

# **Aprendizaje profundo**

## REDES CONVOLUCIONALES

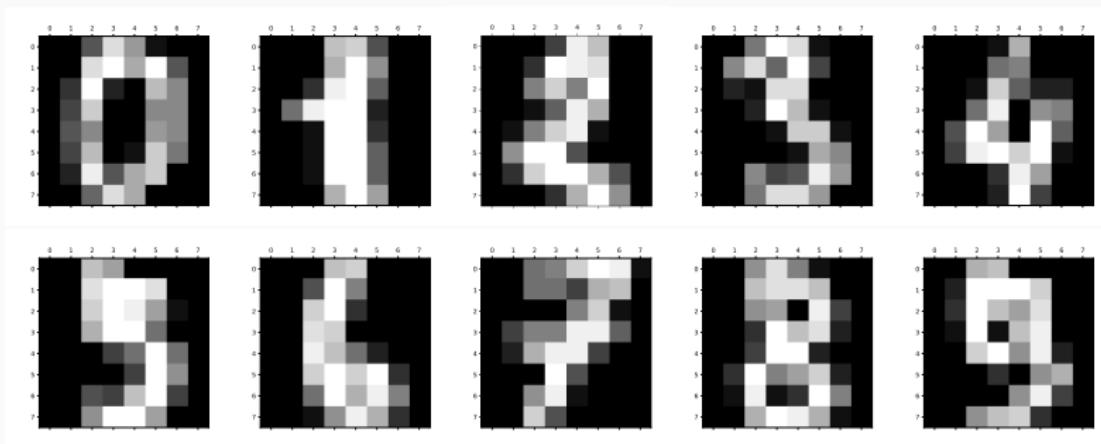
---

Gibran Fuentes-Pineda

Agosto 2019

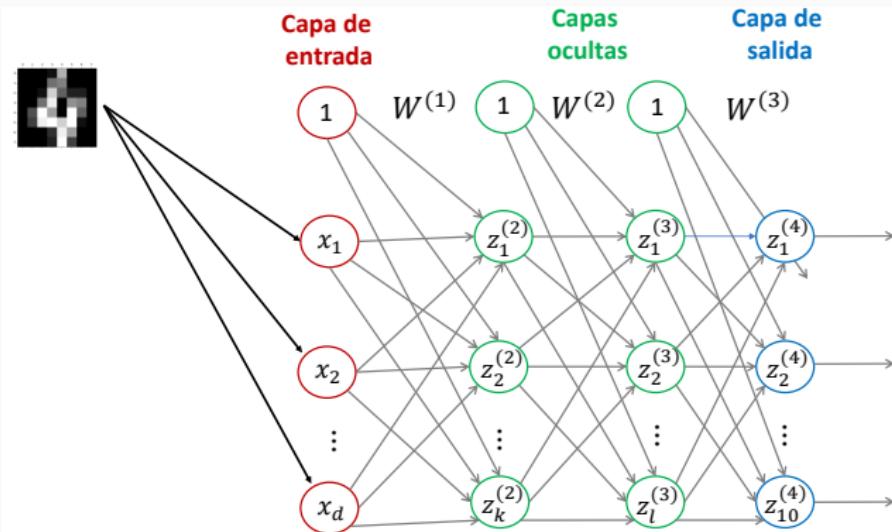
# Clasificación de imágenes

- Por ejemplo, clasificar dígitos escritos a mano



# Clasificación de imágenes con PMC

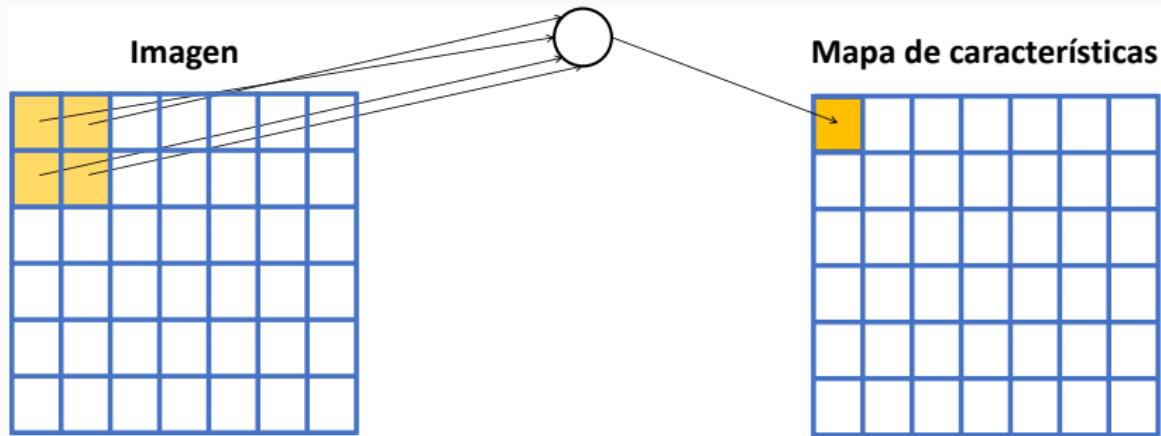
- Redes perceptrón multicapa tienen muchos parámetros
  - Por ej., si tenemos imágenes de  $32 \times 32$  y 1 red con 1 sola capa oculta con 10 neuronas tendríamos  $(32 \times 32 \times 10) + 1$  pesos



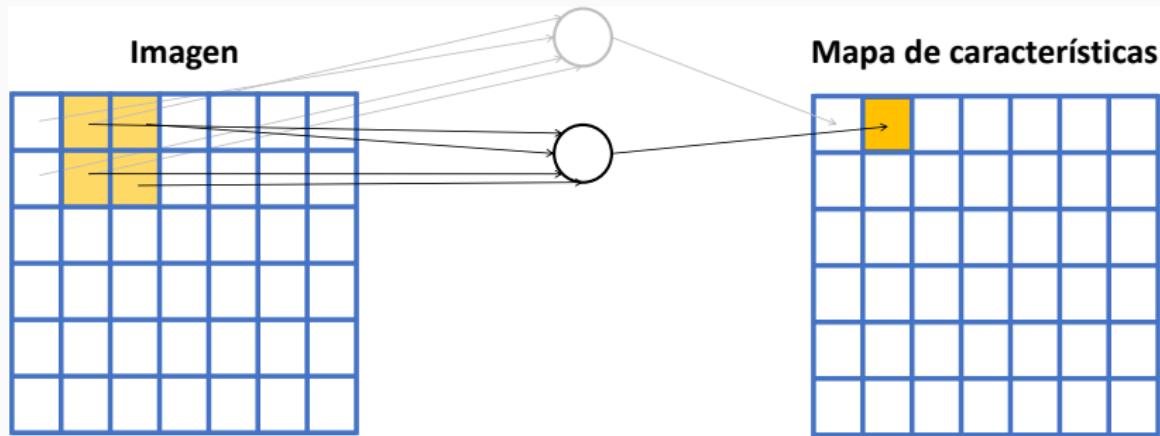
# Capa convolucional

- Pueden verse como un caso especial de una capa densa con 2 variaciones
  1. Conectividad local
  2. Pesos compartidos

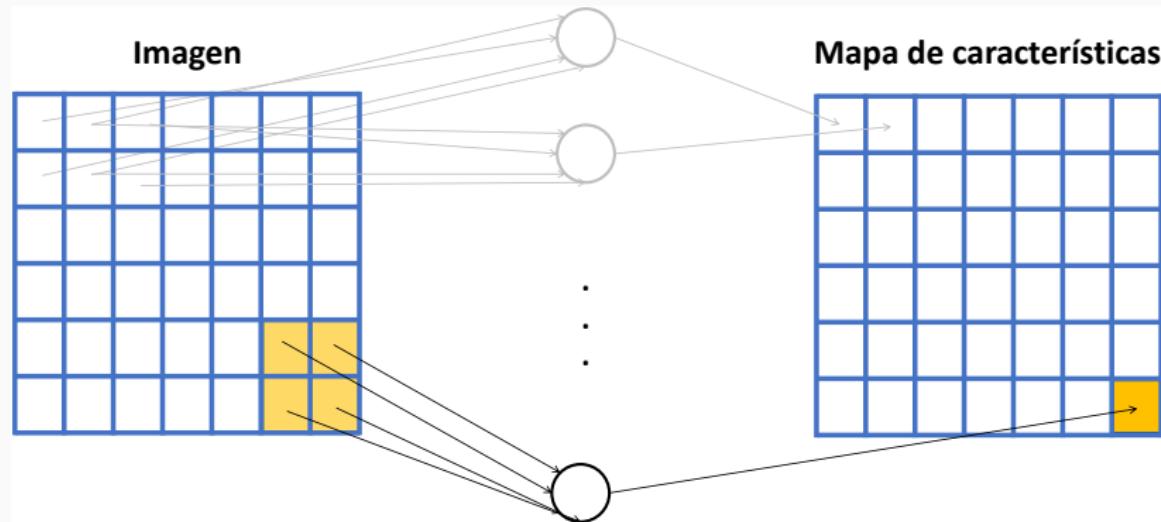
## Conectividad local (1)



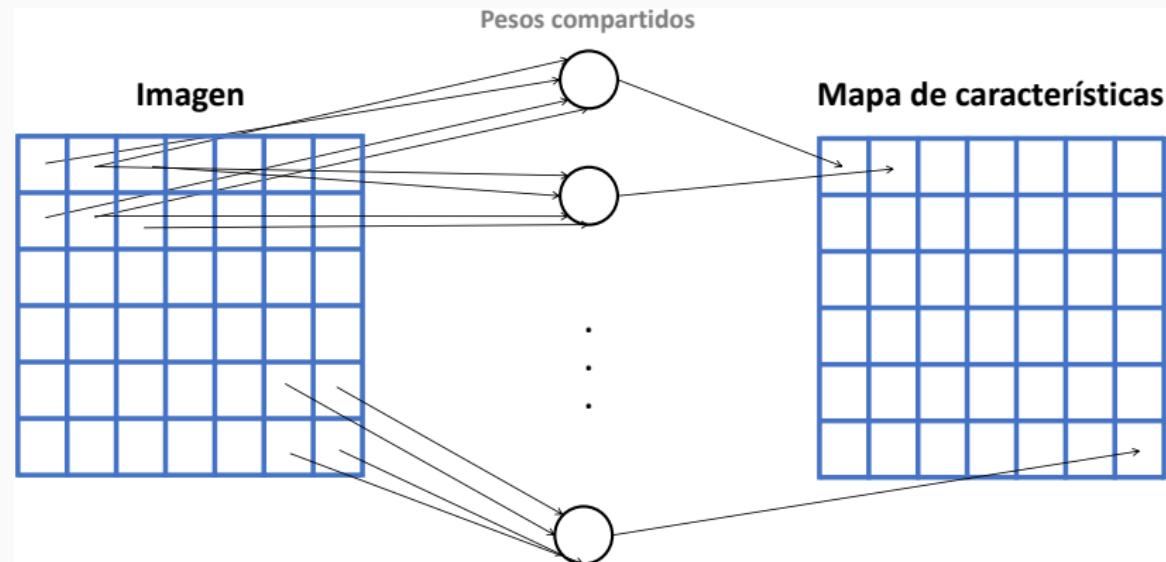
## Conektividad local (2)



## 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

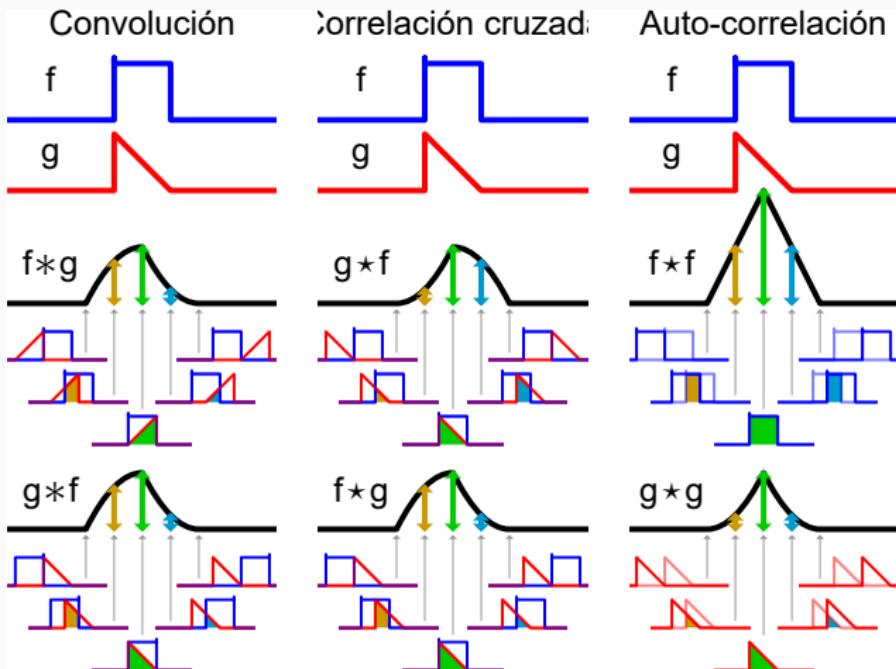


Imagen tomada de Wikipedia (Convolution)

# Convolución de una imagen con un filtro (1)

The diagram illustrates the convolution process between an input image  $I$  and a kernel  $K$ . The input image  $I$  is a 7x7 matrix with values ranging from 0 to 1. The kernel  $K$  is a 3x3 matrix with values 1, 0, 1. The result of the convolution,  $I * K$ , is a 5x5 matrix.

The convolution operation involves sliding the kernel over the input image and performing element-wise multiplication (highlighted by dotted lines) followed by summation (highlighted by dashed lines). The resulting value is placed in the output matrix at the corresponding position.

**Input Image ( $I$ )**

0 <small><math>\times 1</math></small>	1 <small><math>\times 0</math></small>	1 <small><math>\times 1</math></small>	1	0	0	0	0
0 <small><math>\times 0</math></small>	0 <small><math>\times 1</math></small>	1 <small><math>\times 0</math></small>	1	1	1	0	0
0 <small><math>\times 1</math></small>	0 <small><math>\times 0</math></small>	0 <small><math>\times 1</math></small>	1	1	1	1	0
0 <small><math>\times 0</math></small>	0 <small><math>\times 1</math></small>	0 <small><math>\times 0</math></small>	1	1	1	0	0
0 <small><math>\times 0</math></small>	0 <small><math>\times 1</math></small>	0 <small><math>\times 0</math></small>	1	1	0	0	0
0 <small><math>\times 0</math></small>	0 <small><math>\times 1</math></small>	1 <small><math>\times 0</math></small>	1	1	0	0	0
1 <small><math>\times 1</math></small>	1 <small><math>\times 0</math></small>	0 <small><math>\times 0</math></small>	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

## Convolución de una imagen con un filtro (2)

$$\begin{array}{c} \text{I} \\ \begin{array}{|c|c|c|c|c|c|c|} \hline 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ \hline 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ \hline 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ \hline \end{array} \end{array} * \begin{array}{|c|c|c|} \hline 1 & 0 & 1 \\ \hline 0 & 1 & 0 \\ \hline 1 & 0 & 1 \\ \hline \end{array} = \begin{array}{|c|c|c|c|c|} \hline 1 & 4 & 3 & 4 & 1 \\ \hline 1 & 2 & 4 & 3 & 3 \\ \hline 1 & 2 & 3 & 4 & 1 \\ \hline 1 & 3 & 3 & 1 & 1 \\ \hline 3 & 3 & 1 & 1 & 0 \\ \hline \end{array}$$

The diagram illustrates the convolution process between an input image  $I$  and a kernel  $K$ . The input  $I$  is a 7x7 matrix with values 0 or 1. The kernel  $K$  is a 3x3 matrix with values 0 or 1. The result of the convolution,  $I * K$ , is a 5x5 matrix. The convolution step is shown with blue dotted lines indicating the receptive field of each output unit in  $I * K$ . The result is highlighted with a green box around the value 4.

## 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	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

**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

Detailed description: The diagram shows the convolution of a 7x7 input image  $I$  with a 3x3 kernel  $K$ . The result is a 5x5 output image  $I * K$ . The input  $I$  has a red 3x3 receptive field highlighted in the second row, third column. The kernel  $K$  is shown below it. The result  $I * K$  is shown on the right, with a green box highlighting the top-left element (1,4,3,4,1). Dotted lines connect the highlighted elements in  $I$  to the corresponding element in the result. The multiplication operation (\*) is placed between the input and kernel.

## Convolución de una imagen con un filtro (4)

$$\begin{array}{|c|c|c|c|c|c|c|c|} \hline 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ \hline \textcolor{red}{0} & \textcolor{red}{0} & \textcolor{red}{0} & 1 & 1 & 0 & 0 \\ \hline \textcolor{red}{0} & \textcolor{red}{0} & \textcolor{red}{0} & \textcolor{blue}{\times_0} & \textcolor{blue}{\times_1} & 1 & 0 \\ \hline \textcolor{red}{0} & \textcolor{red}{0} & \textcolor{red}{1} & 1 & 0 & 0 & 0 \\ \hline \textcolor{red}{0} & 1 & 1 & 0 & 0 & 0 & 0 \\ \hline 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ \hline \end{array} \quad * \quad \begin{array}{|c|c|c|} \hline 1 & 0 & 1 \\ \hline 0 & 1 & 0 \\ \hline 1 & 0 & 1 \\ \hline \end{array} \quad = \quad \begin{array}{|c|c|c|c|c|} \hline 1 & 4 & 3 & 4 & 1 \\ \hline 1 & 2 & 4 & 3 & 3 \\ \hline 1 & 2 & 3 & 4 & 1 \\ \hline \textcolor{green}{1} & \textcolor{green}{3} & \textcolor{green}{3} & \textcolor{green}{1} & 1 \\ \hline 3 & 3 & 1 & 1 & 0 \\ \hline \end{array}$$

**K**                                    **$I * K$**

# Mapas de activación

- Mapas de activación pueden verse como volúmenes

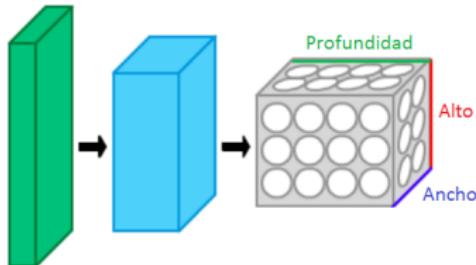


Imagen tomada de Wikipedia (Convolutional Neural Network)

- Dimensiones están dadas por  $H^{\{\ell\}} \times W^{\{\ell\}} \times N_{filters}$ , donde

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

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

## Submuestreo o decimación

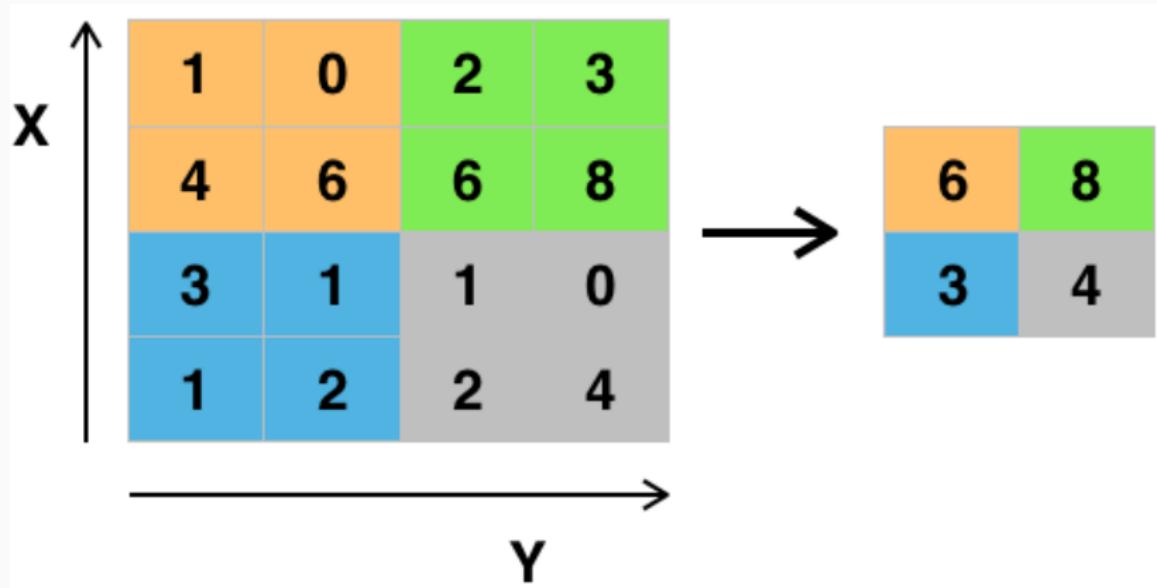


Imagen tomada de Wikipedia (Convolutional Neural Network)

# Retro-propagación en redes convolucionales

- 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 $1 \times 1$

- Funciona como un tipo de decimación en profundidad

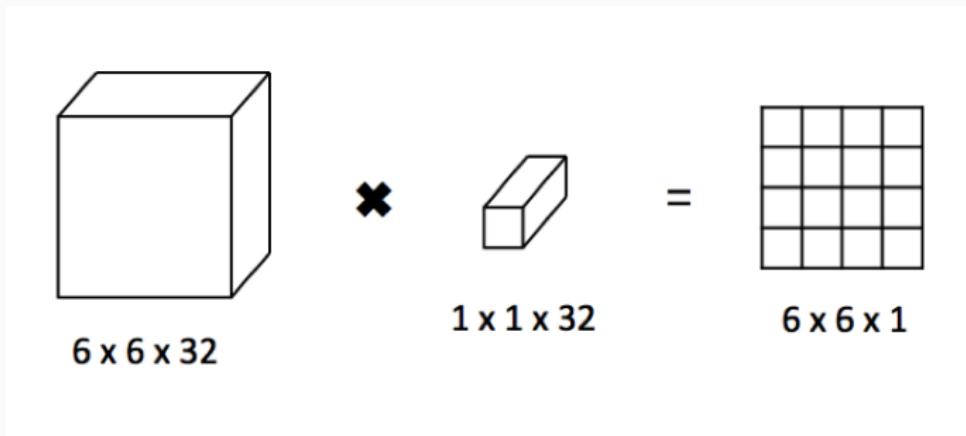


Imagen tomada de

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

# Arquitecturas de redes convolucionales: LeNet

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

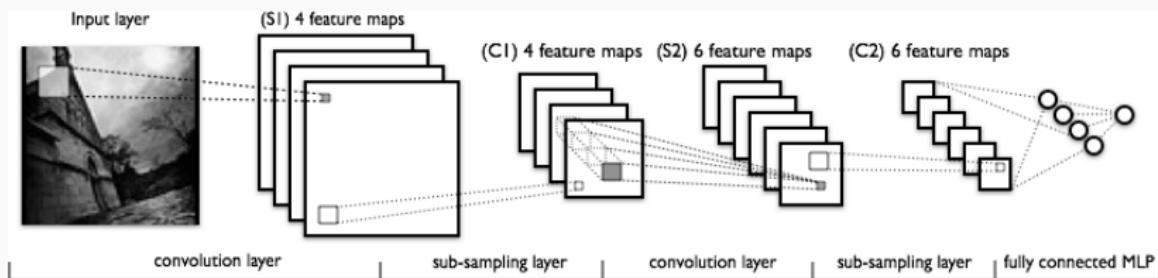


Imagen tomada de <http://deeplearning.net/tutorial/lenet.html>

# Arquitecturas de redes convolucionales: AlexNet

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

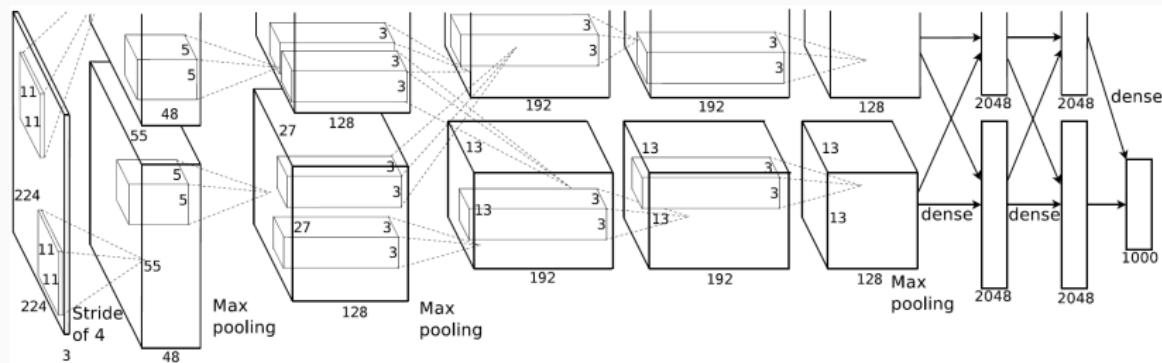


Imagen tomada de Krizhevsky et al. *ImageNet Classification with Deep Convolutional Neural Networks*, 2012

# Arquitecturas de redes convolucionales: ZFNet

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

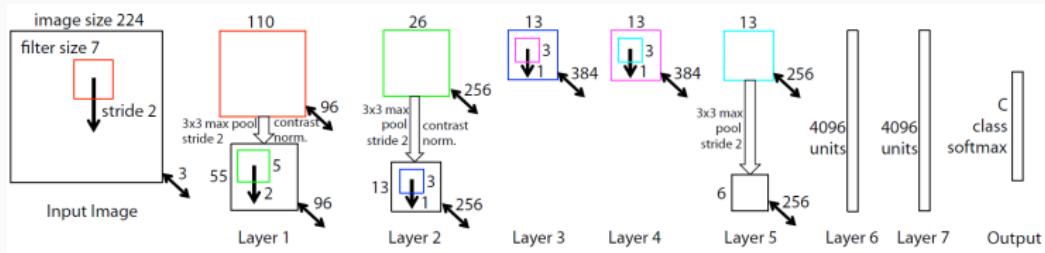


Imagen tomada de Zeiler and Fergus. *Visualizing and Understanding Convolutional Networks*, 2014

# Arquitecturas de redes convolucionales: VGGNet

- 19 capas
- Filtros de  $3 \times 3$  con desplazamientos de 1
- Max-pooling de  $2 \times 2$  con desplazamientos de 2



Imagen tomada de diapositivas de Simonyan (ILSVRC Workshop 2014)

# Arquitecturas de redes convolucionales: GoogleNet

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

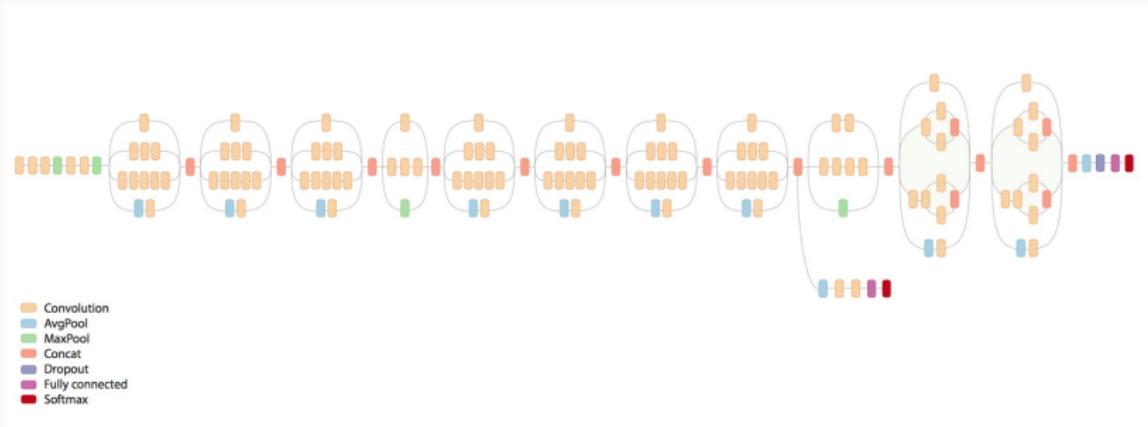
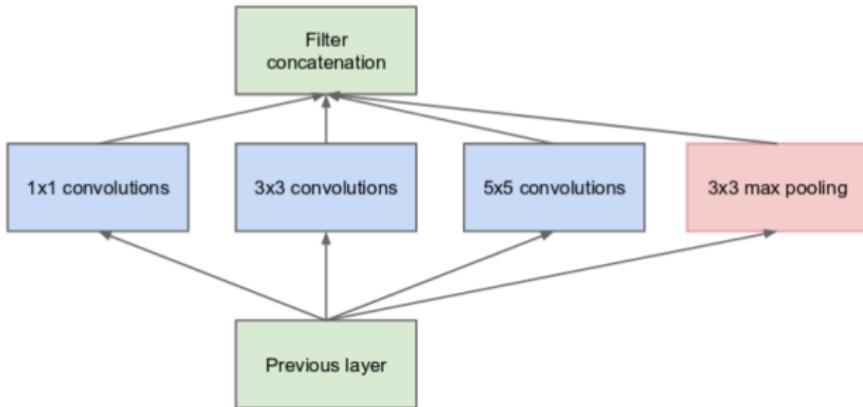


Imagen tomada de <https://research.googleblog.com/2016/03/train-your-own-image-classifier-with.html>

# Bloque Inception



(a) Inception module, naïve version

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

# Sobre profundidad en redes convolucionales (1)

## Revolution of Depth

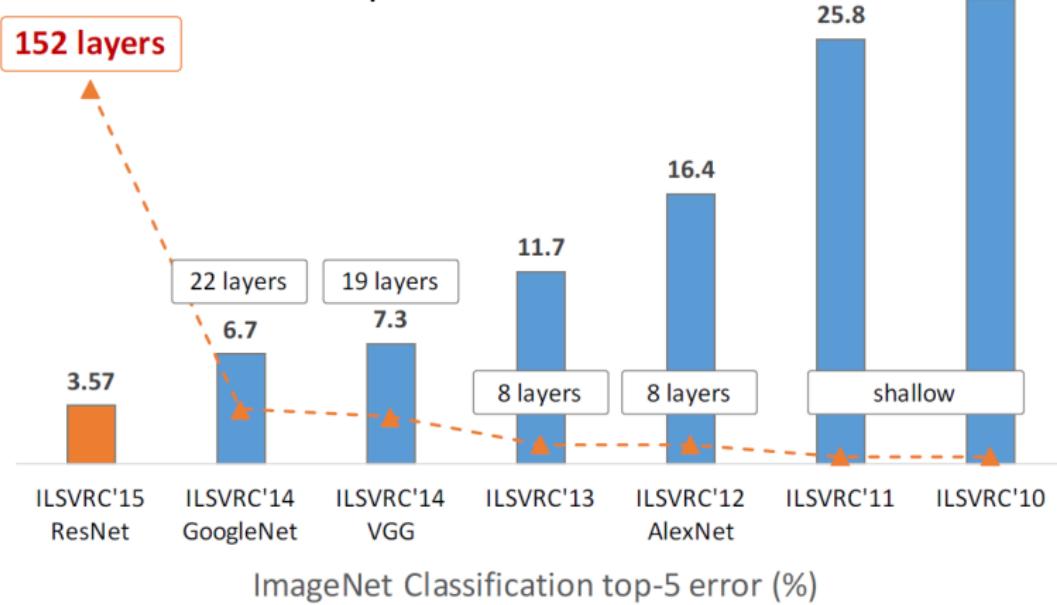


Imagen tomada de diapositivas de Kaiming He (ICML 2016)

## Sobre profundidad en redes convolucionales (2)

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

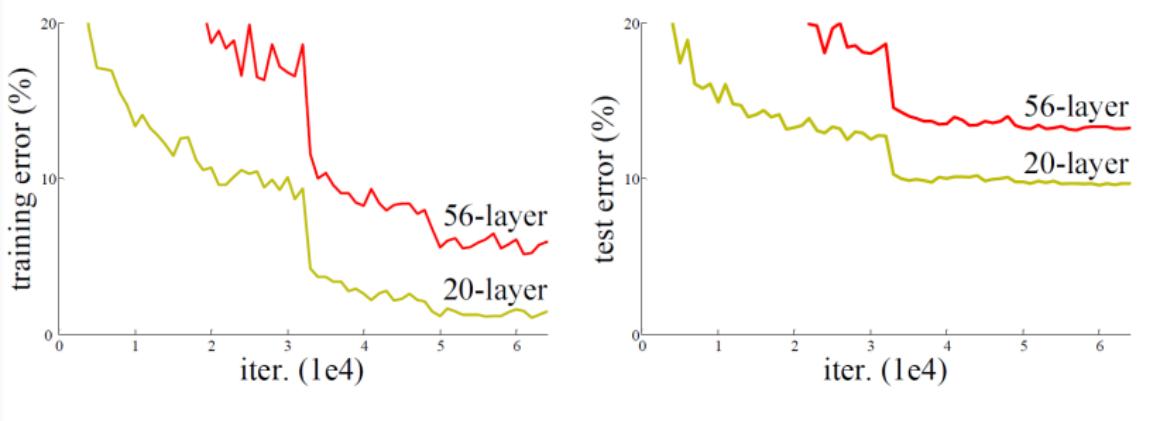


Imagen tomada de He et al. *Deep Residual Learning for Image Recognition*, 2015

# Entrenando redes más profundas: ResNet (1)

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

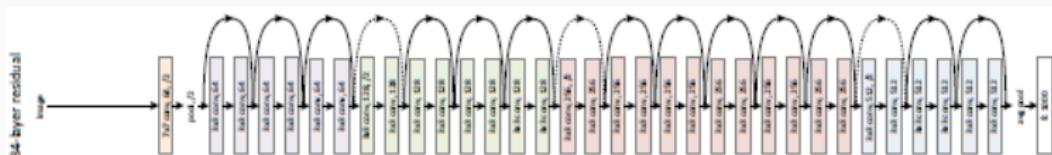
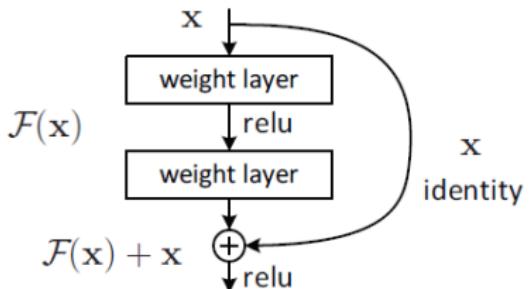


Imagen tomada de He et al. Deep Residual Learning for Image Recognition , 2015

## Entrenando redes más profundas: ResNet (2)

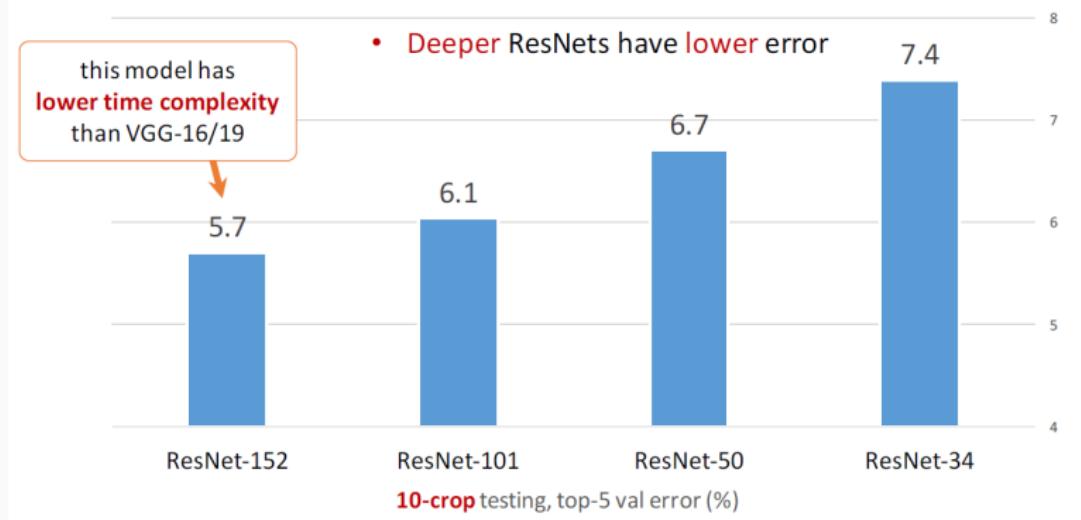


Imagen tomada de diapositivas de Kaiming He (ICML 2016)

# Redes convolucionales 1D

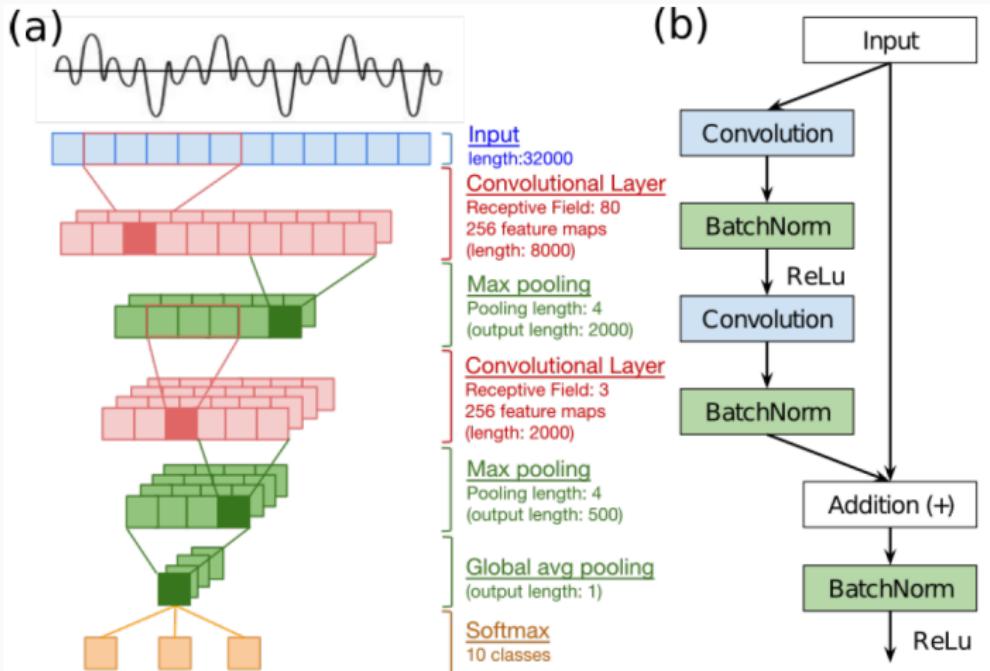


Imagen tomada de Dai et al. *Very Deep Convolutional Neural Networks for Raw Waveforms*, 2016

# Redes convolucionales 3D (1)

- Respuesta en redes convolucionales 2D en una imagen

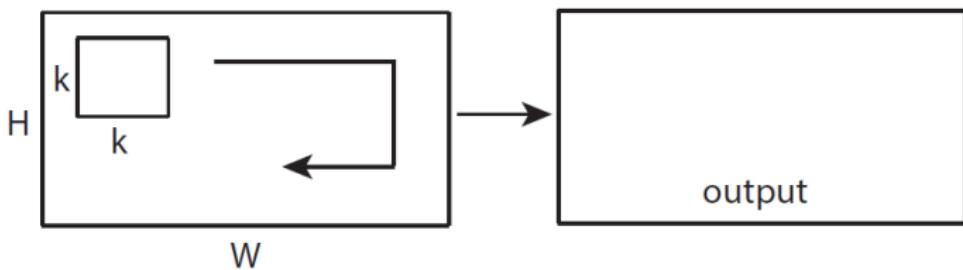


Imagen tomada de Tran et al. *Learning Spatiotemporal Features with 3D Convolutional Networks*, 2015

## Redes convolucionales 3D (2)

- Respuesta en redes convolucionales 2D en varias imágenes

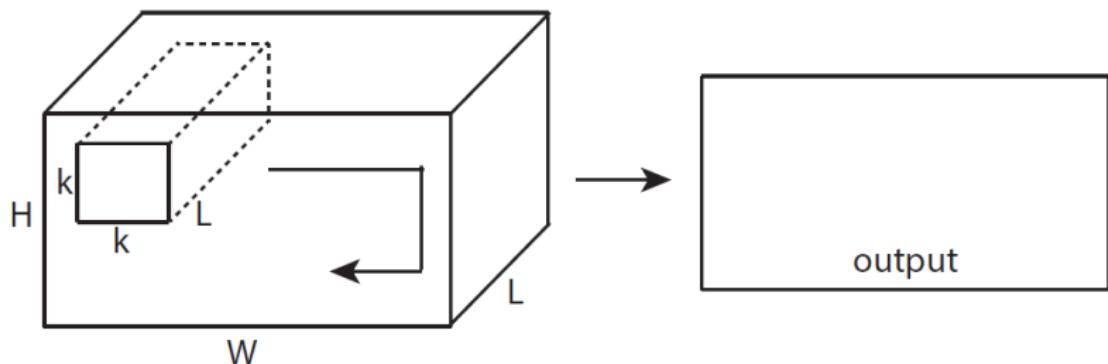


Imagen tomada de Tran et al. *Learning Spatiotemporal Features with 3D Convolutional Networks*, 2015

## Redes convolucionales 3D (3)

- Respuesta en redes convolucionales 3D

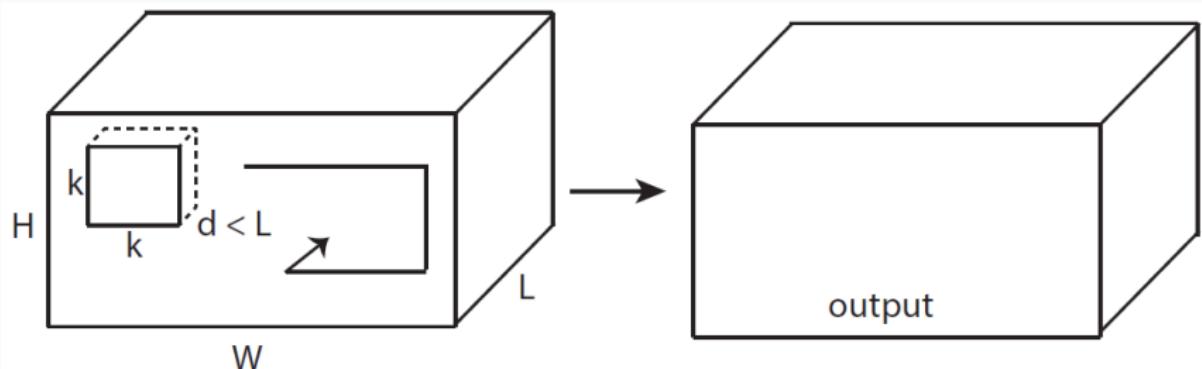


Imagen tomada de Tran et al. *Learning Spatiotemporal Features with 3D Convolutional Networks*, 2015

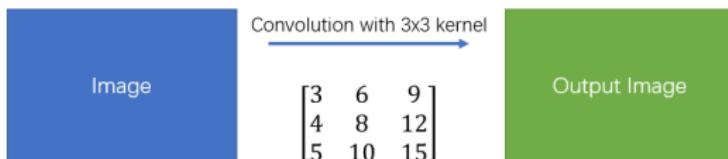
## Filtros separables

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

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

# Convolución separable espacialmente

## Simple Convolution



## Spatial Separable Convolution

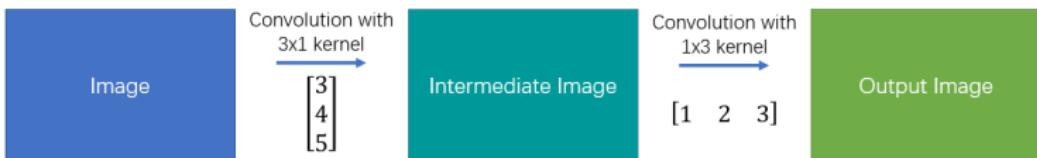


Imagen tomada de <https://towardsdatascience.com/a-basic-introduction-to-separable-convolutions-b99ec3102728>

# Convolución en profundidad

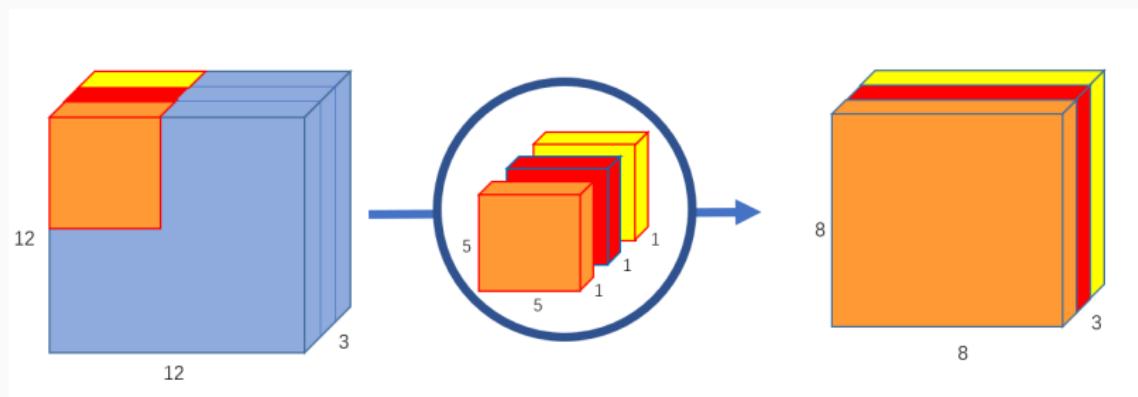


Imagen tomada de <https://towardsdatascience.com/a-basic-introduction-to-separable-convolutions-b99ec3102728>

# Convolución puntual (de $1 \times 1$ )

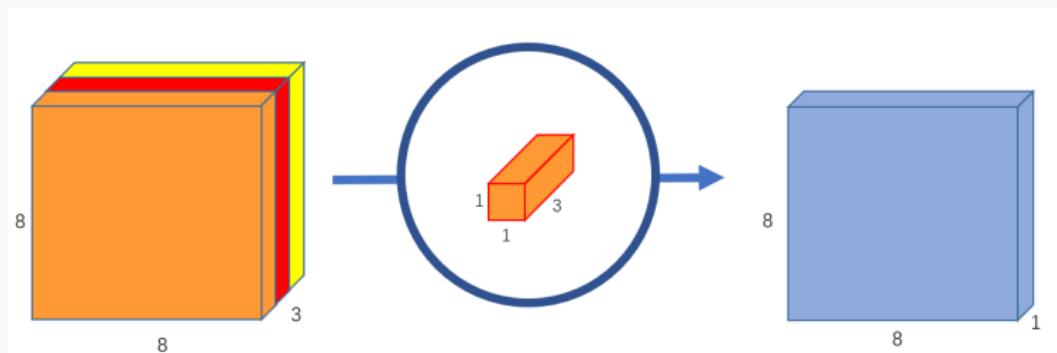


Imagen tomada de <https://towardsdatascience.com/a-basic-introduction-to-separable-convolutions-b99ec3102728>

# Convolución separable en profundidad

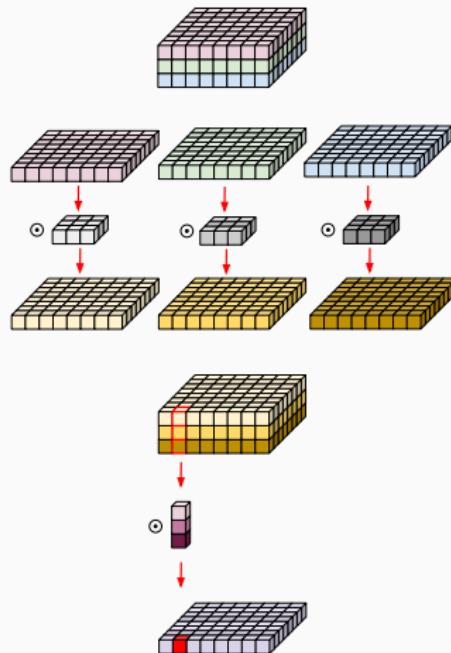


Imagen tomada de <https://eli.thegreenplace.net/2018/depthwise-separable-convolutions-for-machine-learning/>

# Convolución dilatada

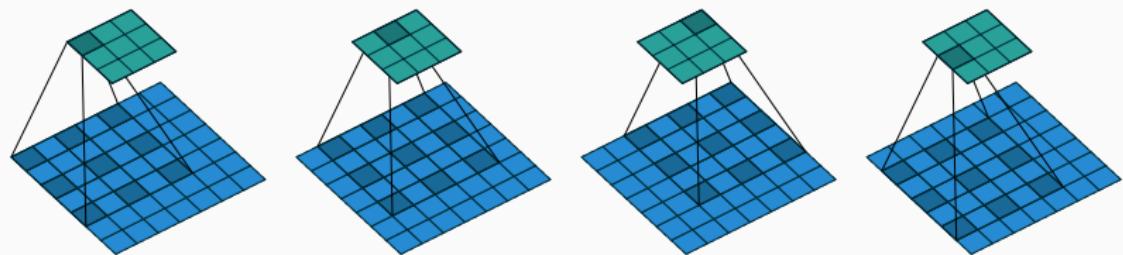


Imagen tomada de Dumoulin and Visin. *A guide to convolution arithmetic for deep learning*, 2018.

# Convolución transpuesta

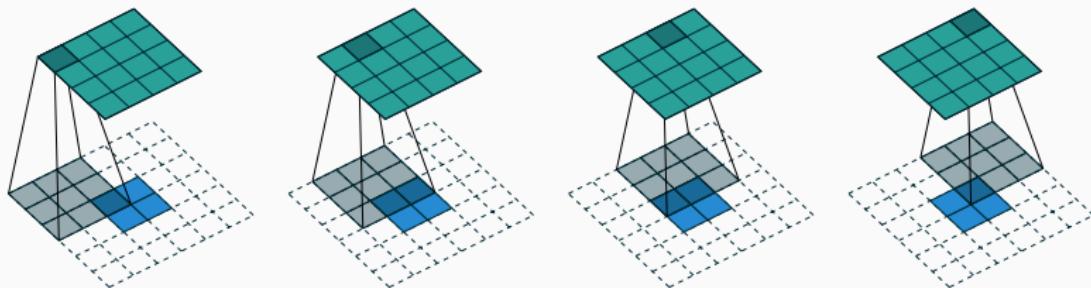


Imagen tomada de Dumoulin and Visin. *A guide to convolution arithmetic for deep learning*, 2018.

# Visualizando respuestas

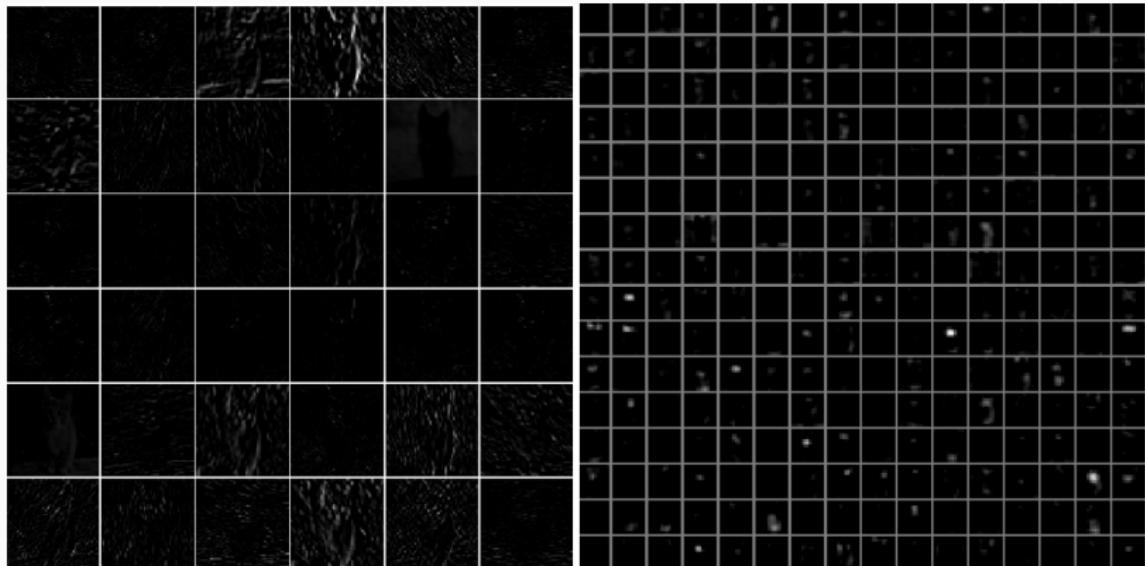


Imagen tomada de <http://cs231n.github.io/understanding-cnn/>

# Visualizando pesos

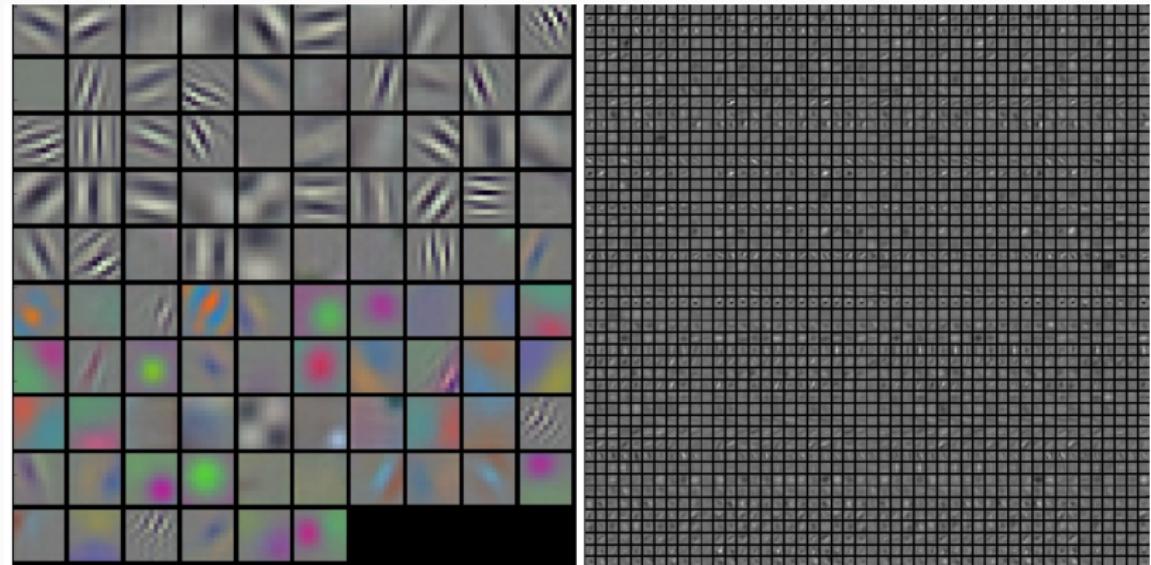


Imagen tomada de <http://cs23in.github.io/understanding-cnn/>

# Visualizando imágenes con respuestas máximas



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

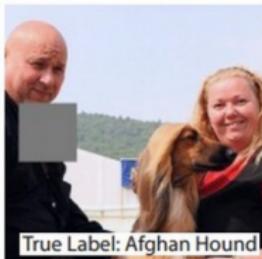
# Visualizando obstrucciones



True Label: Pomeranian



True Label: Car Wheel



True Label: Afghan Hound

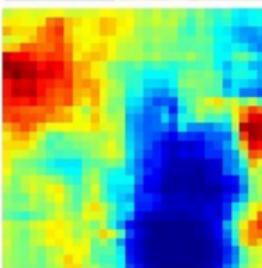
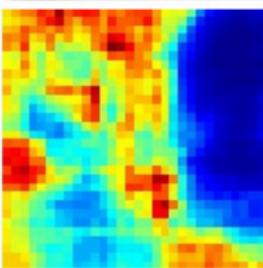
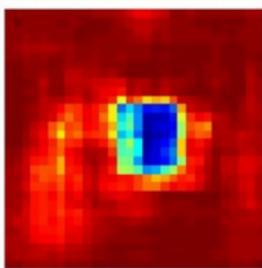


Imagen tomada de <http://cs231n.github.io/understanding-cnn/>

# Convolución transpuesta

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

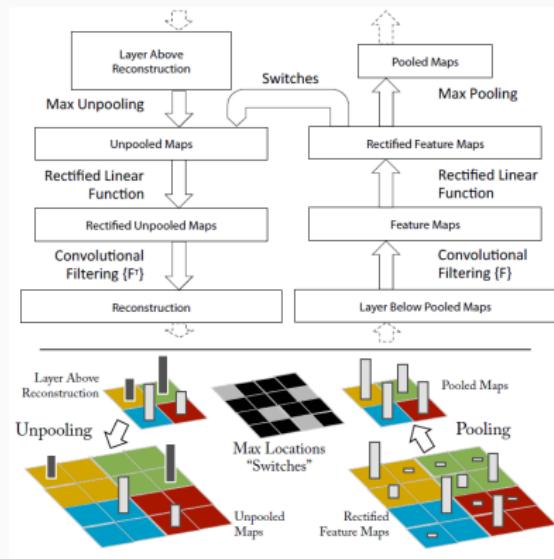


Imagen tomada de Zeiler and Fergus. *Visualizing and Understanding Convolutional Networks*, 2014

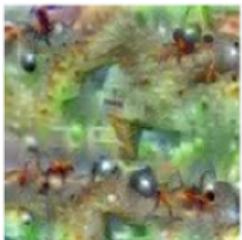
# DeepDream de Google: diferentes clases



Hartebeest



Measuring Cup



Ant



Starfish



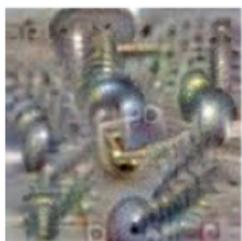
Anemone Fish



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

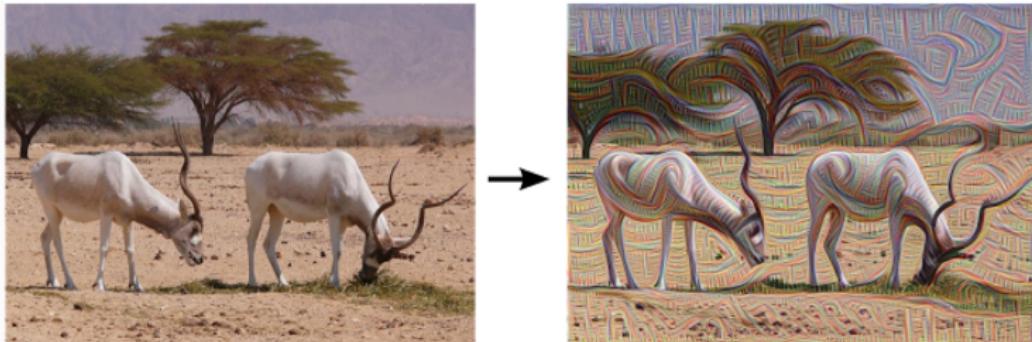


Imagen tomada de <https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html>

# 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



Horizon



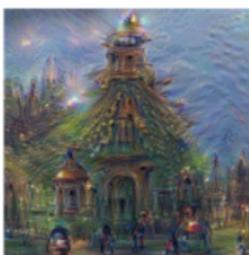
Trees



Leaves



Towers & Pagodas



Buildings



Birds & Insects

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>