

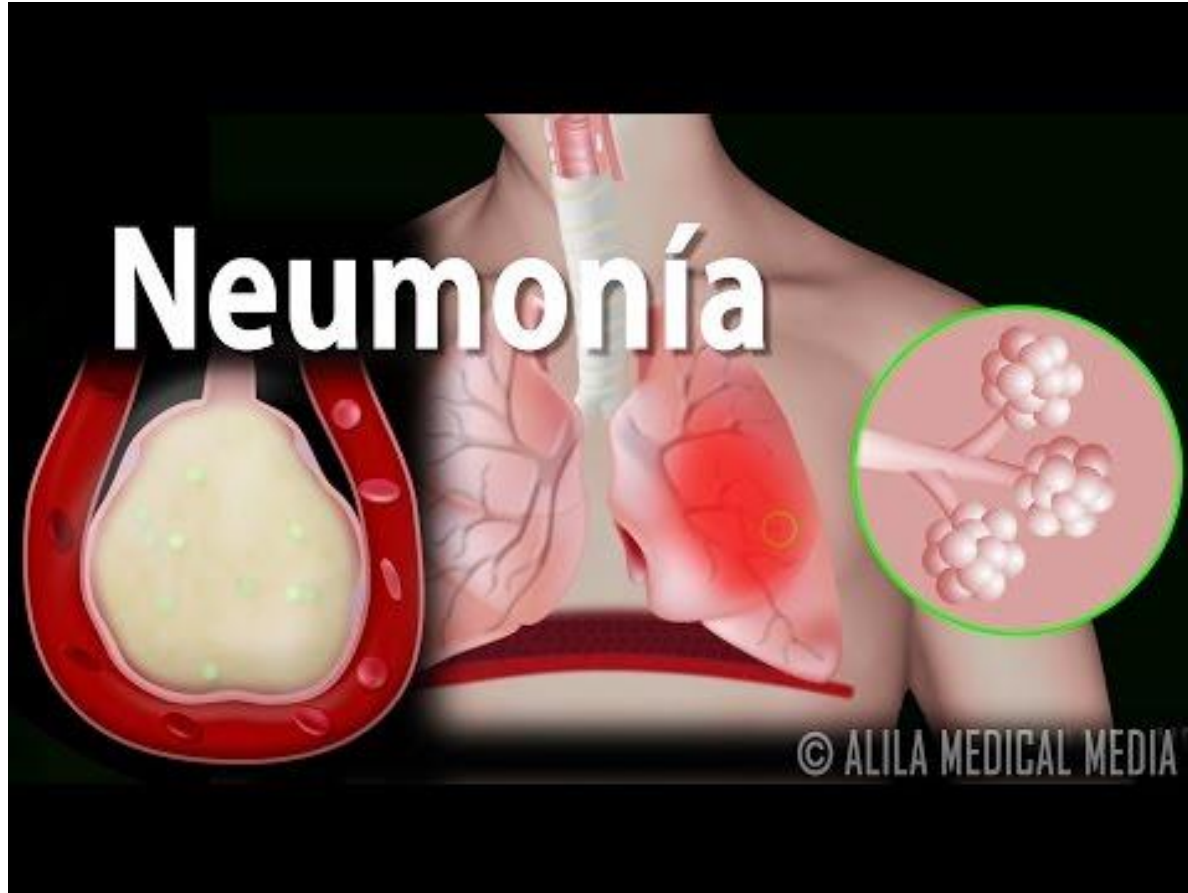
Redes neuronales convolucionales para la detección de neumonía



ACTUMLOGOS

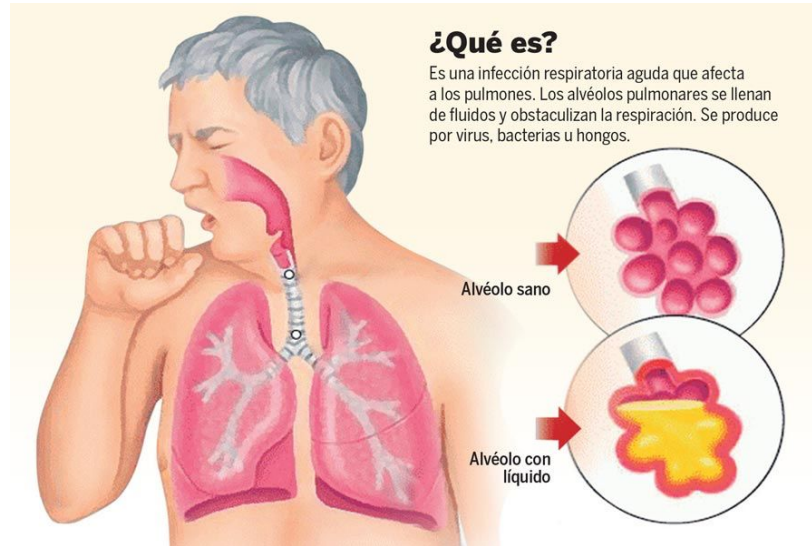
DESARROLLANDO HABILIDADES TECNOLÓGICAS

Introducción



¿Qué es la neumonía?

La neumonía es un tipo de infección respiratoria aguda que afecta a los pulmones. Estos están formados por pequeños sacos, llamados alvéolos, los cuales se llenan de aire al respirar. Los alvéolos de los enfermos de neumonía están llenos de pus y líquido, lo que hace dolorosa la respiración y limita la absorción de oxígeno.



[Neumonía](#)

¿Qué causa la neumonía?

Diversos agentes infecciosos como los son los virus (como lo es el nuevo coronavirus causante del Covid-19), bacterias y hongos, causan neumonía, siendo los más comunes los siguientes:

- *Streptococcus pneumoniae*: la causa más común de neumonía bacteriana en niños.
- *Haemophilus influenzae* de tipo b (Hib): la segunda causa más común de neumonía bacteriana.
- El virus sincitial respiratorio es la causa más frecuente de neumonía vírica.
- *Pneumocystis jiroveci* es una causa importante de neumonía en niños menores de seis meses con VIH/SIDA, responsable de al menos uno de cada cuatro fallecimientos de lactantes seropositivos al VIH.

¿Cómo se diagnostica?

En la neumonía, los alvéolos, que deberían estar llenos de aire, se llenan de líquido o tejido inflamatorio, por lo que en una radiografía, esto se observa de color blanco, mientras que el espacio lleno de aire aparece de color oscuro. La presencia de este color blanco confirma el diagnóstico de la infección.

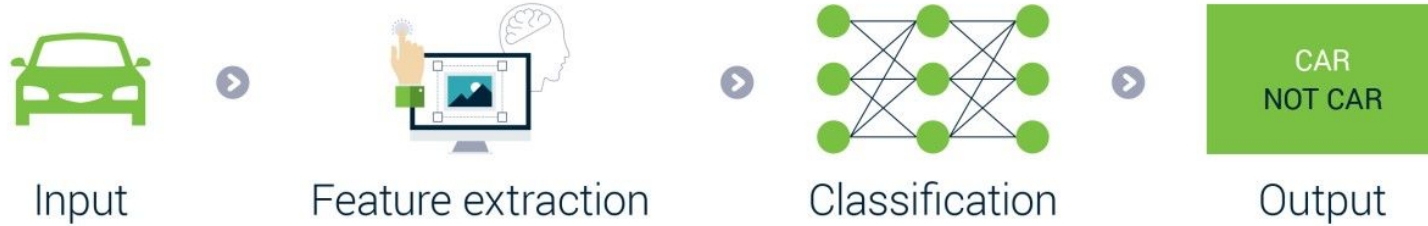


Enfermo

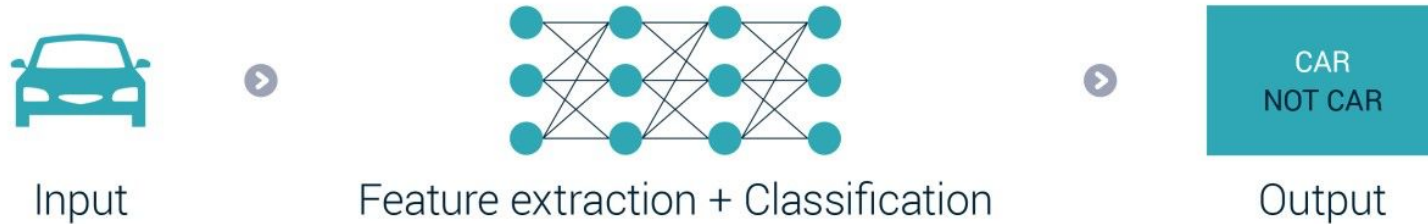
Sano

Inteligencia artificial: ML y DL

Machine Learning



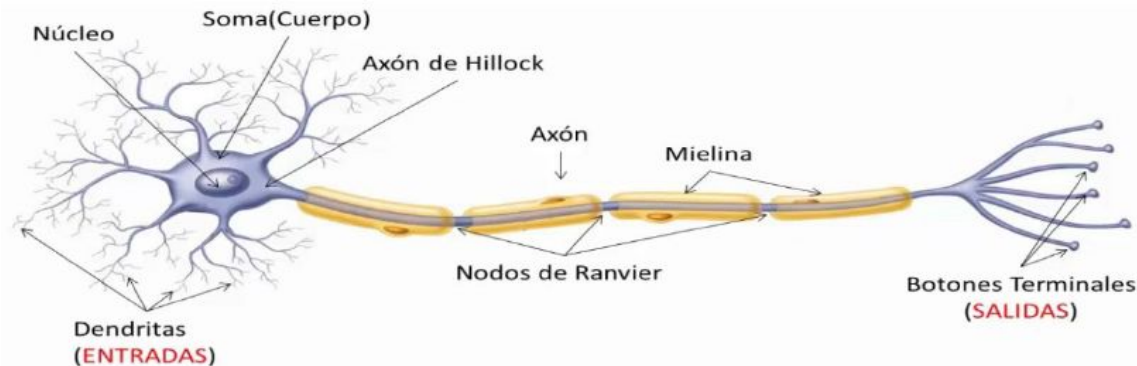
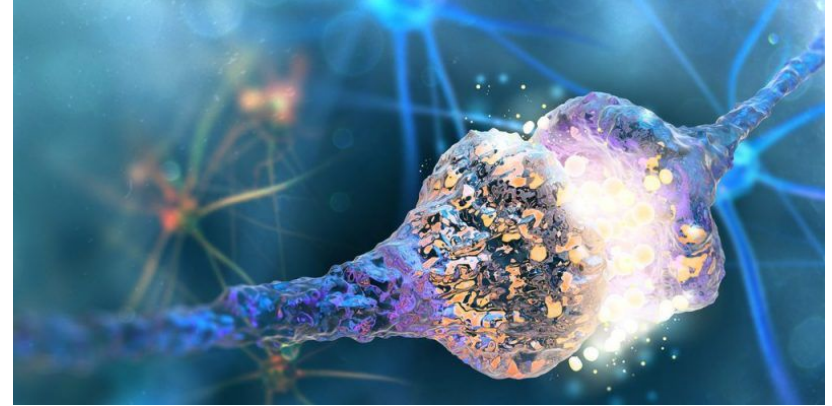
Deep Learning



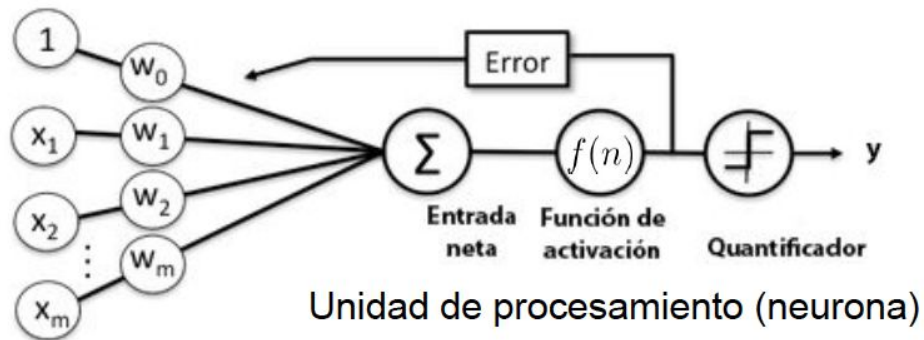
[Difference between Machine Learning y Deep Learning](#)

Redes neuronales artificiales

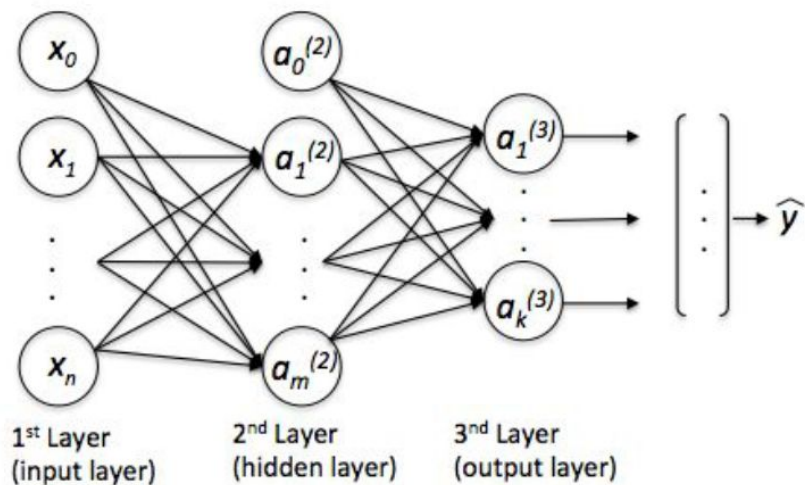
Una red neuronal artificial es un algoritmo que funciona como los científicos consideran que funciona nuestro cerebro. Nuestro sistema nervioso está constituido por células llamadas neuronas, que se comunican unas con otras a través de caminos llamados sinapsis. Estas células forman extensas redes que procesan la información de nuestro cuerpo para la toma de decisiones.



[Sinapsis neuronal](#)



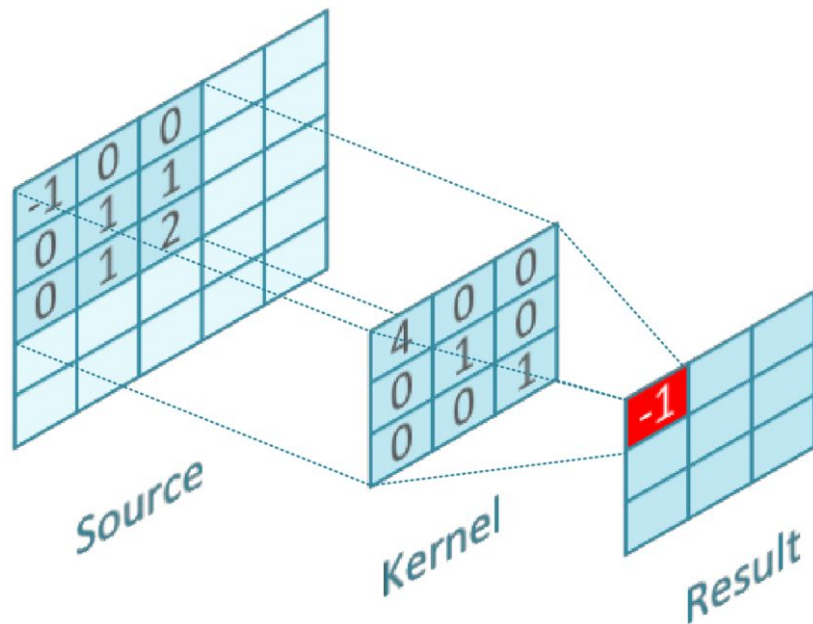
$$\hat{y} = f(n) = f(\mathbf{w}^T \mathbf{x} + b)$$

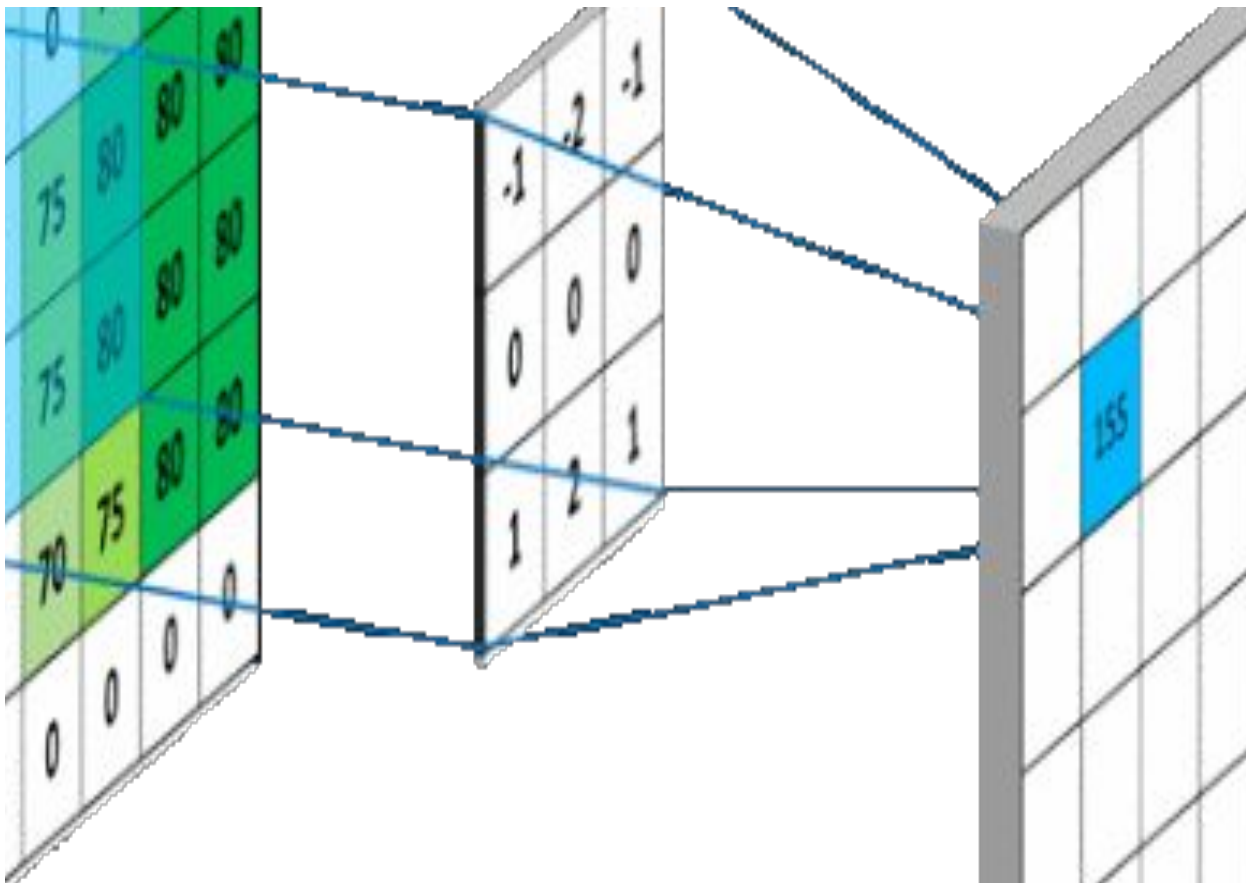


Red neuronal

Convolución

En teoría de señales, la convolución es una de las operaciones más importantes. En particular en el procesamiento digital de imágenes, la operación de convolución permite realizar un procedimiento llamado filtrado. Dicho procedimiento es llevado a cabo “haciendo pasar” una ventana de convolución (*kernel*) sobre toda la imagen y realizando una suma ponderada de sus elementos.





Convolution

Source layer

5	2	6	8	2	0	1	2
4	3	4	5	1	9	6	3
3	9	2	4	7	7	6	9
1	3	4	6	8	2	2	1
8	4	6	2	3	1	8	8
5	8	9	0	1	0	2	3
9	2	6	6	3	6	2	1
9	8	8	2	6	3	4	5

Convolutional
kernel

-1	0	1
2	1	2
1	-2	0

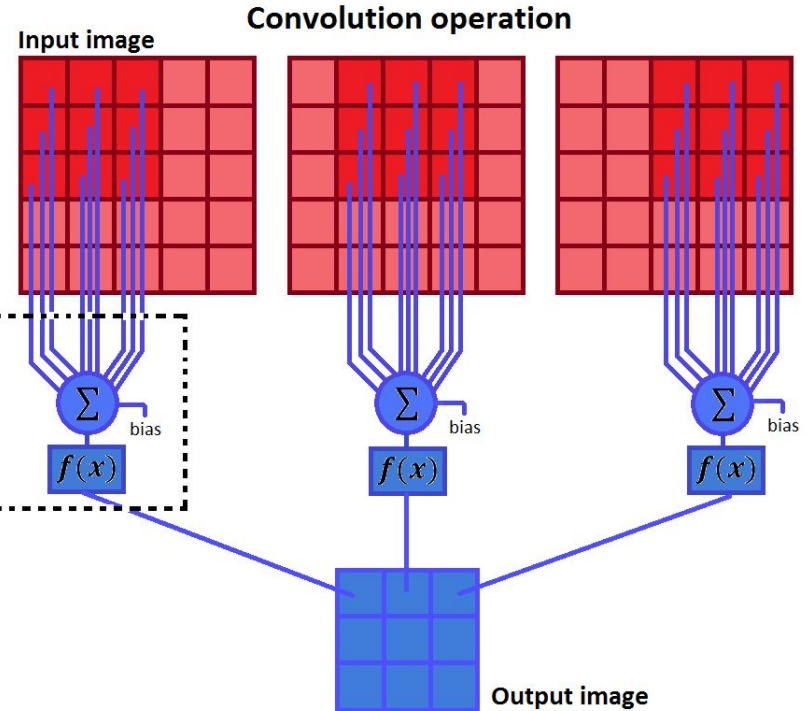
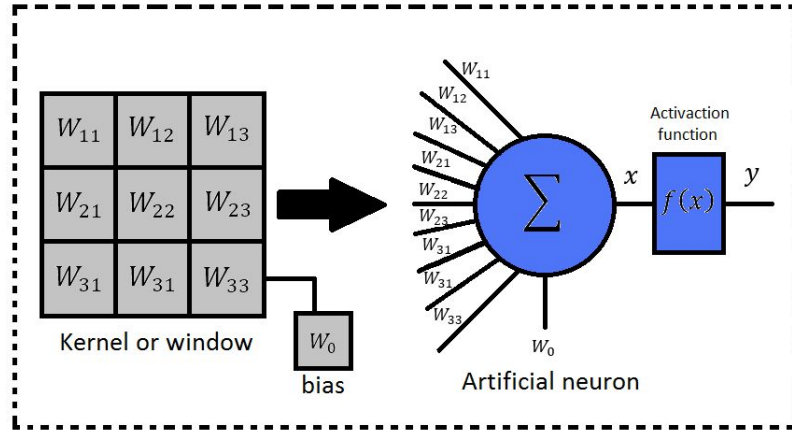
Destination layer

		5					

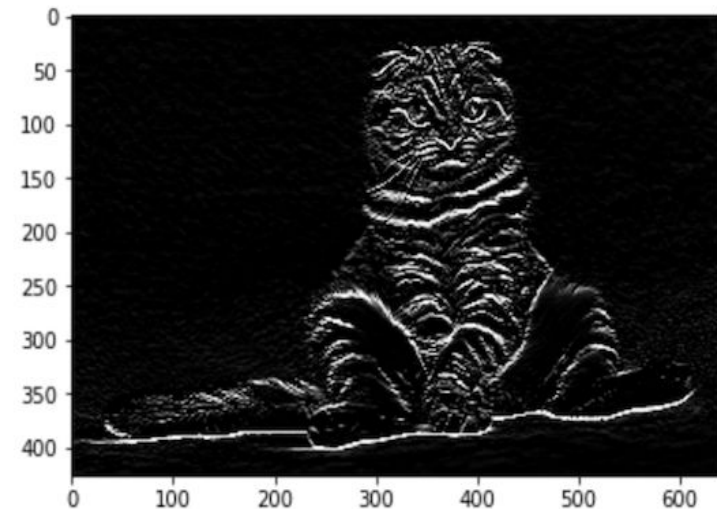
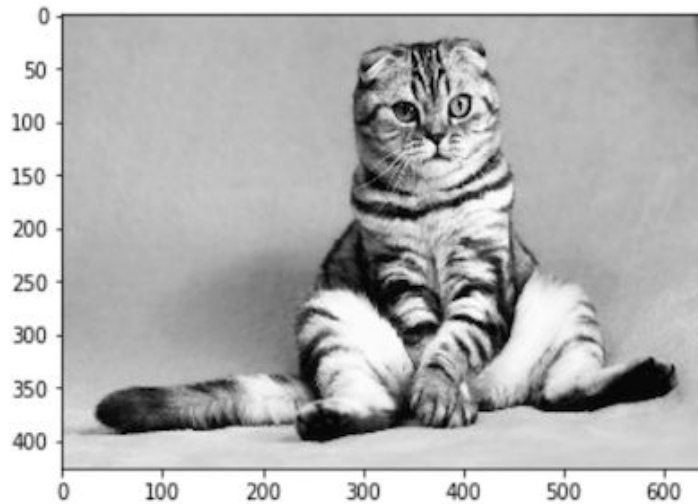
$$\begin{aligned} &(-1 \times 5) + (0 \times 2) + (1 \times 6) + \\ &(2 \times 4) + (1 \times 3) + (2 \times 4) + \\ &(1 \times 3) + (-2 \times 9) + (0 \times 2) = 5 \end{aligned}$$

Red Neuronal Convocucional (Convolutional Neural Network)

Relationship between an artificial neuron and the convolution window

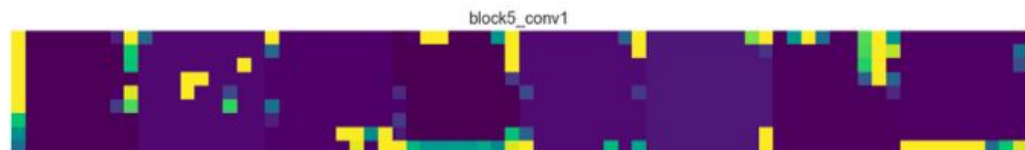
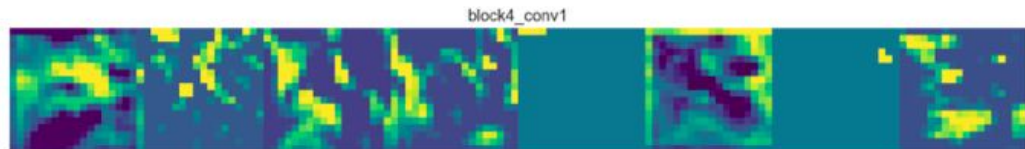
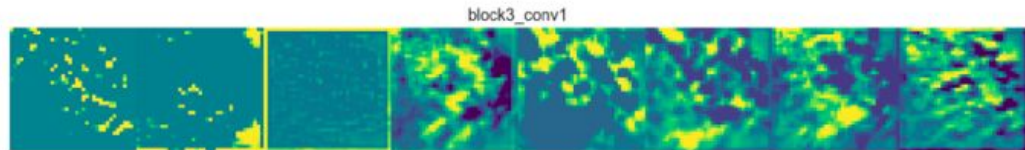
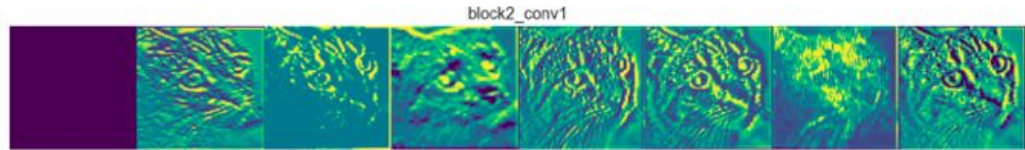
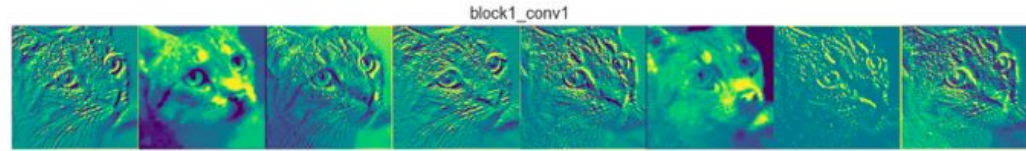


Estos filtros lo que hacen es resaltar características de los datos. En el caso de imágenes, esas características son comúnmente bordes, esquinas, cambio de nivel de color, entre otras. Cada capa convolucional genera nuevas imágenes llamadas *mapas de rasgos*.



Mapas de rasgos (Features Maps)


Imagen original






Lo que vamos a hacer en este taller es utilizar una modificación del dataset disponible en [Kaggle](#). Dicho dataset está compuesto de más de 5000 imágenes de rayos X de pacientes, de dos clases distintas: Neumonía/Normal

Chest X-Ray Images (Pneumonia)

5,863 images, 2 categories

 Paul Mooney • updated 2 years ago (Version 2)

[Data](#) [Tasks \(1\)](#) [Kernels \(567\)](#) [Discussion \(28\)](#) [Activity](#) [Metadata](#) [Download \(1 GB\)](#) [New Notebook](#) ⋮

 **Usability** 7.5  **License** Other (specified in description)  **Tags** online communities, online image galleries, health, biology, image data and 1 more

Description


Context

[http://www.cell.com/cell/fulltext/S0092-8674\(18\)30154-5](http://www.cell.com/cell/fulltext/S0092-8674(18)30154-5)

Normal

Bacterial Pneumonia

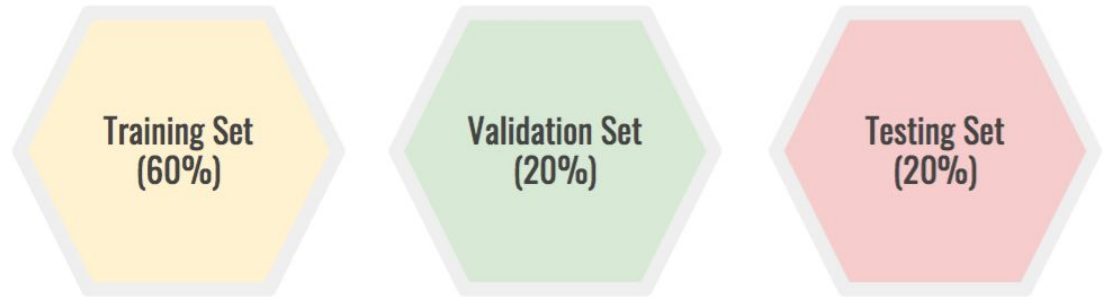
Viral Pneumonia



El dataset original contiene muy pocas imágenes en el conjunto de validación, por lo que se modificó para contar con más datos para validación y así evitar el sobreajuste.



- En este
- Entrenamiento → 60%
 - Validación → 20%
 - Prueba → 20



ó:

Reto 1: En el archivo *R/IAA20-DetecciónNeumonia* y en 5 min, complete las celdas las cuales permiten acceder a Google Drive, copiar localmente el dataset, descomprimirlo y importar las bibliotecas a utilizar.

Tips:

- Acepte los permisos de autorización para acceder a su Drive.
- Seleccione la carpeta de su Drive donde esta el dataset.
- De *Keras* importe *models* y *layers*.
- Consulte en “San Google”

Resultado esperado:

Go to this URL in a browser: <https://accounts.google.com/o/oauth2/a>

Enter your authorization code:

.....

Mounted at /content/drive



Solución:

```
[ ] # Accediendo a Google Drive
    from google.colab import drive
    drive.mount('/content/drive')
```

```
[ ] # Copiamos los archivos de Drive al entorno de Colab
    !cp "/content/drive/My Drive/Datasets/Neumonia_Dataset.zip" "Neumonia_Dataset.zip"
    !unzip -uq "Neumonia_Dataset.zip"
```

```
[ ] # Bibliotecas a emplear
    import os
    import cv2
    import keras
    import numpy as np
    from keras.preprocessing.image import ImageDataGenerator
    from keras import models
    from keras import layers
    import matplotlib.pyplot as plt
```

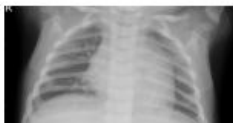
Reto 2: En el archivo *R/IAA20-DetecciónNeumonía* y en 5 min, complete las celdas las cuales permiten seleccionar de forma aleatoria cuatro imágenes de personas sanas y cuatro de personas con neumonía del conjunto de entrenamiento. Posteriormente muestre las imágenes.

Resultado esperado:

Tips:

- Concatene las rutas de las imágenes de entrenamiento, validación y prueba.
- Complete el ciclo for que toma cuatro imágenes de personas sanas y cuatro con neumonía del conjunto de entrenamiento.
- Complete el ciclo for que imprime las imágenes tomadas de forma aleatoria.
- Consulte en “San Google”

Primer fila: personas sin neumonía
Segunda fila: personas con neumonía



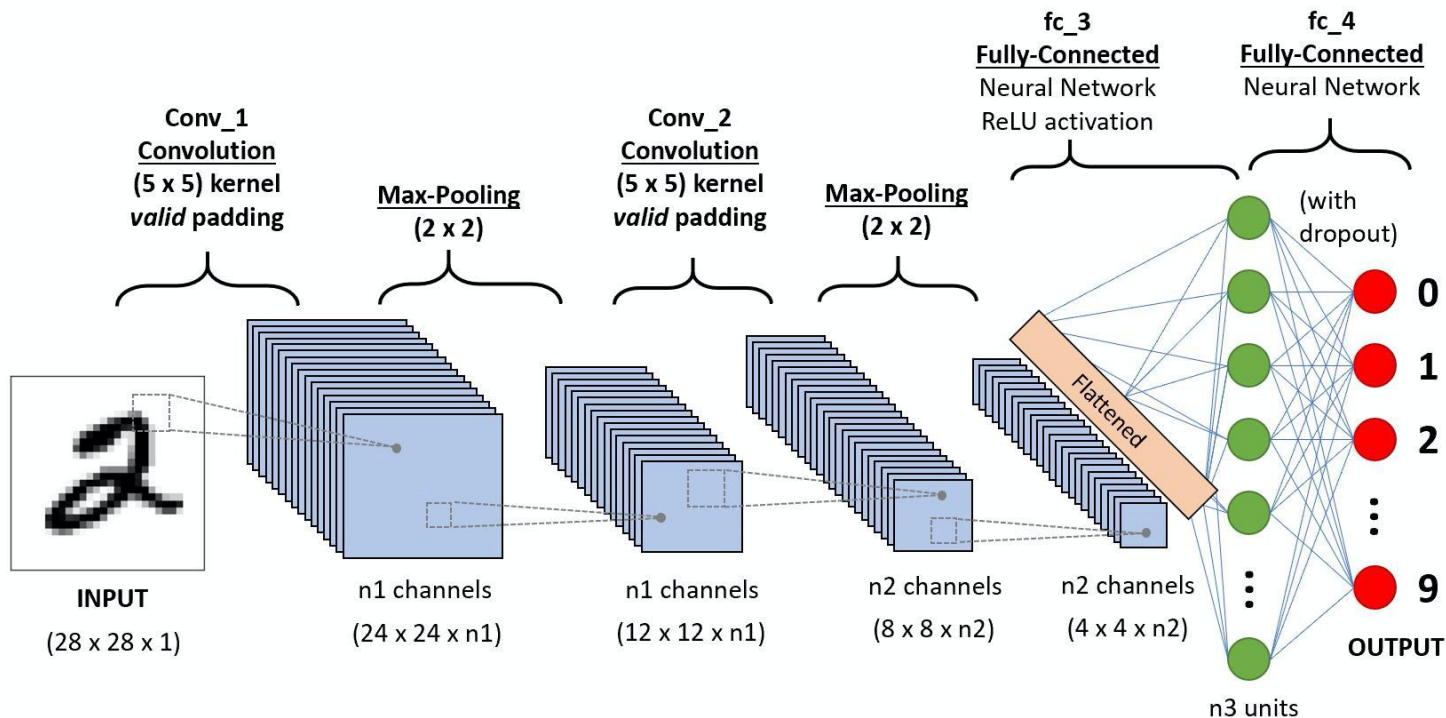
Solución:

```
[ ] # Definir las rutas donde estan las imagenes
train_folder= '/content/Neumonia_Dataset/train'
val_folder = '/content/Neumonia_Dataset/val'
test_folder = '/content/Neumonia_Dataset/test'
```

```
[ ] # Se toman 4 imagenes de forma aleatoria
os.listdir(train_folder)
train_sanas = train_folder+'/NORMAL/'
train_neum = train_folder+'/PNEUMONIA/'
img_sanas = []
img_neumonia = []
for i in range(4):
    num_alea = np.random.randint(len(os.listdir(train_sanas)))
    img_sanas.append(train_sanas + os.listdir(train_sanas)[num_alea])
    img_neumonia.append(train_neum + os.listdir(train_neum)[num_alea])
```

```
[ ] # Se muestran radiografías de personas con y sin neumonía
print('Primer fila: personas sin neumonía')
print('Segunda fila: personas con neumonía')
plt.rcParams['figure.dpi'] = 150
for num_imagen in range(8):
    if num_imagen<4:
        imagen = cv2.imread(img_sanas[num_imagen])
    else:
        imagen = cv2.imread(img_neumonia[num_imagen-4])
    plt.subplot(2,4,num_imagen+1)
    plt.imshow(imagen)
    plt.axis('off')
plt.show()
```

Red Neuronal Convolucional (Convolutional Neural Network)



Reto 3: En el archivo *R/IAA20-DetecciónNeumonia* y en 5 min, complete las celdas las cuales permiten construir la red neuronal convolucional. Esta red está compuesta por tres capas convolucionales, tres de *pooling*, una de flatten y dos capas densas. Muestra los detalle de la arquitectura propuesta.

Resultado esperado:

Tips:

- Las capas convolucionales ocupan funciones de activación ReLU.
- Emplee el método *compile()*
- Utilice el método *summary()*
- Consulte en “San Google”

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 148, 148, 32)	896
max_pooling2d (MaxPooling2D)	(None, 74, 74, 32)	0
conv2d_1 (Conv2D)	(None, 72, 72, 32)	9248
max_pooling2d_1 (MaxPooling2D)	(None, 36, 36, 32)	0
conv2d_2 (Conv2D)	(None, 34, 34, 32)	9248
max_pooling2d_2 (MaxPooling2D)	(None, 17, 17, 32)	0
flatten (Flatten)	(None, 9248)	0
dense (Dense)	(None, 128)	1183872
dense_1 (Dense)	(None, 1)	129
Total params: 1,203,393		
Trainable params: 1,203,393		
Non-trainable params: 0		

Solución:

```
[ ] # Red neuronal convolucional
    cnn = models.Sequential()

    # Capas convolucionales y de pooling
    cnn.add(layers.Conv2D(32, (3, 3), activation="relu", input_shape=(150, 150, 3)))
    cnn.add(layers.MaxPooling2D(pool_size = (2, 2)))
    cnn.add(layers.Conv2D(32, (3, 3), activation="relu"))
    cnn.add(layers.MaxPooling2D(pool_size = (2, 2)))
    cnn.add(layers.Conv2D(32, (3, 3), activation="relu"))
    cnn.add(layers.MaxPooling2D(pool_size = (2, 2)))
    cnn.add(layers.Flatten())

    # Capas densamente conectadas
    cnn.add(layers.Dense(activation = 'relu', units = 128))
    cnn.add(layers.Dense(activation = 'sigmoid', units = 1))

    # Compilar el modelo neuronal
    cnn.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics = ['accuracy'])

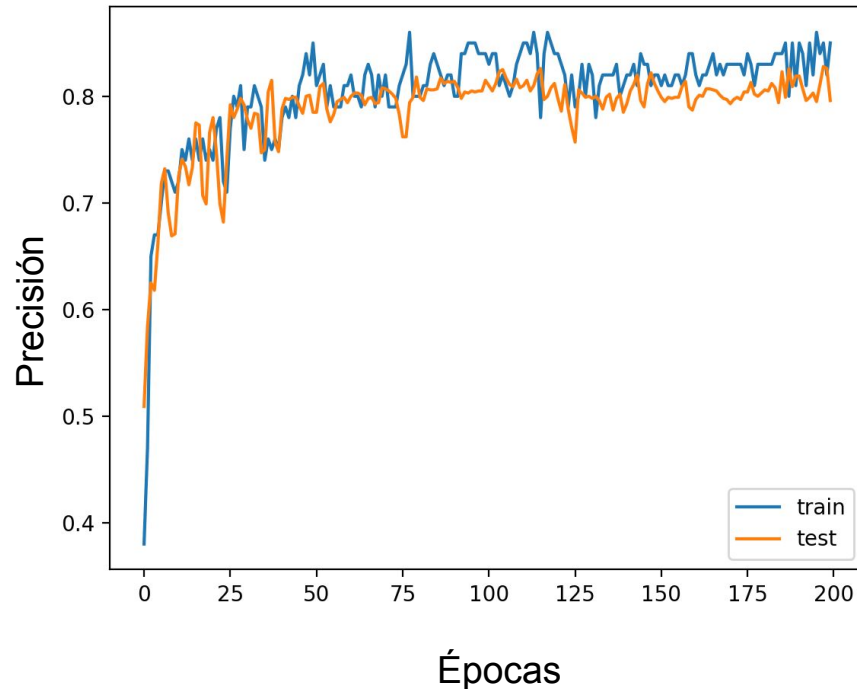
[ ] # Detalle de la red neuronal convolucional
    cnn.summary()
```

¿Qué es el aprendizaje de una red neuronal?

Los seres humanos cuando son bebés, no aprenden a caminar de inmediato. El aprendizaje se realiza durante un cierto periodo de tiempo y después de un proceso de repetición. Se comienza con asimilar pequeños componentes del movimiento del cuerpo humano y se ajusta con retroalimentación del ambiente, hasta que por fin se logra caminar lo suficientemente bien.



De una manera similar, las redes neuronales se entrenan realizando repeticiones llamadas *iteraciones*. Cuando una red procesa todos los datos del conjunto de entrenamiento se le conoce como *época*. Es importante hacer notar que no hay manera de pre-calcular cuántas épocas se necesitan y este es un hiperparámetro del modelo que debe ser ajustado para cada problema.



Reto 4: En el archivo *R/IAA20-DetecciónNeumonia* y en 5 min, complete las celdas las cuales permiten el preprocesamiento de las imágenes. Utilice lotes de 20 imágenes y considere que se trata de un modelo binario. Entrene la CNN utilizando 30 épocas, 100 pasos por época y 50 pasos de validación.

Tips:

- Para entrenar el modelo, utilice el método *fit()*
- Consulte en “San Google”

Resultado esperado:

```
Found 3512 images belonging to 2 classes.  
Found 1172 images belonging to 2 classes.  
Found 1172 images belonging to 2 classes.
```

```
Epoch 20/30  
100/100 [=====] - 36s 357ms/step - loss: 0.1643 - accuracy: 0.9378 - val_loss: 0.1665  
Epoch 21/30  
100/100 [=====] - 35s 353ms/step - loss: 0.1512 - accuracy: 0.9388 - val_loss: 0.1154  
Epoch 22/30  
100/100 [=====] - 36s 358ms/step - loss: 0.1596 - accuracy: 0.9362 - val_loss: 0.1241  
Epoch 23/30  
100/100 [=====] - 36s 357ms/step - loss: 0.1342 - accuracy: 0.9535 - val_loss: 0.1039  
Epoch 24/30  
100/100 [=====] - 36s 362ms/step - loss: 0.1329 - accuracy: 0.9455 - val_loss: 0.1193  
Epoch 25/30  
100/100 [=====] - 36s 358ms/step - loss: 0.1178 - accuracy: 0.9575 - val_loss: 0.1043  
Epoch 26/30  
100/100 [=====] - 36s 355ms/step - loss: 0.1469 - accuracy: 0.9493 - val_loss: 0.1032  
Epoch 27/30  
100/100 [=====] - 36s 363ms/step - loss: 0.1407 - accuracy: 0.9425 - val_loss: 0.1391  
Epoch 28/30  
100/100 [=====] - 36s 357ms/step - loss: 0.1254 - accuracy: 0.9475 - val_loss: 0.1461  
Epoch 29/30  
100/100 [=====] - 36s 359ms/step - loss: 0.1267 - accuracy: 0.9513 - val_loss: 0.1236  
Epoch 30/30  
100/100 [=====] - 36s 360ms/step - loss: 0.1128 - accuracy: 0.9545 - val_loss: 0.1254
```

Solución:

```
[ ] # Preprocesamiento de las imagenes
    train_datagen = ImageDataGenerator(rescale = 1./255,
                                       shear_range = 0.2,
                                       zoom_range = 0.2,
                                       horizontal_flip = True)

    # Normalización de imagenes
    test_datagen = ImageDataGenerator(rescale = 1./255)

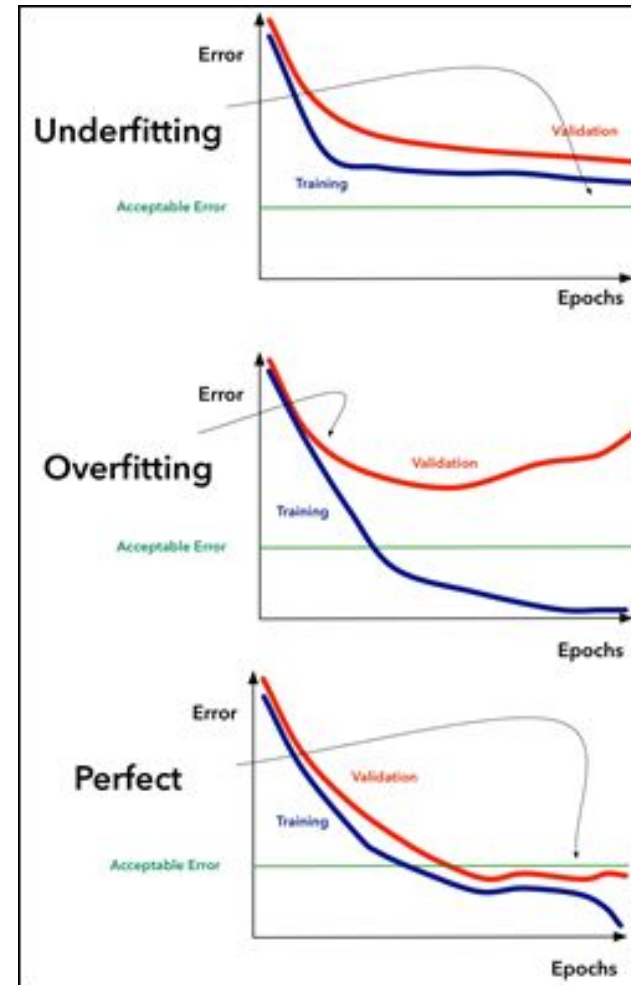
    # Generación de los conjuntos de entrenamiento, validación y prueba
    training_set = train_datagen.flow_from_directory(train_folder,
                                                    target_size = (150, 150),
                                                    batch_size = 20,
                                                    class_mode = 'binary')

    validation_generator = test_datagen.flow_from_directory(val_folder,
                                                           target_size=(150, 150),
                                                           batch_size=20,
                                                           class_mode='binary')

    test_set = test_datagen.flow_from_directory(test_folder,
                                                target_size = (150, 150),
                                                batch_size = 20,
                                                class_mode = 'binary')

[ ] cnn_model = cnn.fit(training_set,
                        steps_per_epoch=100,
                        epochs=30,
                        validation_data=validation_generator,
                        validation_steps=50)
```

Durante el proceso de entrenamiento, se pueden presentar dos circunstancias relacionadas con el desempeño. En primer lugar se tiene el *underfitting* en la cual de función de pérdida de entrenamiento y validación se encuentran por encima de un umbral aceptable. En otras palabras, el modelo ya no está aprendiendo. En el caso del *overfitting* la curva de validación y entrenamiento se empiezan a separar, esto indica que el modelo no está generalizando. Lo ideal es que la curva de validación y entrenamiento se encuentren por debajo de un umbral y no se separen.

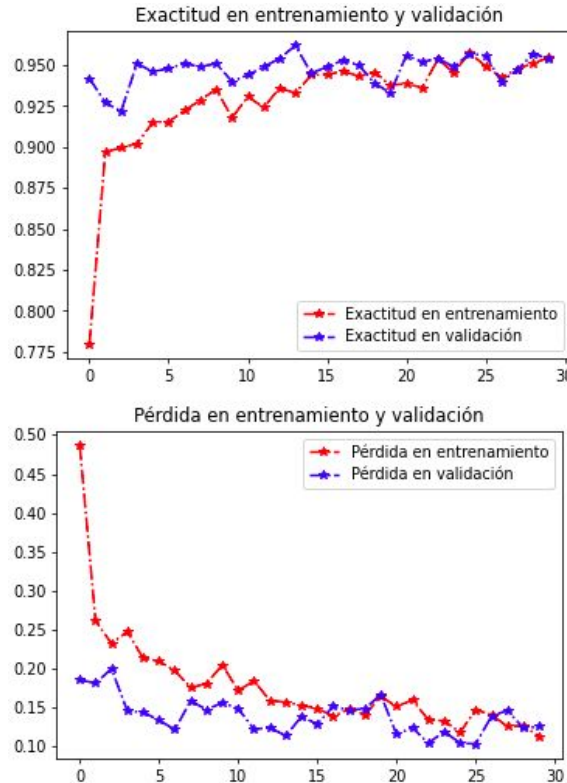


Reto 5: En el archivo *R/IAA20-DetecciónNeumonia* y en 5 min, grafique la exactitud y la pérdida en el entrenamiento y validación.

Tips:

- Utilice el atributo *history* para obtener los valores de exactitud y pérdida.
- En el método plot utilice los valores obtenidos de exactitud y pérdida.
- Consulte en “San Google”

Resultado esperado:



Solución:

```
[ ] # Graficas de la presición y función de perdida

acc = cnn_model.history['accuracy']
val_acc = cnn_model.history['val_accuracy']
loss = cnn_model.history['loss']
val_loss = cnn_model.history['val_loss']

epochs = range(len(acc))

plt.rcParams['figure.dpi'] = 70
plt.plot(epochs, acc, '-.r*', label='Exactitud en entrenamiento')
plt.plot(epochs, val_acc, '-.b*', label='Exactitud en validación')
plt.title('Exactitud en entrenamiento y validación')
plt.legend()
plt.figure()

plt.plot(epochs, loss, '-.r*', label='Pérdida en entrenamiento')
plt.plot(epochs, val_loss, '-.b*', label='Pérdida en validación')
plt.title('Pérdida en entrenamiento y validación')
plt.legend()

plt.show()
```

Reto 6: En el archivo *R/IAA20-DetecciónNeumonia* y en 5 min, complete las celdas las cuales permiten evaluar el desempeño del modelo generado con el conjunto de prueba. Utilice el modelo generado para predecir, dada una imagen, si la persona tiene o no neumonía.

Resultado esperado:

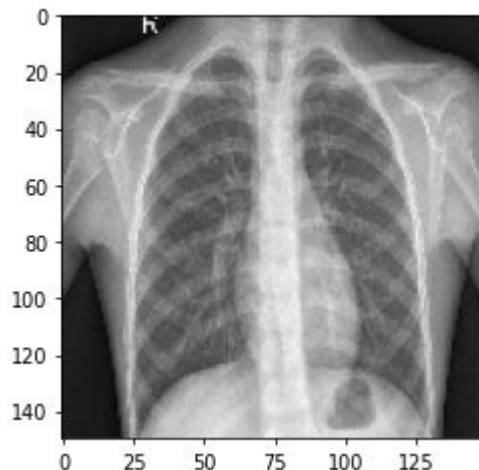
```
50/50 [=====] - 8s 162ms/step - loss: 0.3005 - accuracy: 0.9090
```

```
La exactitud en el conjunto de prueba es: 90.89999794960022 %
```

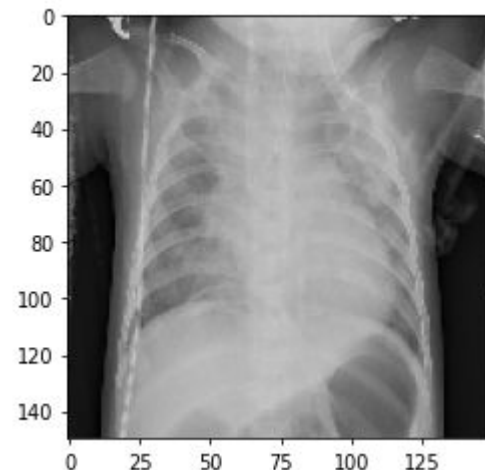
Tips:

- Utilice el método *evaluate()*.
- Utilice el método *resize()* de la biblioteca cv2.
- Emplee el método *reshape()* de la biblioteca numpy.
- Utilice el método *predict()* de la CNN.
- Consulte en “San Google”.

Persona sin neumonia



Persona con neumonia



Solución:

```
[ ] test_accu = cnn.evaluate(test_set,steps=50)
```

```
[ ] print('La exactitud en el conjunto de prueba es: ',test_accu[1]*100, '%')
```

```
[ ] # Predicción sobre una imagen de prueba
```

```
#img_ori = cv2.imread('/content/Neumonia_Dataset/test/NORMAL/IM-0011-0001-0002.jpeg')
```

```
img_ori = cv2.imread('/content/Neumonia_Dataset/test/PNEUMONIA/person119_bacteria_566.jpeg')
```

```
img_ori = cv2.cvtColor(img_ori, cv2.COLOR_BGR2RGB)
```

```
img = cv2.resize(img_ori, (150, 150), interpolation=cv2.INTER_CUBIC)
```

```
imagen_a_probar = np.reshape(img,(1,150, 150, 3))
```

```
predictions = cnn.predict(imagen_a_probar)
```

```
if(predictions == 0):
```

```
    print('Persona sin neumonia')
```

```
else:
```

```
    print('Persona con neumonia')
```

```
plt.imshow(img)
```

```
plt.show()
```


Conclusión Final

- El *deep learning* es una de las áreas de mayor crecimiento en los últimos años y ha provocado una revolución en el comunidad de inteligencia artificial.
- Detrás del aprendizaje profundo se encuentran las redes neuronales artificiales, que son algoritmos que buscan emular el comportamiento de las neuronas reales.
- Bibliotecas como TensorFlow y Keras permiten a los desarrolladores implementar algoritmos de *deep learning* en unas cuantas líneas de código.
- Las redes neuronales convolucionales son el modelo de *deep learning* más utilizado y son ampliamente utilizadas en tareas de visión por computadora.

Cursos



Python para IA

Programa en python aplicaciones de IA

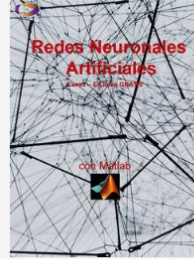
MÁS INFORMACIÓN



PLN para IA

Haz que una maquina converse

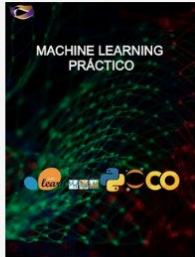
MÁS INFORMACIÓN



Redes Neuronales Artificiales

Aprende los conceptos básicos (GRATIS)

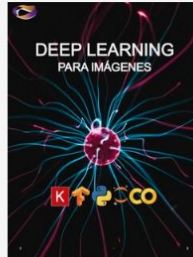
MÁS INFORMACIÓN



Machine Learning Práctico

Programa algoritmos que aprendan de los datos

MÁS INFORMACIÓN



Deep Learning para Imágenes

Haz que una maquina reconozca imágenes

MÁS INFORMACIÓN



Visión por Computadora con Python

Aprende a procesar imágenes (GRATIS)

MÁS INFORMACIÓN

<https://www.actumlogos.com/>

CURSOS

GRATIS

EQUIPO

CONTACTO

BLOG

Participa en la revolución tecnológica



ACTUMLOGOS
DESARROLLANDO HABILIDADES TECNOLÓGICAS

hola@actumlogos.com

wa.me/5215539940156

[Facebook](#)

[LinkedIn](#)