3ª Emisión

Módulo 08

Redes Convolucionales

Edgar Morales Palafox

DGTIC
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

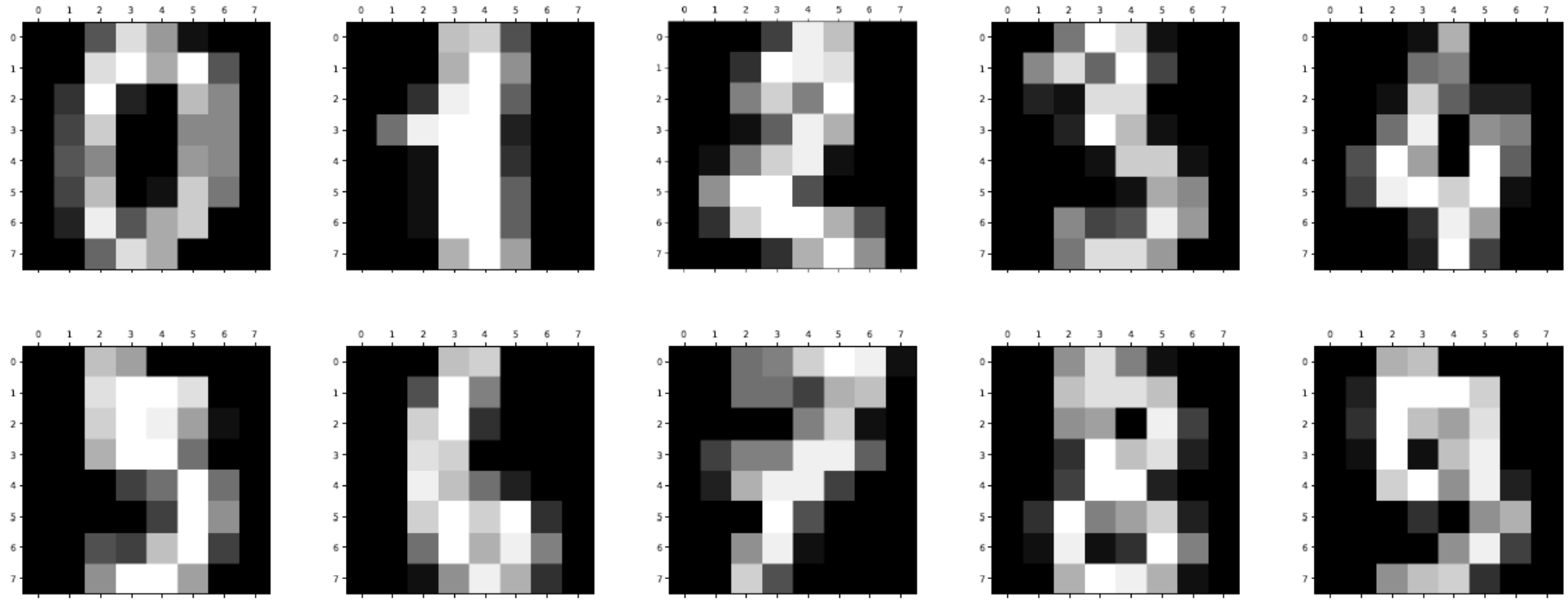
El participante identificará los componentes y el funcionamiento de las redes neuronales convolucionales, así como estrategias para entrenarlas.

Contenido

1. La operación de convolución.
2. Submuestreo.
3. Arquitecturas de redes neuronales convolucionales.
4. Aprendizaje por transferencia.
5. Acrecentamiento de datos
6. Ejemplo práctico: clasificación de imágenes.

La operación de convolución

Para comprender el funcionamiento de las redes convolucionales, usaremos el ejemplo de clasificar los dígitos



Representación de una imagen en pixeles

Matriz de pixeles (dos dimensiones) contiene valores entre 0 - 255 que describen la intensidad de blanco a negro.

=

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	Y
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	8	76	147	255	227	101	10	2	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	178	253	253	253	253	253	253	40	0	0	0	0	0	0	0
8	0	0	0	0	0	0	19	231	227	133	82	164	216	253	217	4	0	0	0	0	0	0
9	0	0	0	0	0	0	89	253	123	0	0	0	78	253	253	5	0	0	0	0	0	0
10	0	0	0	0	0	0	89	216	84	0	0	0	113	253	161	1	0	0	0	0	0	0
11	0	0	0	0	0	0	10	17	0	0	0	25	220	253	32	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	172	253	253	23	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	23	91	142	220	253	253	23	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	87	243	253	253	253	253	253	116	21	0	0	0	0	0	0
15	0	0	0	0	0	0	0	197	253	253	216	194	240	253	253	151	0	0	0	0	0	0
16	0	0	0	0	0	0	0	25	61	165	22	0	45	103	246	251	82	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	231	253	205	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	231	253	224	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	231	253	218	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	12	233	252	94	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	15	180	253	219	0	0	0	0	0	0
22	0	0	0	0	0	0	0	163	94	0	0	16	123	253	253	124	0	0	0	0	0	0
23	0	0	0	0	0	0	71	247	225	35	201	213	253	253	126	26	0	0	0	0	0	0
24	0	0	0	0	0	0	54	242	253	253	253	253	223	109	4	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	65	135	130	135	108	12	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
29																						

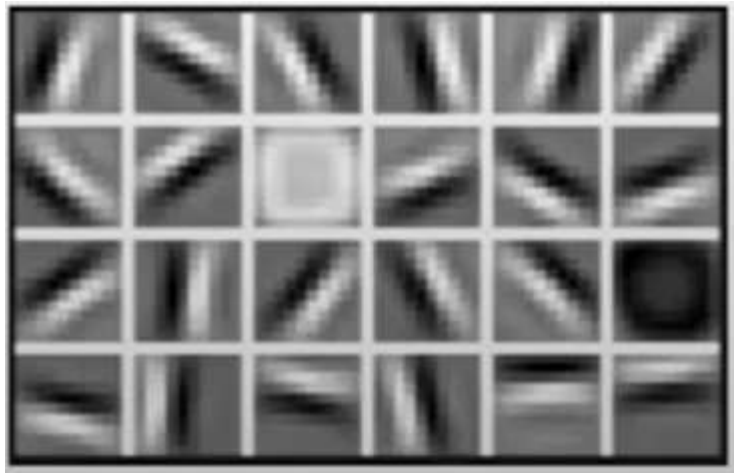
Imagen como dato de entrada

La red toma como entrada los pixeles de una imagen. Si tenemos una imagen con apenas 28×28 pixeles de alto y ancho, esto equivale a utilizar 784 neuronas. Y eso es si sólo tenemos 1 color (escala de grises). Si tuviéramos una imagen a color, necesitaríamos 3 canales RGB (red, green, blue) y entonces usaríamos 28x28x3 = 2352 neuronas. Estas neuronas constituyen nuestra capa de entrada.

¿Cómo reconocer las imágenes?

Buscando patrones en los pixeles, que nos permitan asociarlos para identificar el digito. Por ejemplo el número uno puede tener más patrones de líneas verticales y el número 2 puede tener más patrones de líneas horizontales. A dicho patrón se le llama filtro.

0	1	0
1	1	1
0	1	0

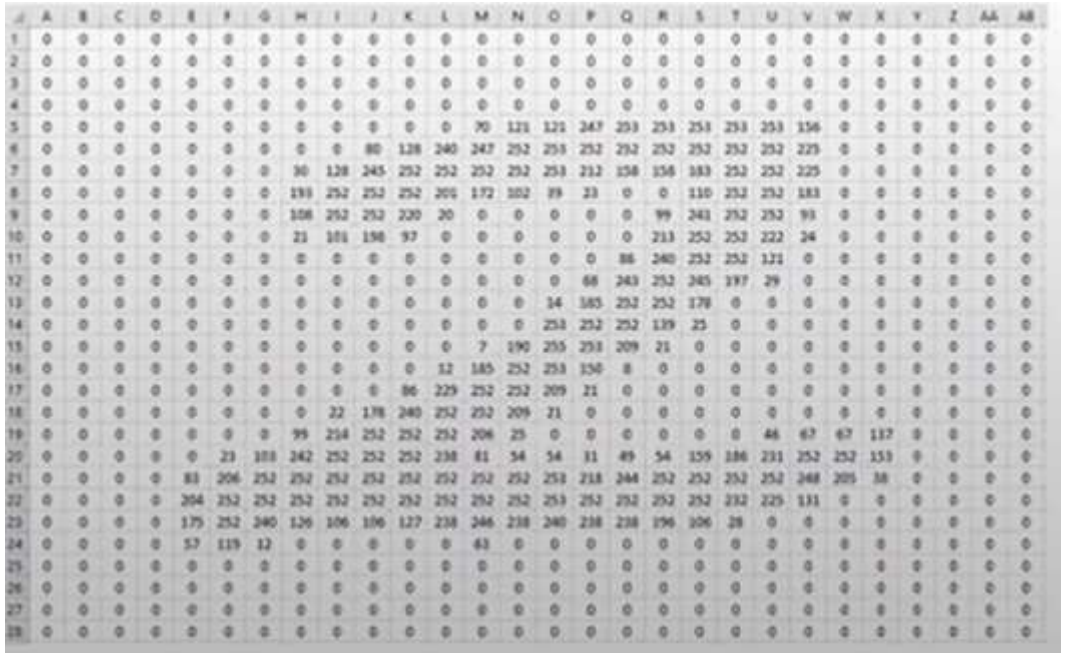


Convolución del filtro

Consisten en tomar «**grupos de pixeles cercanos»** de la imagen de entrada e ir operando matemáticamente (producto escalar) contra una pequeña matriz que se llama **kernel**. Ese kernel supongamos que tiene un tamaño de 3×3 pixeles y con ese tamaño logra «visualizar» todas las neuronas de entrada (de izquierda-derecha, de arriba-abajo) y así logra generar una nueva matriz de salida, que en definitiva será nuestra nueva capa de neuronas ocultas.

Filtro horizontal

- Convolucionamos el filtro de 3X3 con la imagen original.
- Se utiliza un filtro para resaltar los bordes horizontales.
- El “*” representa el operador convolucional.



*

1	1	1
0	0	0
-1	-1	-1

=

¿Cómo funciona la operación convolucional?

Para efectos prácticos consideremos una imagen de 6X6 y se le aplica un filtro para reconocer patrones verticales.

Algoritmo:

- Se coloca el filtro en la parte superior izquierda.
- Se multiplica cada subregión de la imagen con el filtro.
- Se realiza la sumatoria de los resultados de la multiplicación.
- Se repite el filtro recorriendo de uno en uno de izquierda a derecha y de arriba a bajo.

0	0	120	120	0	0
0	120	0	0	120	0
0	0	0	0	120	0
0	0	0	120	0	0
0	0	120	0	0	0
0	120	120	120	120	0

*

1	**1**	**1**
0	**0**	**0**
-1	**-1**	**-1**

=

0(1)+0(1)+120(1) +
0(0)+120(0)+0(0)+
0(-1)+0(-1)+0(-1) =120

Feature Mapping

120	240	120	0
120	0	0	0
-120	-120	0	120
-240	-240	-240	-120

Submuestreo

Como se mencionó para la imagen en escala de grises de 28X28 se requieren 784 neuronas de entrada, si a la imagen se le aplican 32 filtros, en la primer convolución se obtiene una capa oculta de 25 088 neuronas.

Si hiciéramos una nueva convolución el número de neuronas se eleva demasiado y ello implica mayor procesamiento. Para reducir el tamaño de la próxima capa se realiza un submuestreo, obteniendo las características más importantes. Hay varios tipos de submuestreo el más usado es Max-Pooling.

Max - Pooling

Recorreremos cada una de nuestras 32 imágenes de características obtenidas anteriormente de 28x28px de izquierda-derecha, arriba-abajo PERO en vez de tomar de a 1 pixel, tomaremos de “2×2” (2 de alto por 2 de ancho = 4 pixeles) e iremos preservando el valor “más alto” de entre esos 4 pixeles (por eso lo de “Max”). En este caso, usando 2×2, la imagen resultante es reducida “a la mitad” y quedará de 14×14 pixeles. Luego de este proceso de subsamplig nos quedarán 32 imágenes de 14×14, pasando de haber tenido 25.088 neuronas a 6272, son bastantes menos y -en teoría- siguen almacenando la información más importante para detectar características deseadas.

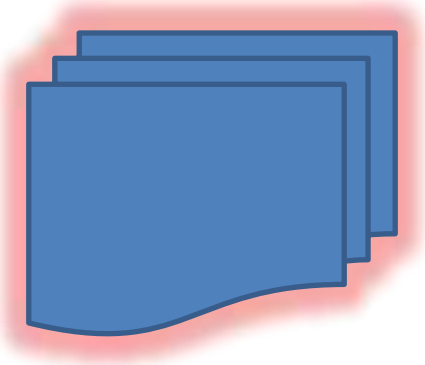
La primer convolución es capaz de detectar características primitivas como lineas ó curvas. **A medida que hagamos más capas con las convoluciones, los mapas de características serán capaces de reconocer formas más complejas**

Primera Convolución

Imagen de entrada 28X28X1

Aplicación De 32 filtros de 3X3

Obtención de 32 features mapping de 28X28X1



Aplicación de Max-Pooling de 2X2

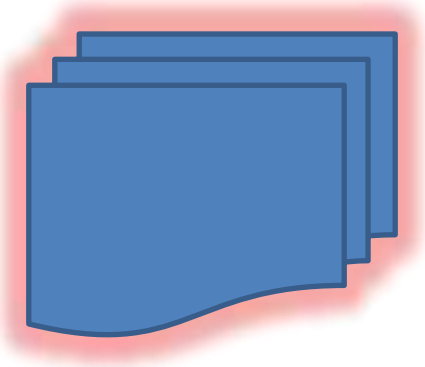
Obtención de 32 salidas de de 14X14X1

Segunda convolución

Imagen de entrada 14X14X1

Aplicación De 64 filtros de 3X3X32

Obtención de features mapping de 14X14X64



Aplicación de Max-Pooling de 2X2

Obtención de salidas de 7X7X64

Tercer convolución

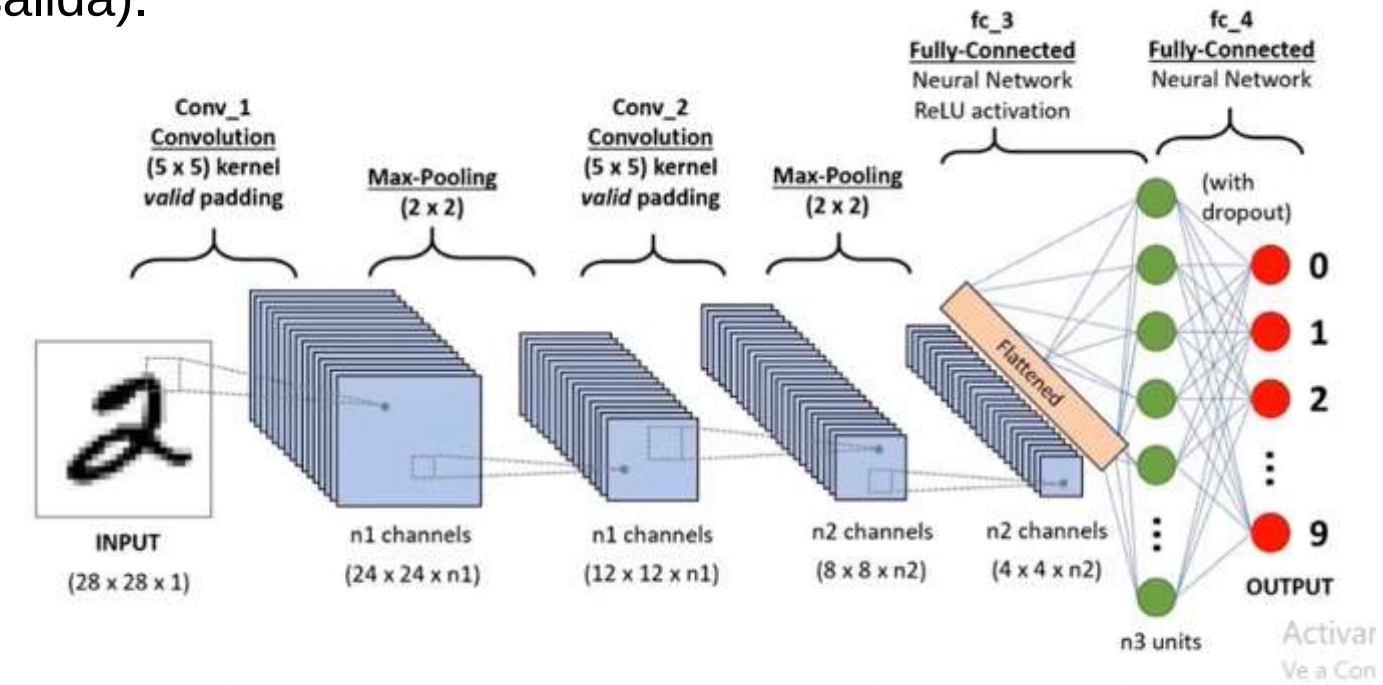
comenzará en tamaño 7×7 pixels y luego del max-pooling quedará en 3×3 con lo cual podríamos hacer sólo 1 convolución más.

En este ejemplo empezamos con una imagen de 28x28px e hicimos 3 convoluciones. Si la imagen inicial hubiese sido mayor (de 224x224px) aún hubiéramos podido seguir haciendo convoluciones.

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).



https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

Arquitecturas de redes convolucionales: LeNet

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

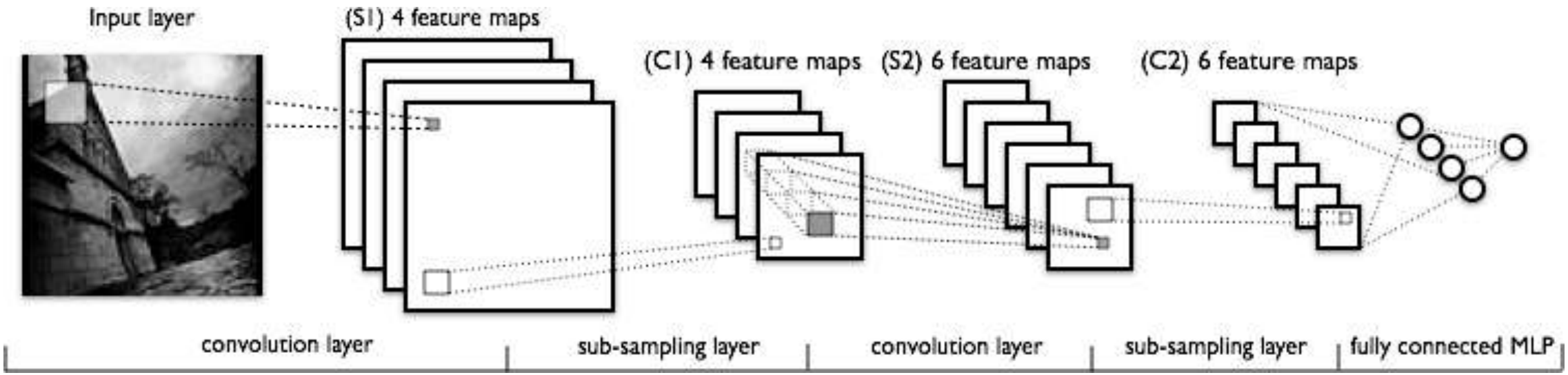


Imagen tomada de http://deeplearning.net/tutorial/lenet.html

Arquitecturas de redes convolucionales: AlexNet

Usa función de activación ReLU

- ▶ Entrenamiento con version optimizada para 2 GPUs
- ▶ Normaliza respuestas
- ▶ Submuestreo con traslape

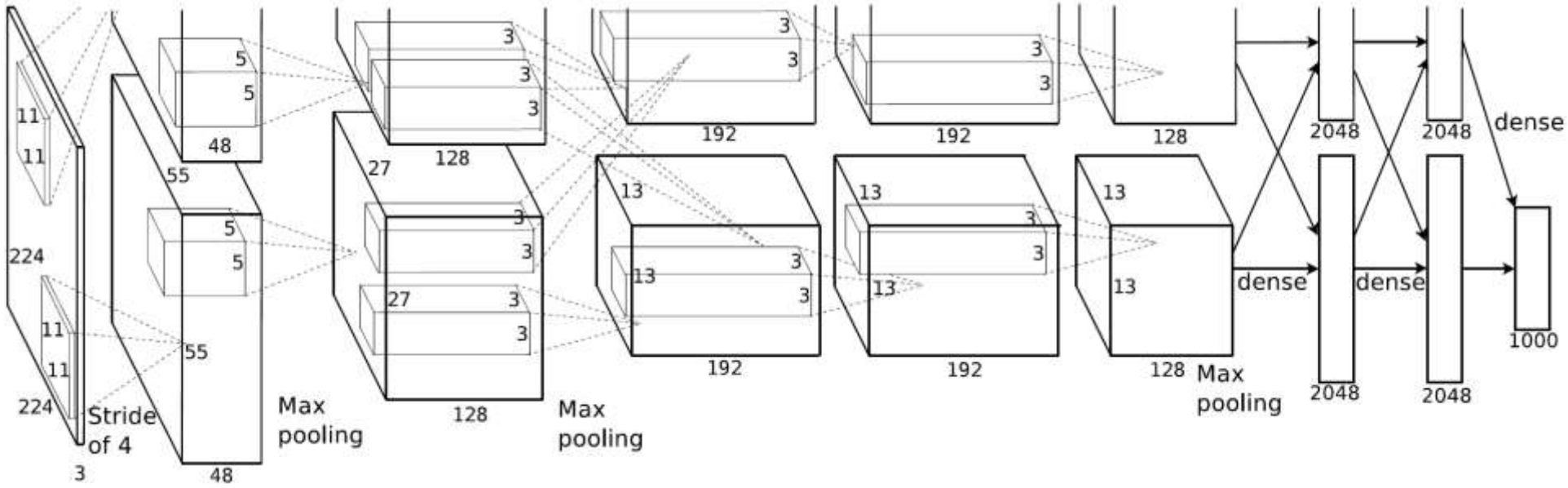


Imagen tomada de Krizhevsky et al. *ImageNet Classification with Deep Convolutional Neural Networks*, 2012

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)

AprenEntrenando redes más profundas: ResNet

- Reformula los mapeos que se desean aprender a residuales
- Más fácil de optimizar los residuales que el mapeo original[1]

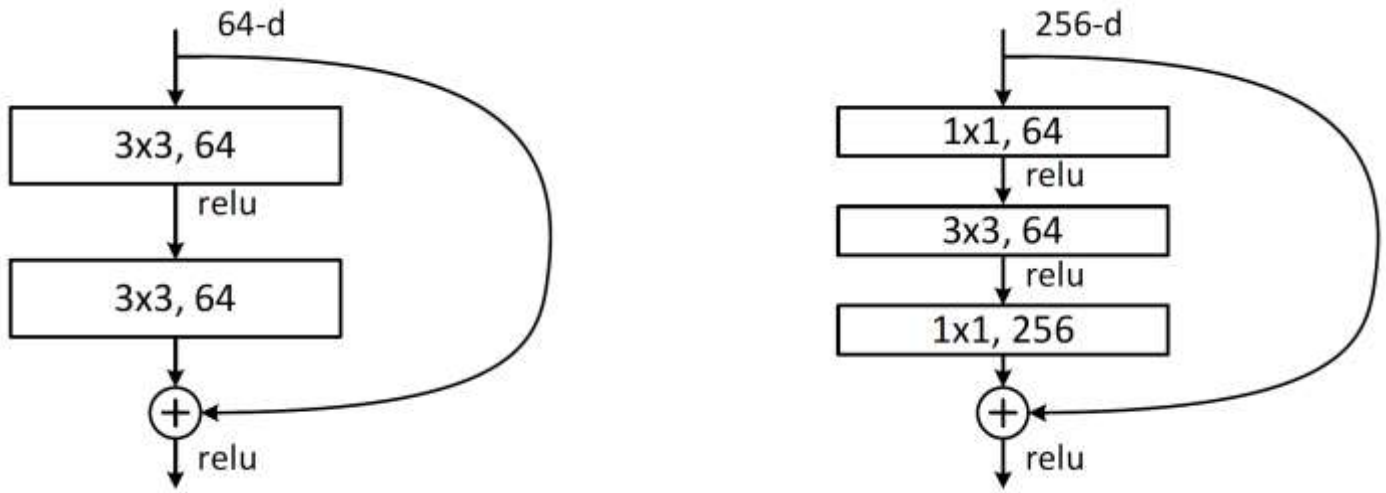


Imagen tomada de He et al. Deep Residual Learning for Image Recognition, 2015

[1]Se 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

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				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	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

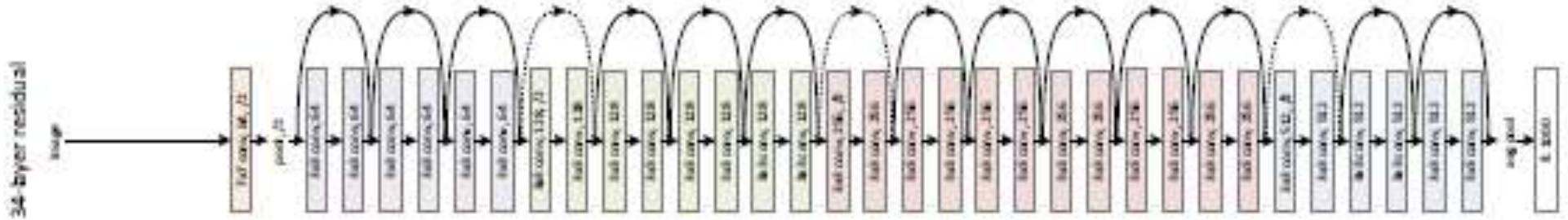
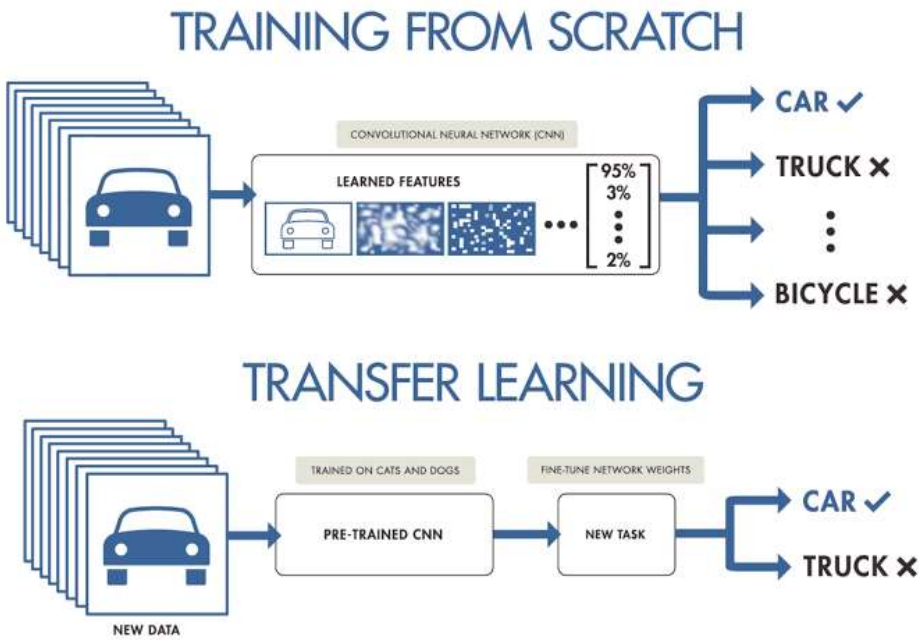


Imagen y tabla tomadas de He et al. *Deep Residual Learning for Image Recognition*, 2015

Aprendizaje por transferencia

Consiste en aprovechar la gran cantidad de información relacionada con la resolución de un problema y utilizarla sobre otro distinto, pero que compartan ciertas características con este. Esto es, modificar patrones ya entrenados, para reconocer otras similares. Los mejores resultados se obtienen en visión artificial y el procesamiento de lenguaje natural.



Supongamos que uno tiene un gran conjunto de datos que comprende imágenes de diferentes tipos de objetos y esa persona tiene que clasificar si la imagen es de un automóvil, un camión o algún otro vehículo. Con el ML tradicional, las imágenes deben pasar a través de una red neuronal convolucional que se va a entrenar por primera vez. Luego, se extraerán las características y, después de otras rondas de CNN, se realizará la clasificación de imágenes. Por otro lado, en el aprendizaje por transferencia se utilizará el modelo CNN preentrenado que habría venido de otro sistema de aprendizaje y dará resultados mucho más precisos para la clasificación de imágenes.

Ventajas del aprendizaje por transferencia

- Se pueden construir modelos más robustos que pueden realizar una amplia variedad de tareas.
- El conocimiento se puede transferir fácilmente de un modelo a otro.
- A veces, los datos no tienen etiquetas, por lo que Transfer Learning puede abordar ese tipo de problemas.
- Los problemas complejos del mundo real se pueden resolver con varias restricciones.

Inconvenientes del aprendizaje por transferencia

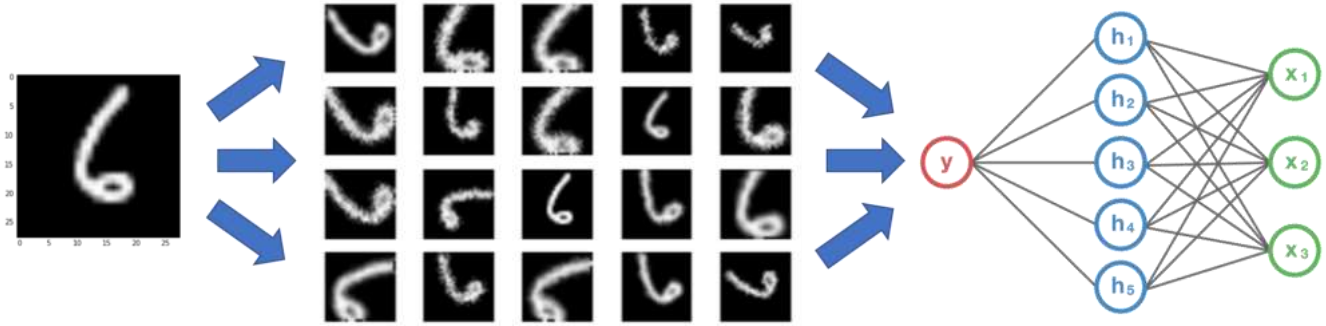
- **Límites de transferencia** : es muy importante en el aprendizaje de transferencia cuantificar la transferencia. Al hacerlo, afecta la calidad de la transferencia y su viabilidad. Este punto es demasiado amplio para entenderlo en este tema, ya que está fuera del alcance del artículo. Para leer más, puede consultar varios artículos de investigación sobre el aprendizaje por transferencia como, por ejemplo, el de Hassan Mahmud y sus coautores, que calcularon la cantidad de transferencia, y también el de Eaton y sus coautores, que presentaron un enfoque novedoso basado en gráficos para medir la transferencia de conocimiento.

Inconvenientes del aprendizaje por transferencia

Transferencia negativa : ha habido muchos casos en los que hablé sobre la mejora en la precisión y el rendimiento con la ayuda de transferir los pesos, sesgos, etc. a otros sistemas de aprendizaje. Pero a veces puede haber una degradación de las actuaciones. Entonces, ahora es fácil adivinar su definición. El aprendizaje de transferencia negativa se refiere a la caída en el rendimiento general del sistema de aprendizaje durante la transferencia de conocimiento desde el origen al destino. Es muy necesario cuidar esta situación y requiere una investigación cuidadosa.

Acrecentamiento de datos

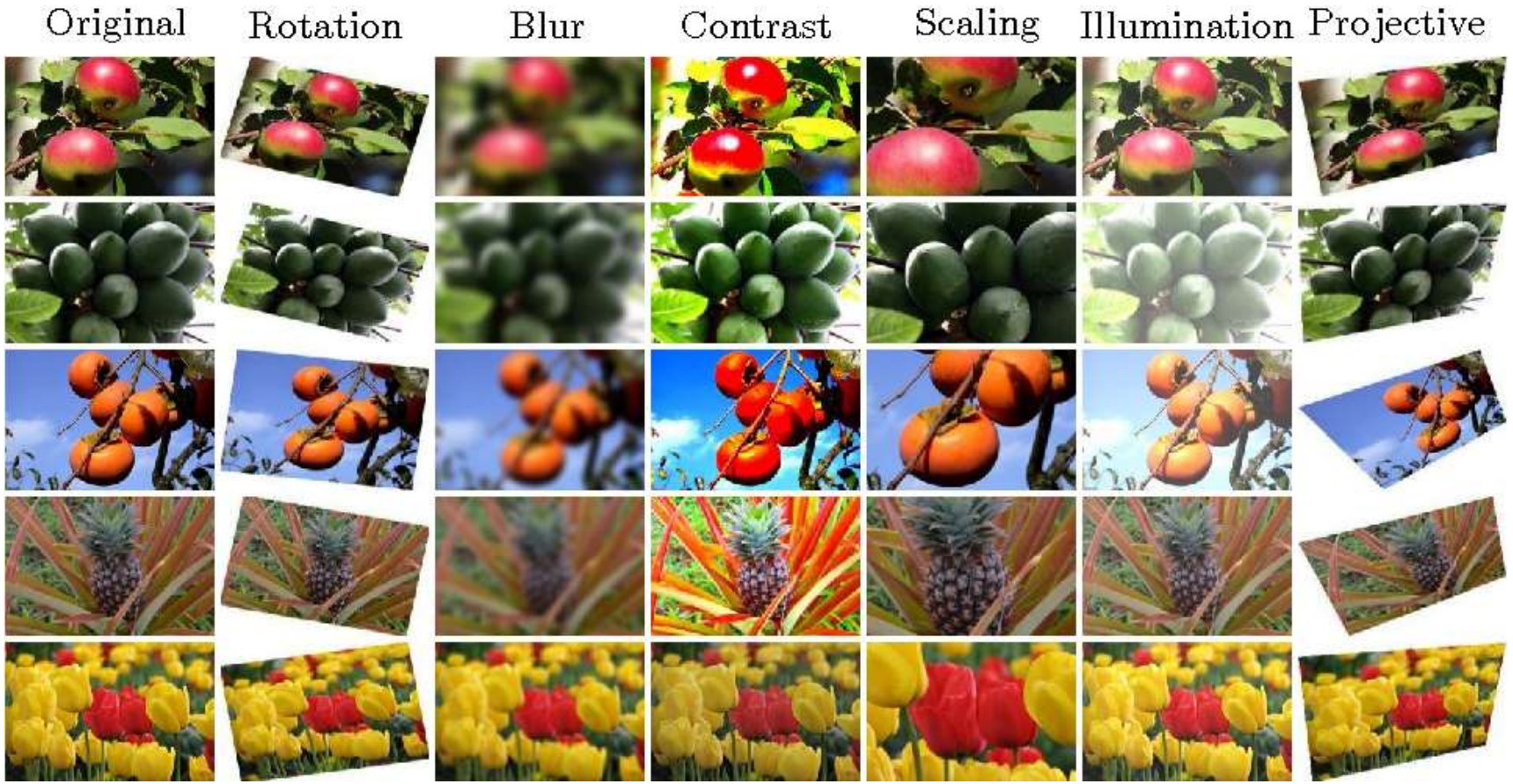
Es un conjunto de técnicas para aumentar artificialmente la cantidad de datos mediante la generación de nuevos puntos de datos a partir de datos existentes. Esto incluye realizar pequeños cambios en los datos o usar modelos de aprendizaje profundo para generar nuevos puntos de datos.



https://www.kdnuggets.com/2018/05/data-augmentation-deep-learning-limited-data.html

Para el aumento de datos, es popular hacer modificaciones simples en los datos visuales. Además, las redes antagónicas generativas (GAN) se utilizan para crear nuevos datos sintéticos . Las actividades clásicas de procesamiento de imágenes para el aumento de datos son:

- relleno
- rotación aleatoria
- re-escalado,
- volteo vertical y horizontal
- traducción (la imagen se mueve a lo largo de la dirección X, Y)
- recorte
- hacer zoom
- oscurecimiento y aclaración/modificación de color
- escala de grises
- cambio de contraste
- agregando ruido
- borrado aleatorio



https://research.aimultiple.com/data-augmentation/

Transformaciones

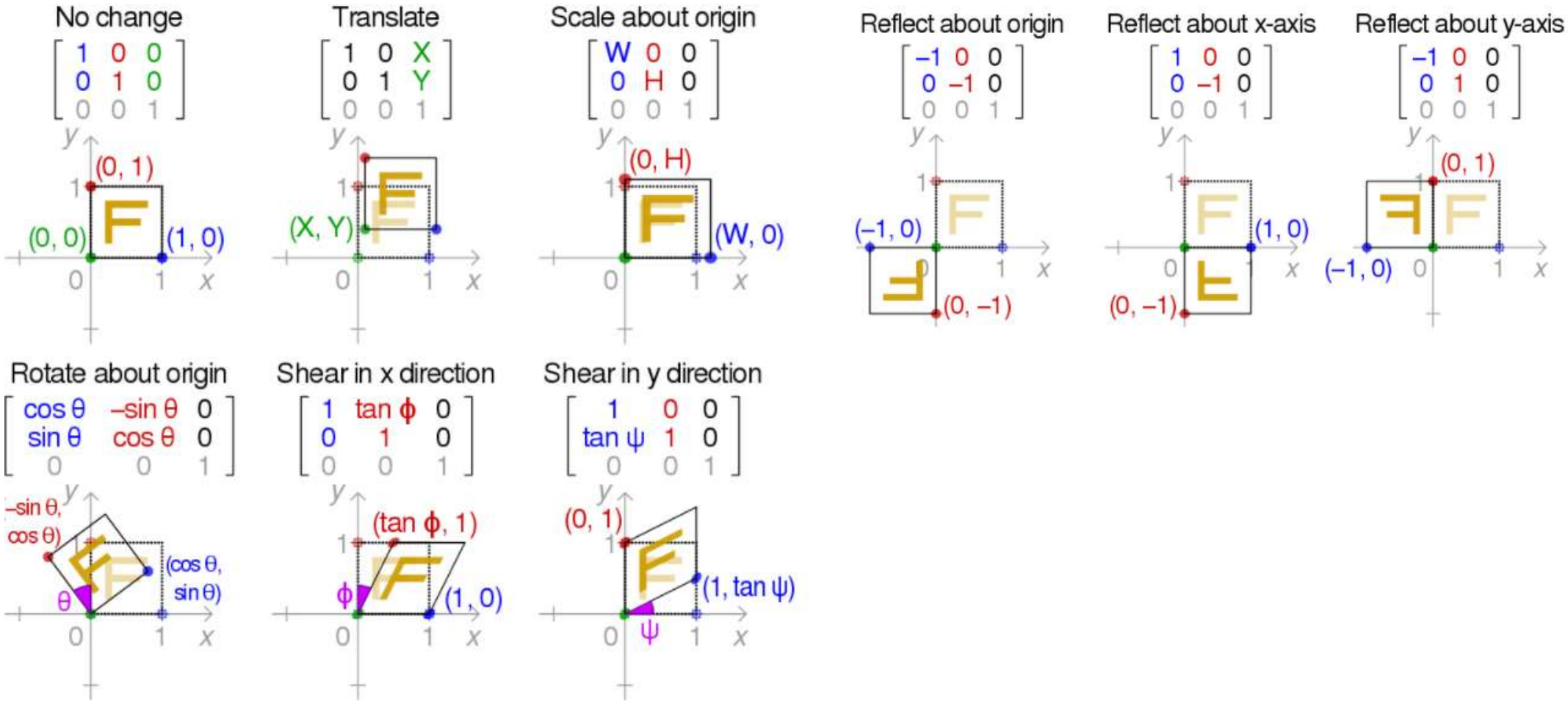


Imagen del usuario de Wikipedia Cmglee (entrada Affine transformation). CC BY-SA 3.0

Ejemplo práctico: clasificación de imágenes con ImageNet

https://colab.research.google.com/drive/1Pww2sLtzZ8movhygC2wlogIe7cgzwUg0#scrollTo=Axjh7btwEDMb

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

Edgar Morales Palafox

Doctor en ciencias de la computación

Edgar_morales_p@yahoo.com.mx

Tels: 55 3104 1600