos

$$Kernel = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$



Y el procesamiento de imágenes de hace así. . .

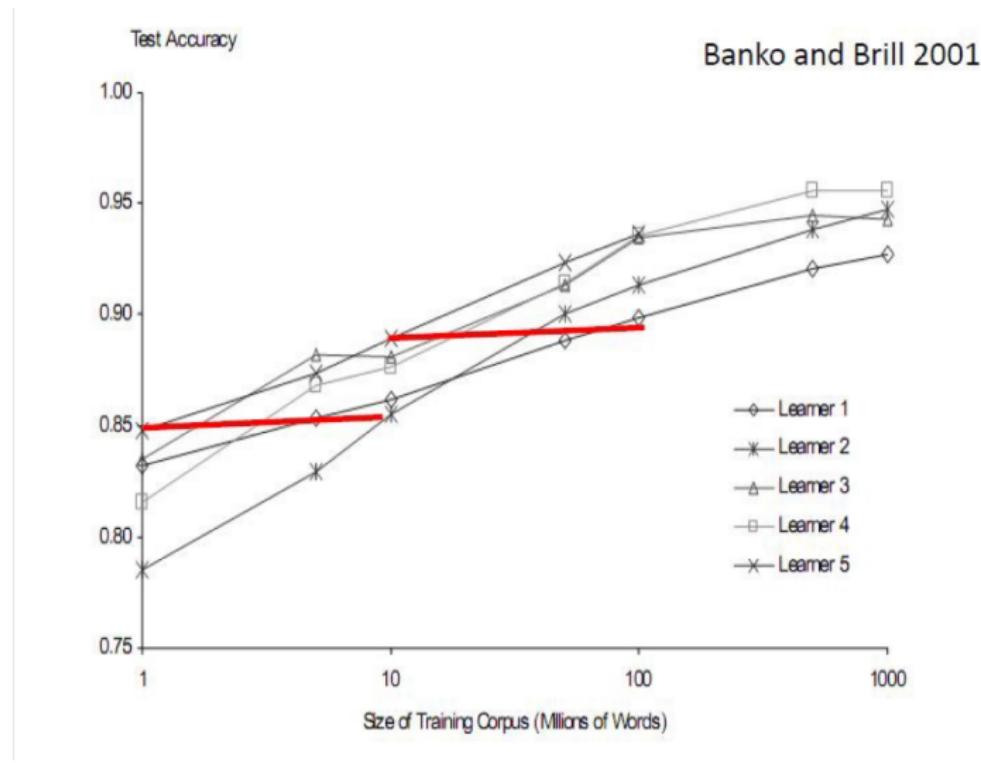
- Primero se seleccionan muchos filtros, dependiendo de lo que queremos
- Se ajustan los filtros para representar las características principales
- Se pueden aplicar filtros a las imágenes filtradas
- Se utilizan otras técnicas (submuestreo, histograma, . . .)
- Se convierte la imagen en un enorme vector de características

¿Y el reconocimiento?

- El reconocimiento implica encontrar una función $h : \mathbb{R}^n \rightarrow C$
- Se supone que debe de existir una función $f : \mathbb{R}^n \rightarrow C$
- Solo se puede aprender si establecemos **una función de pérdida o error**
- Se entrena a partir de un conjunto de datos de los cuales no tenemos información estadística *a priori*
- Lo único que se puede medir, es que tan **aproximadamente probablemente correcto** es el modelo
- Vamos a hablar solo de los modelos neuronales

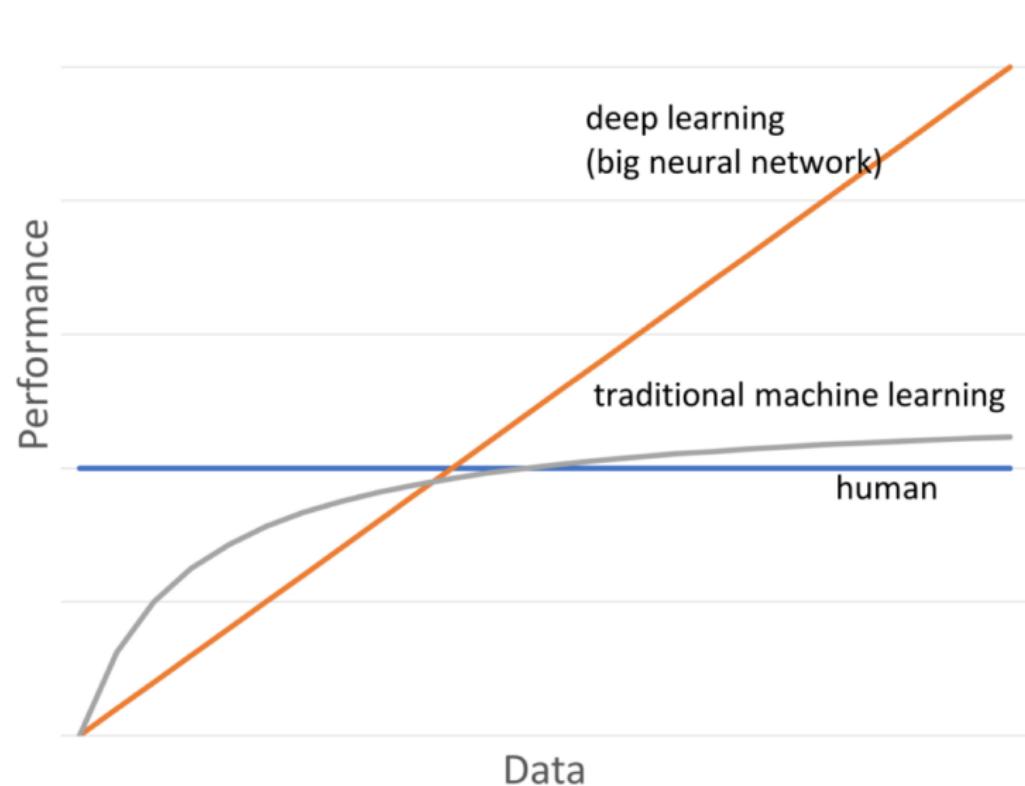
Aprendizaje profundo (DL)

Motivación



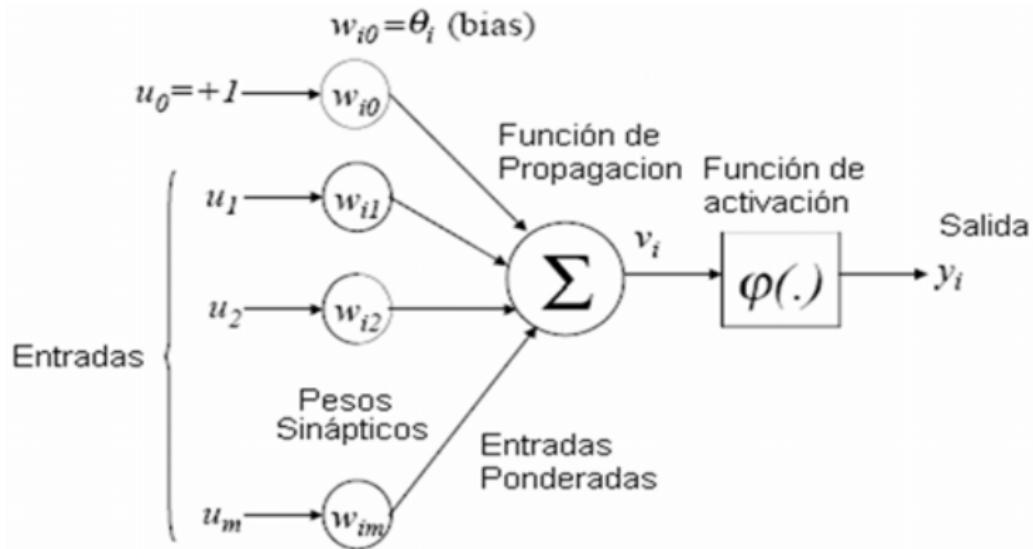
Aprendizaje profundo (DL)

Motivación



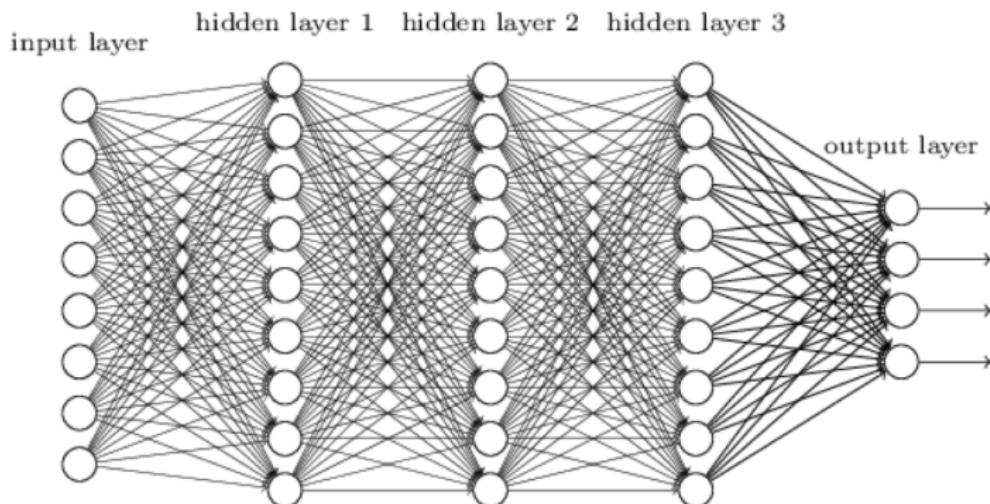
Arquitecturas hacia adelante

Unidad o Neurona



Arquitecturas hacia adelante

Estructura general

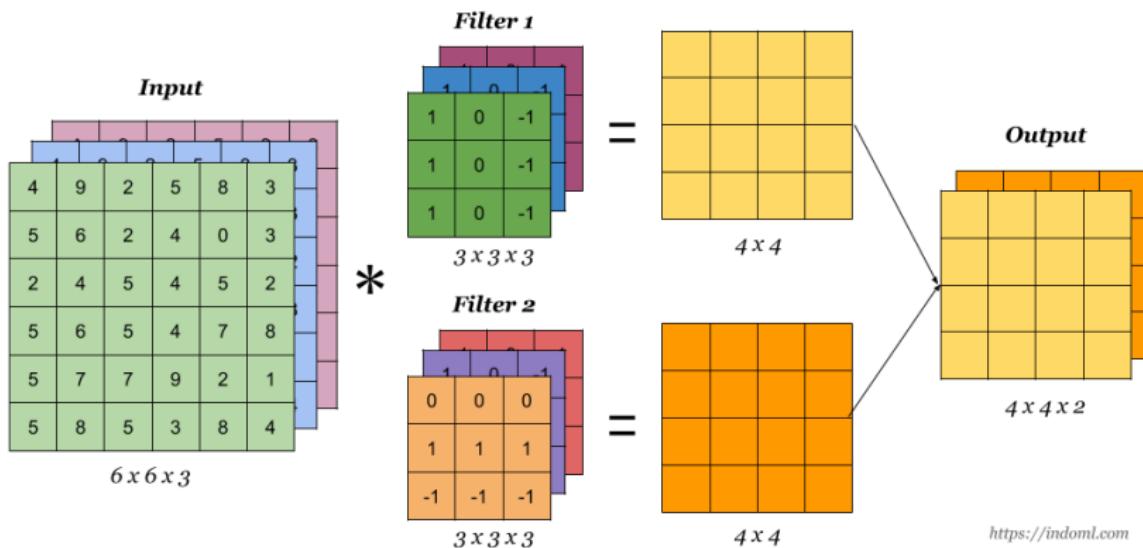


Pero el procesamiento es lo más complicado ¿Qué no?

- El procesamiento es la etapa limitante
- Depende del conocimiento experto que se tenga del problema a tratar
- Es necesario ajustarlo y corroborarlo con muchas imágenes, con expertos en tratamiento de imágenes y con expertos en el problema particular
- ¡Pero tenemos muchísimos datos!
- ¿Y que tal si aprendemos el procesamiento junto con el clasificador?

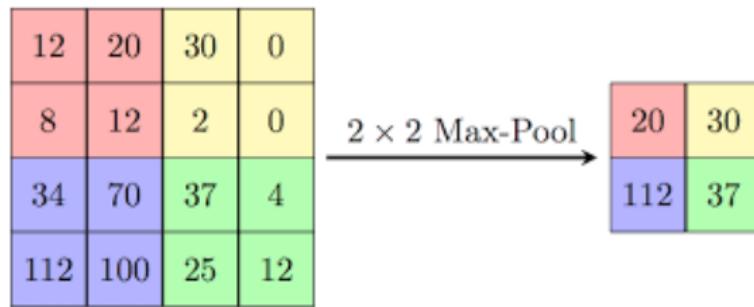
Redes convolucionales (CNN)

Capa convolucional



Redes convolucionales (CNN)

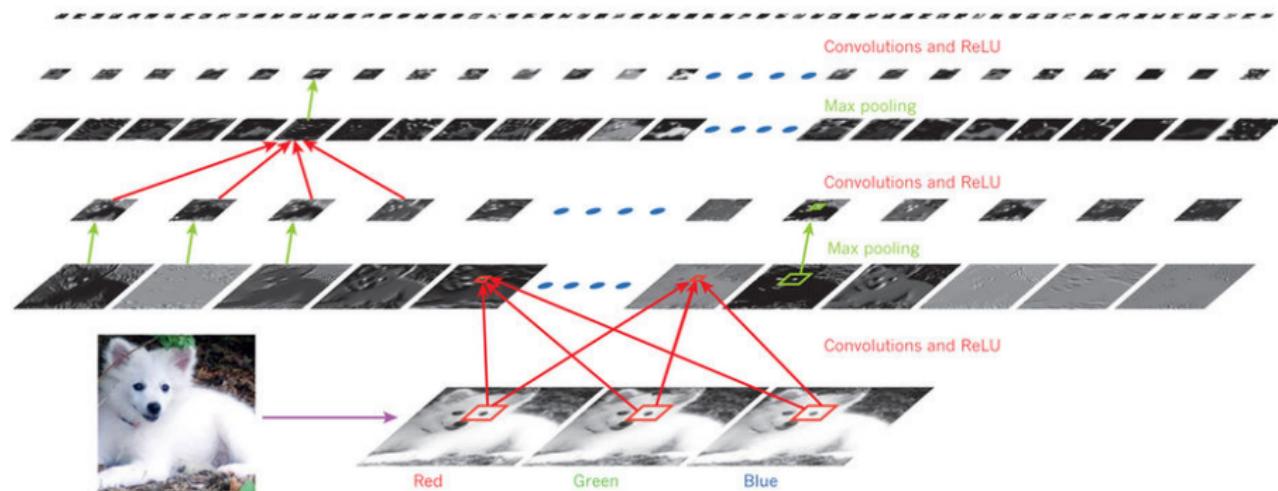
Submuestreo



Redes convolucionales (CNN)

Arquitectura general

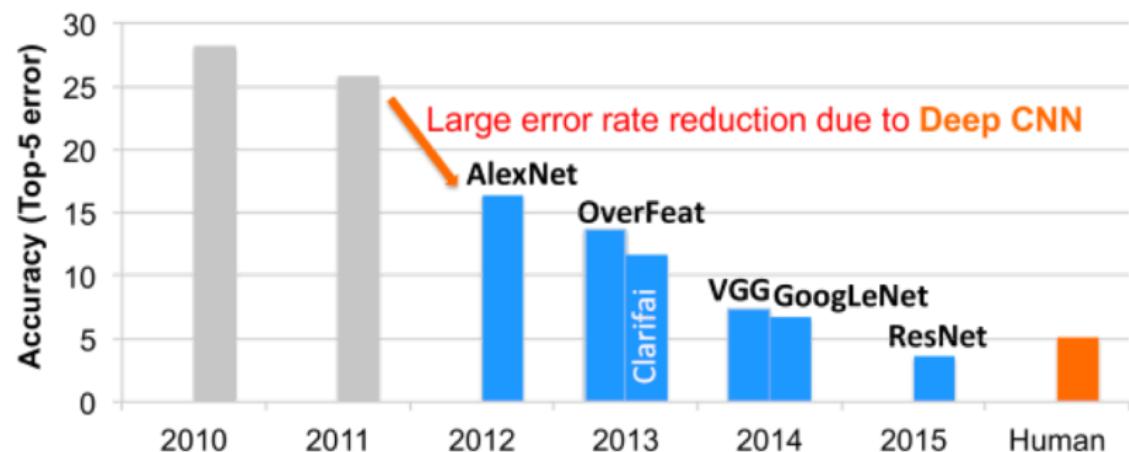
Samoyed (16); Papillon (5.7); Pomeranian (2.7); Arctic fox (1.0); Eskimo dog (0.6); white wolf (0.4); Siberian husky (0.4)



Redes convolucionales (CNN)

Desempeño en tratamiento de imágenes

Resultados aplicados al conjunto de datos *ImageNet*

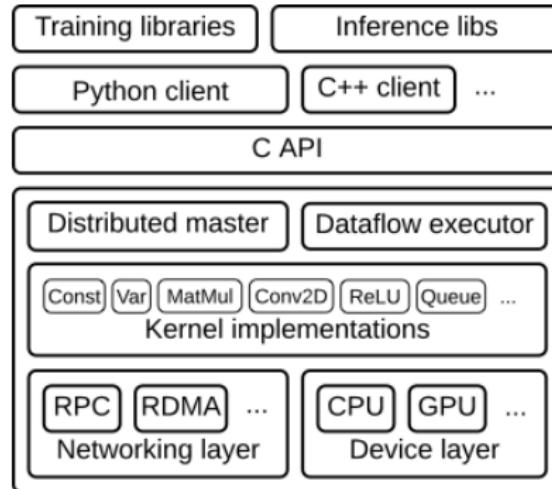




- Desarrollado por *Google* para el proyecto *Google Brain*
- Biblioteca para cómputo numérico basado en graficas de flujo de datos
- Los Vértices representan operaciones y las aristas tensores
- Se adapta a diferentes plataformas

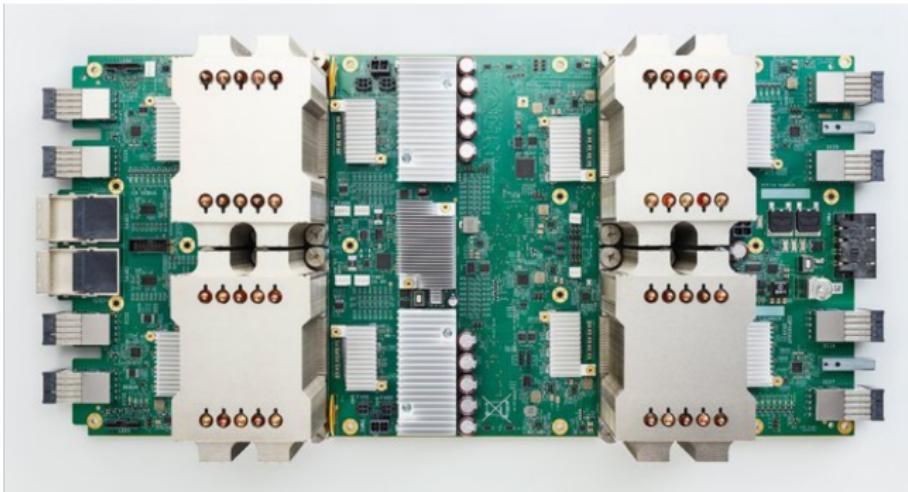
Tensorflow

Arquitectura

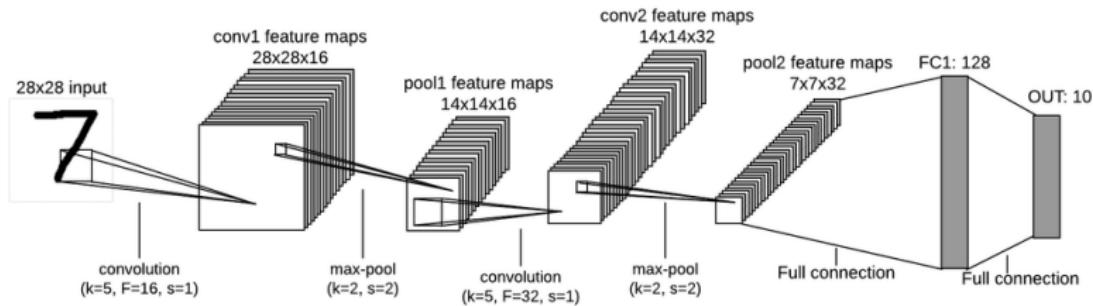


Tensor Processing Unit (TPU)

Unidad de procesamiento específica para TensorFlow



¿Y entonces que vamos a hacer?



Recapitulando

En este taller quisimos poner de manifiesto que:

- No es difícil hacer una CNN con las herramientas existentes
- Las ideas generales del aprendizaje profundo no son esotéricas ni mágicas
- Se requiere de expertos en aprendizaje máquina para un desarrollo profesional
- Es posible seguir y entender lo que realice el equipo de desarrollo
- Además de computación se requieren bases en matemáticas
- Sin aplicación, es un juguetote

... y colorín colorado ...

Muchas gracias por su atención