Deep Learning

Redes Neuronales con TensorFlow

Septiembre 13 – Septiembre 20

Convolucionales

Deep Learning

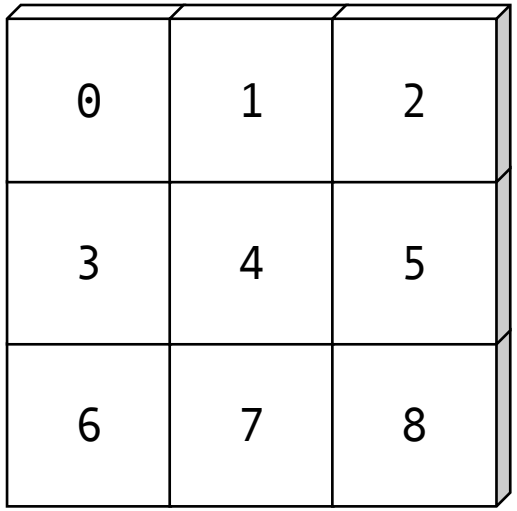
Redes Neuronales con TensorFlow

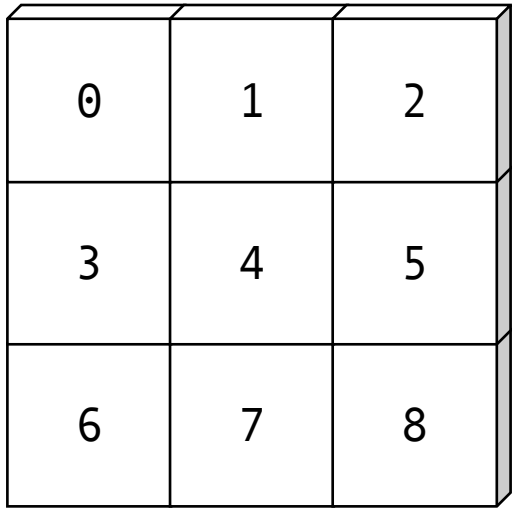
Septiembre 13 – Septiembre 20

Convolucionales

- Kernels
- Convolucionales
- Hiperparámetros
- Image Augmentation

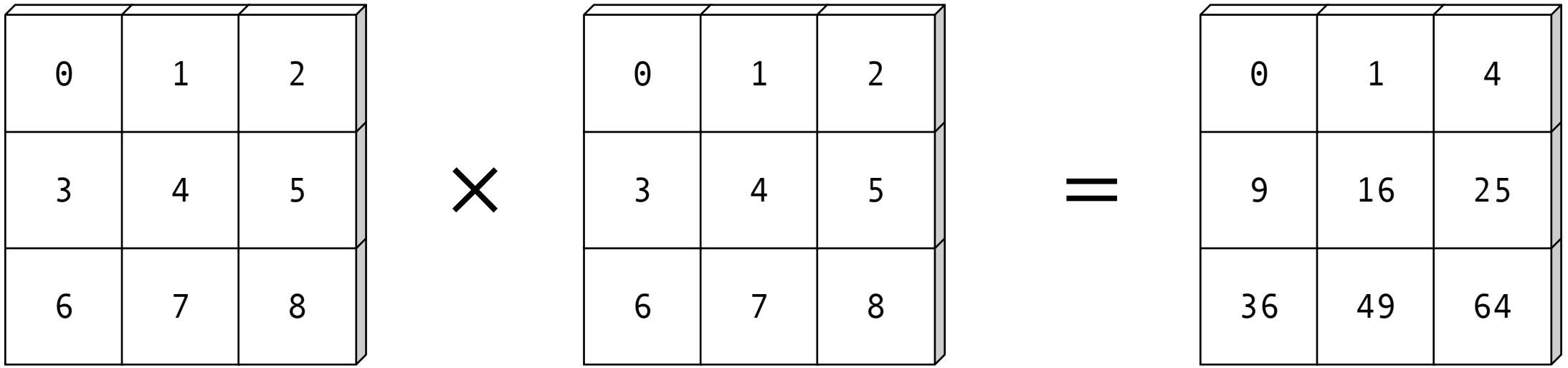
```
import numpy as np

a = np.arange(0, 9).reshape(3,3)
b = np.arange(0, 9).reshape(3,3)
```





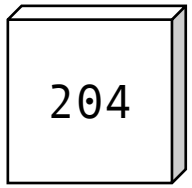
HD INSTITUTO HUMAN DELTA

```
>> import numpy as np

>> a = np.arange(0, 9).reshape(3,3)
>> b = np.arange(0, 9).reshape(3,3)

>> a*b
```

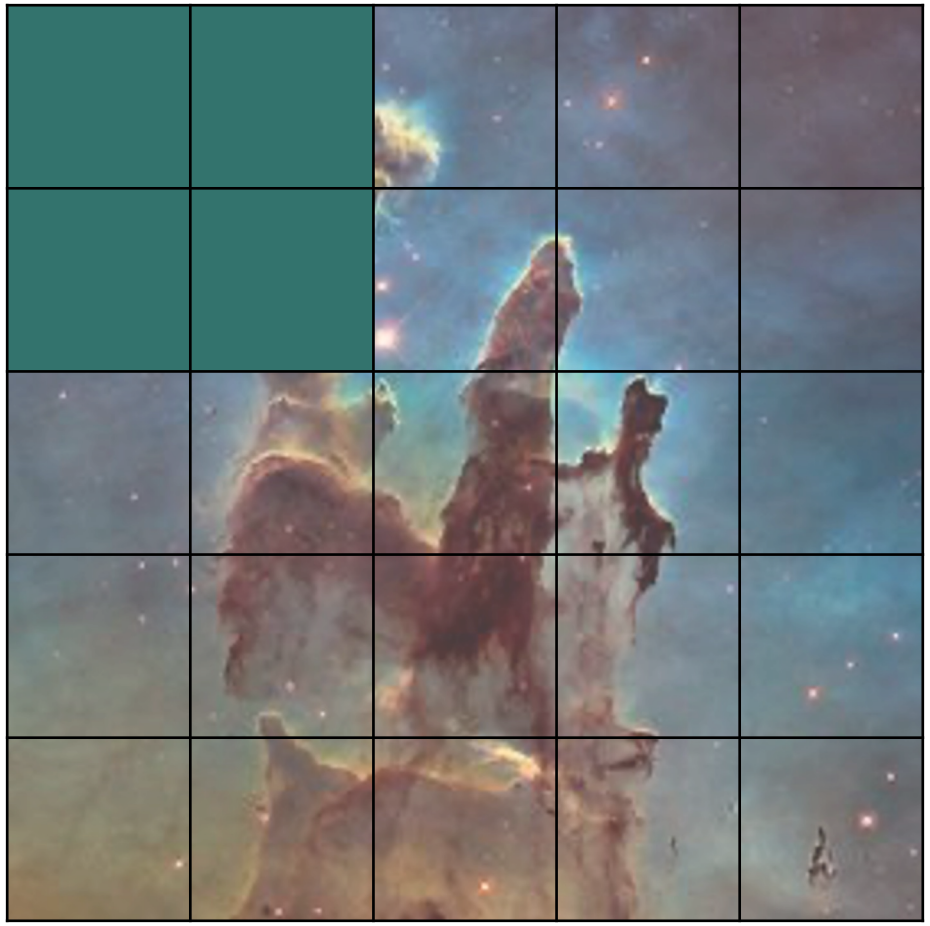


HD INSTITUTO HUMAN DELTA

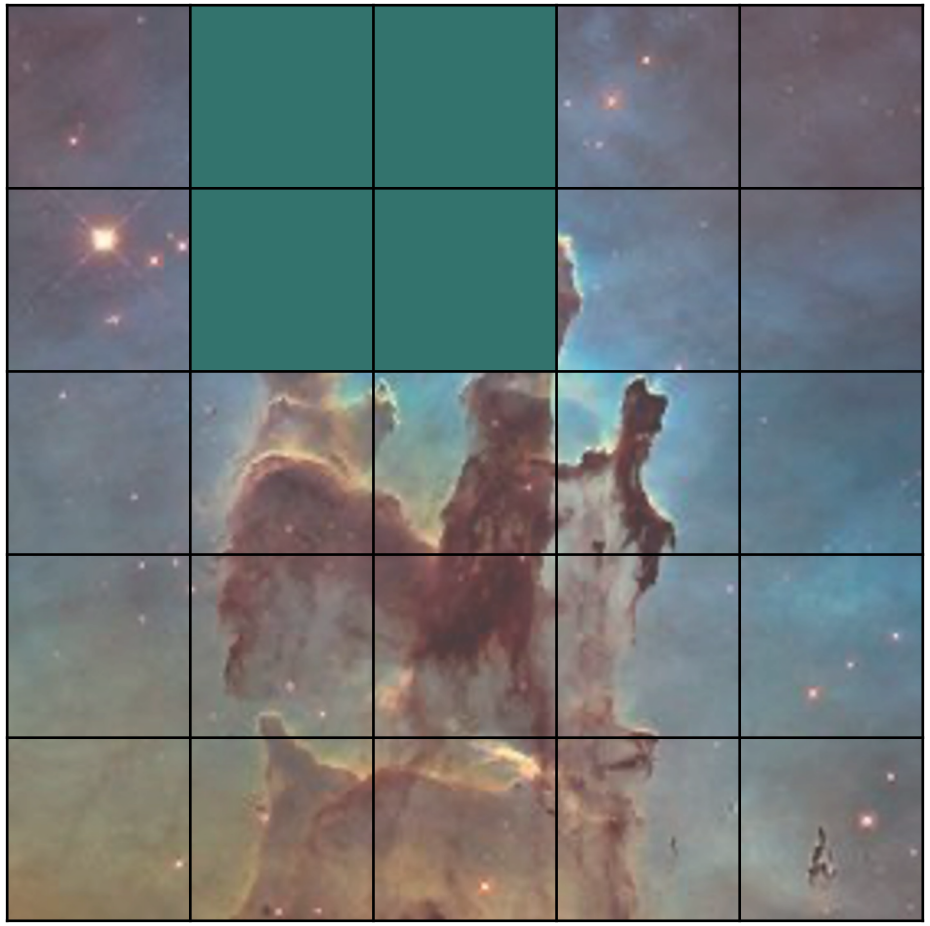
```
>> import numpy as np

>> a = np.arange(0, 9).reshape(3,3)
>> b = np.arange(0, 9).reshape(3,3)

>> a*b

>> (a*b).sum()
```



Kernels



Kernels



Kernels

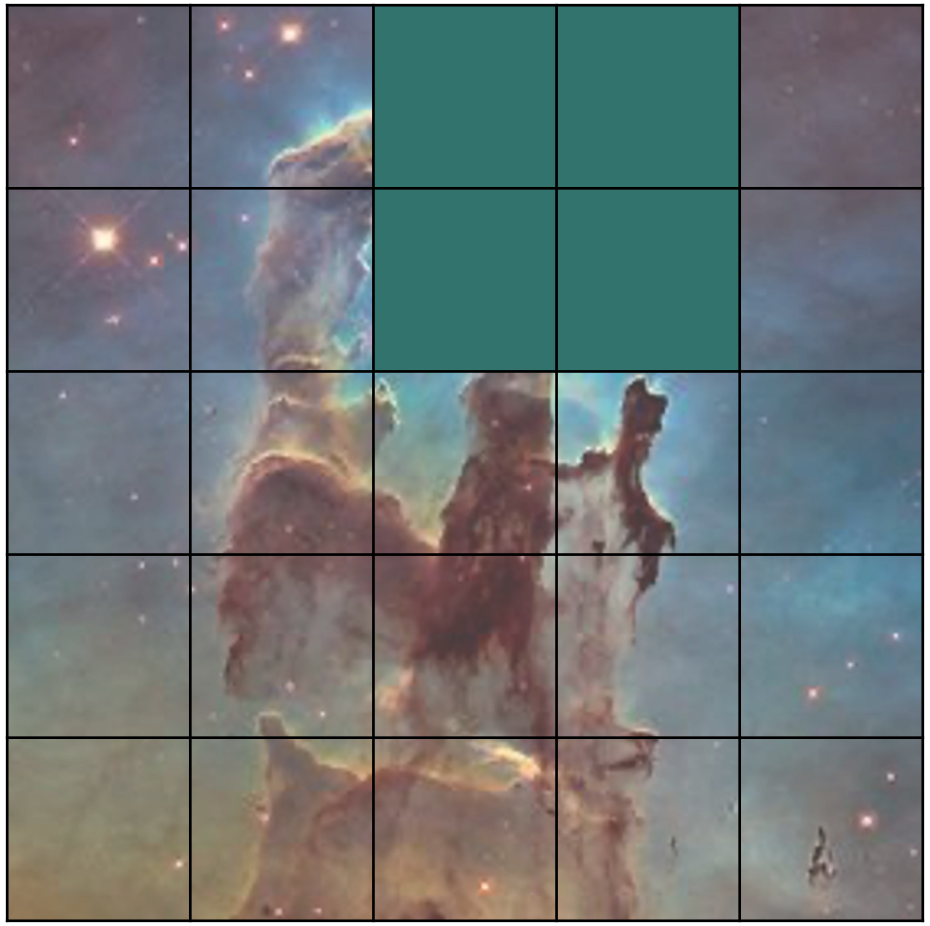


Imagen original

Imagen B/N

Kernels

Imagen original

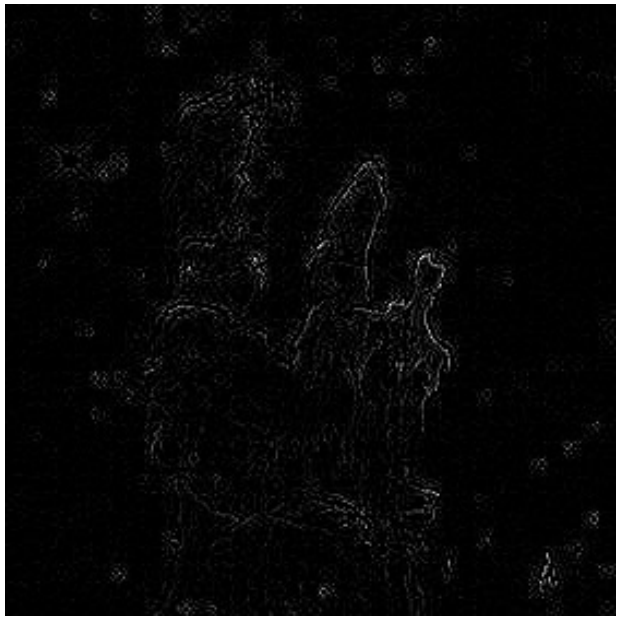
Imagen B/N

Sharpen

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

Imagen original

Imagen B/N



Laplacian

0	1	0
1	4	1
0	1	0

Kernels

Imagen original

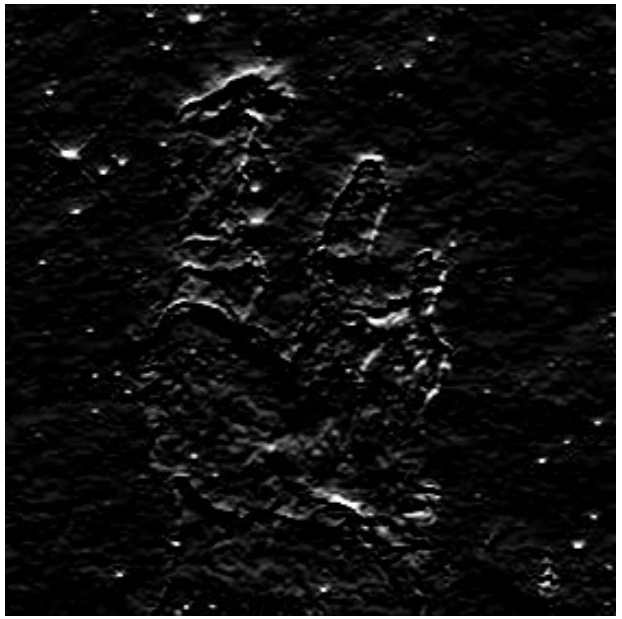
Imagen B/N

Sobel X

-1	0	1
-2	0	2
-1	0	1

Imagen original

Imagen B/N

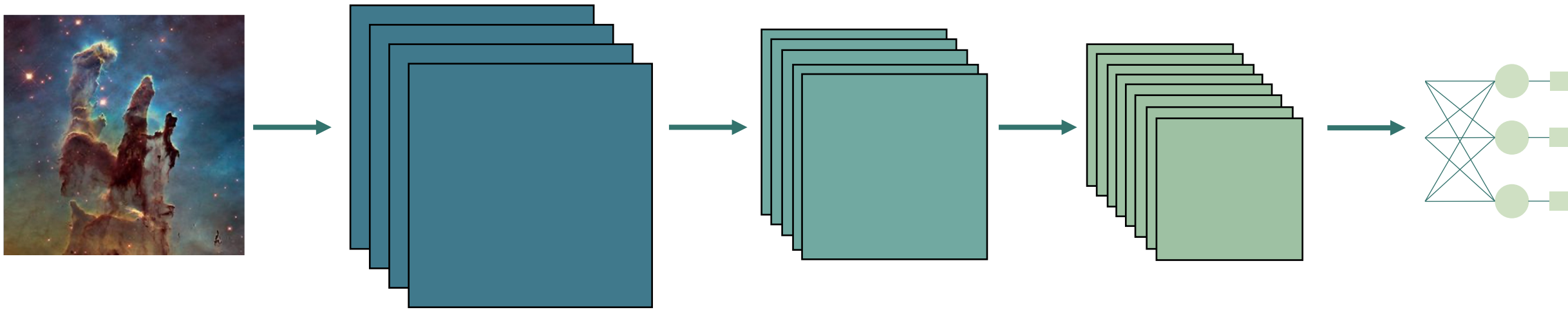


Sobel Y

-1	2	-1
0	0	0
1	2	1

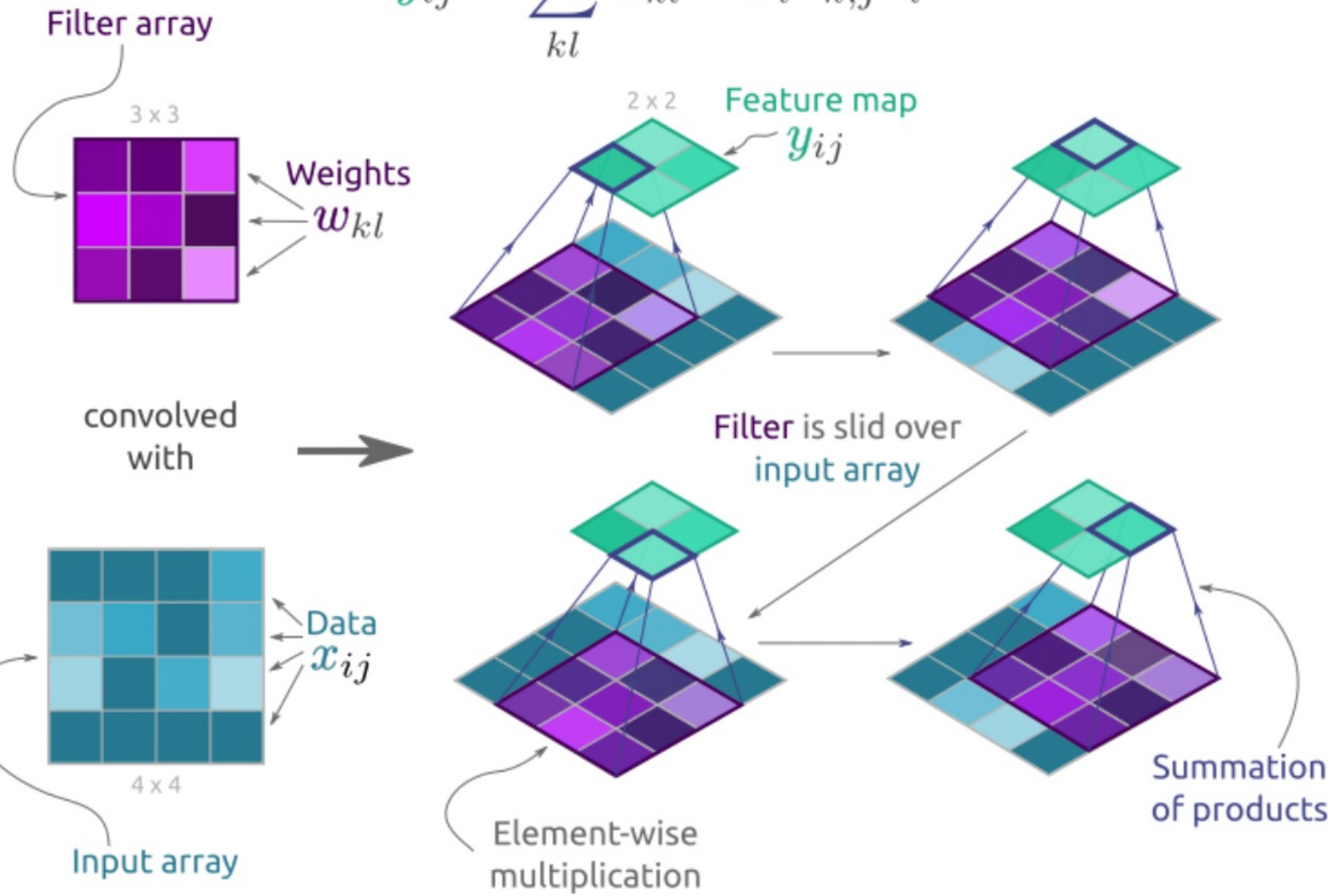
El objetivo de la red neuronal es establecer los valores del kernel

Convolucional



Convolutions in 2D

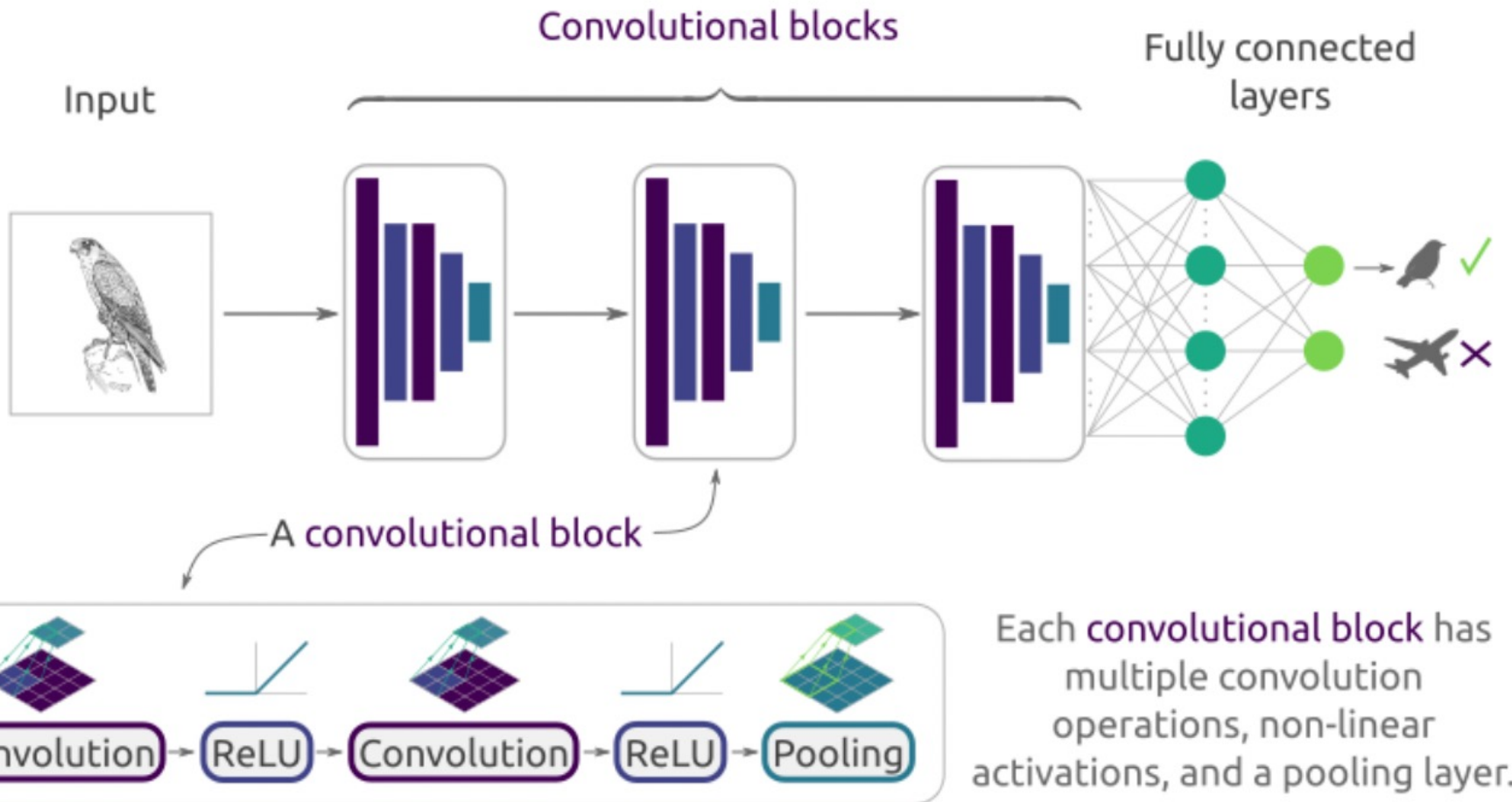
$$y_{ij} = \sum_{kl} w_{kl} \times x_{i-k,j-l}$$



*Tomado de Roy Keyes (2022)

A Basic Convolutional Neural Network

A **simplified convolutional neural network** made of several convolutional blocks for feature extraction and fully connected layers for image classification.

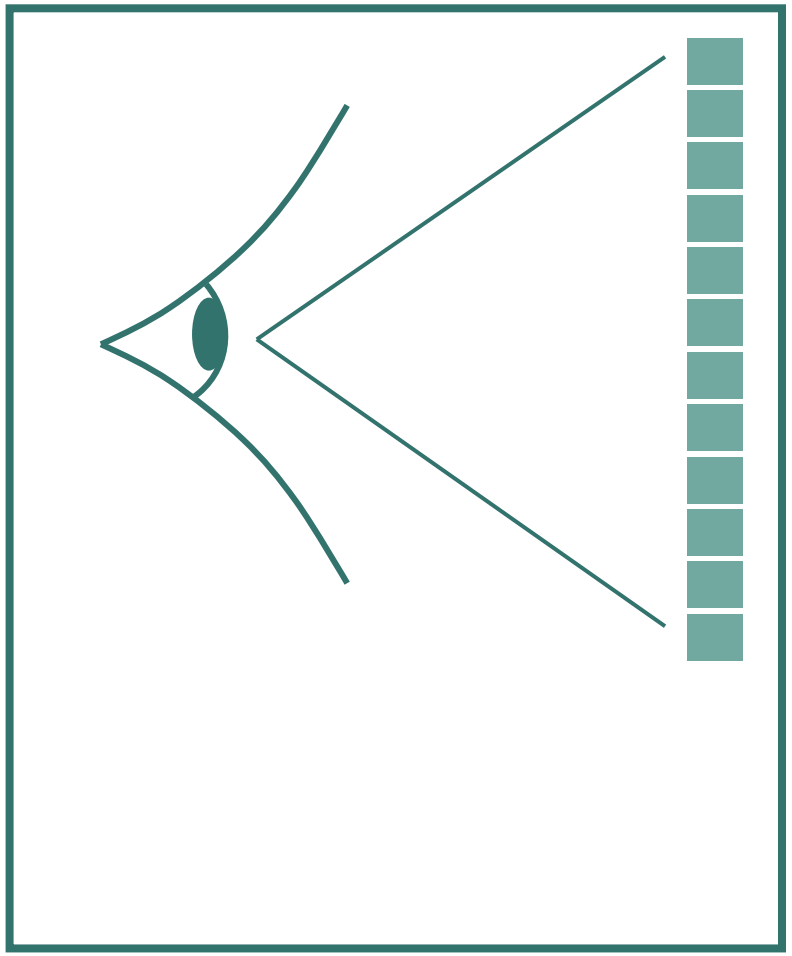


Each **convolutional block** has multiple convolution operations, non-linear activations, and a pooling layer.

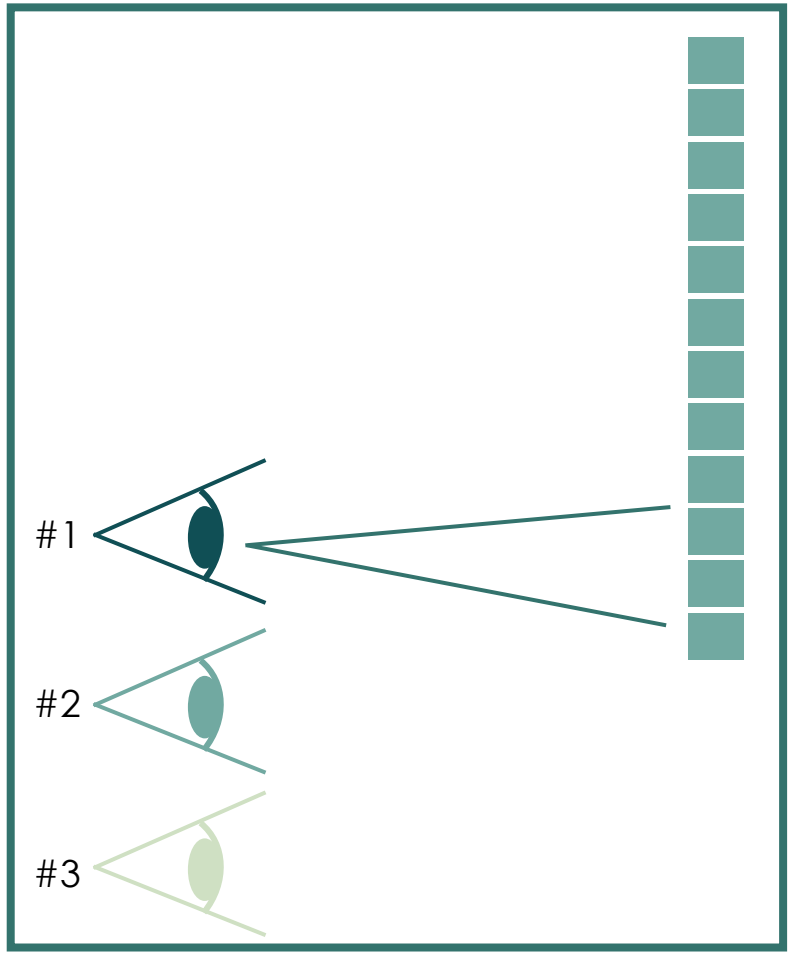
*Tomado de Roy Keyes (2022)

HD INSTITUTO HUMAN DELTA MR

Feedforward Neural Network

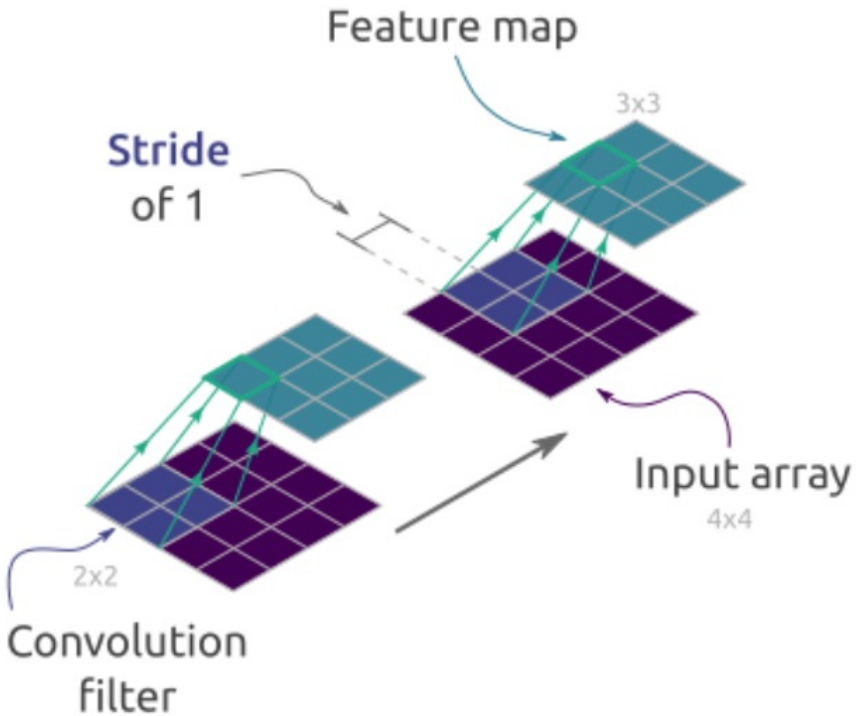


Convolutional Neural Network

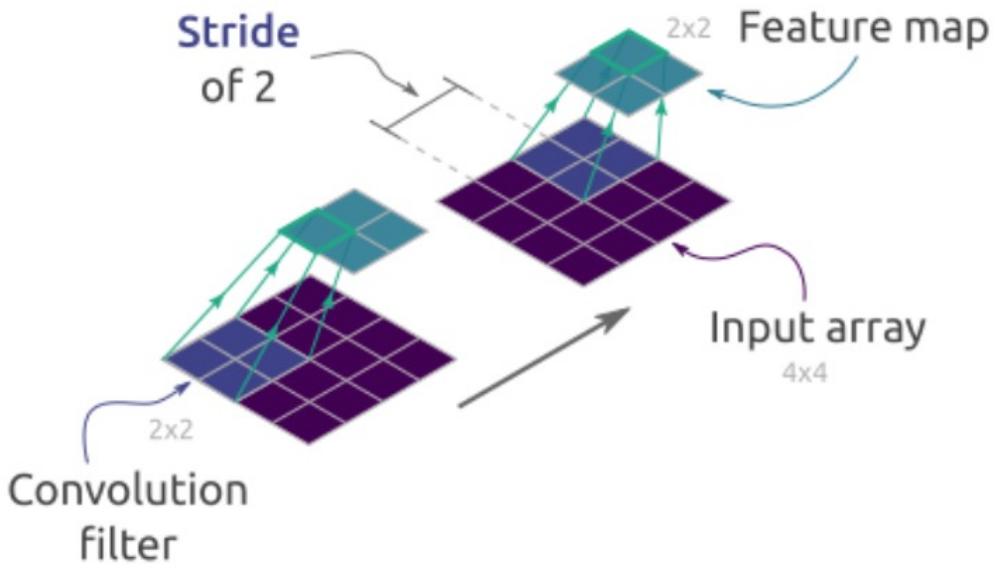


*Tomado de Meor Amer (2022)

Convolution Stride Length



Stride length is how many places the convolution filter is moved between operations and affects the size of the resulting feature map



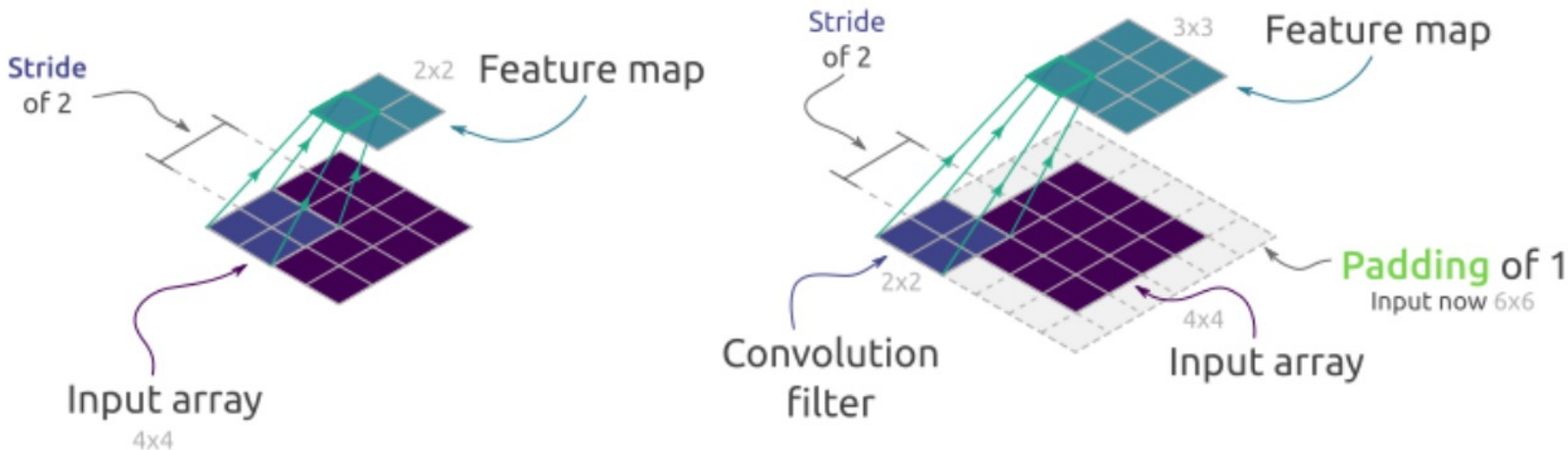
A larger stride results in a smaller feature map array. Here a stride of 1 results in a 3x3 feature map, while a stride of 2 results in a 2x2 feature map.

*Tomado de Roy Keyes (2022)

HD INSTITUTO HUMAN DELTA MR

Convolution Padding

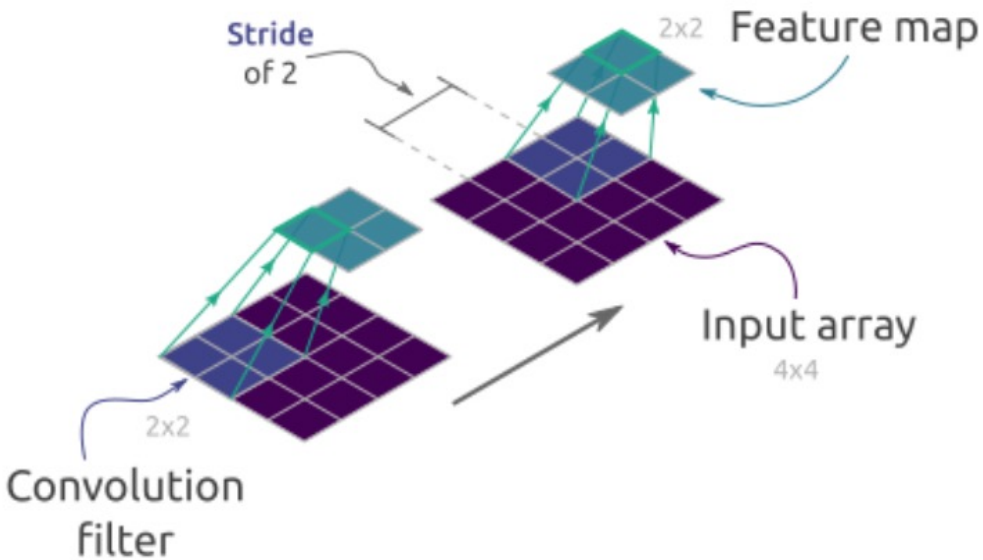
Padding is extra data added to the input array to make the array larger in order to control the size of the resulting feature map



In this example adding a padding of one entry around the input array increases the feature map size from 2x2 to 3x3

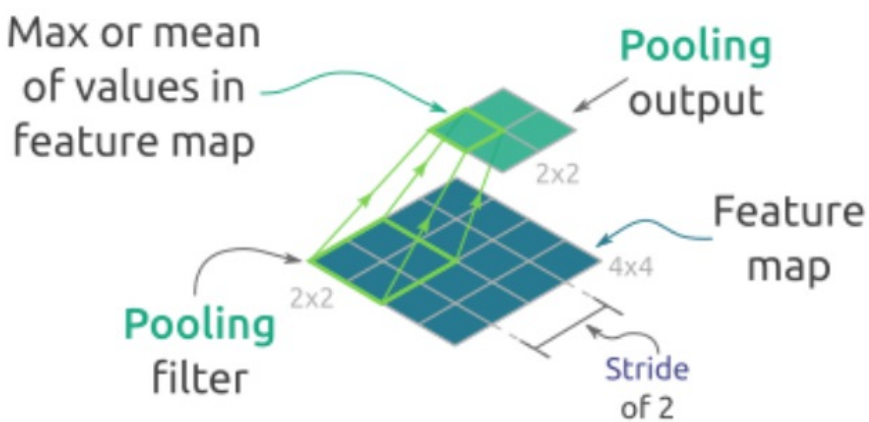
*Tomado de Roy Keyes (2022)

Convolution Pooling

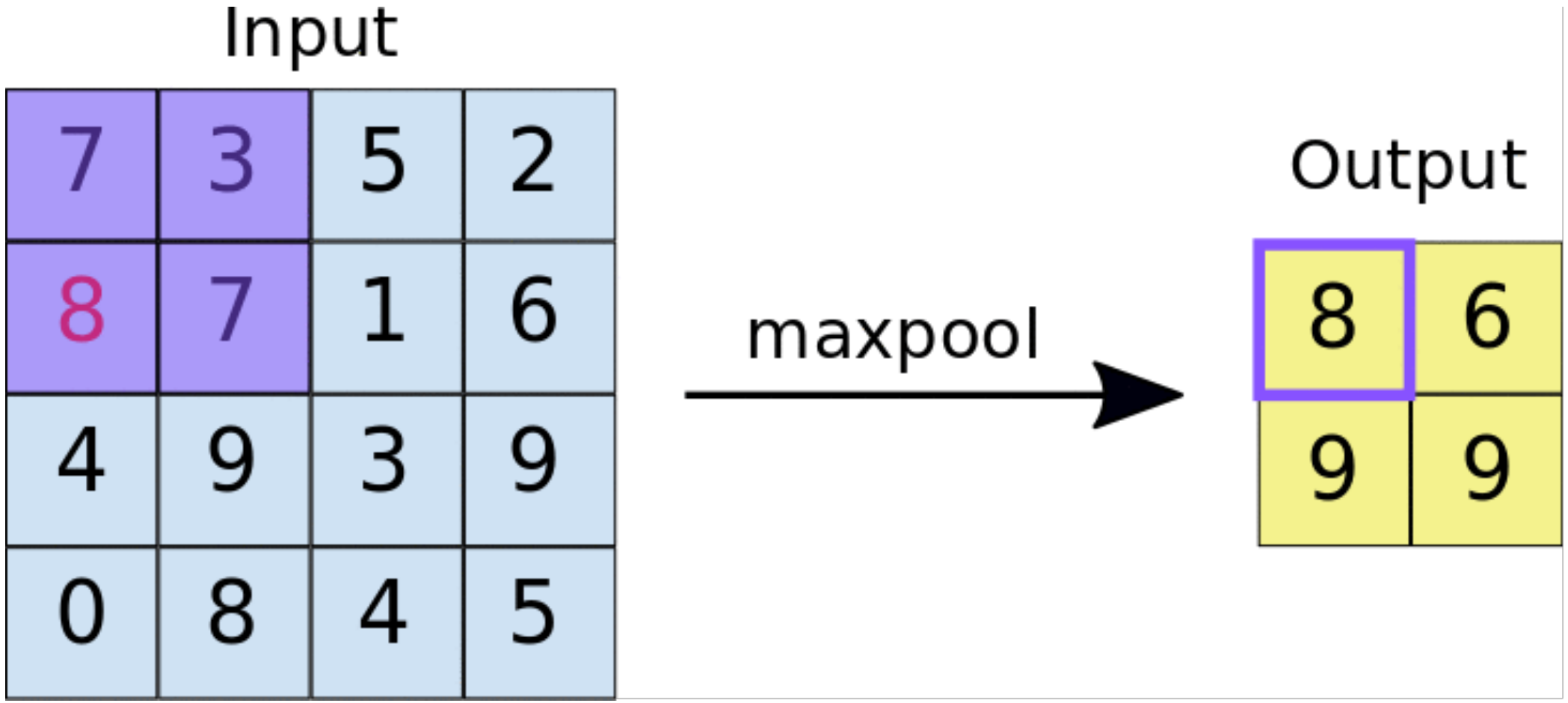


Pooling aggregates the values in the feature map to reduce its dimensions and make the network more robust to postition shift of the object of interest

The most common pooling operations are max pooling, which takes the max value within the pooling filter, and mean pooling, which takes the mean value. 2x2 is a common pooling filter size.



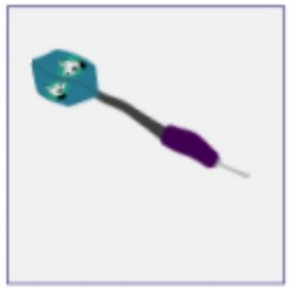
*Tomado de Roy Keyes (2022)


Input
7
3
5
2
8
7
1
6
4
9
3
9
0
8
4
5
maxpool
Output
8
6
9
9

Data Augmentation

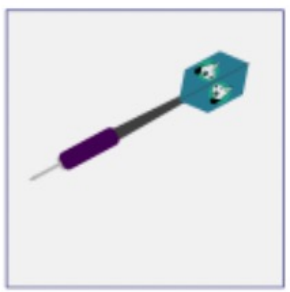
Original

Translation

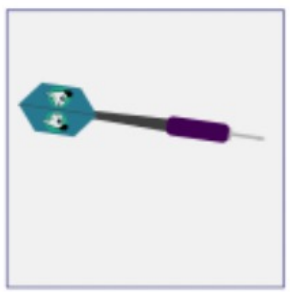


Warping

Data augmentation is the process of artifically increasing the size of your training data set by performing transformations on the original images.



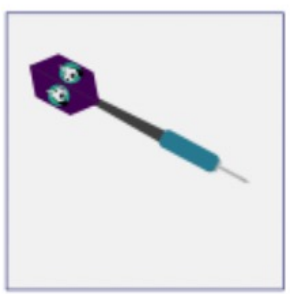
Mirroring



Rotation

Tinting

All kinds of transformations are useable, as long as the original label still applies to resulting image.



Color change

Scaling

Cropping

All of these images are still recognizeable as containing a dart

*Tomado de Roy Keyes (2022)

HD INSTITUTO HUMAN DELTA MR

1. Obtener imágenes

2. Declarar directorios

3. Reescalar / augmentation

4. Flow (crear batches)

1. Obtener imágenes

```
>> !wget .zip
```

2. Declarar directorios

```
>> train_dir = "pizza_sushi/train/"

>> test_dir = "pizza_sushi/test/"
```

3. Reescalar / augmentation

```
>> from tensorflow.keras.preprocessing.image import
ImageDataGenerator

>> train_datagen = ImageDataGenerator(rescale=1/255.)
>> test_datagen = ImageDataGenerator(rescale=1/255.)
```

4. Flow (crear batches)

```
>> train_data = train_datagen.flow_from_directory(directory=train_dir,
                                                  target_size=(224, 224),
                                                  class_mode='binary',
                                                  batch_size=32)
```

1. Obtener imágenes

```
>> !wget .zip
```

2. Declarar directorios

```
>> train_dir = "pizza_sushi/train/"

>> test_dir = "pizza_sushi/test/"
```

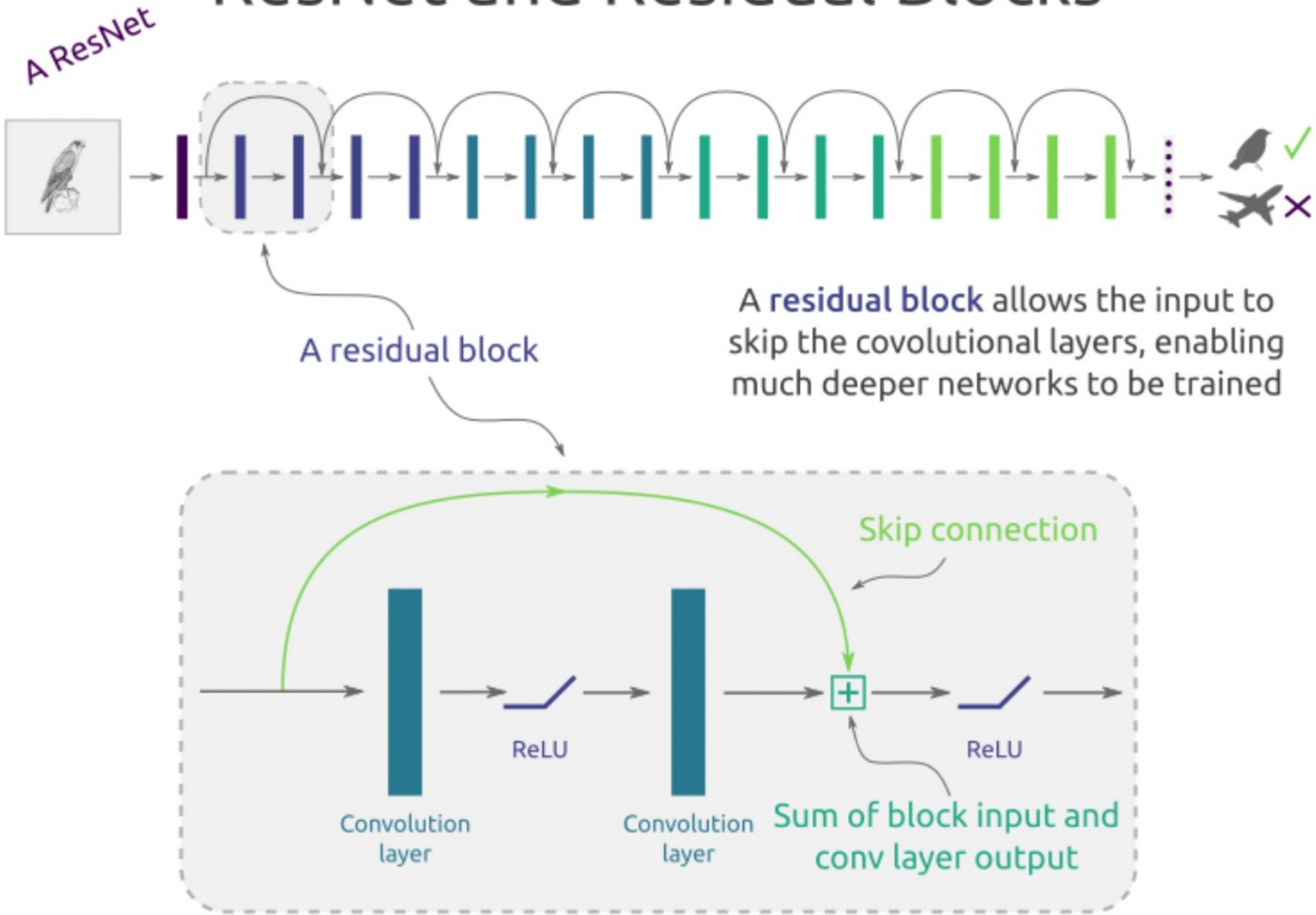
3. Reescalar / augmentation

```
>> from tensorflow.keras.preprocessing.image import ImageDataGenerator

>> train_datagen_augmented = ImageDataGenerator(rescale = 1/255.,
                                                rotation_range = 20,
                                                shear_range = 0.2,
                                                zoom_range = 0.2,
                                                width_shift_range = 0.2,
                                                height_shift_range = 0.2,
                                                horizontal_flip = True)
```

4. Flow (crear batches)

```
>> train_data = train_datagen.flow_from_directory(directory=train_dir,
                                                  target_size=(224, 224),
                                                  class_mode='binary',
                                                  batch_size=32)
```

ResNet and Residual Blocks



*Tomado de Roy Keyes (2022)

Referencias

- Foto de portada: Joshua Hoehne
- Meor Amer, (2022), **“A visual Introduction to Deep Learning”**
- Roy Keyes, (2022), **“Deep Learning”**

GRACIAS

emilio.sandpal@gmail.com

milioe