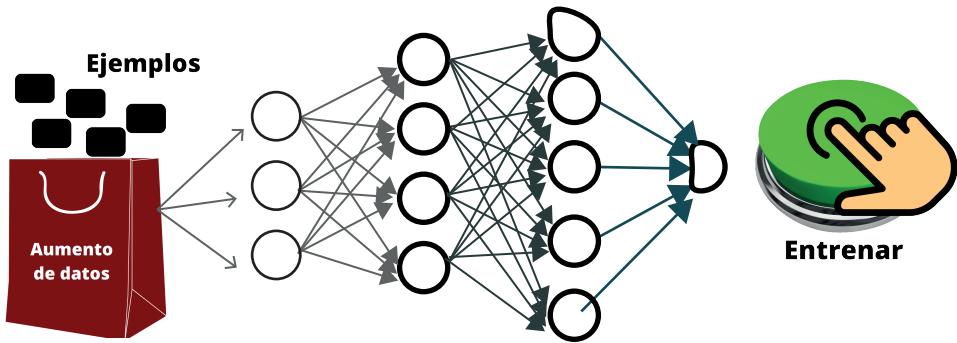


AUMENTO DE DATOS

Una de las causas del sobre entrenamiento es la poca información brindada al modelo, cuando validamos nuevos ejemplos este no será capaz de desempeñarse correctamente. Las bases de datos no siempre son abundantes y basicamente debemos trabajar con la información existente.

Al dar mucha información al modelo, este será expuesto a cada posible aspecto de la distribución de los datos y puede ser que jamás se sobre entrene.



El aumento de datos genera nuevos ejemplos para el entrenamiento a partir de los ya existentes, se hace a través de un número aleatorio de transformaciones. El objetivo es que durante el entrenamiento el modelo no vea dos veces la misma imagen o ejemplo, esto ayudará a exponerlo a más aspectos de los datos.



En keras esto se logra configurando el número de transformaciones utilizando las imágenes leídas por el `ImageDataGenerator`. Algunas de las transformaciones que podemos hacer son:

`rotation_range`: rango de valores en grados (0-180) en el que las imágenes rotarán.

`width_shift`: rango de valores en el que las imágenes serán trasladadas de forma horizontal o vertical.

`zoom_range`: rango de zoom que se aplicará a las imágenes.

`horizontal_flip`: gira la mitad de las imágenes horizontalmente.

A continuación se muestra un ejemplo de un aumento de datos, la base de datos utilizada es de tumores cerebrales y está disponible en el repositorio.

A PROGRAMAR

```
[1] from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing.image import array_to_img
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.preprocessing.image import load_img
import os

[2] DIR_no="/ubicacion_sin_tumores/"
DIR_si="/ubicacion_con_tumores/"
DIR_save_no= "/ubicacion_aumento_sin/"
DIR_save_si= "/ubicacion_aumento_con/"

[3] datagen = ImageDataGenerator(rotation_range=40,
                               width_shift_range=0.2,
                               height_shift_range=0.2,
                               shear_range=0.2,
                               zoom_range=0.2,
                               horizontal_flip=True,
                               fill_mode='nearest')

[4] no= os.listdir(DIR_no)
si= os.listdir(DIR_si)

[5] for i in range(len(DIR_no)):
    imn= os.path.join(DIR_no, no[i])
    ims= os.path.join(DIR_si, si[i])
    img= load_img(imn)
    x= img_to_array(img)
    x= x.reshape((1,)+ x.shape)
    img2= load_img(ims)
    y= img_to_array(img2)
    y= y.reshape((1,)+ y.shape)
    c= datagen.flow(x, batch_size= 1, save_to_dir= DIR_save_no,
                     save_prefix='no_'+ str(z), save_format= 'png')
    r= datagen.flow(y, batch_size= 1, save_to_dir= DIR_save_si,
                     save_prefix='si_'+ str(z), save_format= 'png')
    for tx in range(30):
        c.next()
        r.next()
```

METIENDONOS AL CODIGO

[1] Importar las librerías que vamos a necesitar.

[2] En dos variables guardamos la dirección donde se encuentran las imágenes y en otras dos variables la dirección donde queremos guardar las imágenes del incremento.

[3] Lo primero es generar un `ImageDataGenerator` con las características que vamos a necesitar. podemos configurar con más elementos nuestro `ImageDataGenerator` si es que así lo necesitamos, pero para nuestro ejemplo es más que suficiente con las que estamos colocando.

[4] Tomamos todas las direcciones de los elementos que se encuentran en nuestro directorio raíz, en este caso para las imágenes con y sin tumores.

[5] En el ciclo for es donde ocurre toda la magia. Este ciclo se ejecutará la misma cantidad de imágenes que tiene cada carpeta, para nuestro ejemplo tenemos la ventaja de que las bases de datos con y sin tumores tienen la misma cantidad de elementos por lo cual podemos realizar el incremento al mismo tiempo. Para cada imagen se generan 30 más, y de este proceso se encarga el segundo ciclo for, si deseamos más elementos debemos incrementar el rango del ciclo.

Algunos ejemplos de las nuevas imágenes son:

