

**Trabajo de investigación:**

Preprocesamiento de imágenes para el entrenamiento de redes neuronales destinadas a la detección y clasificación de vehículos

María Carmona Pastor

Máster Universitario en Robótica y Automatización

2023-2024

Índice

[**1. Introducción 4**](#_heading=h.lnxbz9)

[**2. Marco Teórico 4**](#_heading=h.35nkun2)

[**3. Creación y etiquetado del conjunto de datos 4**](#_heading=h.1ksv4uv)

[3.1. Proceso 4](#_heading=h.44sinio)

[Tabla 1. 4](#_heading=h.2jxsxqh)

[Etiquetado 4](#_heading=h.4i7ojhp)

[Reetiquetado 4](#_heading=h.1pxezwc)

[3.2. Herramientas utilizadas 4](#_heading=h.2xcytpi)

[**4. Análisis de limitaciones 4**](#_heading=h.2xcytpi)

[4.1. Problemas detectados 4](#_heading=h.1ci93xb)

[4.2. Obtención de información del dataset 5](#_heading=h.3whwml4)

[**5. Propuesta de aumento de datos 5**](#_heading=h.2bn6wsx)

[**6. Conclusiones 5**](#_heading=h.qsh70q)

[**7. Bibliografía 5**](#_heading=h.3as4poj)

Índice de tablas

[Tabla 1. Distribución de imágenes del conjunto de datos de partida. 9](#_heading=h.2jxsxqh)

Índice de imágenes

[Ilustración 1. Estructura de una red neuronal convolucional [1]. 5](#_heading=h.rzmllu4du7h1)

[Ilustración 2. Tipos de furgonetas por tamaño[5]. 8](#_heading=h.v1zfxeagnkeg)

[Ilustración 3. Citröen C15 [6]. 8](#_heading=h.iqoefz3cbmbo)

[Ilustración 4. Ford Ranger [7]. 9](#_heading=h.z5dnpzysc9tt)

[Ilustración 5. Fiat Fiorino Cargo [8]. 9](#_heading=h.7ck6xpxbota5)

# Introducción

El objetivo de este trabajo es la preparación de un conjunto de datos (en adelante, *dataset*) válido que se utilizará en el proceso de entrenamiento de una red neuronal convolucional destinada al reconocimiento y la clasificación de vehículos. Concretamente, queremos dividir los vehículos detectados en las siguientes clases: moto (M), coche (C), furgoneta ligera (FL), furgoneta pesada (FP), autobús, (A), camión ligero (CL), camión pesado (CP) y camión pesado articulado (CPA).

El preprocesamiento de los datos es una fase crucial a la hora de obtener un buen modelo. Si los datos son incorrectos o están incompletos, el modelo no será capaz de aprender correctamente.

En este trabajo, se procederá al etiquetado, limpieza y análisis de los datos.

# Marco teórico

## Aprendizaje supervisado

Existen numerosas técnicas de aprendizaje profundo o *deep learning*. Estas se pueden dividir en aprendizaje supervisado o no supervisado. La característica definitoria del aprendizaje supervisado es que recibe como entrada *dataset* que llevan asociados una salida esperada. El modelo debe aprender los patrones necesarios para que sus predicciones se ajusten a dicha salida esperada. Por el contrario, en el aprendizaje no supervisado, no se dispone de una salida esperada, si no que el modelo aprende patrones y características presentes en los datos, pero estos patrones no están “predefinidos”, los datos no están etiquetados [1].

Muchas veces, las etiquetas asociadas a los datos en el aprendizaje supervisado se refieren a ciertas clases en las que se quieren clasificar los datos de entrada. Esto se denominan problemas de clasificación, como es el caso del nuestro.

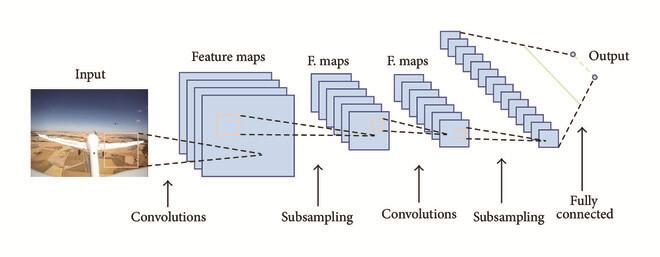
Utilizaremos una red neuronal convolucional (CNN, por sus siglas en inglés), que entra dentro de la categoría del aprendizaje supervisado.

## Redes neuronales convolucionales

Este tipo de modelos tiene un rendimiento superior a otros a la hora de procesar imágenes o entradas de audio [2], motivo por el cual lo hemos escogido para esta tarea.

Las redes neuronales se componen de capas de nodos que están conectados unos a otros. La red recibe como entrada los valores numéricos asociados a cada píxel en una imagen. Las CNNs tienen al menos una capa convolucional y una capa totalmente conectada como capa final, pudiendo tener más capas convolucionales intermedias, así como capas de agrupación [2].

* Capa convolucional: se trata de un filtro o *kernel* que recorre partes de la imagen, y determina si una característica (línea, curva, textura…) existe o no en ese subconjunto de píxeles. El filtro es una matriz, y dependiendo de los valores o pesos que tenga, detectará una característica u otra. El resultado de pasar la imagen por este filtro es una matriz que representa ciertas características de la imagen. En la primera capa se detectarán elementos pequeños y, con cada capa adicional, la red será capaz de detectar características mayores o más abstractas.
* Capa de agrupación: su función es reducir la dimensionalidad del problema. No siempre están presentes. También utiliza un *kernel*, como la capa convolucional, pero en este caso no tiene pesos asociados, si no que representa una función de agregación; por ejemplo, seleccionar el píxel con el valor más alto.
* Capa totalmente conectada: la última capa siempre estará totalmente conectada; es decir, cada nodo de esta capa está directamente conectado a un nodo de la capa anterior. De esta forma la salida de la red tendrá en cuenta todas las características extraídas en las capas anteriores.



##### Ilustración 1. Estructura de una red neuronal convolucional [1].

## YOLOv8

Utilizaremos la arquitectura YOLOv8, de la serie *You Only Look Once*. Esta arquitectura se utiliza para detectar objetos en imágenes en tiempo real, y destaca por su velocidad. Es capaz de realizar tareas de detección, clasificación, segmentación y detección de poses [3].

## Preprocesamiento de datos

El preprocesamiento de datos se refiere a la fase previa al entrenamiento de la red. En esta fase se decide qué datos de entrada se utilizarán en el entrenamiento. También es recomendable realizar un análisis de los datos. Los datos incluyen las imágenes y sus etiquetas.

Dentro del preprocesamiento de datos entran el perfilado, limpieza, reducción, transformación, enriquecimiento y validación de los datos [4]:

* El perfiladode datos se refiere a la obtención de estadísticas que ayudan a comprender la estructura y la calidad de los datos.
* La limpiezade datos consiste en corregir o eliminar datos incorrectos, inexactos, duplicados o corruptos que podrían introducir ruido en el modelo.
* La reducciónde datos consiste en la eliminación controlada de datos con el objetivo de reducir el volumen, pero manteniendo resultados suficientemente parecidos.
* La transformaciónde datos consiste en modificar o extraer nuevos datos que pueden ser beneficiosos según el problema.
* En el enriquecimientose añaden datos provenientes de otras fuentes, o bien derivados de los datos originales. El aumento de datos entraría en esta fase.
* La validación comprueba que los datos cumplen ciertos criterios específicos, garantizando que sean completos, precisos y confiables.

El preprocesamiento es una fase crucial en el desarrollo de modelos de aprendizaje automático, y tiene un impacto muy alto en la calidad y precisión de los resultados finales.

## Requisitos de un buen dataset

A continuación, se describen las características básicas que debe tener un *dataset* para obtener buenos resultados en el entrenamiento:

* Diversidad y representatividad: deben existir datos que representen las máximas situaciones posibles que podrían darse. Por ejemplo, si es posible que el modelo reciba imágenes tomadas de día y de noche, debe haber suficientes ejemplos de los dos casos en el *dataset*.
* Tamaño adecuado: debe haber suficientes datos. Un número aceptable suele ser entre 1000 y 10.000 imágenes.
* Equilibrio de clases: cada clase debe tener un número similar de ejemplos.
* Etiquetado preciso: las etiquetas asociadas a cada imagen deben ser precisas y consistentes.
* Buena calidad de imagen: la resolución de la imagen debe ser aceptablemente alta.

# Creación y etiquetado del conjunto de datos

## Definición de clases

Partimos con ventaja en este problema, dado que en un trabajo anterior, se realizaron experimentos para obtener un modelo capaz de clasificar vehículos en las siguientes categorías:

0: Moto (M)

1: Coche (C)

2: Furgoneta (F)

3: Autobús (A)

4: Camión ligero (CA)

5: Camión pesado (CP)

6: Camión pesado articulado (CPA)

Como se puede observar, cada clase debe llevar un identificador (número entero) asociado, y cada una tiene su abreviatura.

En este trabajo, añadiremos una nueva categoría a la clasificación, dividiendo la clase “Furgoneta” en “Furgoneta ligera” y “Furgoneta pesada”.

0: Moto (M)

1: Coche (C)

2: Furgoneta ligera (FL)

3: Furgoneta pesada (FP)

4: Autobús (A)

5: Camión ligero (CA)

6: Camión pesado (CP)

7: Camión pesado articulado (CPA)

A continuación, se establecen los siguientes criterios para diferenciar entre FL y FP.



##### Ilustración 2. Tipos de furgonetas por tamaño[5].

En la Ilustración 2, vemos una clasificación de furgonetas por tamaño. Decimos que una furgoneta es ligera si se encuentra por debajo del umbral de longitud del tipo L1H1, cualquier cosa por encima será pesada. Si un vehículo es más largo que L1, pero más bajo que H1, se considerará pesado. Si es más alto que H1, pero más corto que L1, se considerará ligero. Es decir, la longitud es el factor predominante, mientras que la altura se utiliza para “desempatar”.

En la práctica, es difícil medir las distancias con exactitud en una foto, de modo que, si no lo vemos claro, se pueden seguir la siguientes reglas:

* Si podemos ver el lateral del vehículo, y vemos que caben 4 ruedas o más entre sus ruedas, es pesada, si no, es ligera.
* Si no podemos ver bien el lateral, discriminamos por altura. Si es difícil determinar si es más o menos alta que el umbral establecido, tomaremos como referencia algún coche que aparezca a una distancia similar en el mismo escenario.

En las Ilustraciones 3 a 5 se muestran ejemplos de furgonetas ligeras.



##### Ilustración 3. Citröen C15 [6].



##### Ilustración 4. Ford Ranger [7].



##### Ilustración 5. Fiat Fiorino Cargo [8].

Una vez tenemos las clases bien definidas, para entrenar un modelo con YOLOv8, es necesario etiquetar las imágenes con un formato determinado. Cada archivo de imagen debe tener un archivo “.txt” asociado, con el mismo nombre que la imagen. Dicho archivo tendrá una línea por cada vehículo presente en la imagen, con el siguiente formato:

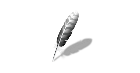
id\_clase x\_centro y\_centro anchura altura

Donde id\_clase se refiere al identificador de clase, (x\_centro, y\_centro) se refiere a las coordenadas, en píxeles, de la *bounding box* del vehículo, contadas desde la esquina superior izquierda de la imagen, y anchura y altura se refieren a las dimensiones de la *bounding box*, también en píxeles.

En el conjunto inicial de datos, tenemos algunas imágenes sin etiquetar, y otras muchas que están etiquetadas, pero estas no incluyen las clases FL y FP, si no que todas las furgonetas están marcadas como FL. El primer paso en la creación del *dataset* será añadir las etiquetas que faltan, y modificar el resto.

## Herramientas utilizadas

Label Studio: un programa destinado al etiquetado de datos, se ha utilizado para crear las etiquetas de las imágenes que no las tenían.

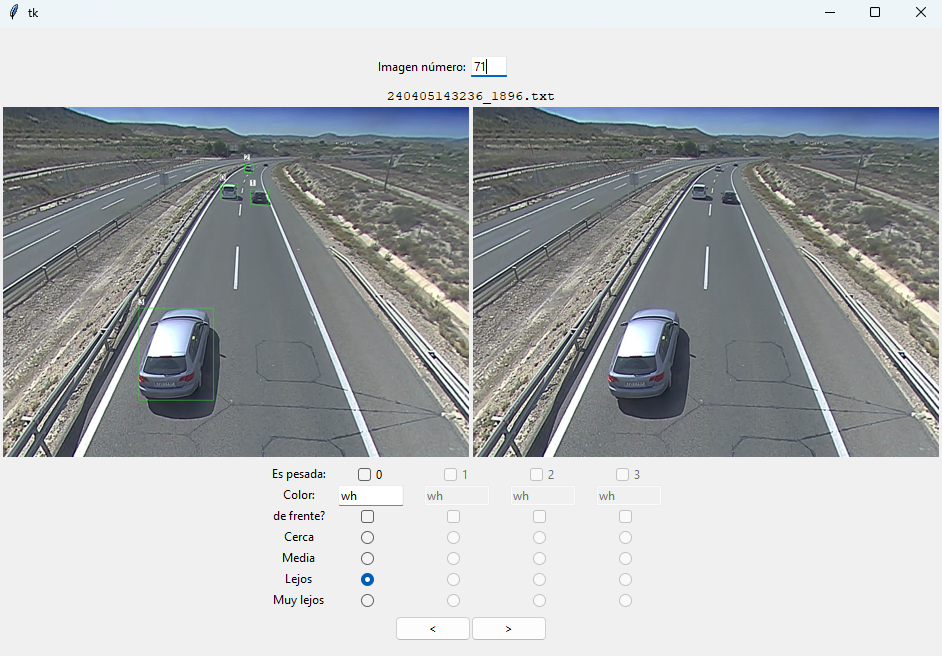
Tkinter: una interfaz en python para el kit de herramientas de GUI Tk, que se utiliza para crear aplicaciones que corren en Windows, Mac OS X y Linux. En este caso, se ha utilizado para crear una herramienta personalizada que permite cambiar las clases de los vehículos en pocos clicks, a partir de las *bounding boxes* ya existentes. También se ha añadido funcionalidad para guardar otras estadísticas de los vehículos, no sólo la clase.

## Proceso

Partimos de tres conjuntos de datos, organizados según la [Tabla 1](#_heading=h.2jxsxqh). Podemos ver que la mayor parte de las imágenes (conjunto *etiquetadas*) ya tenían un archivo con la información, pero no con las clases que necesitamos.

| **Nombre a la carpeta en el repositorio** | **Referencia en este documento** | **Preetiquetadas** | **Nº de imágenes** |
| --- | --- | --- | --- |
| FURGONETAS LIGERAS | *ligeras* | No | 97 |
| FURGONETAS PESADAS | *pesadas* | No | 47 |
| furgonetas\_27\_06\_2023 | *etiquetadas* | Si | 2254 |
| *Tabla 1. Distribución de imágenes del conjunto de datos de partida.* | | | |

### To approach this issue, we must utilize a small, independent script to categorize and relabel the images into the correct format. This script will run in a window, showing the original image and the one containing bounding boxes and will then be able to edit the information contained within each image relating to the different vehicles and their class and size. The data is stored as number sequences inside the image name.



### 

### para solucionar -> creacion del script de etiquetado…

descripcion del ui

how data is stored

### Etiquetado

The labeled images are lacking bounding boxes so the next step is to manually add them utilizing a program called label studio. This furthers our goal by allowing the image collection to be more accurately processed. For the training, the information has to be stored as a yaml file so the transfer is also handled at this stage.

used label studio

paso a formato yaml

# Análisis de limitaciones

## Problemas detectados

Once all the images had been relabeled and processed, their lacking nature became evident in the following fields:

Redundancy:  
 Al ser imagenes de trafico, many of the images were to all intents and purposes redundant, making them low value for training a model. Some pictures had been taken within seconds which made them practically the same while others differed only in a few degrees of change in the horizontal angle of the camera. A big portion of the more than 2000 images was unfortunately redundant in this way.

Ambiguous:

Al ser imagenes de trafico, the resolution is not ideal for training. Many of the images contained ambiguity with some vehicles being only fuzzy rectangles in the distance. A lot of the images also require the context of previous or later images, where the vehicles was or will be closer and therefore more visible. This proved daunting in the labeling stage, leading to a lot of assumptions.

Color variation:

Vans are commonly produced in one of three colors: white, gray, or black. Because of this, there was little to no content relating to other colored vans, which would lead to a huge weakness in the trained model for recognizing and identifying the different types of vans.

Location variation:

The entire 2000+ image dataset is all contained within 5 or 6 locations. Of these, only one of them is within an urban setting and only one is in a complex road network. The others are taken on highways with very non-descript surroundings. The trained model would struggle greatly in a city setting or on a winding road.

Road state:

The dataset is composed mainly of light to no traffic images. There is no content including more than 10 vehicles or with single lane build up. There is nothing with mild to heavy traffic.

Angle variation:

Having been taken in only the aforementioned 5 or 6 locations, the angles at which the different vehicles are photographed are lacking. There is no top or true side view in the dataset and the few raised angles utilized are hardly distinct amongst themselves.

Weather variation:

All images were taken on sunny or clear days. There is no rain, hail, cloud cover, or fog present in any of the images, making it impossible for the model to identify vans correctly through these weather conditions. This proves to be a massive weakness in the final model and is a priority within the data augmentation portion of this project.

Daytime variation:

Only a small portion of the images are taken at night and in none of the locations are there pictures taken at varied times of day such as midday and midnight. This makes training hard, especially in darker conditions where the already failing resolution of the images will be pushed even harder. This also makes it much harder for the model to recognize other aspects of the images like the van color or its distance from the camera.

Class imbalance:

Las imágenes proporcionadas para el entrenamiento son únicamente de furgonetas, lo cual, teniendo en cuenta que el modelo, idealmente, debería poder clasificar correctamente en las 7 clases proporcionadas, es un problema si queremos conseguir un modelo que generalize correctamente.

imagenes redundantes/repetidas

imagenes ambiguas

poca variedad de color

poca variedad de localizacion

poca variedad de angulo

poca variedad meteorologica

poca variedad cronologica

class imbalance

## Obtención de información del dataset

Explicar los añadidos a la interfaz grafica del programilla. Foto gui.

Añadir graphs with the info generated.

tkinter

graficos con info

pie chart

column graphs

# Propuesta de aumento de datos

The most important issue at hand is the lacking variation present in the dataset. Thankfully, most of these failings can be easily resolved with the use of filters or hue/brightness/saturation manipulation. Weather can be easily simulated and the weather, daytime, and color issues can be fixed with simple shifts in the color makeup of the image. Other than the color, we can also tinker with the image itself by cropping it into different sizes or flipping it horizontally.

color variation

weather filters

class balancing with cropping of images

flip horizontal

5.1 dataset filter

# Conclusiones

A pivotal aspect of this pre-training phase is the proper labeling of the dataset. Errors here will propagate into our final model, making it much harder for the training to produce satisfying and effective results. This stage contains the highest risk in human error as well, making it all the more vital that it be handled with the proper attention. Creating the labeling script will ease the flow for the labeling which will, in turn, improve consistency between images and allow for a faster and continuous labeling, minimizing the strain on the human component of the process.

Through labeling the dataset, it became evident that one of the biggest challenges going forward is the lack of diversity. Using different data augmentation methods, we can improve the variation within the dataset to maximize training potential. By manipulating what little content we have we can markedly improve the diversity within the dataset and allow for the model to properly train on as many different road conditions as we can manage.

we will figure this out at 3am

# Trabajo futuro

Se procederá a realizar el aumento de datos especificado, así como el entrenamiento sobre una red pre-entrenada con el *dataset* obtenido.

# Bibliografía

| [1] | A. Carrio, C. Sampedro, A. Rodrigues-Ramos y P. Campoy, «A Review of Deep Learning Methods and Applications for Unmanned Aerial Vehicles,» *Journal of sensors,* vol. 2017, 2017. |
| --- | --- |
| [2] | IBM, «¿Qué son las redes neuronales convolucionales?,» [En línea]. Disponible en: https://www.ibm.com/es-es/topics/convolutional-neural-networks. [Último acceso: 27 agosto 2024]. |
| [3] | Ultralytics, «Ultralytics YOLO Docs» [En línea]. Disponible en: https://docs.ultralytics.com/tasks/. [Último acceso: 2 septiembre 2024]. |
| [4] | F. Khan, «¿Qué es el preprocesamiento de datos? Definición, importancia y pasos,» Astera, 10 mayo 2024. [En línea]. Disponible en: https://www.astera.com/es/type/blog/data-preprocessing/. [Último acceso: 28 agosto 2024]. |
| [5] | espacio FURGO, «Guía para elegir furgoneta: ¿L1 H1, L1 H2, L3 H3, L4 H3?… el tamaño sí que importa» [En línea]. Disponible en: https://www.espaciofurgo.com/guia-para-elegir-furgoneta-l1-h1-l1-h2-l3-h3-l4-h3-el-tamano-si-que-importa/. [Último acceso: 2 septiembre 2024]. |
| [6] | coches.net, «Furgonetas pequeñas: 3 modelos para 9 marcas» [En línea]. Disponible en: https://www.coches.net/noticias/furgonetas-pequenas. [Último acceso: 2 septiembre 2024]. |
| [7] | Ford, «NUEVO FORD RANGER» [En línea]. Disponible en: https://www.ford.es/furgonetas-pick-up/nuevo-ranger#. [Último acceso: 2 septiembre 2024]. |
| [8] | autonoción.com, «¿Buscando una furgoneta pequeña? Aquí las pocas alternativas» [En línea]. Disponible en: https://www.autonocion.com/furgonetas-pequenas-comerciales/. [Último acceso: 2 septiembre 2024]. |

https://roboflow.com/formats/yolov8-pytorch-txt