

Aprendizaje profundo

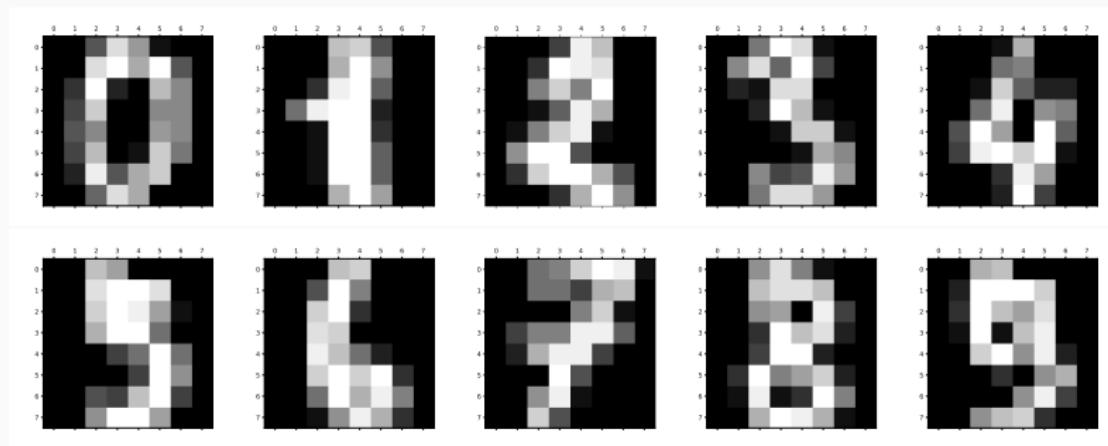
REDES CONVOLUCIONALES

Gibran Fuentes-Pineda

Octubre 2020

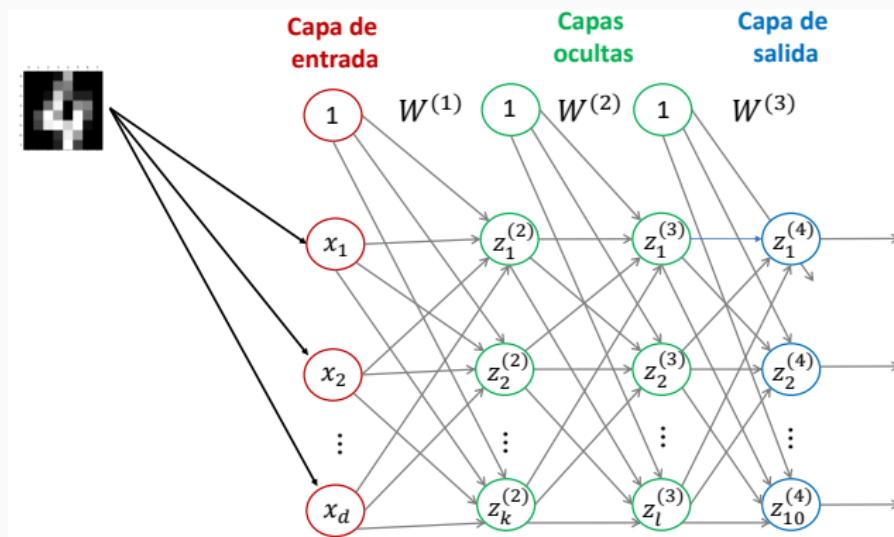
Clasificación de imágenes

- Por ejemplo, clasificar dígitos escritos a mano



Clasificación de imágenes con PMC

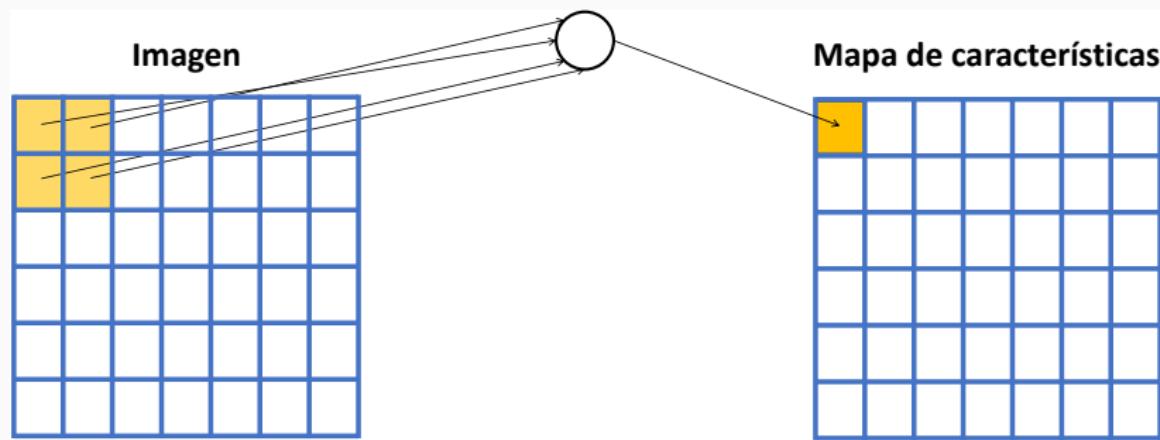
- Redes perceptrón multicapa tienen muchos parámetros
 - Por ej., si tenemos imágenes de 32×32 y 1 red con 1 sola capa oculta con 10 neuronas tendríamos $(32 \times 32 \times 10) + 1$ pesos



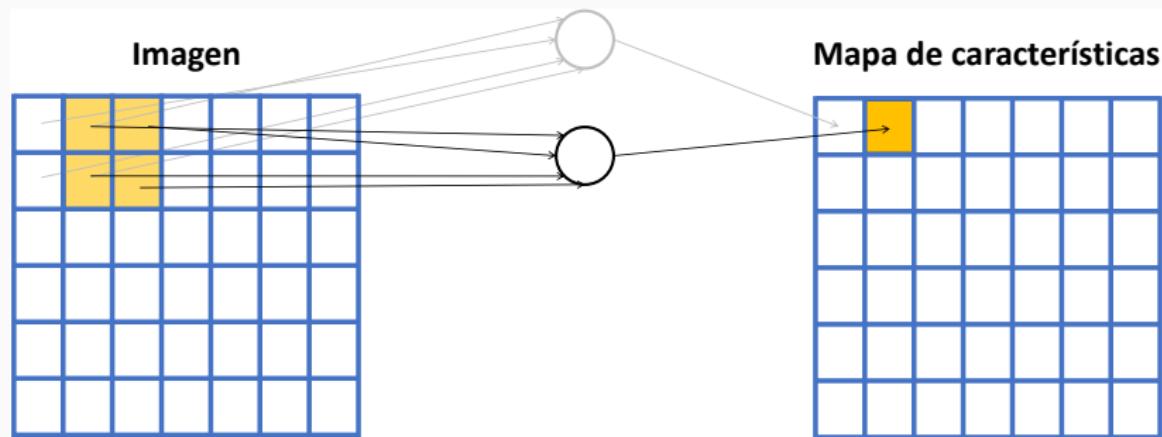
Capa convolucional

- Pueden verse como un caso especial de una capa densa con 2 variaciones
 1. Conectividad local (dispersa)
 2. Pesos compartidos
- Las representaciones obtenidas son más eficientes
- La operación es equivariante

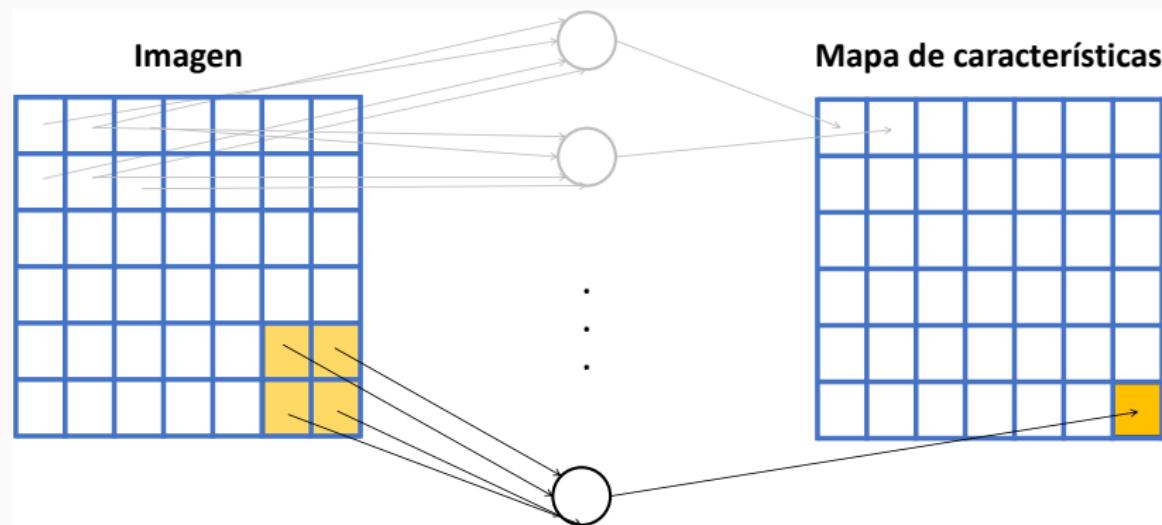
Conectividad local (1)



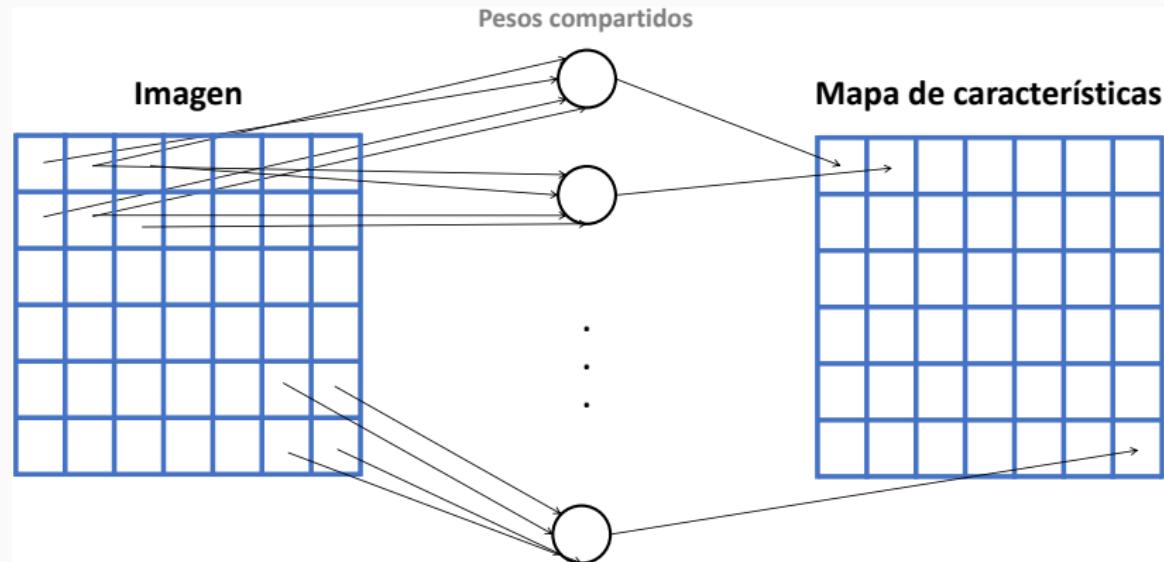
Conectividad local (2)



Conectividad local (3)



Pesos compartidos



Operación de convolución

- Convolución en 1 dimensión
 - Continua $s(t) = (x * k)(t) = \int_m x(m)k(t - m)dm$
 - Discreta $s[i] = (x * k)[i] = \sum_{m=-\infty}^{\infty} x[m]k[i - m]$
- Convolución en 2 dimensiones (discreta)
 - $S[i, j] = (I * K)[i, j] = \sum_m \sum_n I[m, n]K[i - m, j - n]$
 - $S[i, j] = (K * I)[i, j] = \sum_m \sum_n I[i - m, j - n]K[m, n]$
- Correlación cruzada (discreta)
 - $S(i, j) = (I * K)(i, j) = \sum_m \sum_n I(i + m, j + n)K(m, n)$

Operación de convolución

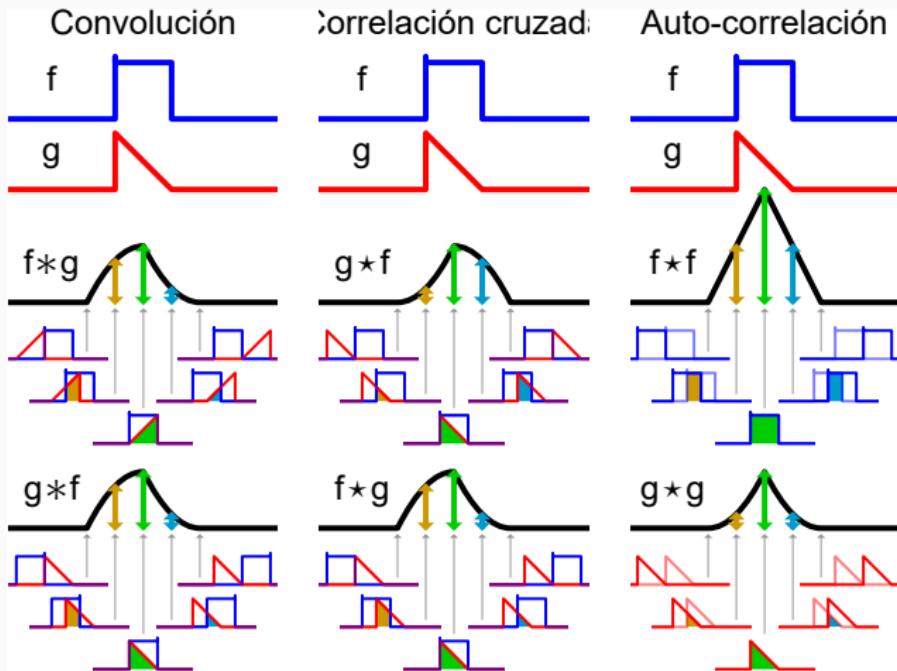


Imagen tomada de Wikipedia (Convolution)

Convolución de una imagen con un filtro (1)

The diagram illustrates the convolution of a 7x7 input image I with a 3x3 kernel K . The result is a 5x5 output image $I * K$.

Input Image (I):

0	1	1	1	0	0	0
0	0	1	1	1	0	0
0	0	0	1	1	1	0
0	0	0	1	1	0	0
0	0	1	1	0	0	0
0	1	1	0	0	0	0
1	1	0	0	0	0	0

Kernel (K):

1	0	1
0	1	0
1	0	1

Output Image ($I * K$):

1	4	3	4	1
1	2	4	3	3
1	2	3	4	1
1	3	3	1	1
3	3	1	1	0

Convolución de una imagen con un filtro (2)

The diagram illustrates the convolution process between an input image I and a kernel K .

Input Image (I):

0	1	1	1	0	0	0
0	0	1	1	1	0	0
0	0	0	1	1	1	0
0	0	0	1	1	1	0
0	0	1	1	0	0	0
0	1	1	0	0	0	0
1	1	0	0	0	0	0

Kernel (K):

1	0	1
0	1	0
1	0	1

Result of Convolution ($I * K$):

1	4	3	4	1
1	2	4	3	3
1	2	3	4	1
1	3	3	1	1
3	3	1	1	0

Convolución de una imagen con un filtro (3)

The diagram illustrates the convolution process between an input image I and a kernel K .

Input Image (I):

0	1	1	1	1	0	0	0
0	0	1	1	1	0	0	0
0	0	0	1	1	1	0	0
0	0	0	1	1	0	0	0
0	0	1	1	0	0	0	0
0	1	1	0	0	0	0	0
1	1	0	0	0	0	0	0

Kernel (K):

1	0	1
0	1	0
1	0	1

Result ($I * K$):

1	4	3	4	1
1	2	4	3	3
1	2	3	4	1
1	3	3	1	1
3	3	1	1	0

The diagram shows the convolution operation step-by-step. A 3x3 window slides across the input image I , applying the kernel K at each position. The result is shown in the output matrix $I * K$. The highlighted red and green regions indicate the receptive fields of the output unit highlighted in green.

Convolución de una imagen con un filtro (4)

$$\begin{array}{|c|c|c|c|c|c|c|c|} \hline 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ \hline \textcolor{red}{0} & \textcolor{red}{0} & \textcolor{red}{0} & \textcolor{red}{1} & 1 & 1 & 0 \\ \hline \textcolor{red}{0} & \textcolor{red}{0} & \textcolor{red}{0} & \textcolor{red}{1} & 1 & 1 & 0 \\ \hline \textcolor{red}{0} & \textcolor{red}{0} & \textcolor{red}{1} & 1 & 0 & 0 & 0 \\ \hline \textcolor{red}{0} & 1 & 1 & 0 & 0 & 0 & 0 \\ \hline \end{array} \quad * \quad \begin{array}{|c|c|c|} \hline 1 & 0 & 1 \\ \hline 0 & 1 & 0 \\ \hline 1 & 0 & 1 \\ \hline \end{array} \quad = \quad \begin{array}{|c|c|c|c|c|} \hline 1 & 4 & 3 & 4 & 1 \\ \hline 1 & 2 & 4 & 3 & 3 \\ \hline 1 & 2 & 3 & 4 & 1 \\ \hline \textcolor{green}{1} & 3 & 3 & 1 & 1 \\ \hline 3 & 3 & 1 & 1 & 0 \\ \hline \end{array}$$

The diagram illustrates the convolution process between a 7x7 input image I and a 3x3 kernel K . The input image I has values ranging from 0 to 1. The kernel K has values 1, 0, 1; 0, 1, 0; 1, 0, 1. The result of the convolution, $I * K$, is shown in the bottom right. The result is highlighted with a green box around the value 1.

Capa convolucional

- Compuesta de filtros distintos
- La salida (mapas de activaciones) es un volumen

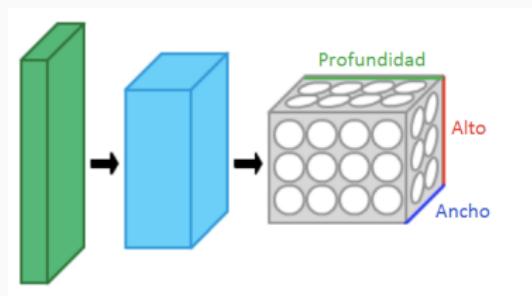
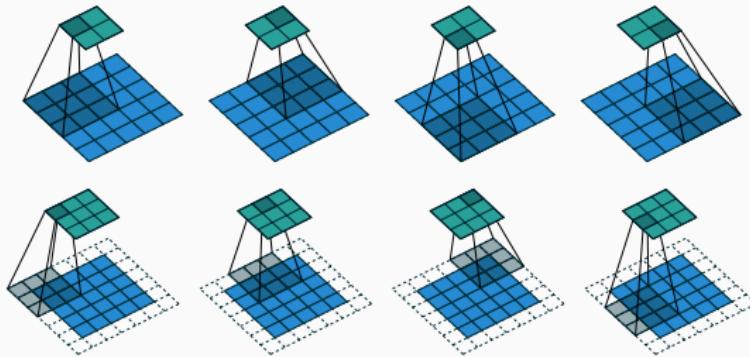


Imagen tomada de Wikipedia (Convolutional Neural Network)

Capa convolucional: hiperparámetros

- Número de filtros $N_{filtros}$, tamaño de cada filtro $F_H \times F_W$, desplazamiento S (*stride*) y relleno (*padding*)



Imágenes creadas por Dumoulin and Visin. A guide to convolution arithmetic for deep learning, 2018.

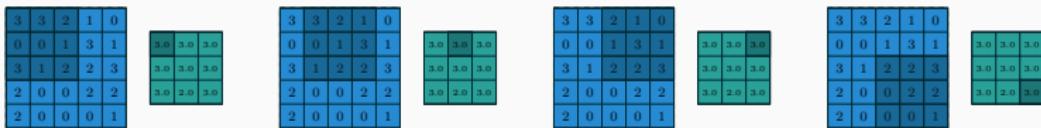
$$H^{\{\ell\}} = \frac{H^{\{\ell-1\}} + 2P^{\{\ell-1\}} - F_H^{\{\ell\}}}{S^{\{\ell\}}} + 1$$

$$W^{\{\ell\}} = \frac{W^{\{\ell-1\}} + 2P^{\{\ell-1\}} - F_W^{\{\ell\}}}{S^{\{\ell\}}} + 1$$

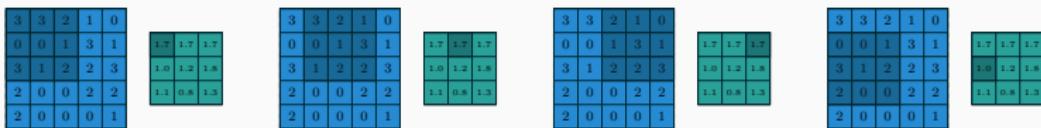
Capa de submuestreo o decimación

- Capa para reducir dimensiones tomando el valor máximo (*max-pooling*) o el promedio (*average pooling*) de un grupo de activaciones usualmente contiguas

Máximo



Promedio



Imágenes creadas por Dumoulin and Visin. A guide to convolution arithmetic for deep learning, 2018.

Capa de submuestreo: hiperparámetros

- Hiperparámetros: alto y ancho de la ventana que define el grupo (F_H, F_W), desplazamiento S , relleno (*padding*) P
- El alto $H^{\{\ell\}}$ y ancho $W^{\{\ell\}}$ de la salida de una capa de submuestreo está dado por

$$H^{\{\ell\}} = \frac{H^{\{\ell-1\}} + 2P - F_H}{S} + 1$$

$$W^{\{\ell\}} = \frac{W^{\{\ell-1\}} + 2P - F_W}{S} +$$

Submuestro global promedio

- Reduce las dimensiones de un volumen de características, tomando únicamente el promedio de cada mapa

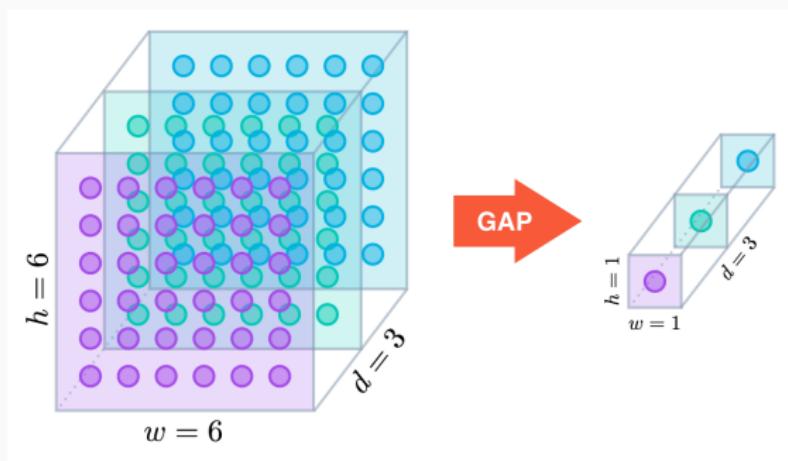


Imagen tomada de <https://alexisbcook.github.io/2017/global-average-pooling-layers-for-object-localization/>

- Hacia adelante
 - Se aplican las operaciones de convolución con correspondiente activación y decimación
- Hacia atrás
 - En la convolución cada neurona actualiza los gradientes por separado y al final se suman para actualizar los pesos compartidos
 - En la decimación se actualiza sólo la neurona ganadora (*max-pooling*)

Convolución de 1×1

- Funciona como un tipo de decimación en profundidad

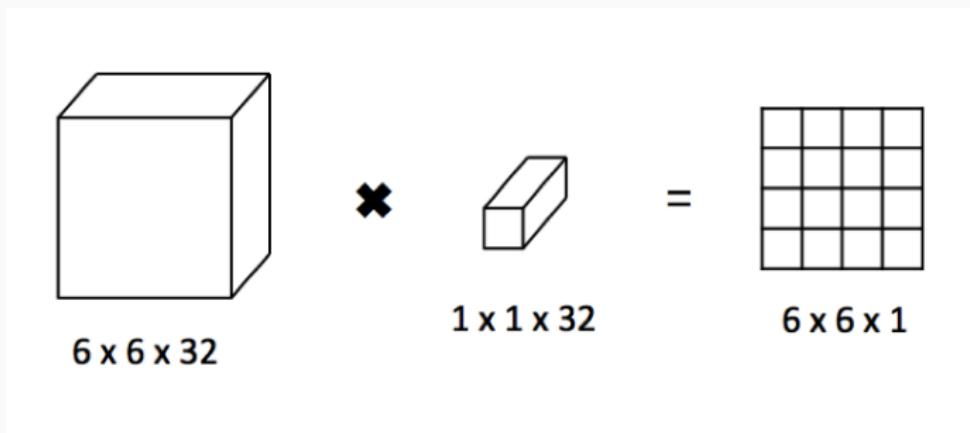


Imagen tomada de <https://medium.com/machine-learning-bites/deeplearning-series-convolutional-neural-networks-a9c2f2ee1524>

Red neuronal convolucional

- Compuesta de dos bloques principales
 1. *Extracción de características*: múltiples capas convolucionales y de submuestreo
 2. *Clasificación*: Una o más capas completamente conectadas (incluyendo la capa de salida)

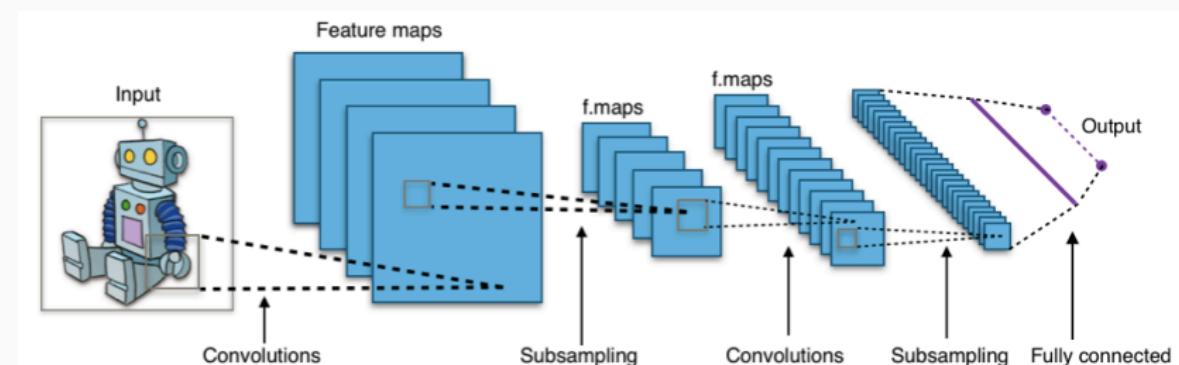


Imagen del usuario Aphex34 de Wikipedia. CC BY-SA 4.0

Arquitecturas de redes convolucionales: LeNet

- Red poco profunda inspirada en la propuesta originalmente por LeCun et al. 1998

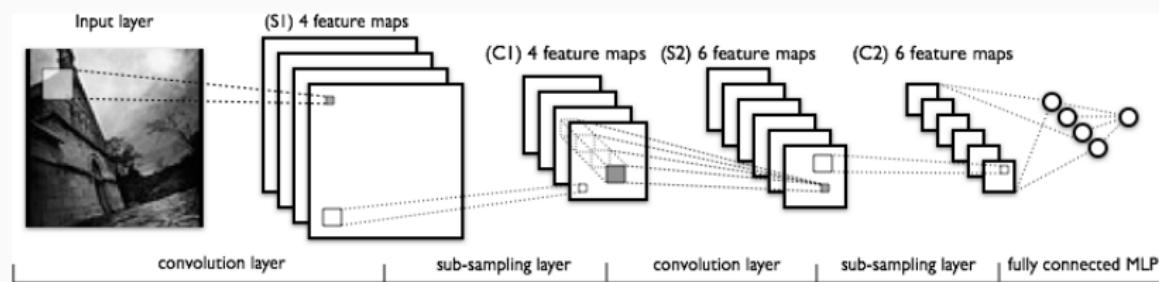


Imagen tomada de <http://deeplearning.net/tutorial/lenet.html>

Arquitecturas de redes convolucionales: AlexNet

- Usa función de activación ReLU
- Entrenamiento con versión optimizada para 2 GPUs
- Normaliza respuestas
- Submuestreo con traslape

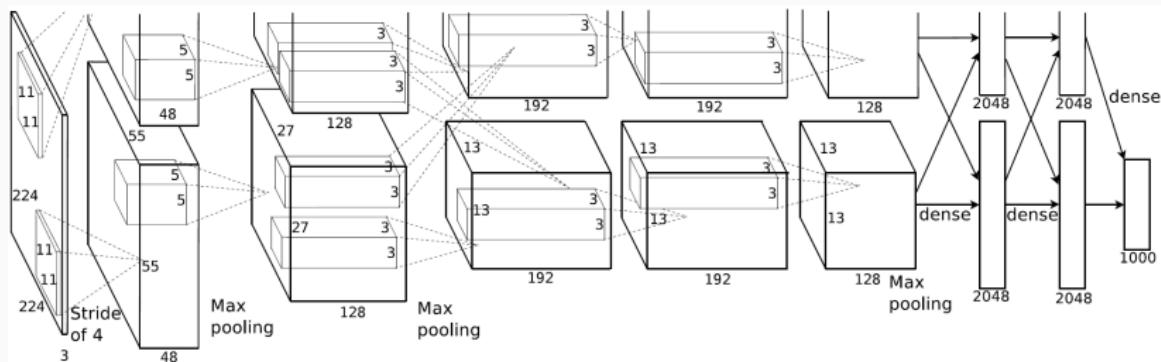


Imagen tomada de Krizhevsky et al. *ImageNet Classification with Deep Convolutional Neural Networks*, 2012

Arquitecturas de redes convolucionales: ZFNet

- Muy similar a AlexNet pero usa filtros más pequeños con menor desplazamiento

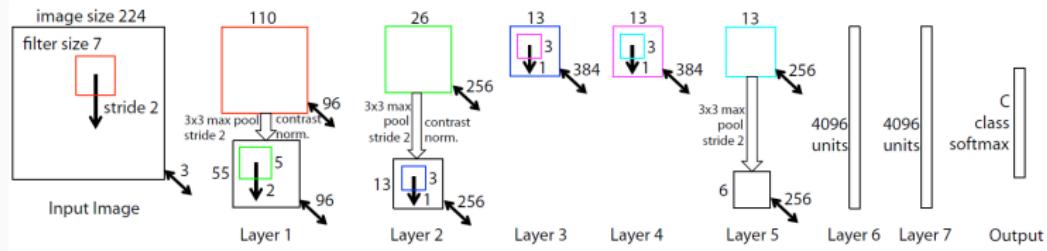


Imagen tomada de Zeiler and Fergus. *Visualizing and Understanding Convolutional Networks*, 2014

Arquitecturas de redes convolucionales: VGGNet

- 19 capas
- Filtros de 3×3 con desplazamientos de 1
- Max-pooling de 2×2 con desplazamientos de 2



Imagen tomada de diapositivas de Simonyan (ILSVRC Workshop 2014)

Arquitecturas de redes convolucionales: GoogleNet (1)

- 22 capas
- Utiliza bloques de capas convolucionales paralelas cuyas salidas se concatenan

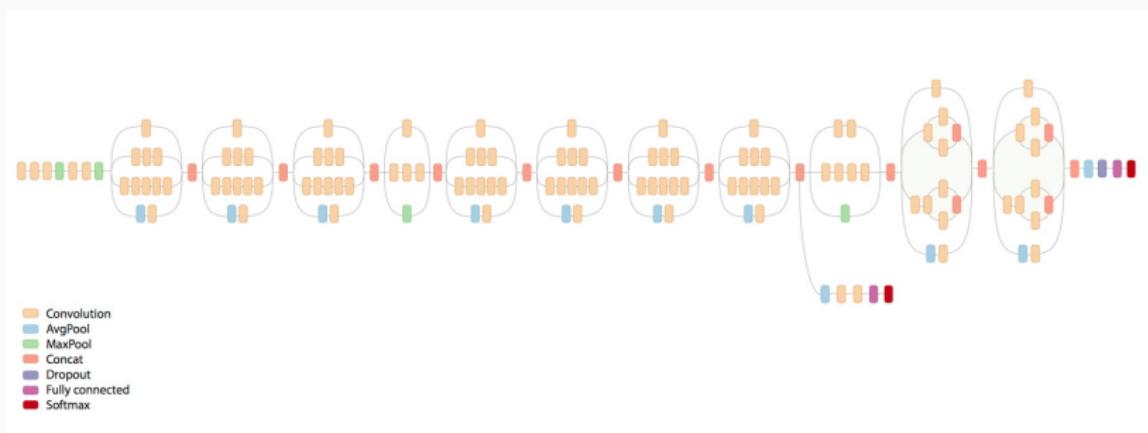


Imagen tomada de <https://research.googleblog.com/2016/03/train-your-own-image-classifier-with.html>

Arquitecturas de redes convolucionales: GoogleNet (2)

type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

Tabla tomada de Szegedy et al. Going deeper with convolutions, 2014

Bloques Inception

Imagen tomada de Szegedy et al. Going deeper with convolutions, 2014

Revolution of Depth

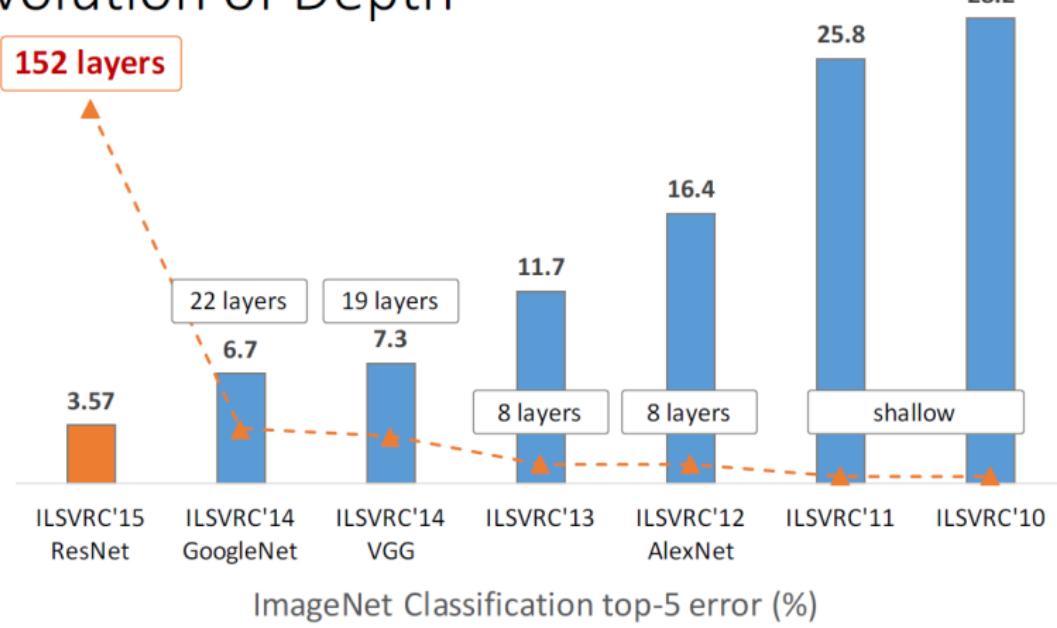


Imagen tomada de diapositivas de Kaiming He (ICML 2016)

Sobre profundidad en redes convolucionales (2)

- Problemas con el desvanecimiento y explosión de respuestas (hacia adelante) y gradientes (hacia atrás)

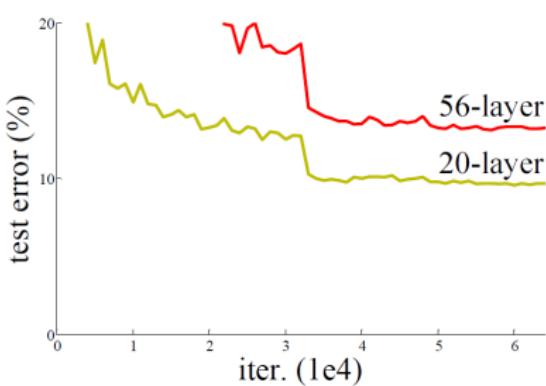
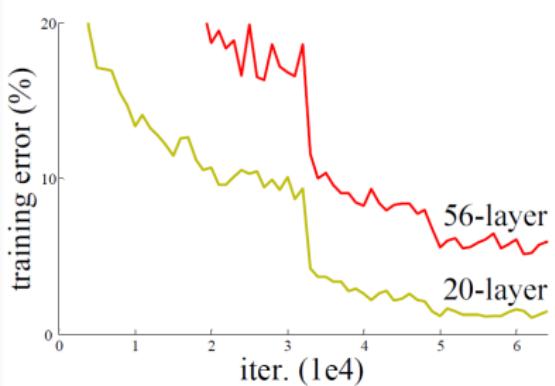


Imagen tomada de He et al. Deep Residual Learning for Image Recognition, 2015

Entrenando redes más profundas: ResNet (1)

- Reformula los mapeos que se desean aprender a residuales
- Más fácil de optimizar los residuales que el mapeo original¹

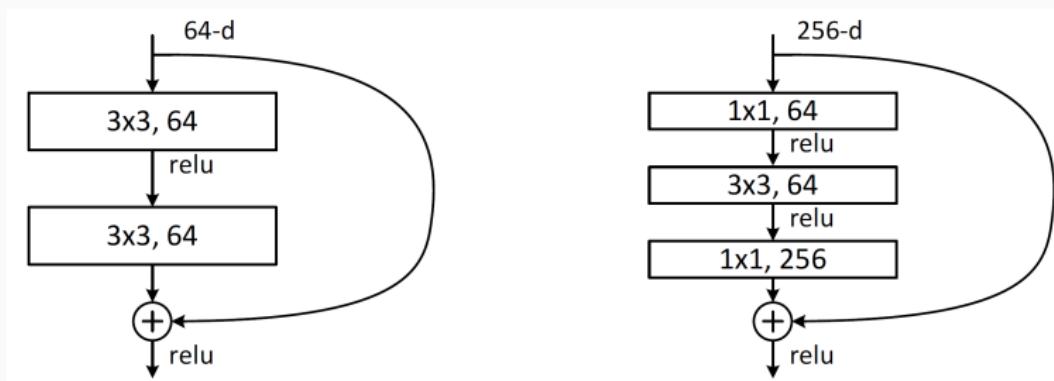


Imagen tomada de He et al. Deep Residual Learning for Image Recognition, 2015

¹Se ha mostrado que son aproximadores universales: Lin y Jegelka. ResNet with one-neuron hidden layers is a Universal Approximator, 2018.

Entrenando redes más profundas: ResNet (2)

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer	
conv1	112×112			7×7, 64, stride 2			
conv2_x	56×56			3×3 max pool, stride 2			
conv3_x	28×28	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	
conv4_x	14×14	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 8$	
conv5_x	7×7	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 23$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 36$	
	1×1			average pool, 1000-d fc, softmax			
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9	

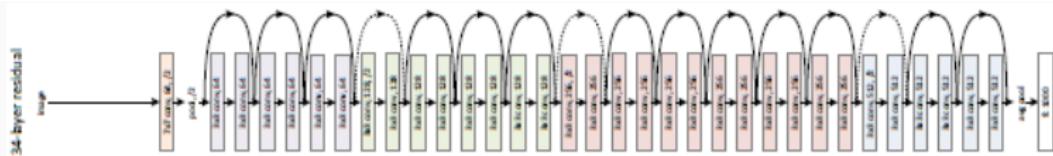


Imagen y tabla tomadas de He et al. Deep Residual Learning for Image Recognition, 2015

Entrenando redes más profundas: ResNet (3)

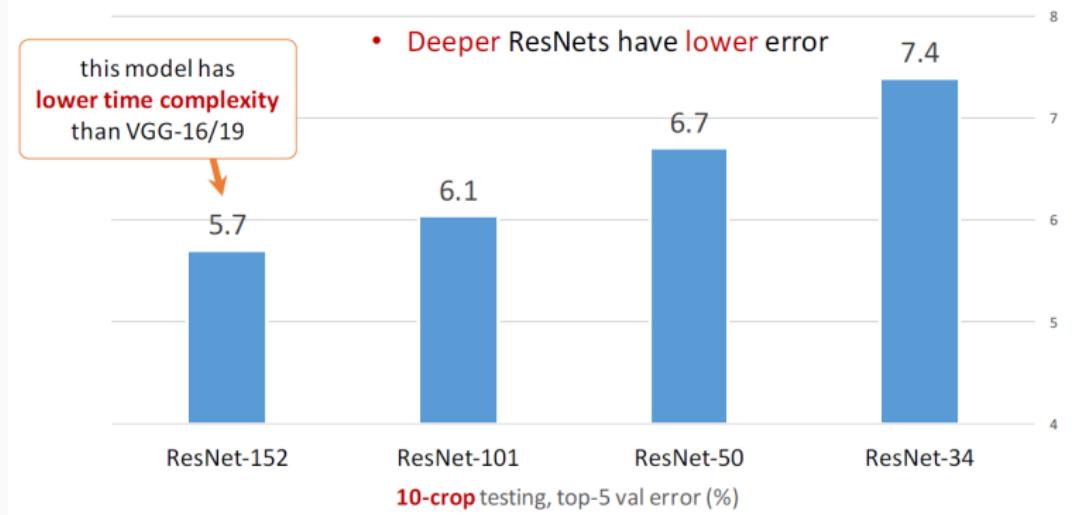


Imagen tomada de diapositivas de Kaiming He (ICML 2016)

Entrenando redes más profundas: DenseNet (1)

- Utiliza bloques que conectan la salida de cada capa con las entradas de capas subsecuentes.

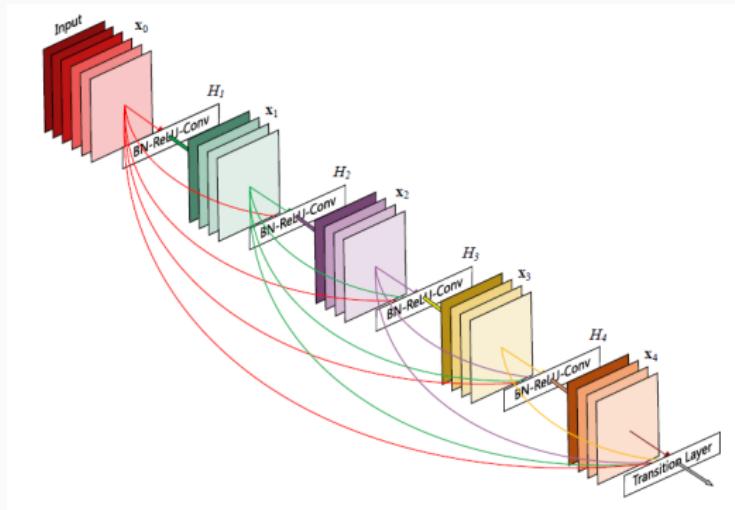


Imagen tomada de Huang et al. Densely Connected Convolutional Networks, 2018

Entrenando redes más profundas: DenseNet (2)

Layers	Output Size	DenseNet-121($k = 32$)	DenseNet-169($k = 32$)	DenseNet-201($k = 32$)	DenseNet-161($k = 48$)
Convolution	112×112			7×7 conv, stride 2	
Pooling	56×56			3×3 max pool, stride 2	
Dense Block (1)	56×56	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$
Transition Layer (1)	56×56			1×1 conv	
	28×28			2×2 average pool, stride 2	
Dense Block (2)	28×28	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$
Transition Layer (2)	28×28			1×1 conv	
	14×14			2×2 average pool, stride 2	
Dense Block (3)	14×14	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 24$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 36$
Transition Layer (3)	14×14			1×1 conv	
	7×7			2×2 average pool, stride 2	
Dense Block (4)	7×7	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 16$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 24$
Classification Layer	1×1			7×7 global average pool	
				1000D fully-connected, softmax	

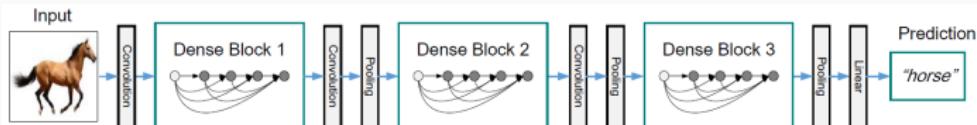


Imagen y tabla tomadas de Huang et al. Densely Connected Convolutional Networks, 2018

Filtros separables

- Filtros que pueden descomponerse como el producto de 2 más simples. Por ej., un filtro 2D puede separarse en 2 filtros 1D.

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \times \begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$$

Convolución espacial separable

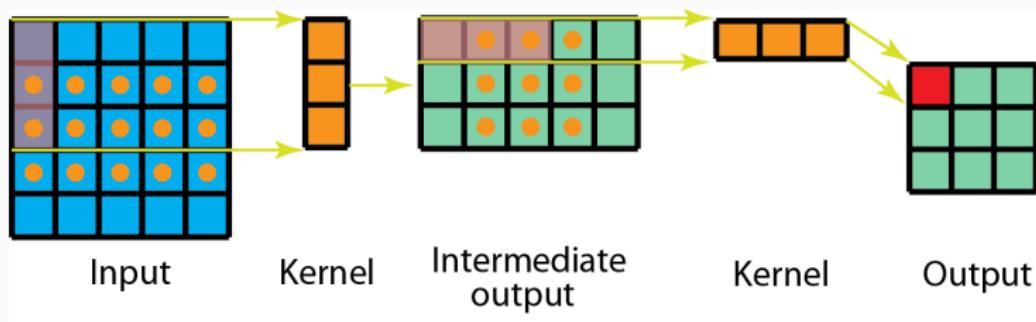


Imagen tomada de Kunlun Bai. A Comprehensive Introduction to Different Types of Convolutions in Deep Learning, 2019.

Convolución en profundidad

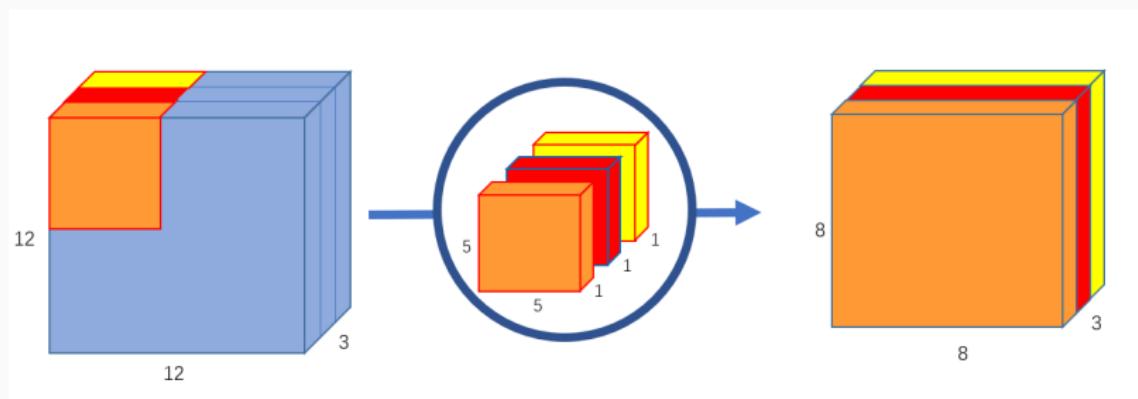


Imagen tomada de Chi-Feng Wang, A Basic Introduction to Separable Convolutions, 2018.

Convolución puntual (de 1×1)

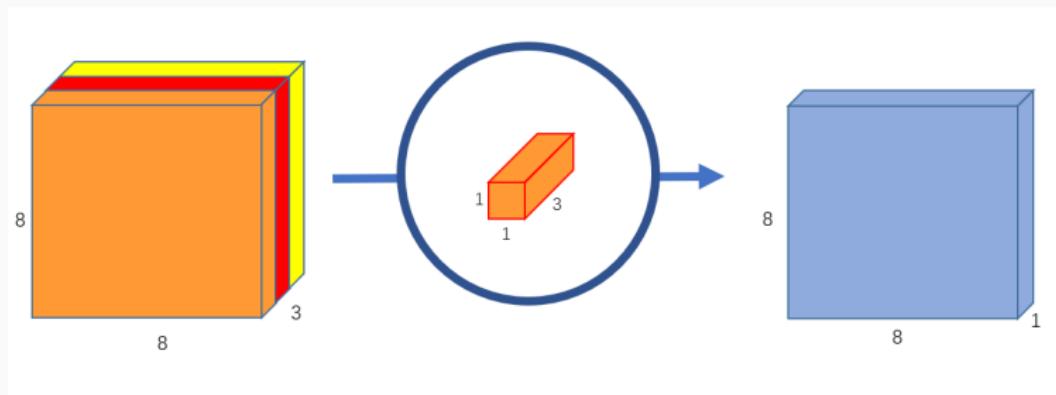


Imagen tomada de Chi-Feng Wang, A Basic Introduction to Separable Convolutions, 2018.

Convolución puntual (de 1×1)

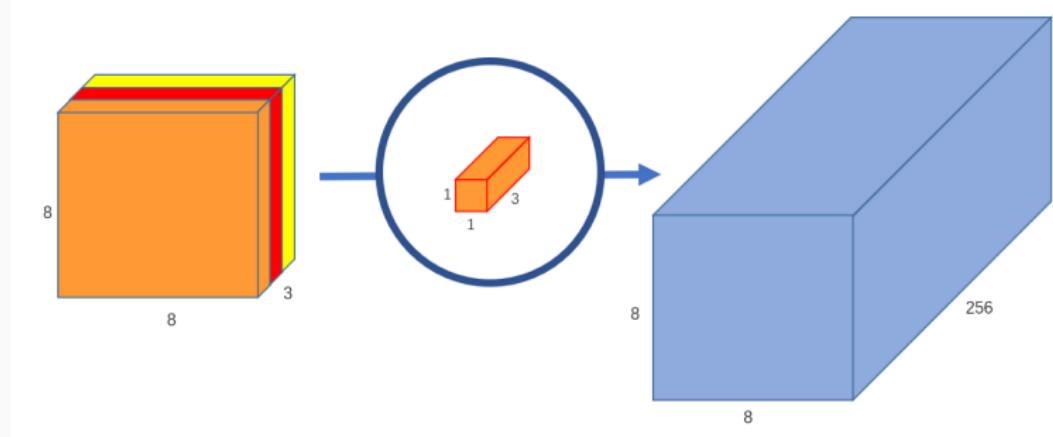


Imagen tomada de Chi-Feng Wang. A Basic Introduction to Separable Convolutions, 2018.

Convolución separable en profundidad

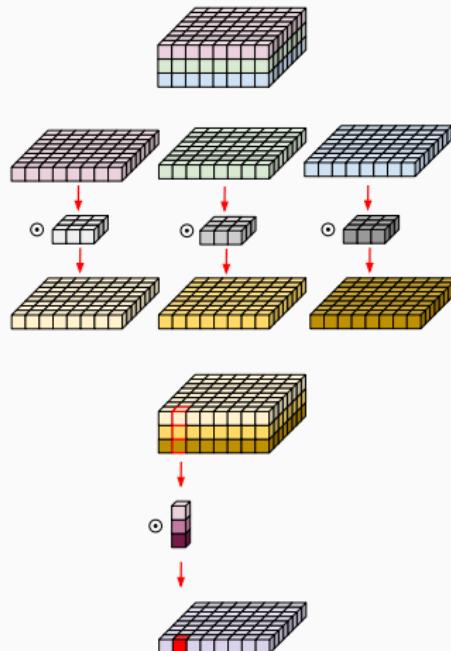


Imagen tomada de Eli Bendersky. Depthwise separable convolutions for machine learning, 2018.

Bloques Inception y convoluciones separables

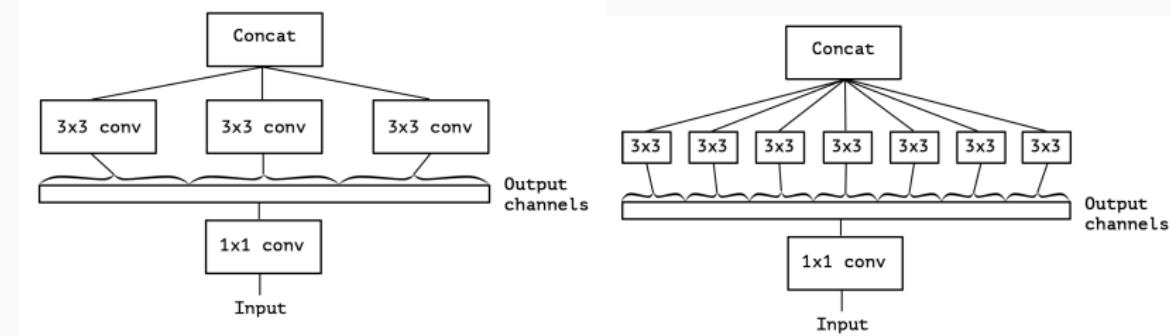


Imagen tomada de F. Chollet. Xception: Deep Learning with Depthwise Separable Convolutions, 2017.

Xception

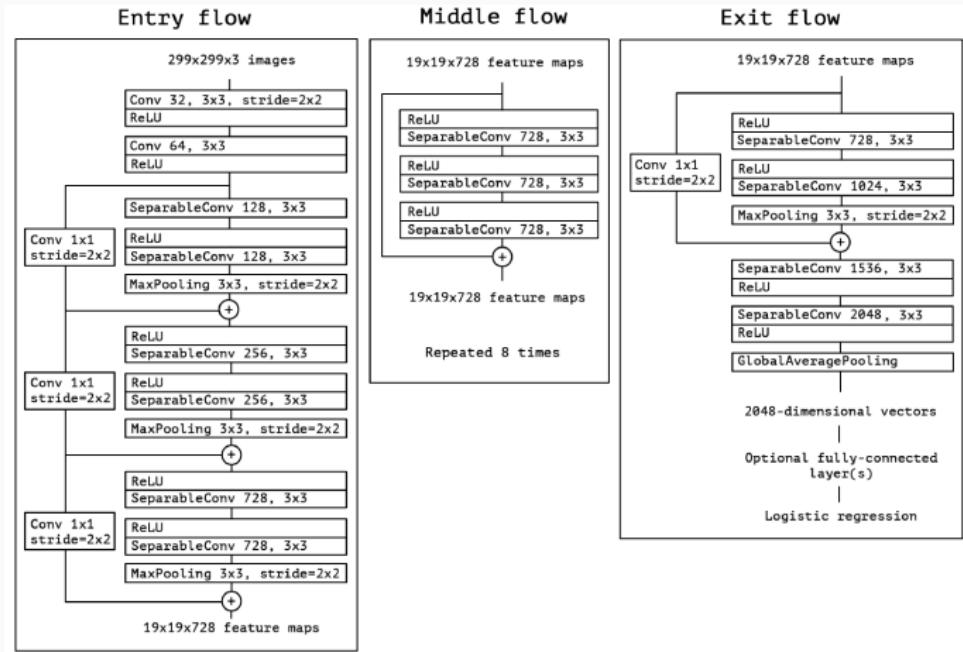


Imagen tomada de F. Chollet. Xception: Deep Learning with Depthwise Separable Convolutions, 2017.

Convolución dilatada

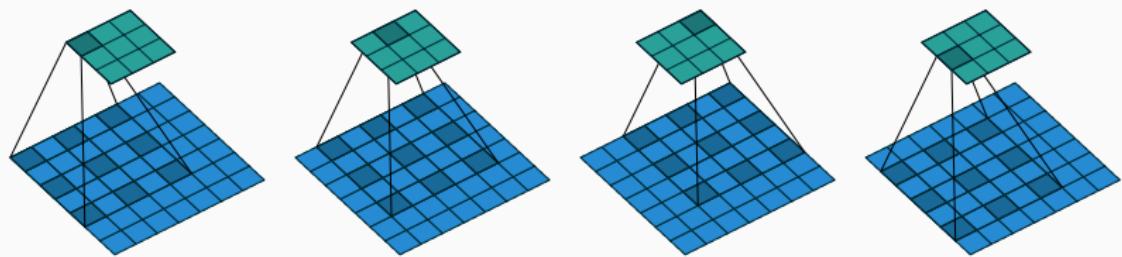


Imagen tomada de Dumoulin and Visin. *A guide to convolution arithmetic for deep learning*, 2018.

Visualizando respuestas

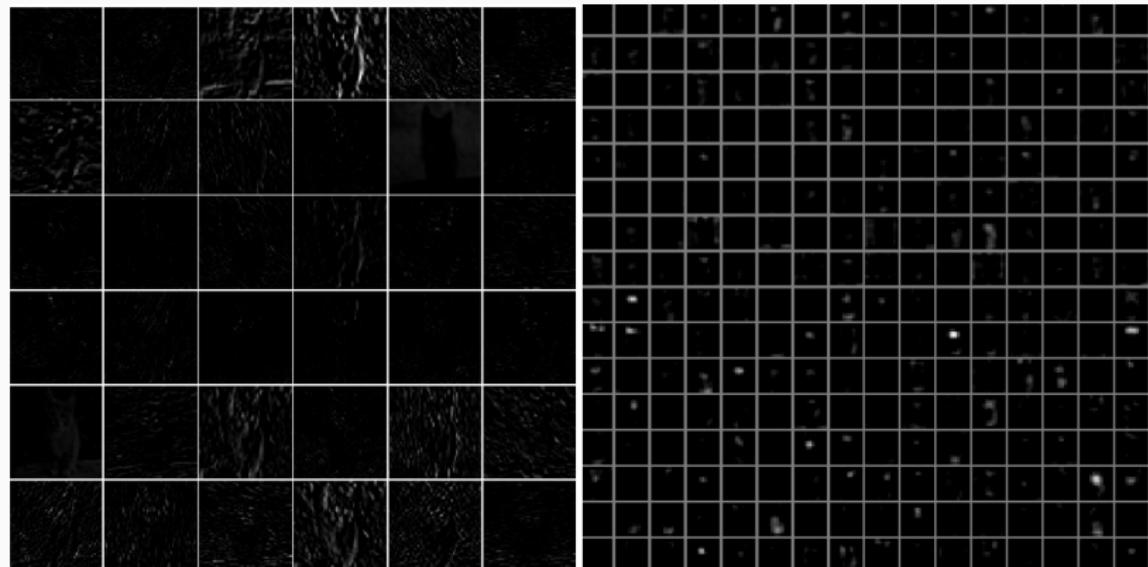


Imagen tomada de <http://cs231n.github.io/understanding-cnn/>

Visualizando pesos

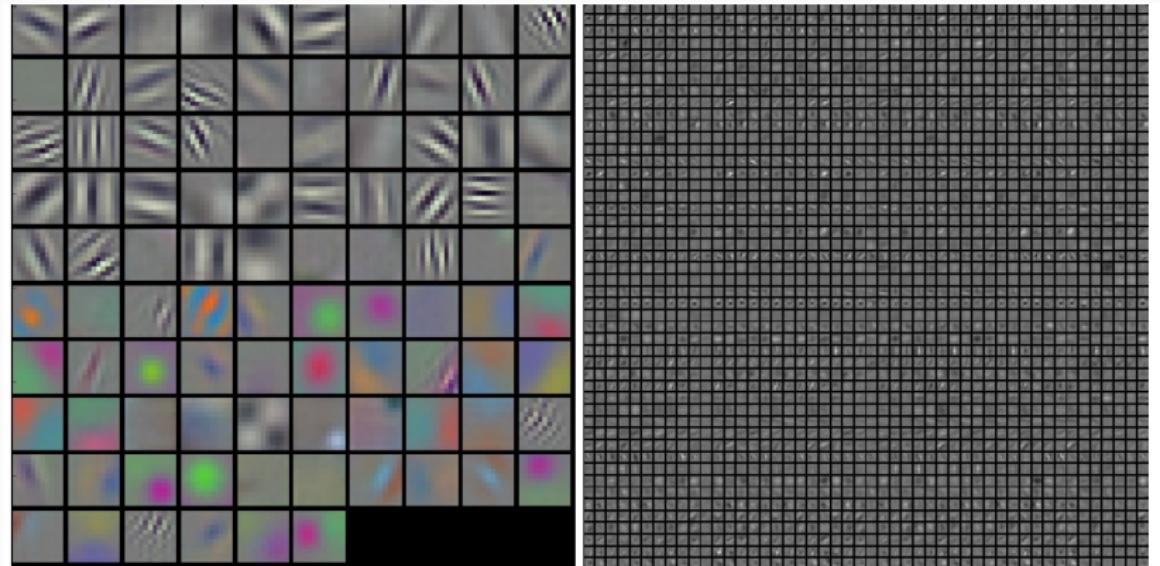


Imagen tomada de <http://cs231n.github.io/understanding-cnn/>

Visualizando imágenes con respuestas máximas



Imagen tomada de <http://cs231n.github.io/understanding-cnn/>

Visualizando obstrucciones

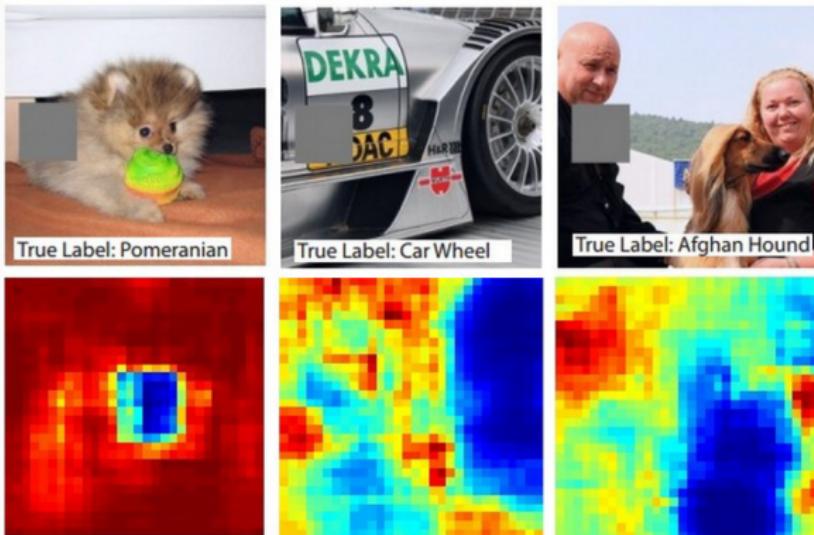


Imagen tomada de <http://cs231n.github.io/understanding-cnn/>

Convolución transpuesta

- Método para visualizar respuestas de diferentes capas en redes neuronales convolucionales

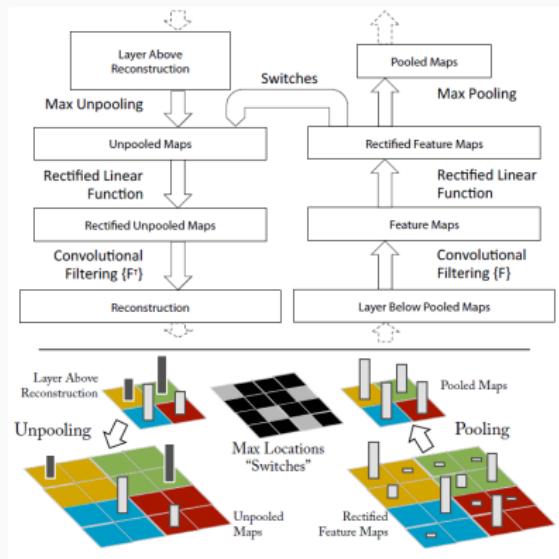


Imagen tomada de Zeiler and Fergus. *Visualizing and Understanding Convolutional Networks*, 2014

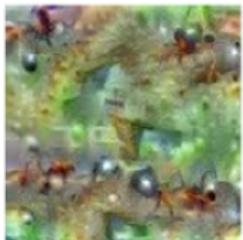
DeepDream de Google: diferentes clases



Hartebeest



Measuring Cup



Ant



Starfish



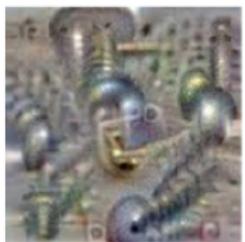
Anemone Fish



Banana



Parachute



Screw

Imagen tomada de <https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html>

DeepDream de Google: características de bajo nivel

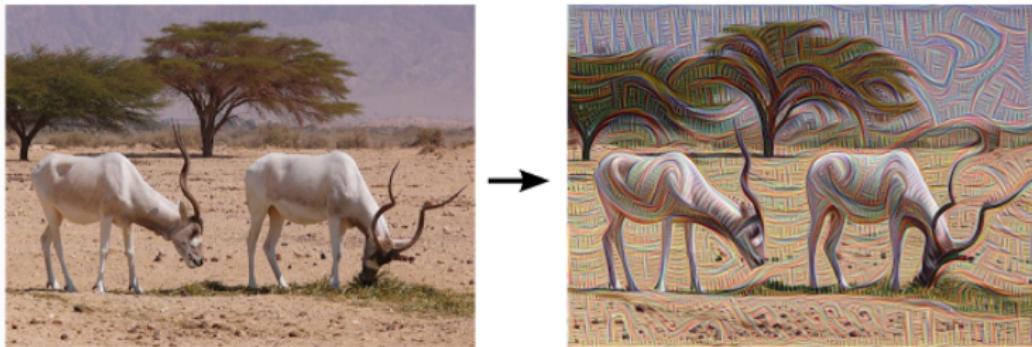


Imagen tomada de <https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html>

DeepDream de Google: características de alto nivel



"Admiral Dog!"



"The Pig-Snail"



"The Camel-Bird"



"The Dog-Fish"

Imagen tomada de <https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html>

DeepDream de Google: influencia de imagen original



Imagen tomada de <https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html>

DeepDream de Google: de forma aleatoria e iterativa



Imagen tomada de <https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html>

Redes convolucionales 1D

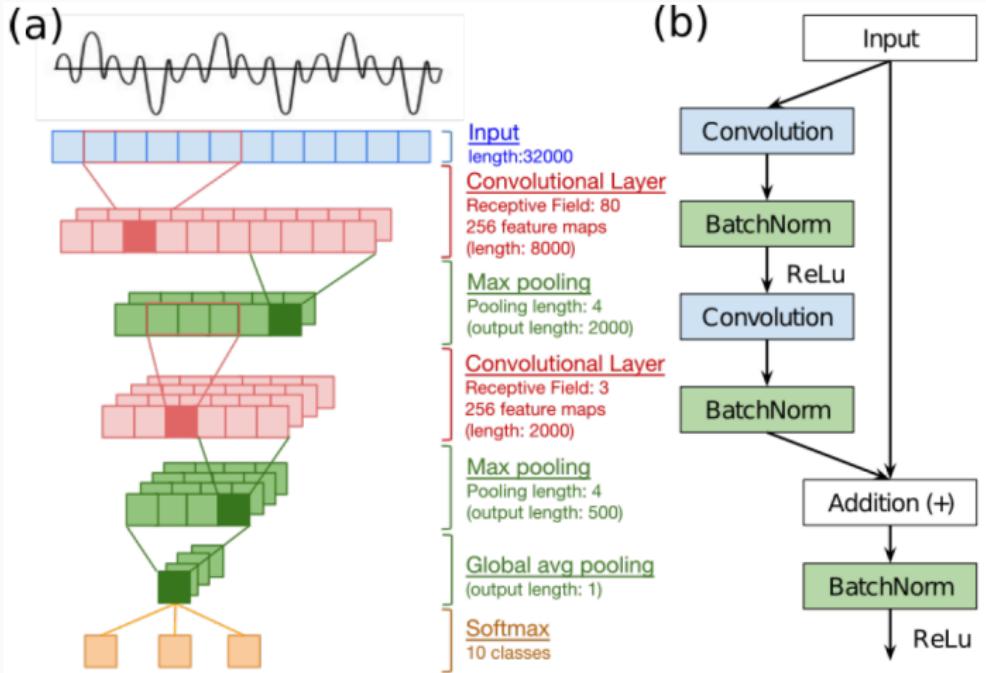


Imagen tomada de Dai et al. *Very Deep Convolutional Neural Networks for Raw Waveforms*, 2016

Redes convolucionales 3D (1)

- Respuesta en redes convolucionales 2D en una imagen

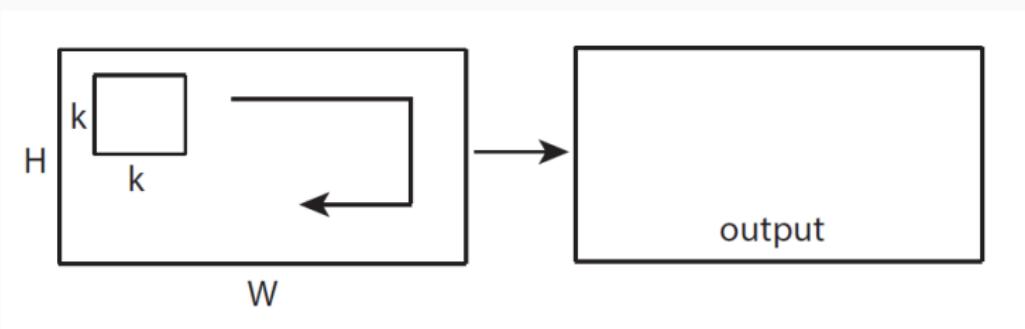


Imagen tomada de Tran et al. *Learning Spatiotemporal Features with 3D Convolutional Networks*, 2015

Redes convolucionales 3D (2)

- Respuesta en redes convolucionales 2D en varias imágenes

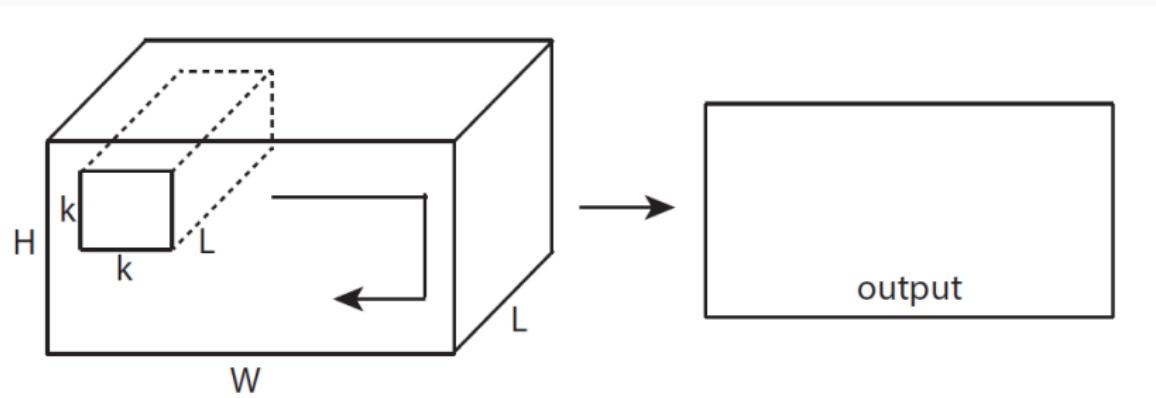


Imagen tomada de Tran et al. *Learning Spatiotemporal Features with 3D Convolutional Networks*, 2015

Redes convolucionales 3D (3)

- Respuesta en redes convolucionales 3D

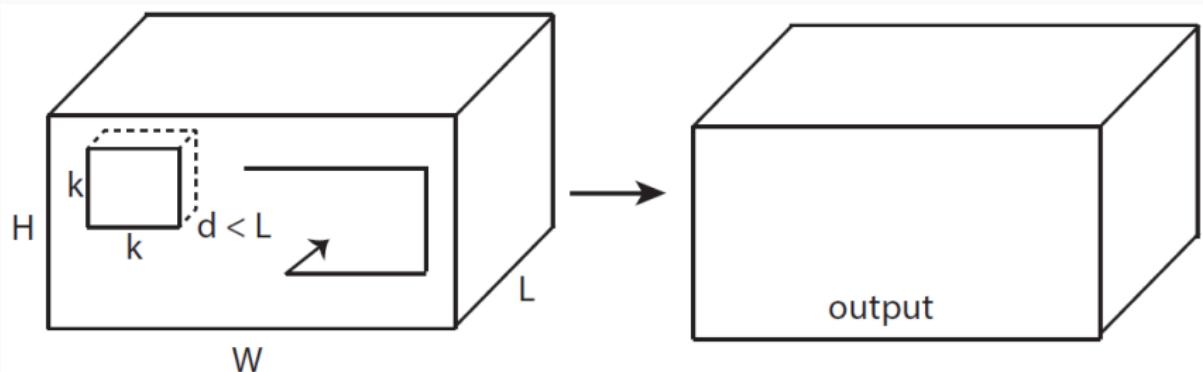


Imagen tomada de Tran et al. *Learning Spatiotemporal Features with 3D Convolutional Networks*, 2015