## Documentación

La idea de emplear una red neuronal la tuvimos a partir de los siguientes notebooks alojados en GitHub.

En ellos se explica desde las bases de las redes neuronales hasta a diferentes tipos de capas y topologías de modelos complejos .

https://github.com/AitorIngelmoMartin/Cuarto\_de\_carrera/tree/main/VISION/BLOQUE%203

https://github.com/AitorIngelmoMartin/Cuarto\_de\_carrera/tree/main/VISION/BLOQUE%204

Una vez entendido la potencia y versatilidad de las redes neuronales, nos pusimos a mirar qué posible relación podíamos tender entre esta tecnología y Matlab.

https://es.mathworks.com/help/vision/ref/trainyolov4objectdetector.html

Link en el cual se hace referencia a un modelo estudiado en los notebooks mencionados anteriormente. Siendo este modelo YOLO. Uno entrenado con miles de millones de imágenes.

A raíz de este link, generamos el script "detectar\_foto.m". En el cual probamos superficialmente la capacidad que tiene YOLO en la detección de una única imagen.

Llegados a este punto, y tras haber comprobado el potencial del modelo, nos pusimos a idear potenciales aplicaciones.

De entre todas ellas, decidimos hacer dos aplicaciones simples de entender, y moderadamente asequibles de implementar. Las cuales son un radar de velocidad y un detector de proximidad para verificar la separación de seguridad del covid.

Aplicaciones realizables gracias a que al detectar un objeto, el modelo nos proporciona el centro de la imagen. Por lo que podemos trabajar con conceptos como el campo y la proximidad.

Bibliografía reseñable:

<https://www.mathworks.com/help/vision/ug/getting-started-with-object-detection-using-deep-learning.html> (Modelos preentrenados para detectar objectos)

<https://www.mathworks.com/help/vision/ref/trainyolov4objectdetector.html> (Entrenar YOLOv4 para objetos no entrenados)

<https://www.mathworks.com/help/deepl> (Datasets de Matlab)