

# Data augmentation para imágenes

Métodos Generativos, curso 2025-2026

---

Guillermo Iglesias, [guillermo.iglesias@upm.es](mailto:guillermo.iglesias@upm.es)

Jorge Dueñas Lerín, [jorge.duenas.lerin@upm.es](mailto:jorge.duenas.lerin@upm.es)

Edgar Talavera Muñoz, [e.talavera@upm.es](mailto:e.talavera@upm.es)

5 de noviembre de 2025

Escuela Técnica Superior de Ingeniería de Sistemas Informáticos | UPM



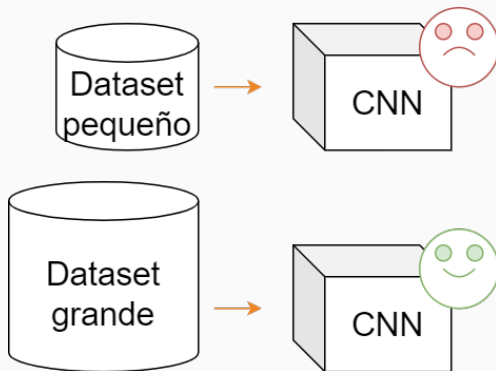
# Introducción al Data Augmentation

---

# Motivación

Cuando se realiza un **entrenamiento** con redes neuronales uno de los aspectos que más **limitan su rendimiento** es la falta de datos.

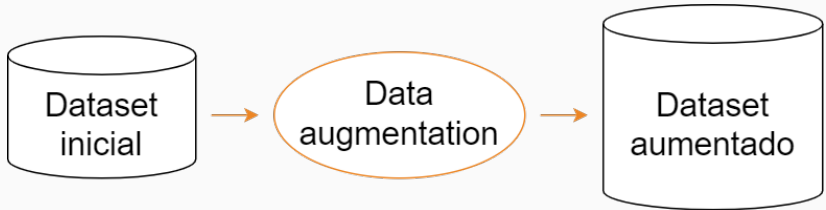
Cabe recordar que las redes neuronales son modelos **altamente exigentes** con la disponibilidad de datos. La **convergencia**, **rendimiento**, **resistencia al overfitting** o **estabilidad** son algunos de los aspectos sensibles a la variabilidad en los datos.



# ¿Qué es el data augmentation?

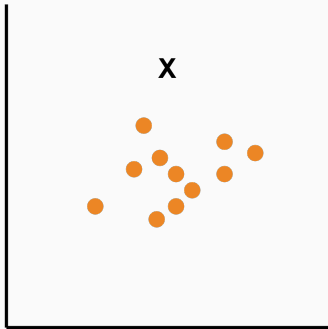
El proceso de **data augmentation** tiene como objetivo **aumentar sintéticamente** el tamaño del dataset que se usa. Para ello se aplican diferentes **transformaciones** al conjunto de datos que se tiene.

Con ello se obtienen **más datos** a partir de la información que se tiene originalmente. La **calidad** de estos datos dependerá de los algoritmos utilizados, el objetivo que se busca es **replicar** lo más fielmente la distribución de datos original.



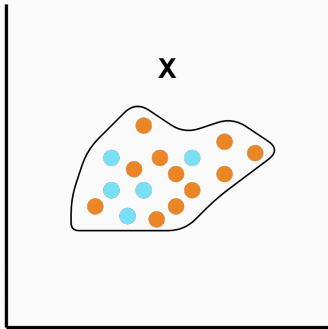
# Data augmentation como replica de la distribución de datos

Desde un punto de vista matemático, se pretende replicar la distribución de datos original, con cierta variabilidad en los nuevos datos, pero manteniendo su distribución original. Se busca generar nuevas instancias indistinguibles de las originales.



# Data augmentation como replica de la distribución de datos

Desde un punto de vista matemático, se pretende replicar la distribución de datos original, con cierta variabilidad en los nuevos datos, pero manteniendo su distribución original. Se busca generar nuevas instancias indistinguibles de las originales.



# Ventajas de uso de data augmentation

En el campo de las **redes neuronales artificiales** el uso de técnicas de aumento de datos permite **mejorar el rendimiento** en ciertos aspectos:

- **Overfitting**: Una de las principales causas de este problema es que los modelos son capaces de “**aprender**” los casos con los que fueron entrenados.
- **Mejor rendimiento**: Al conseguir una distribución de datos más rica se consiguen mejores **precisiones** a la hora de clasificar, al permitir a la red una mejor **generalización**.

# Implementaciones del data augmentation

Las técnicas de data augmentation, se utilizan para **todo tipo de datos**, independientemente de su origen. Sin embargo, **no todas las técnicas** pueden ser aplicadas a **todos los conjuntos** de datos de la misma manera.

Existen algoritmos de **data augmentation** específicos para distintos tipos de datos:

- **Señales**
- **Imágenes**
- **Datos tabulares**

Como norma general, las técnicas aplicadas deben ser cuidadosamente **estudiadas** para adaptarse al **problema** que se trate concreto.



## Data augmentation para imágenes

---

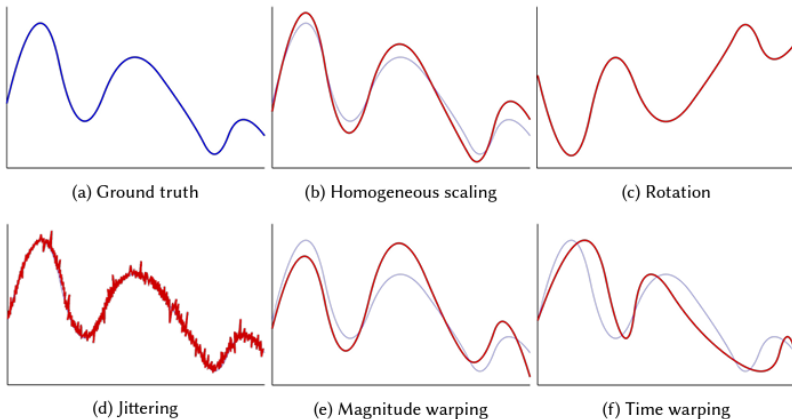
Existen un conjunto de técnicas de data augmentation que se pueden aplicar de manera general a las imágenes. Una de las ventajas de las imágenes es su fácil comprensión debido a que los resultados pueden ser validados de manera visual.

Sin embargo, cada problema es distinto y no siempre será posible aplicar todas las técnicas a cualquier problema. Siempre debe ser analizado cómo modifican los algoritmos el conjunto de datos.

Mantener un equilibrio entre variabilidad y similitud al conjunto inicial es un proceso complicado.

# Relaciones entre distintas técnicas de data augmentation

Por otra parte, ciertas técnicas de data augmentation pueden ser utilizadas para **distintos tipos de datos**. Por ejemplo existen grandes **similitudes** entre datos de **imágenes** y **señales**.



Existen una gran cantidad de algoritmos distintos para aumentar el número de imágenes de un dataset.

En este curso, se analizarán las siguientes:

- Recorte
- Volteo horizontal
- Volteo vertical
- Rotación
- Translación
- Cambios en brillo
- Cambios en contraste
- Cambios en matiz
- Cambios en la calidad

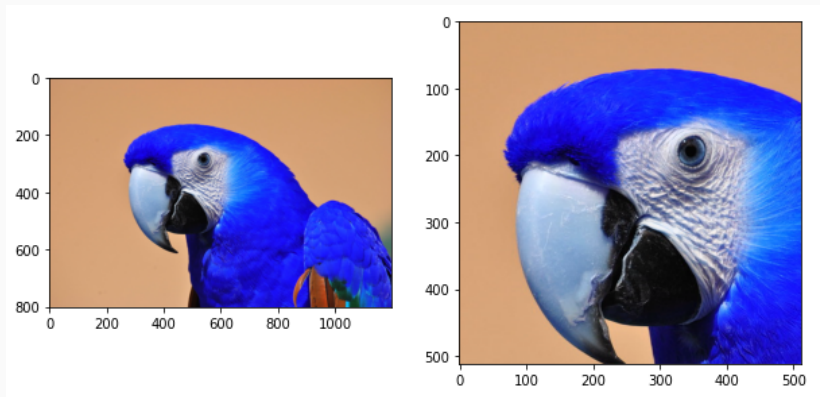
# Notebook de ejemplo, data augmentation

El siguiente notebook contiene las implementaciones de los algoritmos de data augmentation que se estudiarán.



- 1.3-DataAugmentation.ipynb

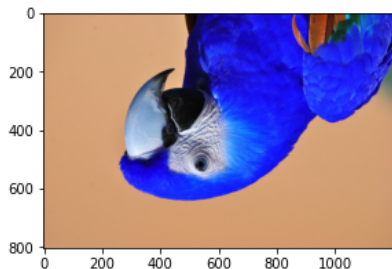
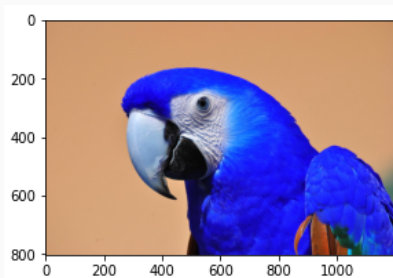
## Documentación de la capa RandomCrop



\* aplicar el formato “horizontal”

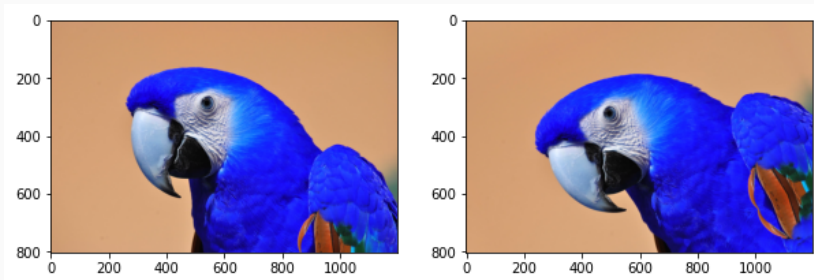
# Volteo horizontal

## Documentación de la capa RandomFlip



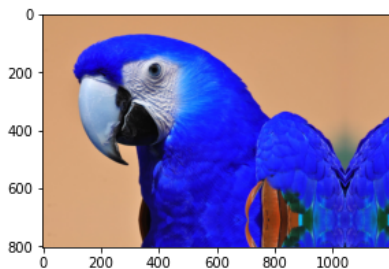
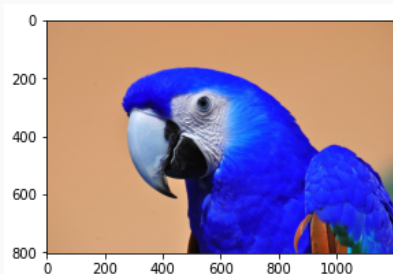
\* aplicar el formato “vertical”

## Documentación de la capa RandomRotation

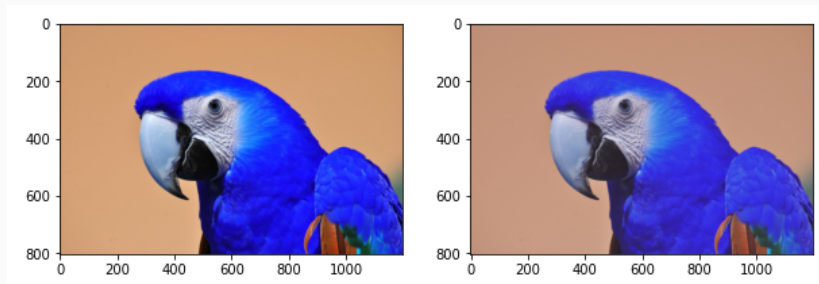




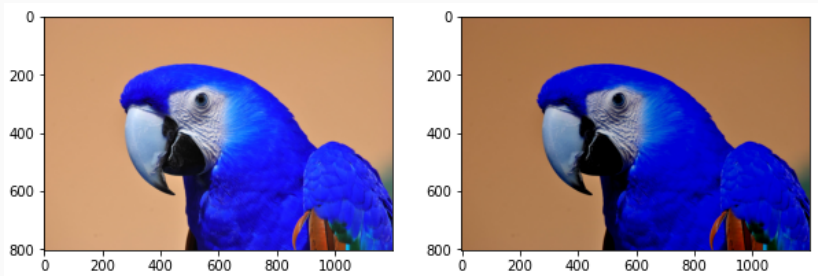
## Documentación de la capa RandomTranslation



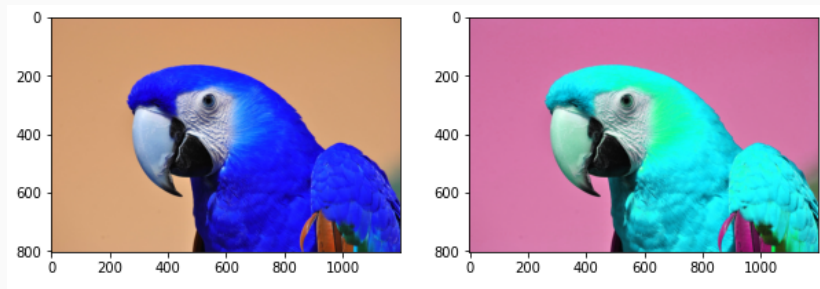
## Documentación de la capa RandomContrast



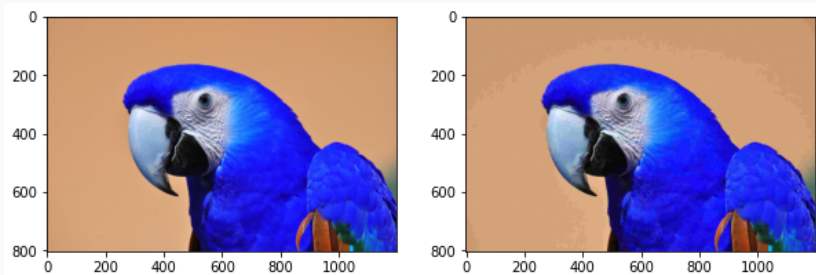
## Documentación de la capa `stateless_random_brightness`



## Documentación de la capa `stateless_random_hue`



## Documentación de la capa `stateless_random_jpeg_quality`



Existen dos posibilidades a la hora de realizar la implementación de los métodos de aumento de datos a la hora de realizar **entrenamientos** con redes neuronales.

- Capas de aumento de datos **dentro del modelo**.
- Aumento de datos aplicado directamente al **dataset**.

# Capas de data augmentation dentro del modelo

A la hora de crear un modelo con las librerías **Tensorflow/Keras** se pueden aplicar nuevas capas de aumento de datos en la **entrada** del modelo.

Esto hará que cada vez que el modelo **reciba las imágenes** como entrada, les aplique aleatoriamente las **transformaciones** deseadas.

```
model = tf.keras.Sequential([  
    # Add the preprocessing layers you created earlier.  
    resize_and_rescale,  
    data_augmentation,  
    layers.Conv2D(16, 3, padding='same', activation='relu'),  
    layers.MaxPooling2D(),  
    # Rest of your model.  
])
```

# Data augmentation al dataset

Otra alternativa es la de aplicar el aumento de datos **después de cargar el dataset**

```
aug_ds = train_ds.map(  
    lambda x, y: (resize_and_rescale(x, training=True), y))
```

*\* para más información se puede consultar la siguiente [guía oficial de Keras](#)*



- [1] Guillermo Iglesias, Edgar Talavera, Ángel González-Prieto, Alberto Mozo, and Sandra Gómez-Canaval.

**Data augmentation techniques in time series domain: a survey and taxonomy.**

*Neural Computing and Applications*, 35(14):10123–10145, 2023.

- Autor original de las diapositivas: Guillermo Iglesias Hernández