

1a
Emisión

Módulo 8

Introducción al Deep Learning

Redes convolucionales

Instructor: Gibran Fuentes Pineda



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Dirección General de Cómputo y de Tecnologías de información y Comunicación
Dirección de Docencia en TIC



Educación
Continua
1971 - 2021

Objetivo del tema



- ▶ El participante identificará los componentes y el funcionamiento de las redes neuronales convolucionales, así como estrategias para entrenarlas.

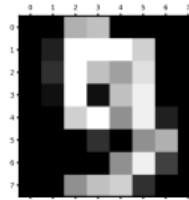
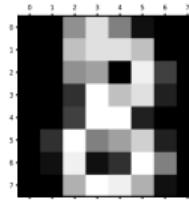
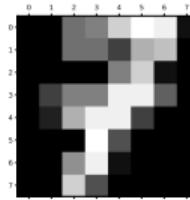
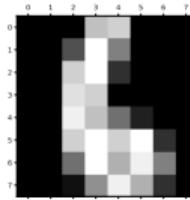
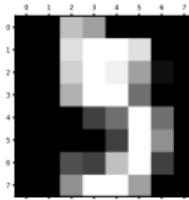
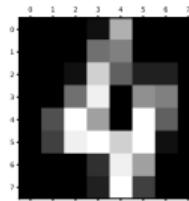
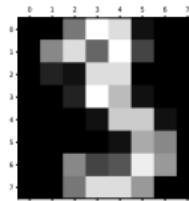
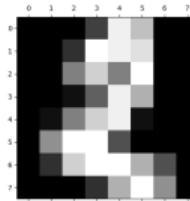
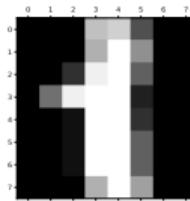
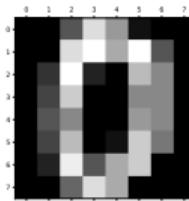
Sub-temas



- 6.1 Motivación
- 6.2 La operación de convolución
- 6.3 Submuestreo
- 6.4 Retropropagación en capas convolucionales y de submuestreo
- 6.5 Arquitecturas de redes neuronales convolucionales
- 6.6 Aprendizaje por transferencia
- 6.7 Acrecentamiento de datos
- 6.8 Ejemplo práctico: clasificación de imágenes

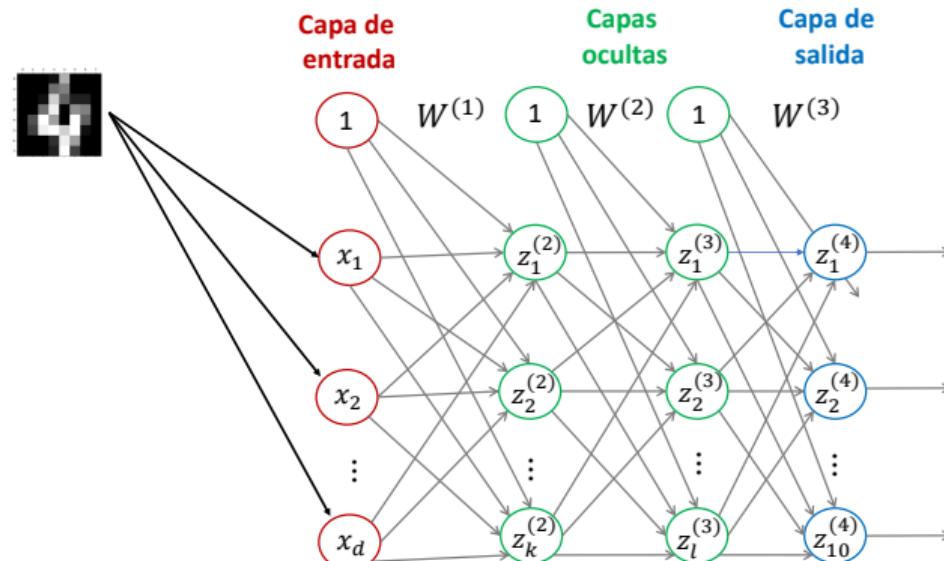
Motivación: clasificación de imágenes

- ▶ Por ejemplo, clasificar dígitos escritos a mano



Motivación: 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



Operación de convolución: definición

- ▶ 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)

- $$S(i, j) = (I * K)(i, j) = \sum_m \sum_n I(i + m, j + n)K(m, n)$$

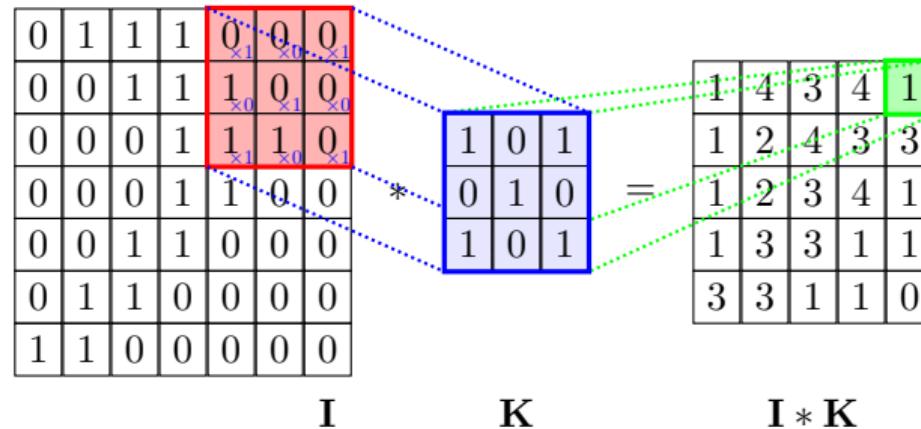
Operación de convolución: imágenes

$$\begin{matrix} \begin{matrix} 0 & 1 & 1 \\ \times 1 & \times 0 & \times 1 \\ 0 & 0 & 1 \end{matrix} & \begin{matrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{matrix} \\ I & \end{matrix} * \begin{matrix} \begin{matrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{matrix} \\ K \end{matrix} = \begin{matrix} \begin{matrix} 1 & 4 & 3 & 4 & 1 \\ 1 & 2 & 4 & 3 & 3 \\ 1 & 2 & 3 & 4 & 1 \\ 1 & 3 & 3 & 1 & 1 \\ 3 & 3 & 1 & 1 & 0 \end{matrix} \\ I * K \end{matrix}$$

Operación de convolución: imágenes

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

Operación de convolución: imágenes



Operación de convolución: imágenes

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

K **$I * K$**

Operación de convolución: capas convolucionales

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

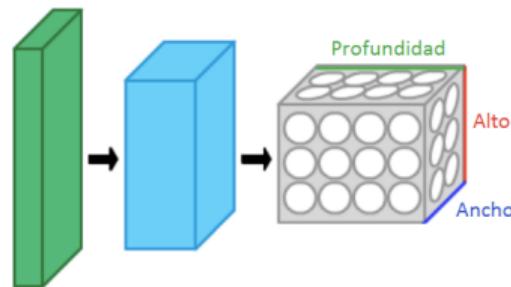
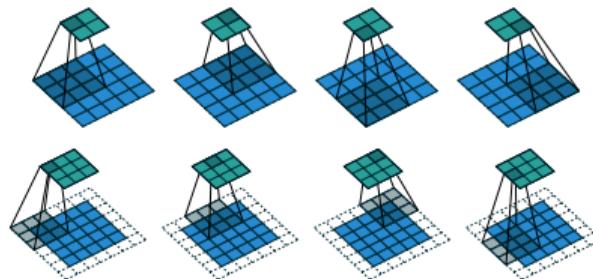


Imagen tomada de Wikipedia (Convolutional Neural Network)

Operación de convolución: 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.

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

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

Operación de convolución: dilatada

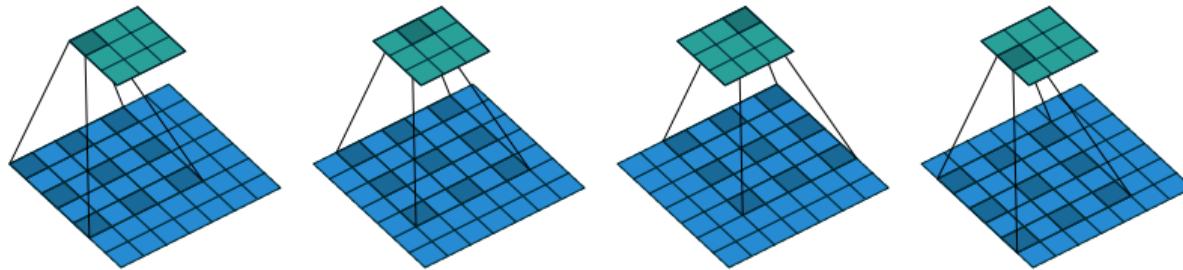


Imagen tomada de Dumoulin and Visin. *A guide to convolution arithmetic for deep learning*, 2018.

Operación de convolución: filtros de 1×1

- ▶ Funciona como un tipo de decimación en profundidad

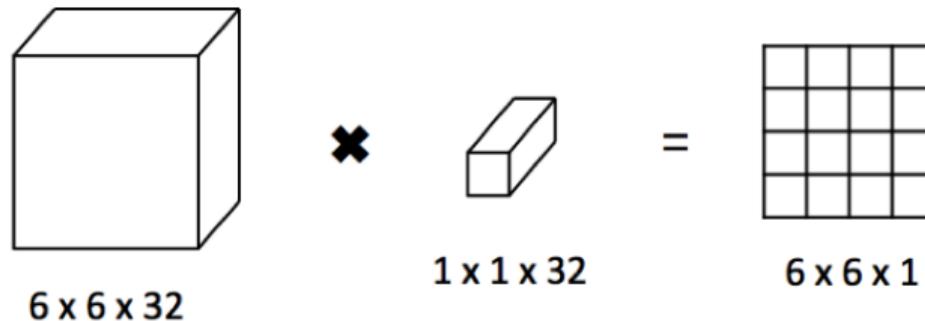


Imagen tomada de <https://medium.com/machine-learning-bites/deeplearning-series-convolutional-neural-networks-a9c2f2ee1524>

Operación de convolución: filtros separables

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

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

Operación de convolución: separable espacialmente

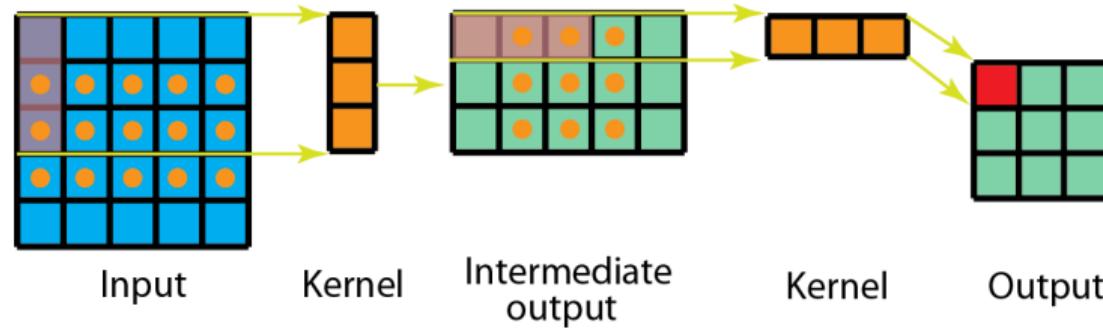


Imagen tomada de Kunlun Bai. A Comprehensive Introduction to Different Types of Convolutions in Deep Learning, 2019.

Operación de convolución: separable en profundidad (1)

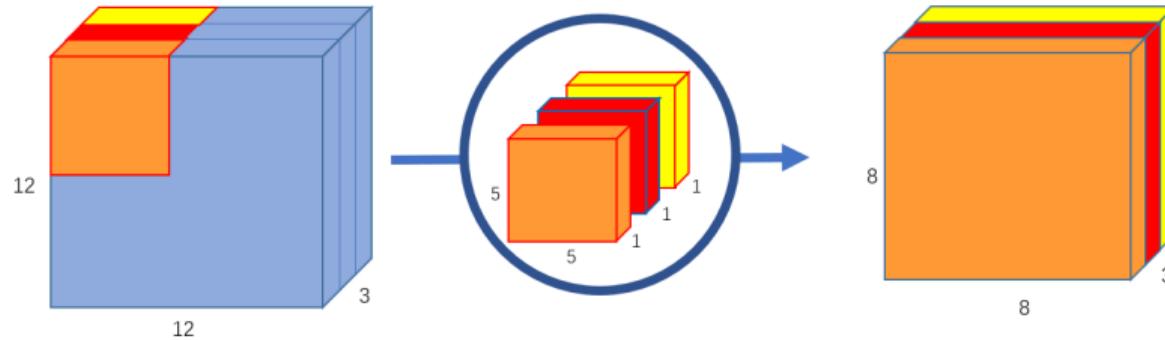


Imagen tomada de Chi-Feng Wang. A Basic Introduction to Separable Convolutions, 2018.

Operación de convolución: separable en profundidad (2)

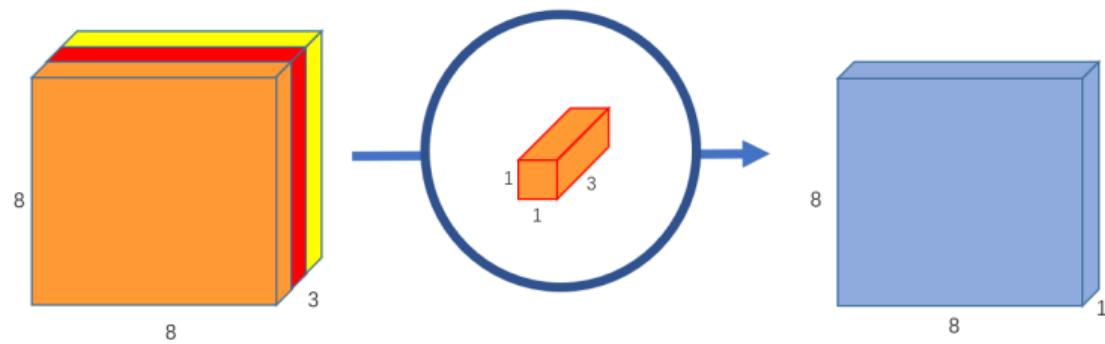


Imagen tomada de Chi-Feng Wang. A Basic Introduction to Separable Convolutions, 2018.

Operación de convolución: separable en profundidad (3)

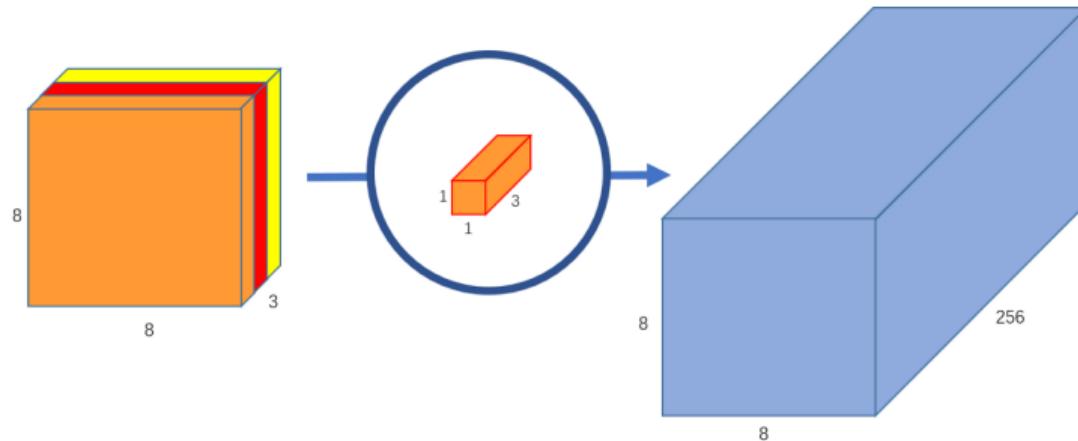


Imagen tomada de Chi-Feng Wang. A Basic Introduction to Separable Convolutions, 2018.

Operación de convolución: separable en profundidad (4)

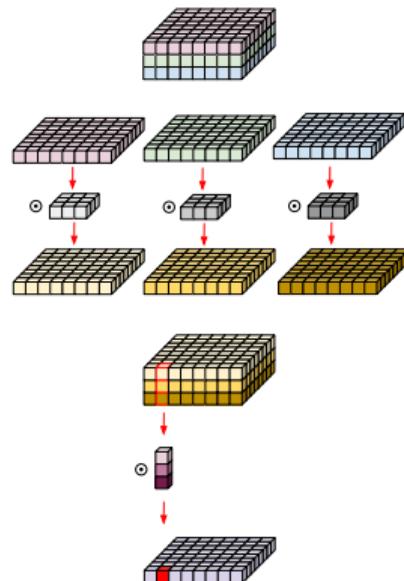


Imagen tomada de Eli Bendersky. Depthwise separable convolutions for machine learning, 2018.

Operación de convolución: procesando datos 1D

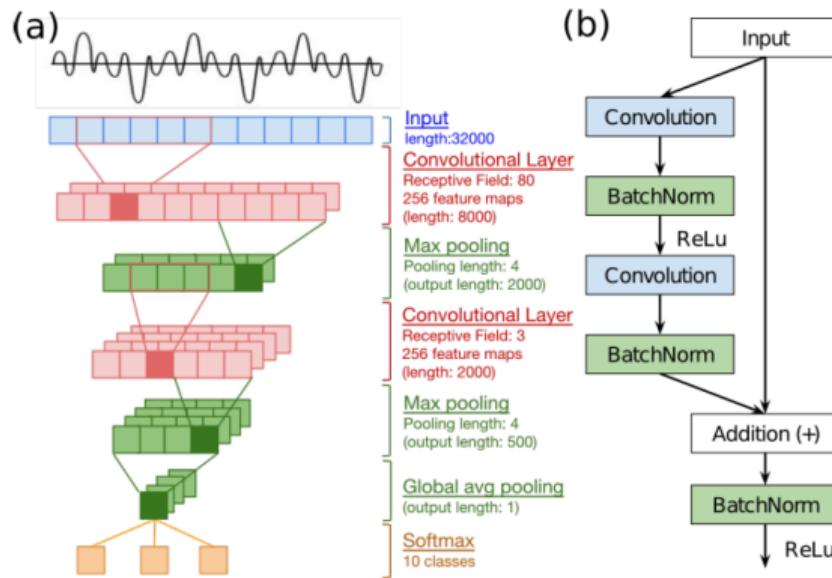


Imagen tomada de Dai et al. *Very Deep Convolutional Neural Networks for Raw Waveforms*, 2016

Operación de convolución: procesando datos 3D (1)

- ▶ Respuesta en redes convolucionales 2D en una imagen

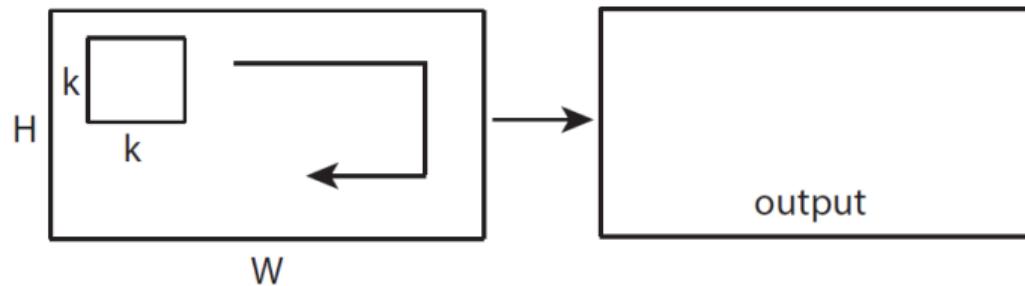


Imagen tomada de Tran et al. *Learning Spatiotemporal Features with 3D Convolutional Networks*, 2015

Operación de convolución: procesando datos 3D (2)

- ▶ Respuesta en redes convolucionales 2D en varias imágenes

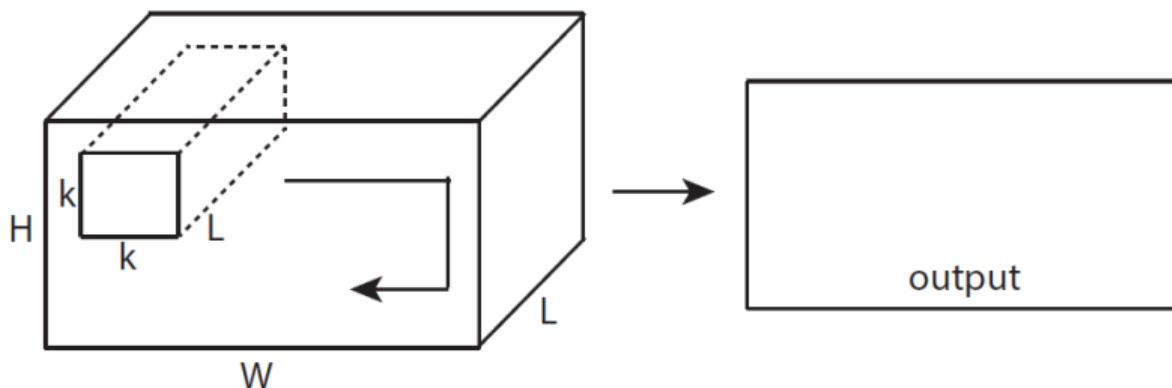


Imagen tomada de Tran et al. *Learning Spatiotemporal Features with 3D Convolutional Networks*, 2015

Operación de convolución: procesando datos 3D (3)

- ▶ Respuesta en redes convolucionales 3D

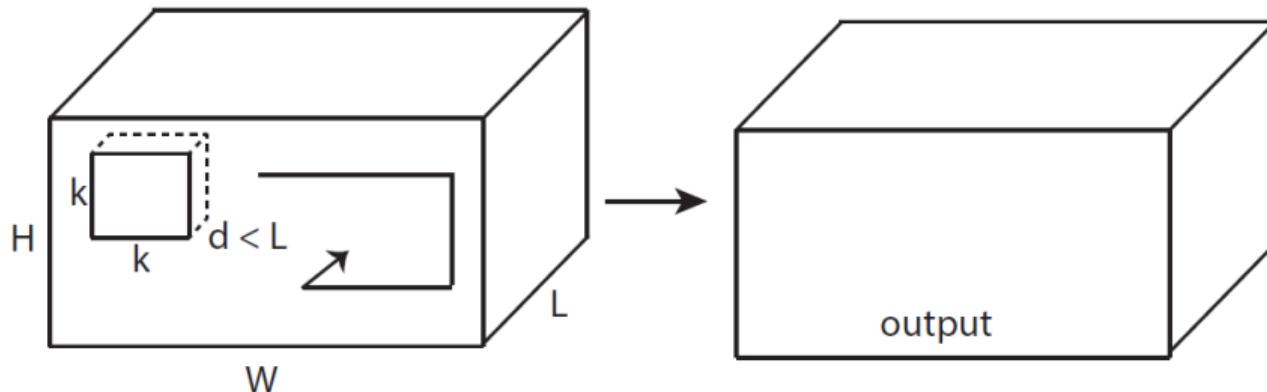


Imagen tomada de Tran et al. *Learning Spatiotemporal Features with 3D Convolutional Networks*, 2015

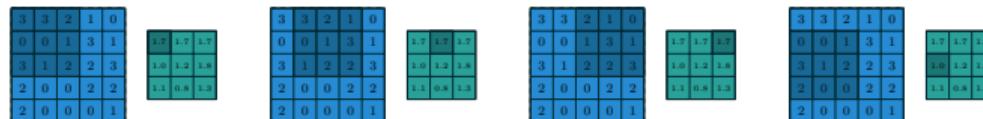
Submuestreo: operación y variantes

- ▶ 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.

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

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

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

Submuestro: promedio de mapa de activaciones

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

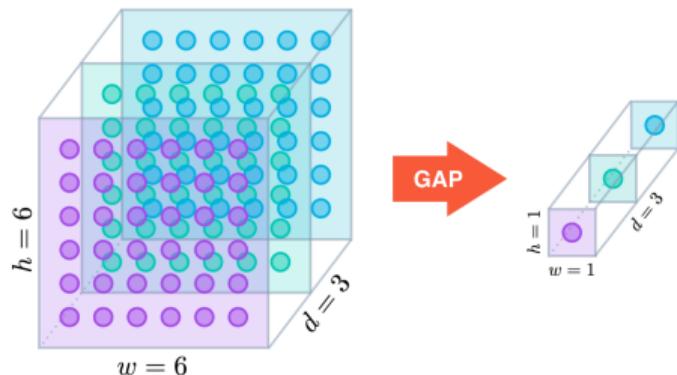


Imagen tomada de <https://alexisbcook.github.io/2017/global-average-pooling-layers-for-object-localization/>

Retro-propagación en capas convolucionales y de submuestreo

- ▶ 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*)

Arquitecturas de redes neuronales convolucionales

- ▶ 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)

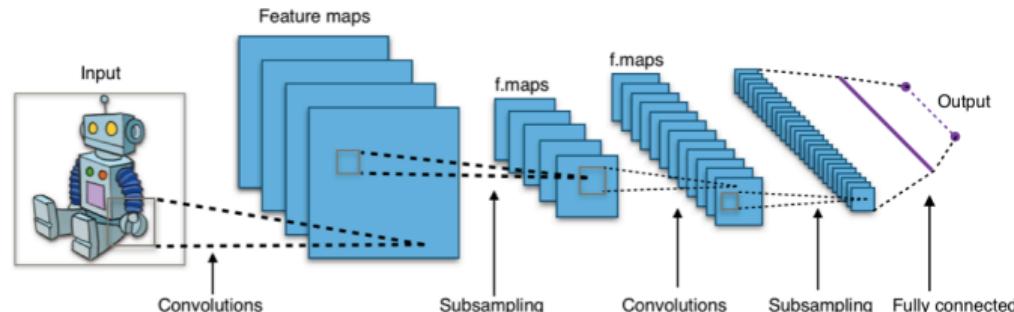


Imagen del usuario Aphex34 de Wikipedia. CC BY-SA 4.0

Arquitecturas de redes convolucionales: LeNet

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

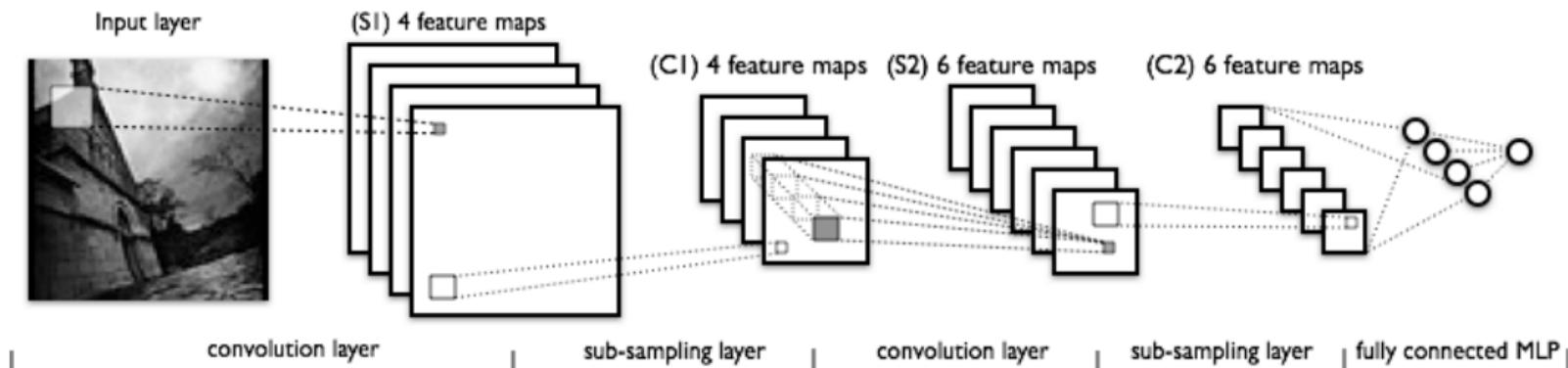


Imagen tomada de <http://deeplearning.net/tutorial/lenet.html>

Arquitecturas de redes convolucionales: AlexNet

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

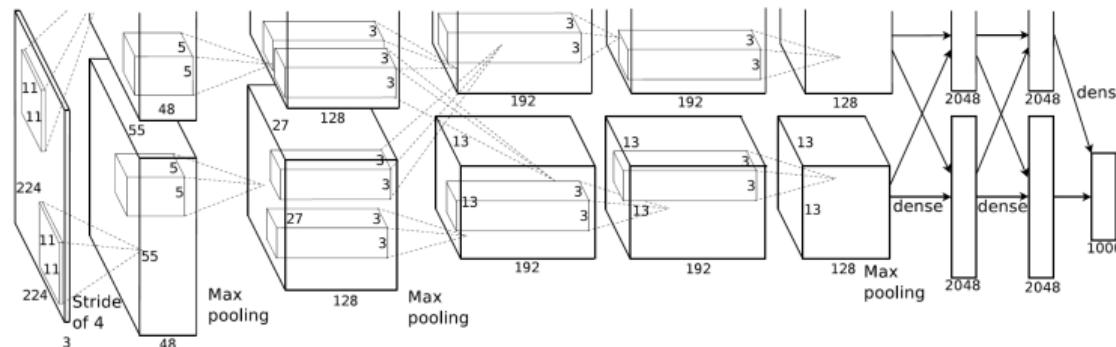


Imagen tomada de Krizhevsky et al. *ImageNet Classification with Deep Convolutional Neural Networks*, 2012

Arquitecturas de redes convolucionales: 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)

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

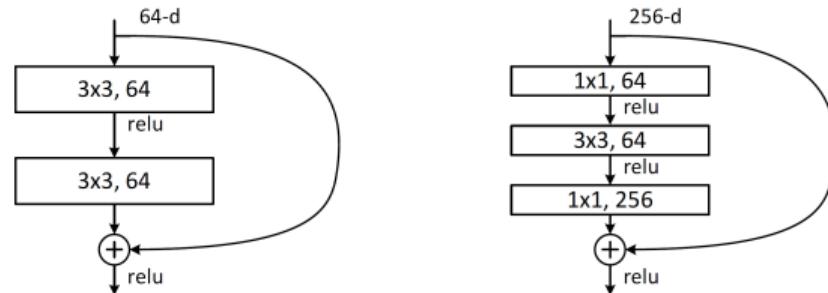


Imagen tomada de He et al. Deep Residual Learning for Image Recognition, 2015

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

Arquitecturas de redes neuronales convolucionales: ResNet (1)

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112			7×7, 64, stride 2		
				3×3 max pool, stride 2		
conv2_x	56×56	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
conv3_x	28×28	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 8$
conv4_x	14×14	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 23$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 36$
conv5_x	7×7	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$
	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

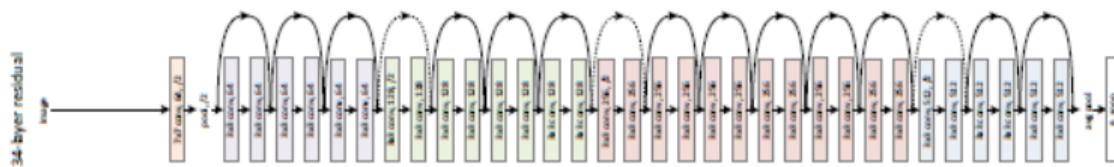


Imagen y tabla tomadas de He et al. Deep Residual Learning for Image Recognition, 2015

Aprendizaje por transferencia

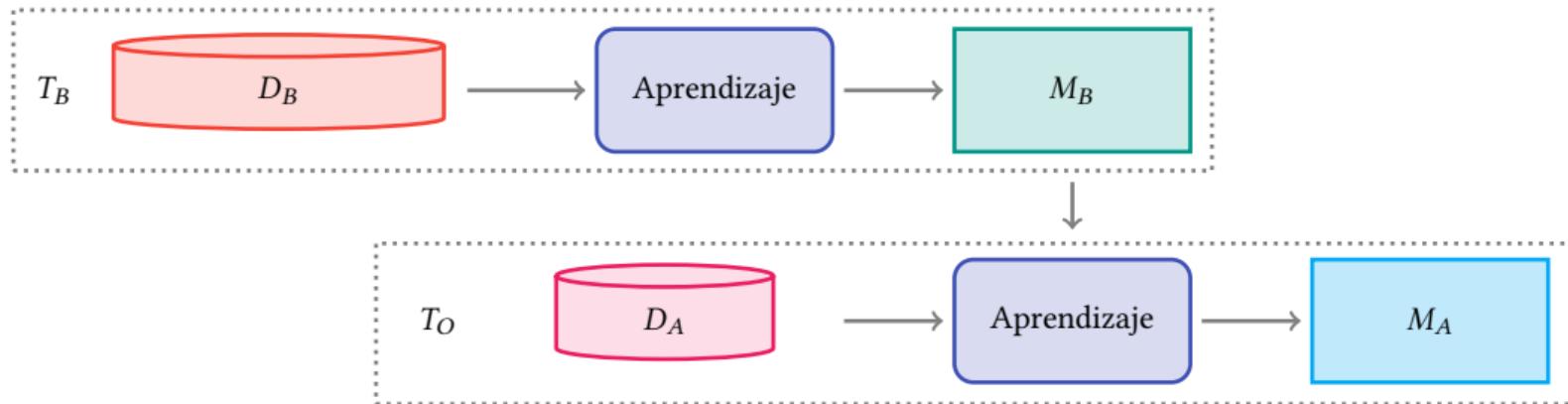


Imagen cortesía de Berenice Montalvo

Aprendizaje por transferencia: desempeño en distintas tareas (1)

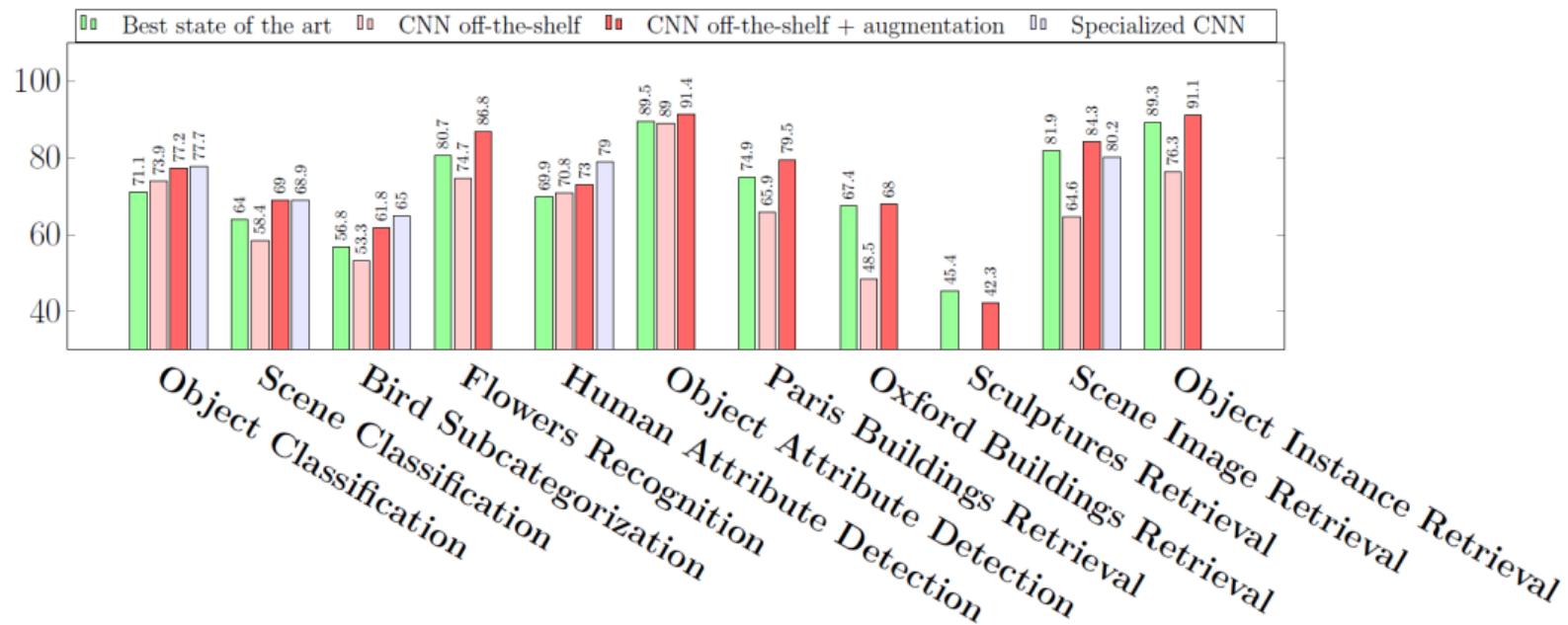


Imagen tomada de Razavian et al. CNN Features off-the-shelf: an Astounding Baseline for Recognition, 2014

Aprendizaje por transferencia: desempeño en distintas tareas (2)

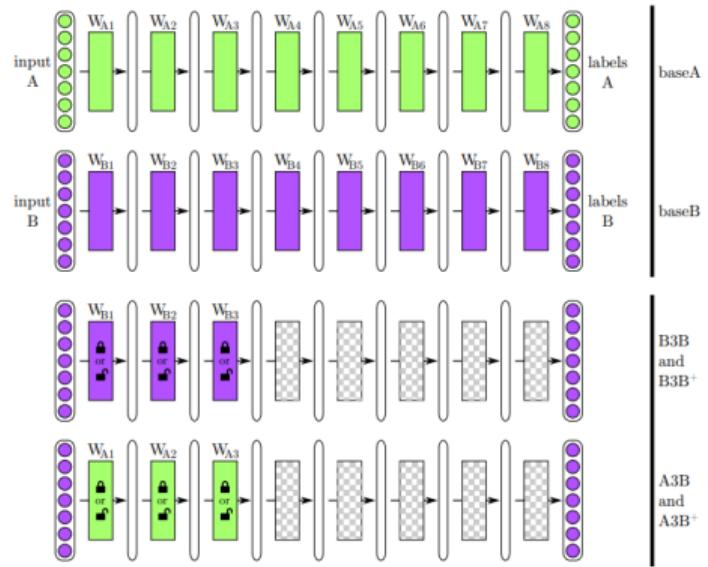


Imagen tomada de Yosinski et al. How transferable are features in deep neural networks?, 2014

Aprendizaje por transferencia: desempeño en distintas tareas (3)

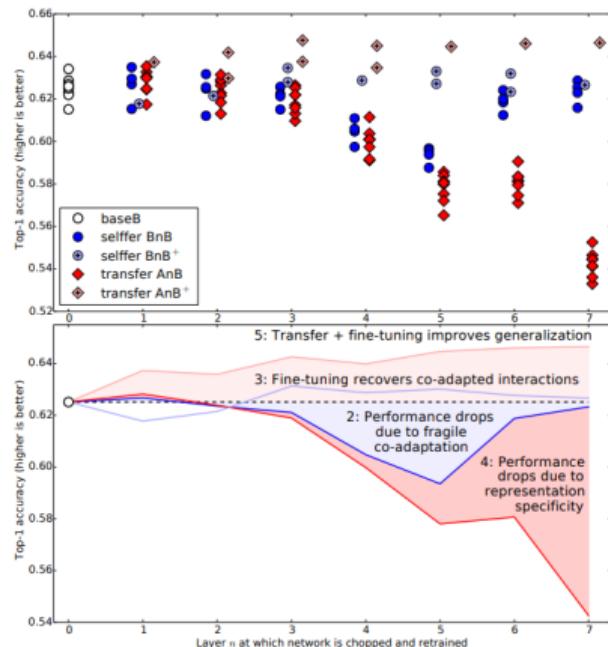


Imagen tomada de Yosinski et al. How transferable are features in deep neural networks?, 2014

Acrecentamiento de datos: transformaciones

$$\begin{bmatrix} 1 & 0 & 0 & v_x \\ 0 & 1 & 0 & v_y \\ 0 & 0 & 1 & v_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- ▶ Traslación

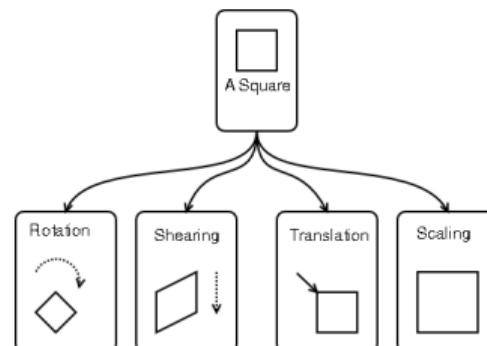


Imagen tomada de <https://people.gnome.org/~mathieu/libart/libart-affine-transformation-matrices.html>

Acrecentamiento de datos: transformaciones

$$\begin{bmatrix} v_x & 0 & 0 \\ 0 & v_y & 0 \\ 0 & 0 & v_z \end{bmatrix}$$

- ▶ Traslación
- ▶ Rescalado

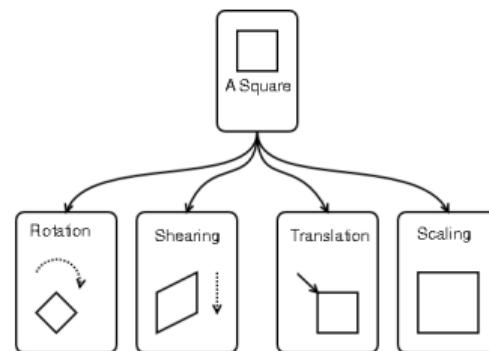


Imagen tomada de <https://people.gnome.org/~mathieu/libart/libart-affine-transformation-matrices.html>

Acrecentamiento de datos: transformaciones

$$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- ▶ Traslación
- ▶ Rescalado
- ▶ Giro

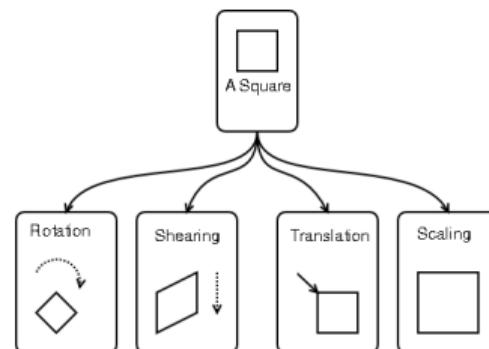


Imagen tomada de <https://people.gnome.org/~mathieu/libart/libart-affine-transformation-matrices.html>

Acrecentamiento de datos: transformaciones

- ▶ Traslación
- ▶ Rescalado
- ▶ Giro
- ▶ Rotación

$$\begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

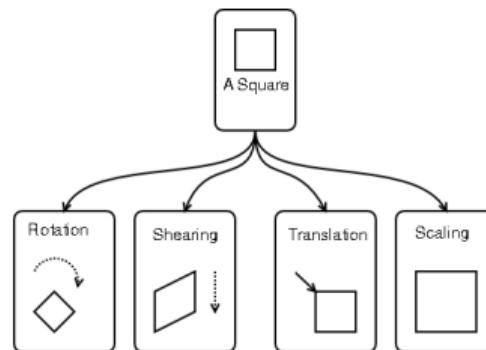


Imagen tomada de <https://people.gnome.org/~mathieu/libart/libart-affine-transformation-matrices.html>

Acrecentamiento de datos: transformaciones

- ▶ Traslación
- ▶ Rescalado
- ▶ Giro
- ▶ Rotación
- ▶ Deformación de corte

$$\begin{bmatrix} 1 & c_x = 0.5 & 0 \\ c_y = 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

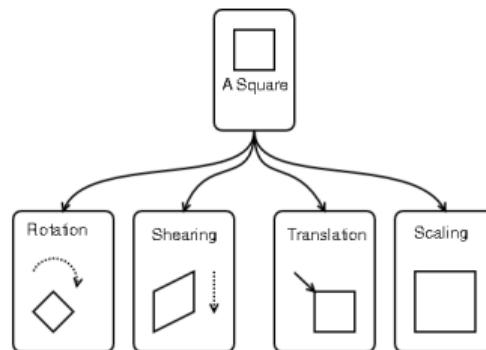


Imagen tomada de <https://people.gnome.org/~mathieu/libart/libart-affine-transformation-matrices.html>

Acrecentamiento de datos: transformaciones

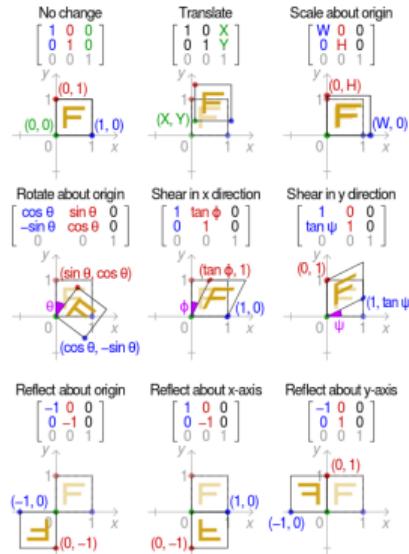


Imagen del usuario de Wikipedia Cmglee (entrada Affine transformation). CC BY-SA 3.0

Acrecentamiento de datos: ejemplos

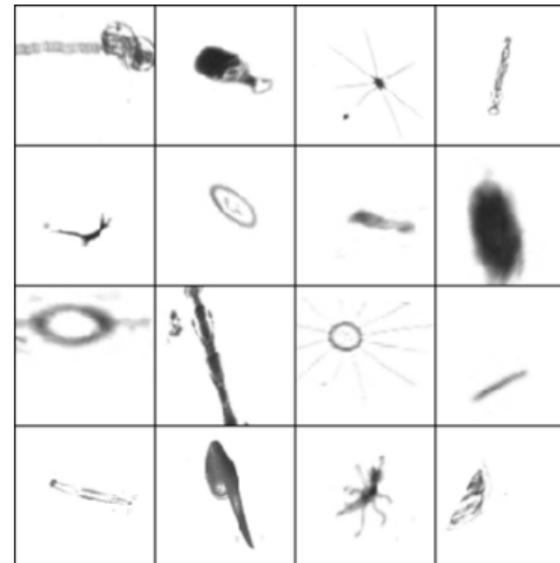
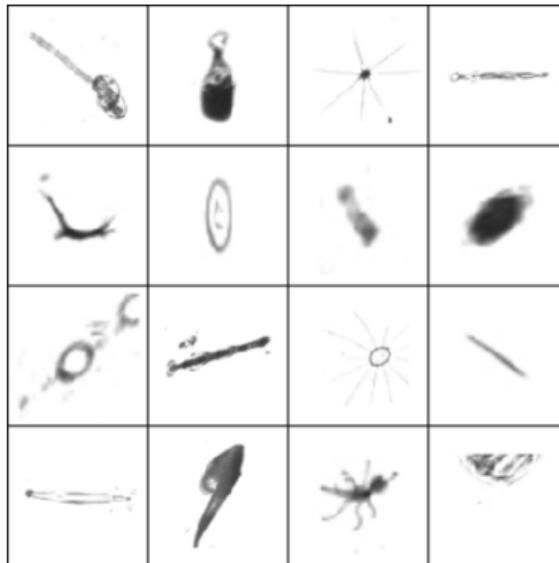


Imagen tomada de Dieleman. Classifying plankton with deep neural networks, 2015.

Ejemplo práctico: clasificación de imágenes con ImageNet



IMAGENET 14,197,122 images, 21841 synsets indexed

Explore Download Challenges Publications Updates About

Not logged in. Login | Signup

ImageNet is an image database organized according to the WordNet hierarchy (currently only the nouns), in which each node of the hierarchy is depicted by hundreds and thousands of images. Currently we have an average of over five hundred images per node. We hope ImageNet will become a useful resource for researchers, educators, students and all of you who share our passion for pictures.

Click here to learn more about ImageNet. Click here to join the ImageNet mailing list.

What do these images have in common? Find out!

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

Referencias



- ▶ Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. (2021). Dive into deep learning. Capítulos 6 y 7. Disponible en <http://www.d2l.ai/>.
- ▶ Murphy, K. P. (2022). Probabilistic Machine Learning: An introduction. MIT Press. Capítulo 13. Disponible en <https://probml.github.io/pml-book/book1.html>.
- ▶ Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press. Capítulo 9. Disponible en <https://www.deeplearningbook.org/>.
- ▶ Nielsen, M. (2019). Neural Networks and Deep Learning. Capítulo 6. Disponible en <http://neuralnetworksanddeeplearning.com/index.html>.
- ▶ Amidi, A. & Amidi, S. Disponible en <https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks>.
- ▶ Wang, J., Turko, R., Shaikh, O., Park, H., Das, N., Hohman, F., Kahng, M., & Chau P. CNN Explainer. Disponible en <https://poloclub.github.io/cnn-explainer/>.

Contacto

Gibran Fuentes Pineda
Investigador Asociado "C", IIMAS, UNAM

gibranfp@unam.mx

LinkedIn: @gibranfuentespineda