UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

Graduado en Ingeniería del Informática

Reconocimiento y seguimiento de objetos mediante redes neuronales para el entrenamiento individual en el baloncesto

Object detection and tracking using neural networks for individual basketball training

Realizado por
Mario Martínez Campuzano

Tutorizado por
Miguel Ángel Molina Cabello
Rafael Marcos Luque Baena

Departamento
Lenguajes y Ciencias de la Computación

MÁLAGA, julio de 2023

UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

GRADUADO EN INGENIERÍA INFORMÁTICA

Reconocimiento y seguimiento de objetos mediante redes neuronales para el entrenamiento individual en el baloncesto

Object detection and tracking using neural networks for individual basketball training

Realizado por
Mario Martínez Campuzano

Tutorizado por
Miguel Ángel Molina Cabello
Rafael Marcos Luque Baena

Departamento
Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA
MÁLAGA, JULIO DE
2023

Resumen

Este documento tiene como objetivo el estudio de un campo en pleno desarrollo y de investigación creciente como es el de reconocimiento de imágenes. Son numerosos los trabajos que se centran en esta rama de la informática para investigar y desarrollar nuevas herramientas que ayuden a la sociedad en problemas más o menos complejos. En el mundo del deporte también ocurre esto y es aquí donde desarrollaremos este trabajo. Cómo puede ayudar la tecnología más avanzada al deportista, de cualquier nivel, para mejorar sus habilidades y tener un impacto positivo en su equipo.

Así, este trabajo se centrará en el apoyo que las nuevas tecnologías pueden darle a los jugadores de baloncesto en el desarrollo de su técnica individual. El bote, el pase y el tiro son los pilares fundamentales del juego y a los que el jugador de baloncesto más tiempo dedica a mejorar. En concreto el bote es uno de los recursos técnicos que no requiere, por norma general, más de un jugador para entrenar. Se busca así que el jugador pueda entrenar y mejorar en cualquier lugar usando un balón y una cámara de vídeo. Con esto el jugador podrá ver la ejecución de distintos ejercicios y recibir una valoración sobre esta.

El trabajo comienza con el estudio de las Redes Neuronales Convolucionales que se encargarán de analizar las imágenes de un jugador de baloncesto, con un balón y frente a una cámara. Los resultados de este proceso de detección de objetos servirán para el posterior proceso de seguimiento de objetos durante un vídeo. Para ello se hará uso de la red neuronal YOLO y del algoritmo de Deep Sort para poder hacer el seguimiento.

Palabras clave: baloncesto, redes neuronales convolucionales, inteligencia artificial, detección de objetos en vídeos, seguimiento de objetos en vídeo.

Abstract

The aim of this document is to study a field that is in full development and growing in research, such as image recognition. There are many works that focus on this branch of computer science to study and develop new tools that help society with more or less complex problems. This is also happening in the world of sport and this is where we will develop this work. How can the most advanced technology help the athlete, at any level, to improve their skills and have a positive impact on their team?

So this work will focus on how new technologies can help basketball players develop their individual technique. Dribbling, passing and shooting are the fundamental pillars of the game and the skills that basketball players spend the most time improving. In particular, dribbling is one of the technical skills that does not generally require more than one player to train. The aim is to enable the player to train and improve anywhere with a ball and a video camera. The player will be able to see how he performs different drills and receive an evaluation.

The work starts with the study of convolutional neural networks. These will be responsible for analysing images of a basketball player and a ball and in front of a camera. The results of this object detection process will be used for the subsequent object tracking process during a video. The YOLO neural network and the Deep Sort algorithm will be used for tracking.

Keywords: basketball, convolutional neuronal networks, artificial intelligence, image recognition, object tracking.

Índice

1. Introducción **7**

1.1. Contexto 7

1.1.1. Nuevas tecnologías en el deporte 7

1.1.2. Técnica individual en el baloncesto 9

1.2. Motivación 9

1.3. Objetivos 10

1.4. Estructura del documento 10

2. Estudio del arte **11**

2.1. Inteligencia Artificial 11

2.2. Redes Neuronales 14

2.2.1. Red Neuronal Biológica 14

2.2.2. Neurona Artificial 15

2.2.3. Red Neuronal Artificial 16

2.2.4. Redes Neuronales Convolucionales 17

2.3. Red Neuronal YOLO 20

2.3.1. Historia de YOLO 22

2.4. Tecnologías utilizadas 24

3. Conjunto de datos **25**

3.1. Acerca del conjunto de datos 25

3.1.1. Conjunto de datos COCO 25

3.1.2. Conjunto de datos Open Images v7 26

3.1.3. Conjunto de datos propio 27

4. Metodología de trabajo **31**

4.1. Metodología ágil e incremental 31

4.2. Método científico 31

4.3. Otros métodos 33

5. Análisis de especificaciones y Diseño del sistema **35**

5.1. Requisitos del programa final 35

6. Llegar al modelo final **37**

6.1. YOLO-v7 37

6.1.1. Primeras pruebas con YOLO-v7 37

6.1.2. Seguimiento con YOLO-v7 40

6.2. YOLO-v8 43

6.2.1. Primeras pruebas con YOLO-v8 43

6.3. Fine-tunning 47

6.3.1. Entrenamiento inicial: conjuntos de imágenes completos 47

6.3.2. Reentrenamiento 1: imágenes del objeto pelota 47

6.3.3. Reentrenamiento 2: pocas imágenes de todas las clases 48

6.3.4. Reentrenamiento 3: aumentar imágenes de todas las clases 48

6.4. Elaboración del conjunto de imágenes propio 49

6.5. Resultados 50

6.5.1. Problemas de compatibilidad con las tarjetas gráficas 51

7. Valoración ejecución de ejercicios **53**

7.1. Algoritmo de seguimiento 53

7.1.1. Detección 53

7.1.2. Estimación 53

7.1.3. Asociación 53

7.1.4. Creación y destrucción de las identidades 54

7.1.5. Parámetros de nuestro modelo 54

7.2. Ejercicios a evaluar 54

7.2.1. Ejercicio 1 contador de botes 55

7.2.2. Ejercicio 2 contador botes con la misma mano 57

7.2.3. Ejercicio 3 contador cambios de mano 60

7.3. Limitaciones 62

8. Conclusiones y Líneas Futuras **63**

8.1. Conclusiones 63

8.2. Líneas Futuras 64

8.2.1. Mejorar la detección de objetos 64

8.2.2. Introducir nuevos ejercicios de técnica individual 65
8.2.3. Reconocer acciones 65
8.2.4. Ampliar el rango de uso 65
8.2.5. Convertirlo en una red social 66

Apéndice A. Manual de Instalación **67**
A.1. Preparación del entorno 67

Apéndice B. Manual de Ejecución **71**
B.1. Ejecución 71

Referencias **73**

1. Introducción

1.1. Contexto

1.1.1. Nuevas tecnologías en el deporte

El deporte es uno de los pilares fundamentales en el ocio de la sociedad actual: entretiene, motiva, inspira, etc. Además de todo esto aporta valores como el compañerismo y el esfuerzo que ayudan a las personas en su día a día. Son muchas las emotivas historias de aficionados y jugadores que han sido inspirados y motivados para superar situaciones complicadas de su vida apoyados en el deporte. Los jugadores y jugadoras son los actores principales y dedican la mayor parte de su vida a entrenar para mejorar sus habilidades individuales. Las nuevas tecnologías están ayudando, a través de la Inteligencia Artificial, a mejorar el rendimiento de los equipos con estadísticas individuales y colectivas durante los partidos. Sin embargo, no son tantas las herramientas que ayuden al jugador en su desarrollo individual.

En la mejor liga de baloncesto del mundo, la NBA, las nuevas tecnologías se dedican al procesamiento de datos durante los partidos para aportar la mayor cantidad de información posible a los entrenadores. Históricamente se han tomado estadísticas individuales, en la temporada 1973-74 se contabilizaron por primera vez los rebotes, robos y tapones, en la 1977-78 las pérdidas y en la 1979-80 debuta la línea de tres puntos en la NBA. Sin embargo, desde hace poco tiempo y debido al avance en potencia y portabilidad de los dispositivos móviles y ordenadores, estas estadísticas se analizan en profundidad y hoy día ha tomado mucho peso lo que se denomina la estadística avanzada [1]. La idea es, a través de las estadísticas individuales, calcular (ver Figura 1) otros valores que indiquen con mayor precisión el impacto de un jugador en el juego. Entre las estadísticas avanzadas encontramos las *deflections* (balones desviados por la defensa), eficiencia ofensiva y defensiva, efectividad de un jugador, tiros efectivos y verdaderos, etc. A continuación, en la Figura 1 se muestra una fórmula mediante la cuál se calcula el ratio de eficiencia de un jugador o *player efficiency rating*, es decir un valor de la estadística avanzada que usa las estadísticas tradicionales para sumar aquellas que son positivas, restar las negativas y ponderarlo en función del tiempo jugado.

```
uPER = (1 / MP) *
     [ 3P
     + (2/3) * AST
     + (2 - factor * (team_AST / team_FG)) * FG
     + (FT *0.5 * (1 + (1 - (team_AST / team_FG)) + (2/3) * (team_AST / team_FG)))
     - VOP * TOV
     - VOP * DRB% * (FGA - FG)
     - VOP * 0.44 * (0.44 + (0.56 * DRB%)) * (FTA - FT)
     + VOP * (1 - DRB%) * (TRB - ORB)
     + VOP * DRB% * ORB
     + VOP * STL
     + VOP * DRB% * BLK
     - PF * ((lg_FT / lg_PF) - 0.44 * (lg_FTA / lg_PF) * VOP) ]
```

Figura 1: Fórmula PER [2]

Además de generar datos durante los partidos para su posterior análisis, comienzan a ser muchas las tecnologías que se están desarrollando, aprovechando los avances en el análisis de imágenes, para obtener información de los partidos automáticamente. Como ejemplo de esto, La Liga de Fútbol profesional, en alianza con Microsoft, presentó en 2021 la herramienta *Beyond Stats*. A través de más de 19 cámaras diferentes que se han instalado en los estadios de los equipos de la liga se recogen las posiciones de árbitros, jugadores y balón. Se captan hasta 25 imágenes por segundo para generar 3.500.000 de datos por partido [3]. Posteriormente, a través de *MediaCoach* se procesa la información mediante dos flujos [4]:

- *Tracking*, información sobre las posiciones de los jugadores, balón y árbitros con las que se pueden elaborar métricas como número de sprints, velocidades máximas o distancias recorridas.
- *Eventing*, información sobre las acciones técnico-tácticas como remates, centros al área y cambios de orientación, sustituciones, faltas, fueras de juego, etc.

Además de usar las nuevas tecnologías para apoyar a entrenadores y jugadores en la toma de decisiones, la Inteligencia Artificial también está revolucionando la forma que tenemos de consumir el deporte. Así, la NBA presentó a través de su aplicación una herramienta para escanear tu cuerpo en 360 grados e intercambiarlo por el jugador que tú quieras dentro de un partido. Así, tu cuerpo escaneado será el que aparezca realizando las jugadas más espectaculares del partido [5].

1.1.2. Técnica individual en el baloncesto

La técnica individual es el pilar fundamental de cualquier deporte, son las herramientas que después un jugador puede usar durante el partido. Si durante el encuentro un jugador lee una situación, como dar un pase a un compañero que está en mejor posición que él, pero no tiene una buena técnica individual, el pase no llegará a su destino o llegará tarde. Es por esto que un jugador sin técnica es como un buen pintor sin saber usar los pinceles y pinturas.

El bote, el pase y el tiro son las tres ramas fundamentales de la técnica individual. El bote permite al jugador avanzar y mejorar la posición que tiene en un momento específico, el pase le permite compartir con los compañeros el balón y el tiro es lo que les permite completar un movimiento exitosamente. Desde los inicios, la técnica individual ha tenido siempre un entrenamiento basado en la repetición de las acciones. Sin embargo, en los últimos años ha tomado peso un concepto como es el de transferencia, es decir, no solo enseñar cómo hacer una acción técnica sino cuándo realizarla. Así los entrenamientos están introduciendo durante gran parte del tiempo un elemento variable que obligue al jugador a ejecutar correctamente una acción técnica teniendo que elegir correctamente cuál de ellas realizar.

1.2. Motivación

Los numerosos avances que se están presentando gracias a la Inteligencia Artificial están abriendo nuevas ramas de investigación que buscan mejorar la vida de las personas. Cada vez estamos más interconectados y poseemos dispositivos con altas capacidades de cómputo que nos permiten desarrollar tareas que hace pocos años parecían inimaginables. Por esto y dada la pasión del autor de este documento por el deporte, y por el baloncesto en particular, nace la posibilidad de unir estos dos mundos para apoyar a los jóvenes que estén iniciándose o aquellos que quieran seguir mejorando sus habilidades.

No se pretende sustituir al entrenador de baloncesto sino aportar una herramienta más que motive al jugador a seguir entrenando y haga más amenas las horas dedicas a repetir ciertos gestos técnicos.

Las técnicas de reconocimiento de objetos en imágenes y el seguimiento de objetos en vídeos nos permiten extraer información muy valiosa y poder interactuar desde un contexto real con el usuario.

1.3. Objetivos

Este trabajo tiene por objeto realizar un programa capaz de seguir a una persona y un balón de baloncesto durante un ejercicio que realice frente a una cámara de vídeo para poder otorgar una retroalimentación o *feedback* de si ha realizado bien la tarea o no.

Para ello se utilizarán modelos de Inteligencia Artificial entrenados para tal fin. Nos basaremos en modelos de aprendizaje profundo a través del uso de Redes Neuronales Convolucionales, caracterizados por resolver problemas de clasificación de imágenes. Así la idea principal es encontrar el modelo que mejor se adapte a la detección del balón y la persona, permitiéndonos seguir la posición de estos objetos y obtener una valoración del movimiento ejecutado.

1.4. Estructura del documento

Este documento sigue una estructura lineal con varias partes diferenciadas:

- En primer lugar, en el capítulo actual, se pretende contextualizar el documento, explicar las motivaciones que han llevado a su realización y lo que se pretende con este.
- A continuación, se pasará a intentar explicar de manera clara y directa los fundamentos y conceptos teóricos en los que se apoya el trabajo. La Inteligencia Artificial, las Redes Neuronales y YOLO son algunos de los temas tratados.
- Una vez realizada la fundamentación teórica se pasará a explicar los distintos conjuntos de datos que se han usado durante el desarrollo de todas las pruebas de este trabajo.
- Posteriormente, se expone la metodología de trabajo seguida durante el proyecto.
- Además, en el siguiente punto se exponen las especificaciones y el diseño del sistema.
- Tras todo esto, se organizarán todas las distintas pruebas que se han ido realizando para la obtención del modelo que nos ayudará al objetivo principal de este trabajo.
- Así llegaremos a la parte fundamental del trabajo, explicar el resultado final de este dónde se exponen qué ejercicios se valorarán y cómo.
- Por último, se incluyen unas conclusiones sobre el trabajo y cómo puede mejorar con unas posibles futuras líneas de actuación.
- Indicar que al final del documento se encuentra la bibliografía utilizada.

2. Estudio del arte

2.1. Inteligencia Artificial

Cuando hablamos de Inteligencia Artificial (o IA) enseguida pensamos en algo relativamente nuevo, creado en estos últimos años y que está revolucionando la manera que tenemos de tratar con las máquinas. Sin embargo, aunque ahora la IA acapare la mayoría de las noticias tecnológicas, ya en 1950 Alan Turing, aunque no tuvo todo el impacto por entonces, publicó un artículo icónico en el que expresaba en las primeras líneas: "¿Pueden pensar las máquinas?" [6] Así expuso el denominado "Juego de la imitación" que explicó de la siguiente manera: se necesitan tres personas: un hombre, una mujer y un interrogador. El interrogador, que se encuentra en una habitación aparte, debe determinar quién es el hombre y quién es la mujer. Para ello, este irá identificando a través de preguntas, que serán respondidas por escrito, un perfil A y un perfil B. Cuándo acabe con las preguntas deberá asociar a cada perfil el sexo que él crea oportuno. Es decir, el interrogador preguntará: "A, ¿Puede decirme de que color tiene el pelo?", Así A, que siempre durante el juego será la mujer o el hombre, debe responder por escrito el color de su pelo para que el interrogador vaya creando el perfil.

Con este juego quería luego hacernos pensar sobre qué ocurriría si sustituimos a la mujer o al hombre por una máquina. Podría entonces el interrogador discriminar entre el humano y la máquina, así vuelve a cuestionar la pregunta inicial "¿Pueden pensar las máquinas?" En la Figura 2 se muestra un ejemplo del Test de Turing donde se sustituye a un jugador por una máquina.

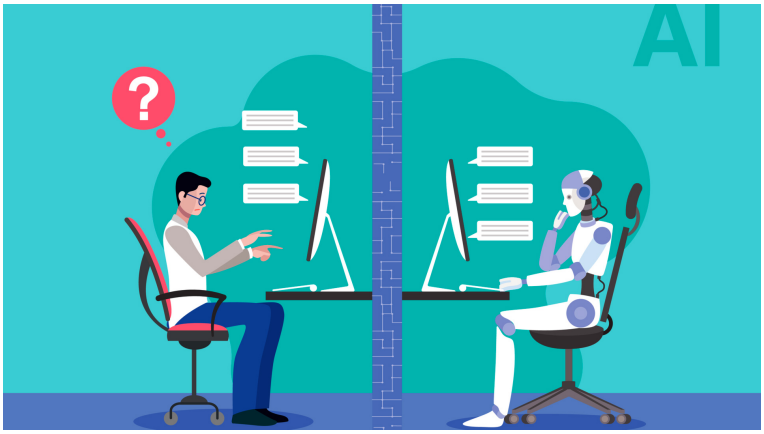

Figura 2: Ejemplificación del Test de Turing [7]

Las aportaciones a esta rama de la informática vienen desde mucho atrás con los aportes

de George Boole y Gottlob Frege con la Lógica Booleana y la Lógica de Primer Orden respectivamente, en 1847 y 1879. Warren McCulloch y Walter Pitts, en 1943, presentan un modelo de neuronas artificiales cuando ni siquiera existía el término de Inteligencia Artificial.

Desde entonces hemos visto avances de todo tipo [8]:

- La creación en 1990 de los Agentes Inteligentes, capaces de percibir el entorno y procesar la información para actuar de manera ordenada buscando maximizar un resultado.
- En la década de los 2000, los Chat Bots comienzan a ganar premios por ser muy parecidos a los humanos.
- En 2011 un ordenador de IBM ganó el concurso de preguntas y respuestas Jeopardy! [9].
- En 2014 un ordenador ganó el test de Turing haciendo creer al interrogador que era un humano [10].
- En 2016 un ordenador ganó al campeón mundial del juego Go [11].

Como hemos visto la IA pretende dar una respuesta afirmativa a la que planteó Alan Turing en su artículo, su objetivo es simular la inteligencia humana en las máquinas. Sin embargo, aunque el objetivo parece claro no hay una definición exacta y común sobre la IA, así en numerosos artículos de revistas encontramos varias de ellas [12]:

- Stuart Russell y Peter Norvig en su libro «Artificial Intelligence: A Modern Approach» la definen como: "la IA es el estudio de los agentes que reciben percepciones del entorno y realizan acciones".
- Patrick Winston define la IA como "algoritmos activados por restricciones, expuestos por representaciones que soportan modelos que vinculan el pensamiento, la percepción y la acción".
- Jeremy Achin en 2017 dio su propia definición: "La inteligencia artificial es un sistema informático capaz de realizar tareas que normalmente requieren inteligencia humana… muchos de estos sistemas de IA se basan en el Machine Learning, otros en el Deep Learning y otros en cosas muy aburridas como las reglas".

Al inicio de esta rama, apoyados en la lógica computacional, la Inteligencia Artificial seguía reglas específicas y concretas que desencadenaban una acción particular. Sin embargo, los

entornos, que son los encargados de alimentar las reglas, no son estáticos ni mucho menos previsibles por lo que este enfoque enseguida se quedó obsoleto, aunque sigue siendo objeto de un gran campo de investigación.

Así en la Inteligencia Artificial surge un nuevo enfoque como es el enfoque del Aprendizaje Automático o Machine Learning [12]. No es algo distinto sino que es una categoría dentro de la IA. Esto consiste en que ahora las reglas que se seguirán no serán dadas, sino que la máquina será capaz de encontrar patrones recurrentes en un conjunto de datos [13]. Para ello es necesario un conjunto de datos de entrenamiento que será procesado por un algoritmo para aprender de él y obtener estos patrones.

Por consiguiente, necesitamos un conjunto de entrenamiento con datos etiquetados para que el algoritmo sepa que características debe identificar. Hay una amplia variedad de algoritmos de Machine Learning aunque podemos exponer dos de los más importantes:

- Algoritmos de regresión, capaces de encontrar la relación entre los datos, por ejemplo, para calcular un valor real, en la mayoría de los casos, en función de otros. Hay distintos tipos: regresión lineal, logística y máquina de vectores de soporte según el tipo y complejidad del valor a predecir.

- Algoritmos de asociación, capaces de descubrir patrones entre los datos o lo que llamamos las reglas de asociación, si X entonces Y.

- Redes Neuronales, una red con varias capas donde entran unos datos, se calculan en una capa intermedia ciertos valores y se extraen unas probabilidades asociadas a cada posible salida.

Además de estos algoritmos encontramos varios tipos de aprendizajes en función de cómo se tratan los datos:

- Aprendizaje supervisado, los datos vienen etiquetados, es decir el conjunto viene dado con los datos de entrada y salida.

- Aprendizaje no supervisado, en este los datos no están etiquetados sino que solo tenemos los datos de entrada. Es usado con grandes cantidades de datos para buscar patrones entre ellos.

- Aprendizaje por refuerzo, en este caso se deja actuar al algoritmo para que vaya aprendiendo, este irá probando con distintos enfoques y valorando la solución para ir aprendiendo de los errores.

Destacar que por ejemplo el logro que mencionamos anteriormente donde una máquina ganó al mejor jugador del mundo del juego Go, se consiguió a través del aprendizaje por refuerzo. Además, las técnicas de Machine Learning las encontramos en la mayoría de los algoritmos de recomendaciones de contenido en plataformas como Youtube, Instagram, Twitter; en los motores de búsqueda como Google; los asistentes personales como Siri, Google Assistant y Amazon Alexa que usan el Machine Learning para el tratamiento de lenguaje natural o en los correos electrónicos para detectar los correos SPAM.

En 1986 Geoffrey Hinton introduce una nueva rama en el Machine Learning denominada Deep Learning [13] o Aprendizaje Profundo. Su nombre lo debe al uso de las Redes Neuronales Profundas donde se usan una gran cantidad de capas intermedias para tratar los datos. La idea original es la de imitar el comportamiento del cerebro humano. Aunque con una sola capa intermedia se pueden llegar a crear buenas predicciones añadir capas intermedias, en su justa medida, ayuda a ganar precisión. [14]

2.2. Redes Neuronales

A continuación, dedicaremos parte del trabajo a entender el mundo de las Redes Neuronales, donde más se apoya este trabajo para su desarrollo.

La Inteligencia Artificial y las Redes Neuronales, como hemos visto, se inventaron hace décadas, pero la falta de capacidad de cálculo de los ordenadores de la época no hicieron posible ir implementando todos los avances.

2.2.1. Red Neuronal Biológica

Las neuronas son las células que componen el sistema nervioso. Estas se encargan de recibir, procesar y transmitir información a modo de señales químicas y eléctricas. Como se puede ver en la Figura 3 para cumplir su función las neuronas tienen una forma particular. Su estructura tiene tres partes marcadas:

- Dendritas, donde se reciben los impulsos eléctricos de las otras neuronas.
- Soma, en él se procesa la información recibida por las dendritas.

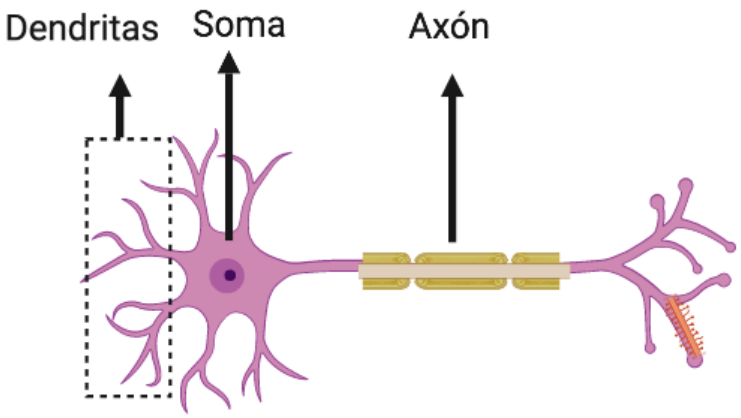

Figura 3: Estructura de una neurona [15]

- Axón, es la parte de la neurona que conecta con las otras y por donde se transmite la información procesada por el Soma.

Las Redes Neuronales Biológicas o Circuito Neuronal son un conjunto de conexiones sinápticas que se producen como resultado de la unión de unas neuronas con otras. En el cerebro encontramos aproximadamente cien mil millones de neuronas que forman una red como la que se puede ver en la Figura 4.

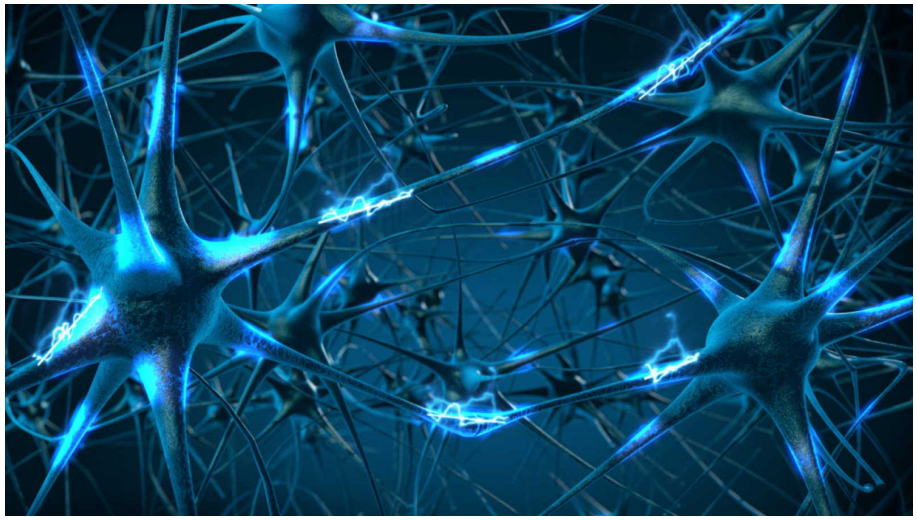

Figura 4: Red Neuronal Biológica en 3D [16]

2.2.2. Neurona Artificial

La Neurona Artificial, como hemos comentado anteriormente, basada en la Neurona biológica, intenta imitar su estructura. Esta neurona, también llamada perceptrón, tiene varias partes diferenciadas como se puede ver en la Figura 5 [17]:

- Las entradas, por donde se recibe la información o bien del conjunto de datos o bien de otras neuronas.
- Los pesos sinápticos, son los encargados de ponderar cada entrada otorgándole a estas la importancia justa.
- Función suma, encargada de sumar las entradas multiplicadas por el peso sináptico de cada una de ellas.
- Función de activación, toma un único valor, el de la función suma, y realiza una operación matemática sobre él para devolver un valor que será tomado como salida.
- Salida, valor que devuelve la neurona tras procesar los datos.

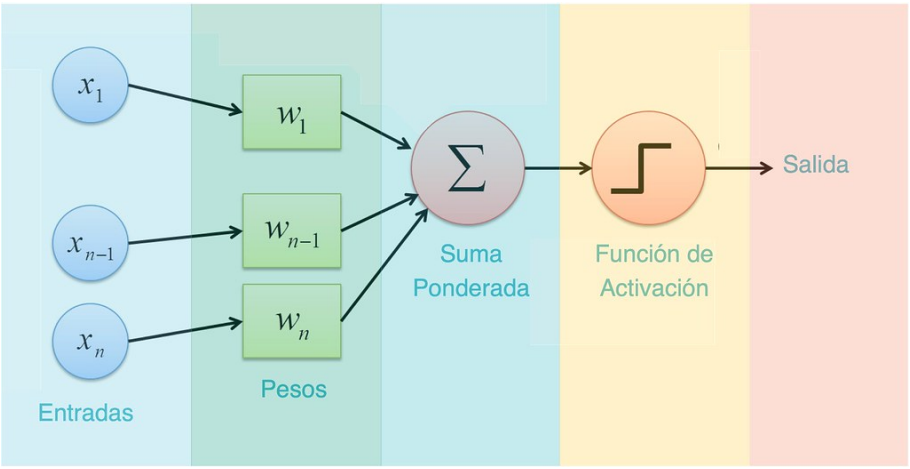

Figura 5: Partes de un perceptrón [17]

Para seguir con la comparación con las neuronas biológicas vemos como las entradas podrían ser las Dendritas, los pesos, la función suma y de activación compondrían el Soma y la salida sería el Axón.

2.2.3. Red Neuronal Artificial

La Red Neuronal Artificia sigue el mismo comportamiento que las biológicas y se basan en interconectar las neuronas a través de sus entradas y las salidas para generar un resultado final.

Las Redes Neuronales tienen tres niveles o capas, ver Figura 6. La primera capa es la capa de entrada o *input layer*, las capas intermedias o *hidden layers* son las capas ocultas y la última

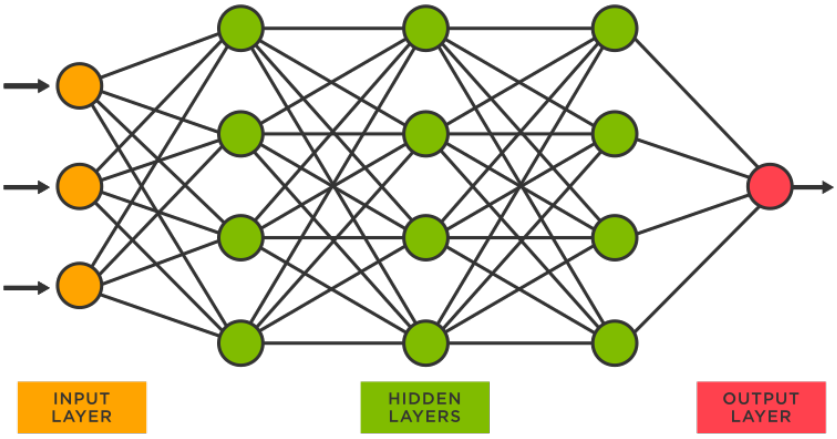

Figura 6: Red Neuronal con varias capas intermedias [18]

capa es la capa de salida o *output layer*. La primera capa recibe como entrada los datos del conjunto y pasa su salida a la primera capa oculta. Dentro de las capas ocultas, pueden ocurrir dos cosas: que solo haya una capa y tras recibir los datos de la capa de entrada pase a la capa de salida, o que haya más de una capa oculta y entonces las salidas de una capa son las entradas de la siguiente hasta llegar así a la capa de salida.

Las redes neuronales pueden recibir distintos tipos de datos en las entradas: píxeles de una imagen, datos económicos, médicos, etc. La importancia de tener un conjunto de datos estructurado es fundamental para obtener un buen comportamiento de la red.

Una vez tenemos el conjunto de entrada la red procede a calcular los pesos de las diferentes capas para poder dar la respuesta deseada al conjunto de datos que está etiquetado previamente, a esto es lo que llamamos entrenamiento. La red recibe unos datos de entrada y calcula un valor de salida, si coincide con el esperado los pesos reciben un *feedback* positivo y cambian poco, si por el contrario no coinciden con el valor esperado entonces recibe un *feedback* negativo y los pesos cambian sus valores.

2.2.4. Redes Neuronales Convolucionales

Uno de los problemas más tratados por la Inteligencia Artificial es sobre la clasificación de imágenes y el reconocimiento de objetos en las mismas. Al comienzo se usaban las Redes Neuronales tradicionales para tratar el problema. Sin embargo, cuando aumentaron el tamaño de las imágenes se volvió un problema inabordable por estas Redes Neuronales.

Así surgió una nueva Red Neuronal enfocada a este tipo de problema, las Redes Neuronales Convolucionales (CNN o ConvNets) [19]. Estas asumen ciertas características de las entradas

para poder reducir el número de variables y hacer el problema tratable computacionalmente. En el año 2012 una CNN consiguió ganar el concurso de reconocimiento de imágenes de ImageNet [20].

Estas Redes Neuronales se componen principalmente de tres capas: la capa convolucional, la capa de agrupación o pooling y la capa totalmente conectada. La idea es ir uniendo estas capas para llegar a clasificar en la imagen cierto objeto.

La primera capa y la de mayor importancia en la red es la capa convolucional, que además es la encargada de dar nombre a la red. Está capa tiene como misión extraer características de la imagen al mismo tiempo que reducir su dimensión. Su funcionamiento es través de unos filtros que recorren la imagen de izquierda a derecha y de arriba a abajo para obtener una salida en cada paso que forme un mapa de activación. Imaginemos que tenemos una imagen de 32x32 píxeles y que la representamos con una matriz de 32x32x3 donde la profundidad es 3 para representar el color del píxel en RGB. Un posible filtro podría ser uno de 5x5x3 (la profundidad siempre debe coincidir). Él es el encargado de recorrer la imagen original y para cada paso hacer el producto escalar entre los valores del filtro (o kernel) y los píxeles de la imagen, el resultado serán los valores que formen el mapa de activación. En la siguiente Figura se muestra el procedimiento de la aplicación de un filtro.

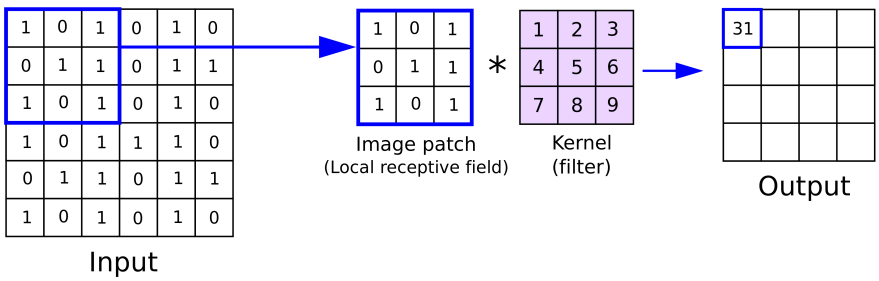

Figura 7: Aplicación filtro en CNN [21]

En la figura 8 vemos como se aplica un filtro de 5x5x3 sobre una imagen de 32x32x3, de esta manera se genera un mapa de activación (Paso 1), también se recorre de arriba a abajo generando el segundo mapa de activación (Paso 2). Es habitual para obtener diferentes características de la imagen aplicar varias capas convolucionales a la imagen (Paso 3). Después de cada capa de convolución se aplica una transformación de unidad lineal rectificada (ReLU). Cuando aplicamos más de una capa de convolución la estructura se vuelve jerárquica ya que cada una ve los píxeles más característicos de la anterior (Paso 4), ver figura 9 [20].

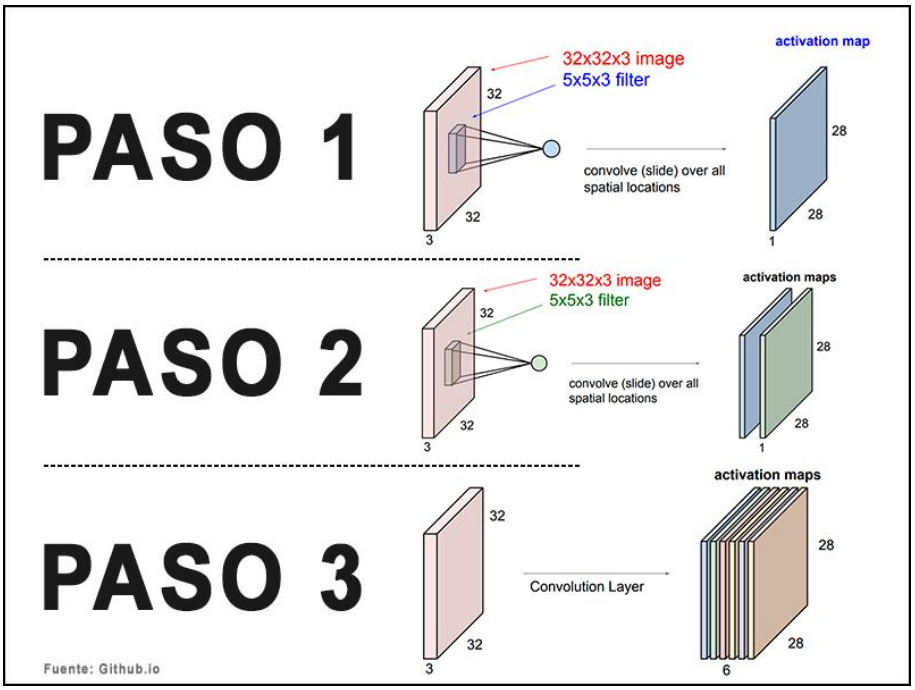

Figura 8: Pasos capa convolucional CNN [20]

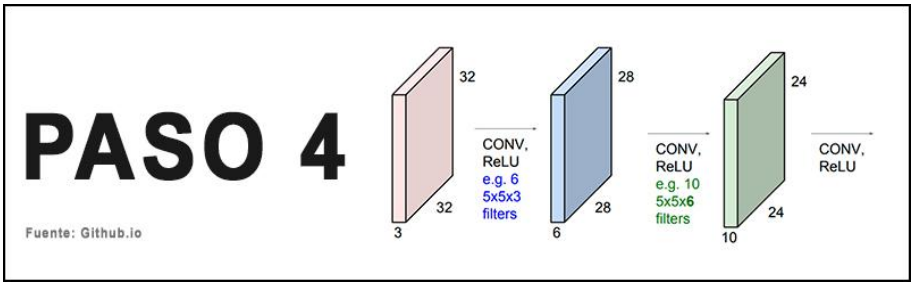

Figura 9: Capa de agrupamiento en CNN [20]

La siguiente capa es la capa de pooling o de agrupación. Es una capa en la que se pierde algo de información de la imagen, por ende, el problema se vuelve mucho más eficiente computacionalmente y aumenta significativamente el rendimiento. Esta capa también aplica un filtro a la imagen, pero a diferencia del visto anteriormente este se basa en dos tipos de agrupamiento:

- Agrupación máxima, seleccionar el píxel más alto de los que están en ese momento en el filtro.
- Agrupación media, calcular el valor medio de los que en ese momento están en el filtro.

La última capa es la capa totalmente conectada o fully connected. Esta capa tiene como objetivo leer las características extraídas por las capas anteriores y obtener resultados. Esto se

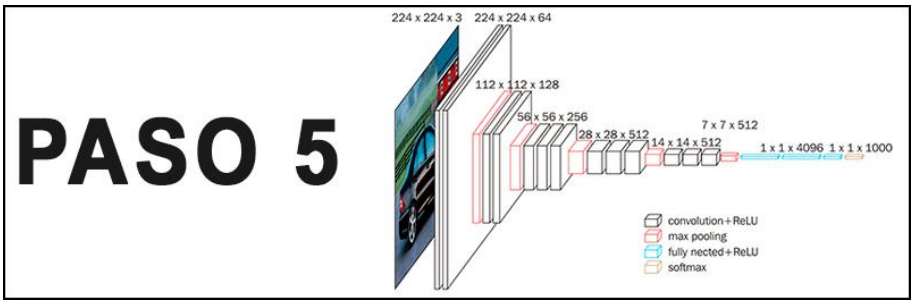

Figura 10: Ejemplo de una red CNN y sus capas [20]

debe a que realmente necesitamos una neurona para cada detección que queramos hacer y se active por ejemplo la neurona para la detección del coche o de la bicicleta, etc. La función de activación softmax es la utilizada en última estancia para activar la neurona.

En la figura 10 vemos un ejemplo completo de una red convolucional y como se compone con las diferentes capas.

2.3. Red Neuronal YOLO

Como hemos dicho anteriormente la detección de objetos es una de las tareas principales en la que se está poniendo el foco con la Inteligencia Artificial. Esta comprende la identificación y localización de objetos en imágenes y vídeos. Hasta ahora hemos visto como se realiza la clasificación y a continuación veremos la detección.

Uno de los primeros enfoques que encontramos es unir un enfoque por regiones y a su vez las Redes Neuronales Convolucionales [22], esto lo podemos ver en la Figura 11. Se trata de dividir la imagen en 2000 regiones y procesar cada una de ellas con una CNN.

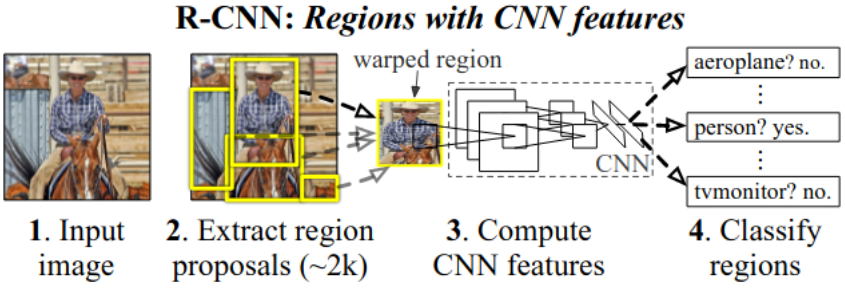

Figura 11: Enfoque por regiones en una red CNN [22]

Podemos diferenciar dos categorías en los algoritmos de detección de objetos según cuántas veces se procese la imagen: single-shot detectors y two-stage detectors.

- **Single-shot detectors**, procesan la imagen una sola vez y obtienen la clasificación. Bastante eficientes, por ello son usados en detecciones en tiempo real.
- **Two-stage detectors**, se procesa la imagen dos veces: la primera sirve para generar posibles detecciones y la segunda confirma las predicciones anteriores para dar un resultado más preciso.

La selección de uno u otro vendrá dada por las características y requisitos de la aplicación. Como este trabajo se sostiene sobre YOLO, a continuación nos centraremos en él.

YOLO está dentro de las categorías de single-shot detectors dentro de los algoritmos de detección de imágenes. Es decir, YOLO solo procesa la imagen una vez buscando ser más eficiente en la búsqueda de objetos. YOLO usa una Red Neuronal Convolucional Profunda que sigue una arquitectura como la siguiente:

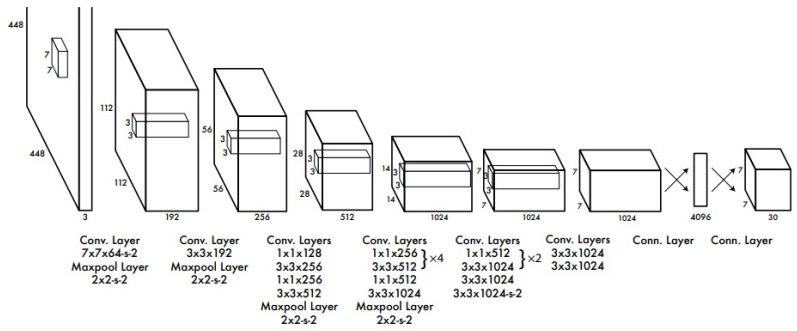

Figure 3: The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1×1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224×224 input image) and then double the resolution for detection.

Figura 12: Ejemplo de una red CNN y sus capas [23]

En la Figura 12 se ve la composición de esta red con 24 capas convolucionales y 2 capas totalmente conectadas. Las primeras 20 capas fueron entrenadas con el dataset de 1000 clases de ImageNet. Luego se combinaron con varias capas convolucionales y totalmente conectadas para mejorar el rendimiento.

Cuando YOLO intenta detectar los objetos genera unas cajas delimitadoras o *bounding box*, esto es un rectángulo que limita el espacio en el que se cree está el objeto detectado. Como vemos en la Figura 13, YOLO divide la imagen en una cuadrícula de SxS, así si el centro de un objeto cae sobre una casilla de esa cuadrícula, esa casilla se convierte en la responsable de detectar el objeto. Con esto, esta casilla genera N *bounding box* y le otorga una puntuación o confianza, es decir cuanto cree él que efectivamente ese *bounding box* es el objeto en cuestión.

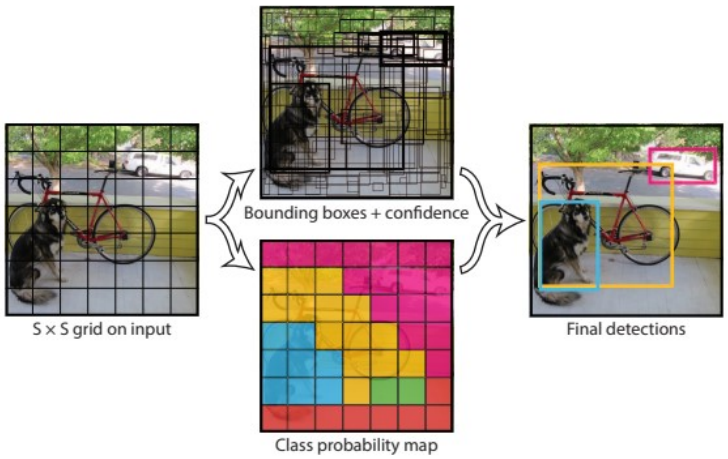

Figure 2: The Model. Our system models detection as a regression problem. It divides the image into an $S \times S$ grid and for each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities. These predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor.

Figura 13: Proceso sobre una imagen YOLO [23]

Es posible que un solo objeto tenga más de un *bounding box* posible, YOLO se quedará con aquel que mayor valor de confianza tenga.

Tras todo esto es posible que en la imagen se hayan otorgado varios *bounding box* para un mismo objeto y que varíen en tamaño, posición, etc. Para esto YOLO implementa NMS o *non-maximun suppression*, un algoritmo encargado de eliminar los *bounding box* redundantes o que no aporten más información al resultado final.

2.3.1. Historia de YOLO

Aunque en el título del apartado hablamos de historia, realmente YOLO tiene una vida relativamente corta y su primera aparición se data en 2015 cuando se publicó YOLO, hasta 2023 con la aparición de la última versión YOLOv8, como se puede ver en la Figura 14.

Tras la aparición de YOLO un año más tarde aparece YOLO-v2 o YOLO-9000. Esta nueva versión introdujo varias modificaciones que buscaban mejorar el rendimiento de la red:

- El esqueleto de la Red Neuronal Convolucional cambia a Darknet19.
- Se introduce el concepto de *anchor boxes* por el que se busca predefinir unos *bounding box* para luego unificarlo con los detectados para obtener los definitivos.

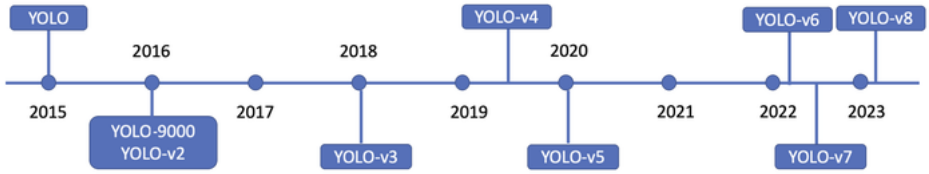

Figure 1. Timeline of You Only Look Once (YOLO) variants.

Figura 14: Timeline YOLO [24]

En el año 2018 nace YOLO-v3 que mejoraba en las siguientes características:

- Se cambia a Darknet59 con 59 capas convolucionales, es una variante ResNet diseñada específicamente para tareas de detección de objetos.
- Se mejora el concepto *anchor boxes* ya que ahora el tamaño de estas es variable.

En 2019 es presentada la siguiente versión YOLO-v4, comentar que está versión no es considerada como una versión oficial de YOLO. Algunas de sus mejoras son:

- Introduce la arquitectura de red CSPNet *Cross Stage Partial Network*, una variante de ResNet.

En 2020 se publica, por parte de Ultralytics, YOLO-v5. Este introdujo una serie de mejoras:

- Se cambia de red a EfficientNet, una estructura más compleja que las anteriores.
- Se cambia el método para generar las *bounding boxes* por *dynamic anchor boxes*.

En 2022 surge YOLO-v6 donde de nuevo se cambia de red por EfficienteRed-L2 y de método para los *anchor boxes* a *dense anchor boxes*.

A los pocos meses de la versión anterior nace YOLO-v7. La principal novedad de esta versión es el uso de diferentes formas y tamaños para detectar objetos diferentes, en concreto son 9 los que se usan.

En 2023 se presenta YOLO-v8, también soportado por Ultralytics, este abandona los *anchor boxes* por lo que se intenta predecir directamente el centro de los objetos. Además, esta versión es más amigable con el programador y tiene una estructura sobre Python más ordenada y limpia. En la Figura 15 se ve la comparación de YOLOv-8 con sus predecesores.

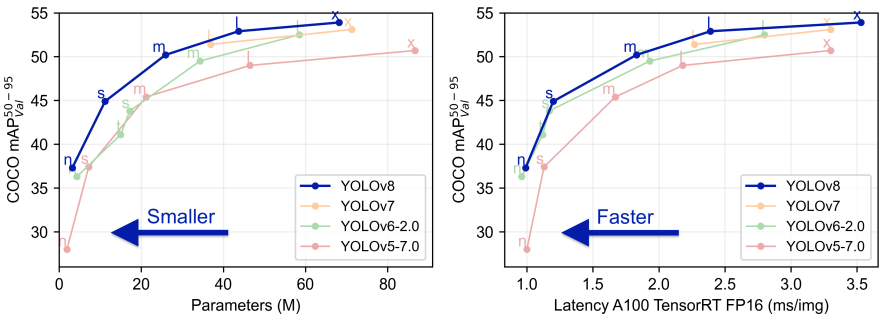

Figura 15: YOLO-v8 en comparación con sus predecesores [25]

2.4. Tecnologías utilizadas

En este trabajo se han utilizado las siguientes tecnologías:

- [26] Python es un lenguaje de programación de alto nivel, interpretado y multiparadigma. Su filosofía se centra en la legibilidad del código y en su sencillez. Es uno de los lenguajes de programación más populares por sus amplios casos de uso desde desarrollo de aplicaciones hasta trabajos de Big Data.

- [27] Conda es un gestor de paquetes y un sistema gestor de entornos. Es ampliamente usado para poder instalar en una misma máquina distintos entornos que contengan distintas versiones de una misma librería sin que se molesten entre ellas.

- [28] Roboflow es un software con herramientas para el manejo de los conjuntos de datos para la las tareas de detección y clasificación de imágenes. Permite crear conjuntos de imágenes y etiquetarlas, a tráves de su herramienta online, incluso puede entrenar un conjunto de imágenes después de haberlo etiquetado. Además, almacena conjuntos de imágenes públicos que puedes usar en otros proyectos. Permite la colaboración entre usuarios para equipos que trabajen en un mismo proyecto.

3. Conjunto de datos

3.1. Acerca del conjunto de datos

Para la elaboración de las diferentes pruebas de este trabajo hemos necesitado usar una serie de imágenes seleccionadas de distintos conjuntos de datos públicos y propios. Dentro de los conjuntos de imágenes, las clases que para este trabajo son relevantes son la clase persona (person) y la clase pelota (sports ball). A continuación, se describen los conjuntos de datos utilizados.

3.1.1. Conjunto de datos COCO

COCO [29] es un conjunto de datos soportado por la Common Visual Data Foundation o CVDF, Facebook y Microsoft entre otros, [30] que cuenta con más de 330.000 imágenes y 80 categorías de objetos. Formado por un conjunto de personas con trayectorias profesionales en Google, Nvidia y Facebook entre otras empresas e instituciones.

Algunos ejemplos de imágenes que encontramos son los siguientes:

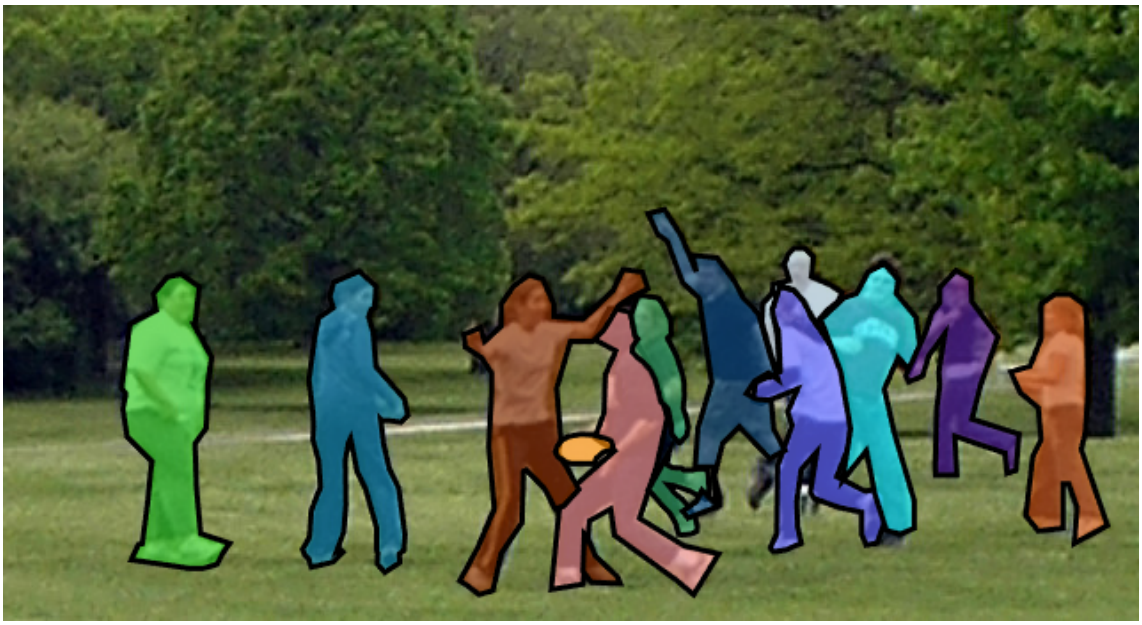

Figura 16: Imagen etiquetada de COCO con personas [29]

En estas imágenes, Figura 16 y 17, las personas y pelotas están segmentadas, es decir en vez de detectarlas y marcarlas con un*bounding box* la zona en la que está el objeto se realiza una segmentación para que la detección no sea cuadricular sino que tenga la forma del objeto

Figura 17: Imagen etiquetada de COCO con una persona y una pelota [29]

en cuestión.

3.1.2. Conjunto de datos Open Images v7

Open Images es un conjunto de datos público que cuenta con más de 9 millones de imágenes. Es el conjunto de imágenes más grande que podemos encontrar. Tiene más de 16 millones de *bounding box* en 1.9 millones de imágenes. Su idea es proporcionar un único conjunto de imágenes donde encontrar las anotaciones para clasificación de imágenes, detección de objetos, detección visual de relaciones, segmentación, etc. que permita estudiar todas estas tareas juntas para fomentar su desarrollo.

Algunos ejemplos que podemos encontrar son las siguientes, Figura 18 y 19:

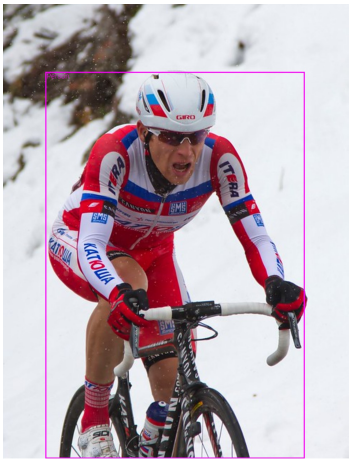

Figura 18: Imagen etiquetada de Open Images con personas [31]

Figura 19: Imagen etiquetada de Open Images con una pelota [31]

3.1.3. Conjunto de datos propio

Durante el desarrollo del trabajo hemos necesitado elaborar dos conjuntos de datos con imágenes propias para poder elaborar las distintas pruebas que nos han hecho llegar al resultado final.

Para realizar estos conjuntos de datos se grabaron varios vídeos donde aparece una persona botando un balón de baloncesto en frente de una cámara de vídeo. Así se elaboró un ejecutable de python capaz de obtener los fotogramas de los vídeos.

Para elaborar los conjuntos de imágenes se utilizó la herramienta Roboflow [28]. Esta herramienta es online y permite subir un conjunto de imágenes, etiquetarlas y exportarlas entre multitudes de formatos.

En la figura 20 se muestra la herramienta de Roboflow para subir las imágenes que queramos añadir al conjunto de datos posteriormente.

Una vez añadidas las imágenes podemos empezar a anotar en las imágenes los distintos objetos. Para ello Roboflow tiene una herramienta donde podemos crear el *bounding box* sobre el objeto en cuestión y etiquetarlo con la clase pertinente. Además de poder crear los *bounding box*, podemos usar otras herramientas como un asistente para anotaciones, herramienta para auto-detectar clases, etc. ver figura 21

En nuestro caso, para el primer conjunto de imágenes, que llamaremos Dataset 1, era para probar la herramienta Roboflow y se procedió a etiquetar solo los objetos pelota o *sports ball*. Se usaron 3 vídeos diferentes de los que se obtuvieron 490 imágenes. Tras esto al exportar el conjunto de imágenes para poder entrenarlo con YOLO se decidió aplicar un proceso de

augmentation mediante el que se aplican unos efectos a las imágenes ya etiquetadas para poder aumentar el número de imágenes del conjunto.

En el segundo conjunto de imágenes, que llamaremos Dataset 2, se introdujo además de la clase pelota la clase persona o *person*. Se repitió el proceso anterior aunque para este entrenamiento se usaron más vídeos para partir de 808 imágenes.

Dataset	**Imágenes totales**	**Clases etiquetadas**
Dataset 1	1219	*sports ball*
Dataset 2	2020	*sports ball, person*

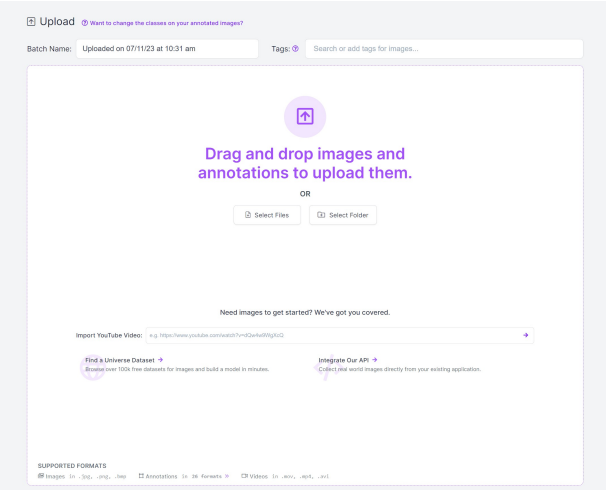

Figura 20: Pantalla para subir imágenes en Roboflow [28]

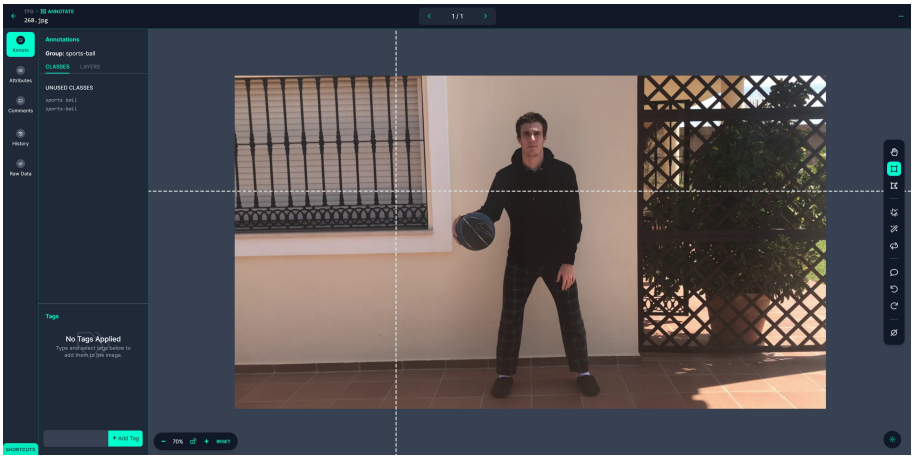

Figura 21: Etiquetar las imágenes en Roboflow [28]

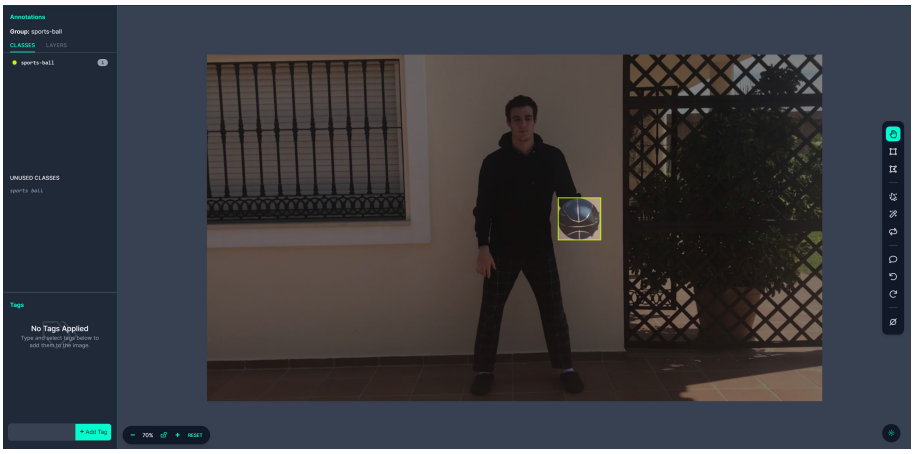

Figura 22: Dataset 1 ejemplo imagen etiquetada [28]

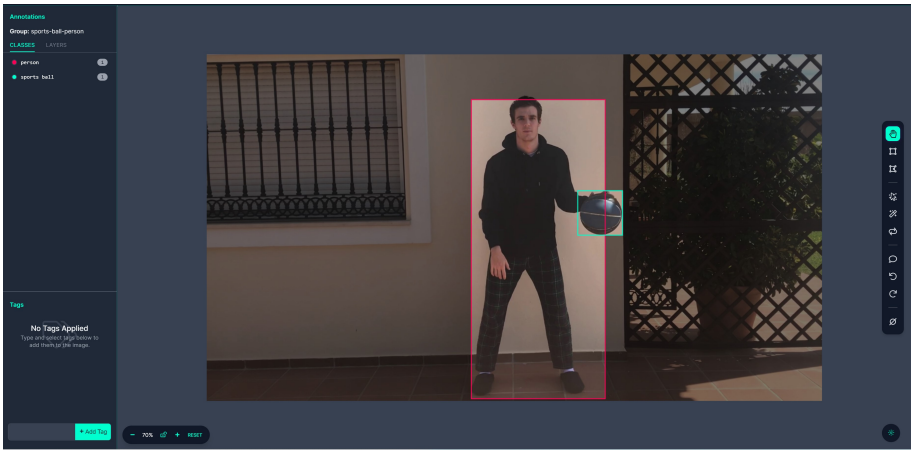

Figura 23: Dataset 2 ejemplo imagen etiquetada [28]

En la Figura 20 se muestra la pantalla para subir las imágenes a Roboflow. En las Figuras 21, 22 y 23 se muestran capturas de pantalla del procedimiento para etiquetar imágenes.

4. Metodología de trabajo

Para la elaboración de este trabajo se ha seguido una metodología influida por varios conceptos y principios propios de algunas metodologías como la metodología ágil e incremental, así como el método científico. Se han seguido también prácticas adecuadas para el desarrollo software, con un desarrollo extensible y bien documentado.

4.1. Metodología ágil e incremental

La metodología ágil se basa en ir añadiendo funcionalidades al proyecto en cada iteración o ciclo del desarrollo. Así, las metodologías ágiles, fundamentadas en las incrementales, tienen como objetivo ir introduciendo en el software diferentes "piezas" para satisfacer las necesidades del cliente ya que entienden el desarrollo a modo de entregas funcionales del proyecto final [32]. Una metodología que surgió debido a la variabilidad que podría ver en las necesidades de un proyecto desde que comienza hasta que acaba. Esta metodología tiene una serie de características [33]:

- Adaptabilidad a los cambios que puedan surgir durante el desarrollo de las partes del software y no negarse a ellos.
- Dar más valor a la comunicación con el cliente antes que la negociación contractual inicial.
- Busca la participación e implicación de los desarrolladores y por ende aumentar la calidad del proyecto final.
- División del trabajo por equipos y roles junto con reuniones organizadas y planificadas entre ellos.
- Se pretende ahorrar en costes debido a la baja posibilidad de fracaso absoluto.

4.2. Método científico

El método científico es el método por excelencia de la ciencia usado con el fin de obtener nuevos conocimientos siguiendo una serie de pasos: observación sistemática, medición, experimentación y la formulación, análisis y modificación de hipótesis [34]. Este método busca

reducir al máximo la subjetividad en los resultados de una investigación. Sin embargo, no todas las ciencias pueden utilizar este método, por ejemplo, las Ciencias Humanas y Sociales dada la variabilidad de su entorno hace que los fenómenos que son objeto de estudio no puedan repetirse controladamente. Este método tiene varias etapas o fases definidas como se muestra en la Figura 24:

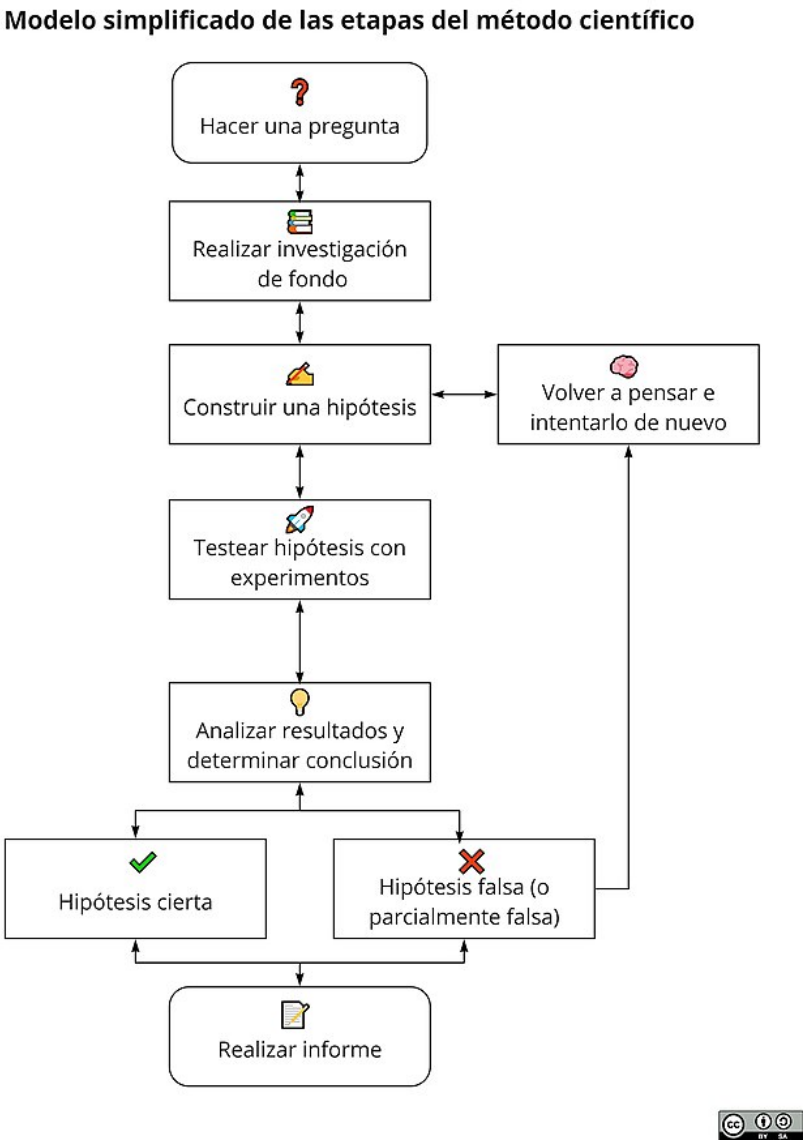

Figura 24: Fases del método científico [35]

Entre estas fases encontramos algunas que destacan por su importancia en el método:

- Observación, obtener diferente información de un fenómeno: frecuencia, dimensiones, etc.
- Hipótesis, se formula un enunciado que será objeto de estudio.

- Experimental, se intenta probar la hipótesis formulada.

4.3. Otros métodos

En cuanto a la elaboración de código software se deben seguir una serie de pautas para garantizar que el código generado es de calidad.

- Favorecer la legibilidad del código para que cualquier programador, aunque no sepa que se hace en el código, pueda entenderlo.
- No reproducir fragmentos de código idénticos (usar funciones en su lugar) para recortar la longitud del código.
- Mantener un orden y formato a la hora de estructurar el documento, nombrar las variables, etc.
- Código reutilizable y escalable.
- Comentar el código adecuadamente para favorecer su interpretación.

5. Análisis de especificaciones y Diseño del sistema

En este punto se muestran los diagramas de secuencia del modelo final. Aunque lo explicaremos con más detalle en secciones siguientes se evaluarán tres ejercicios referentes al bote y a la técnica individual. El Ejercicio 1 contador de botes, es un ejercicio donde sencillamente se espera que el programa cuente los botes. El Ejercicio 2 contador de botes con la misma mano pretende contar botes solo si se realizan con la misma mano. Por último, el Ejercicio 3 contador de cambios de mano se encarga de contabilizar los cambios que se realizan seguidamente, sin botes intermedios.

5.1. Requisitos del programa final

El primer diagrama de secuencia muestra los pasos de la ejecución del Ejercicio 1 contador de botes en la que se espera que el usuario suba el vídeo al programa y este le devuelva el número de botes realizados.

Requisito	Ejecutar Ejercicio 1 contador de botes
Descripción	Ejecución y análisis del Ejercicio 1 contador de botes

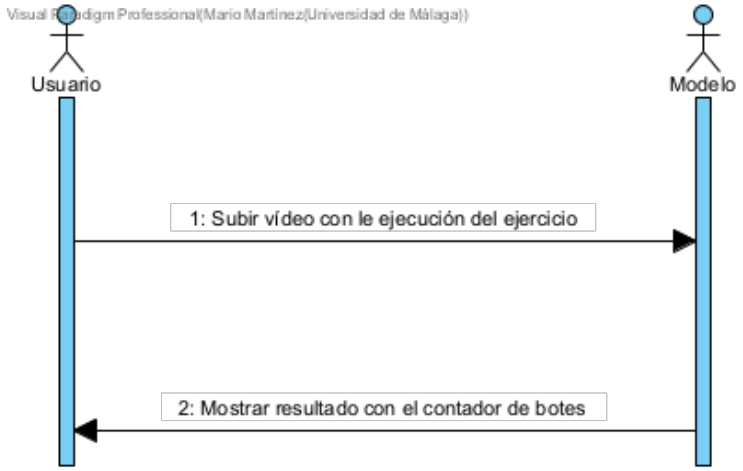

Figura 25: Diagrama de Secuencia para el requisito “Ejecutar Ejercicio 1 contador de botes”

El otro diagrama que se muestra refleja la secuencia de la ejecución del Ejercicio 2 contador de botes con la misma mano, donde solo se cuentan los botes y se le muestran al usuario si se realizan los botes siempre con la misma mano, por el contrario, se le mostrará un mensaje de error.

Requisito	Ejecutar Ejercicio 2 contador botes con la misma mano
Descripción	Ejecución y análisis del Ejercicio 2 contador botes con la misma mano

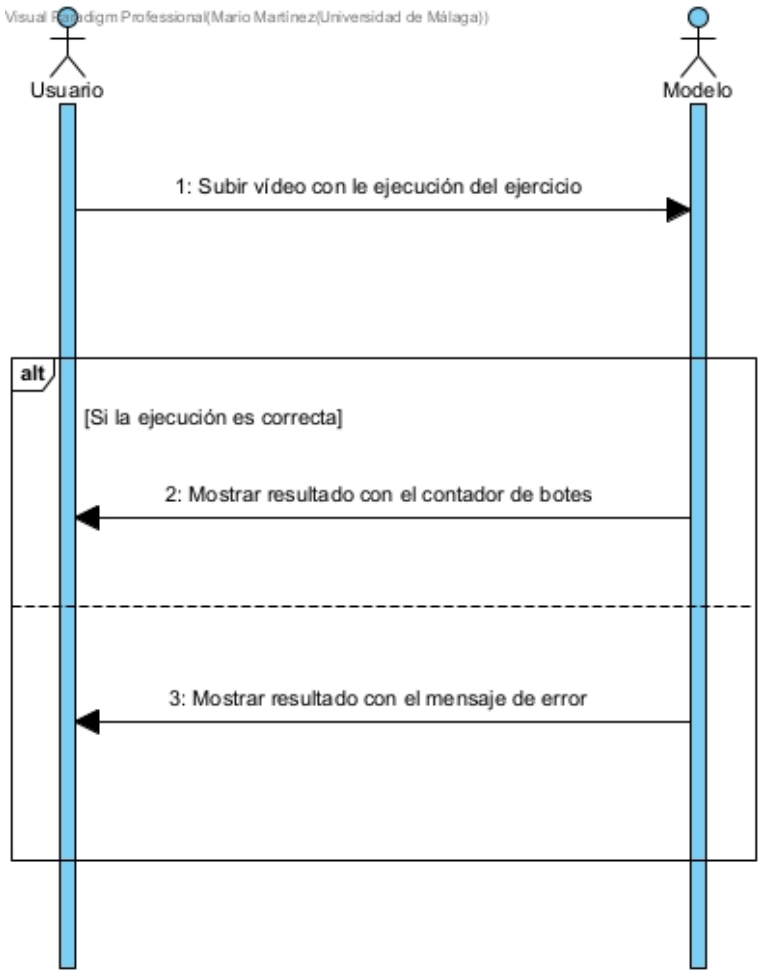

Figura 26: Diagrama de Secuencia para el requisito “Ejecutar Ejercicio 2 contador botes con la misma mano”

6. Llegar al modelo final

A continuación, detallaremos todas las pruebas que hemos ido realizando para alcanzar un modelo que detectara el objeto persona y pelota durante la mayor parte del tiempo. Como hemos comentado antes usaremos YOLO para detectar objetos en las imágenes y para entrenar modelos.

Además es importante destacar una serie de consideraciones previas. Primero, todos los vídeos usados para probar el resultado de los modelos son distintos a los utilizados para entrenar los modelo. Segundo, los vídeos tiene siempre la misma resolución de 1920x1080 a 30 fotogramas por segundo.

6.1. YOLO-v7

6.1.1. Primeras pruebas con YOLO-v7

Al comienzo de este trabajo se optó por implementar para las tareas de reconocimiento de imágenes YOLO-v7. Para comenzar a trabajar con este modelo es tan fácil como clonar el repositorio y ejecutar una serie de comandos que permitan ejecutar la detección o el entrenamiento.

La primera idea fue probar el repositorio de YOLO-v7 [36]. Para ello se clonó en una máquina local y se realizó una prueba básica con las imágenes de prueba que otorga el propio repositorio como se puede ver en las Figuras 27 y 28.

Figura 27: Reconocimiento de objetos en imagen de prueba YOLO-v7 [36]

Figura 28: Reconocimiento de objetos en imagen de prueba YOLO-v7 [36]

Para ejecutar este comando basta con seguir la documentación de este modelo y ejecutar desde la terminal y en la carpeta donde se encuentra el archivo **detect.py** el siguiente comando:

Tras estas pruebas se pasó a probar diferentes modelos preentrenados con los vídeos que en este trabajo se van a tratar: una persona y un balón frente a una cámara, ver figura 31. El resultado no fue malo, aunque tampoco fue bueno del todo. El objeto persona era detectado en todo momento. Sin embargo, el objeto pelota había varios fotogramas en los que se perdía y no se detectaba. Por ejemplo, uno de los vídeos cuenta con 140 fotogramas de los cuales en unos 40 no se detecta ningún objeto pelota cuando realmente aparece en todo momento, ver Figura

```
python detect.py --weights yolov7.pt --conf 0.25 --img-size 640 --source
inference/images/bus.jpg --device cpu
```

Figura 29: Comando para ejecutar YOLO-v7 [36]

30. Esto puede no ser un problema y corregirse cuando apliquemos algoritmos para hacer el seguimiento.

```
video 1/1 (106/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 bench, 1 tennis racket, Done. (613.2ms) Inference, (1.0ms) NMS
video 1/1 (107/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 tennis racket, Done. (652.0ms) Inference, (2.0ms) NMS
video 1/1 (108/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 sports ball, 1 tennis racket, Done. (627.9ms) Inference, (1.0ms) NMS
video 1/1 (109/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, Done. (618.3ms) Inference, (1.0ms) NMS
video 1/1 (110/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 sports ball, Done. (622.9ms) Inference, (1.0ms) NMS
video 1/1 (111/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 sports ball, Done. (618.3ms) Inference, (3.0ms) NMS
video 1/1 (112/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 sports ball, Done. (618.0ms) Inference, (2.0ms) NMS
video 1/1 (113/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 sports ball, Done. (628.6ms) Inference, (0.5ms) NMS
video 1/1 (114/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 sports ball, Done. (621.5ms) Inference, (1.5ms) NMS
video 1/1 (115/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 sports ball, Done. (622.8ms) Inference, (2.0ms) NMS
video 1/1 (116/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 sports ball, Done. (621.6ms) Inference, (1.0ms) NMS
video 1/1 (117/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 bench, Done. (620.2ms) Inference, (2.0ms) NMS
video 1/1 (118/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, Done. (626.8ms) Inference, (1.0ms) NMS
video 1/1 (119/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, Done. (618.2ms) Inference, (2.0ms) NMS
video 1/1 (120/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, Done. (611.0ms) Inference, (1.0ms) NMS
video 1/1 (121/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, Done. (627.1ms) Inference, (2.0ms) NMS
video 1/1 (122/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 sports ball, Done. (615.2ms) Inference, (2.0ms) NMS
video 1/1 (123/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 sports ball, Done. (633.9ms) Inference, (1.0ms) NMS
video 1/1 (124/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 bench, 1 sports ball, Done. (612.3ms) Inference, (2.0ms) NMS
video 1/1 (125/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 bench, 1 sports ball, Done. (616.6ms) Inference, (1.6ms) NMS
```

Figura 30: Resultado fotograma a fotograma de las detecciones con yolov7.pt

Figura 31: fotograma de un vídeo propio con yolov7.pt

Este vídeo, que se muestra en las figuras 30 y 31 fue procesado con el modelo *yolov7.pt*, un modelo ya preentrenado. En YOLO existen varios modelos ya preentrenados con los que puedes probar según las especificaciones de los vídeos y la aplicación donde se vayan a usar, ver Figura 32.

Para probar diferentes rendimientos se probó con *yolov7x-pt* otro modelo preentrenado. Así, con los mismos vídeos que probamos con *yolov7.pt* se repitió la detección con este nuevo modelo y los resultados fueron parecidos, como se puede ver en la Figura 33. El objeto persona

MS COCO

Model	Test Size	AP^{test}	$AP_{50}{}^{test}$	$AP_{75}{}^{test}$	batch 1 fps	batch 32 average time
YOLOv7	640	51.4%	69.7%	55.9%	161 *fps*	2.8 *ms*
YOLOv7-X	640	53.1%	71.2%	57.8%	114 *fps*	4.3 *ms*
YOLOv7-W6	1280	54.9%	72.6%	60.1%	84 *fps*	7.6 *ms*
YOLOv7-E6	1280	56.0%	73.5%	61.2%	56 *fps*	12.3 *ms*
YOLOv7-D6	1280	56.6%	74.0%	61.8%	44 *fps*	15.0 *ms*
YOLOv7-E6E	1280	56.8%	74.4%	62.1%	36 *fps*	18.7 *ms*

Figura 32: Modelos para YOLO-v7 [36]

estaba en todo momento detectado mientras el objeto pelota no llegaba a estar reconocido durante gran parte de la detección, ver Figura 34.

```
video 1/1 (16/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 potted plant, Done. (797.3ms) Inference, (0.0ms) NMS
video 1/1 (17/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 potted plant, Done. (750.4ms) Inference, (0.0ms) NMS
video 1/1 (18/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 potted plant, Done. (734.8ms) Inference, (0.0ms) NMS
video 1/1 (19/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 potted plant, Done. (719.2ms) Inference, (15.6ms) NMS
video 1/1 (20/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 sports ball, 1 potted plant, Done. (718.8ms) Inference, (0.0ms) NMS
video 1/1 (21/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 potted plant, Done. (796.9ms) Inference, (0.0ms) NMS
video 1/1 (22/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 potted plant, Done. (734.8ms) Inference, (15.6ms) NMS
video 1/1 (23/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, Done. (750.0ms) Inference, (0.0ms) NMS
video 1/1 (24/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, Done. (750.0ms) Inference, (0.0ms) NMS
video 1/1 (25/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 bowl, Done. (703.1ms) Inference, (0.0ms) NMS
video 1/1 (26/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 sports ball, Done. (812.5ms) Inference, (0.0ms) NMS
video 1/1 (27/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 sports ball, Done. (750.0ms) Inference, (0.0ms) NMS
video 1/1 (28/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 sports ball, Done. (703.5ms) Inference, (0.0ms) NMS
video 1/1 (29/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 sports ball, Done. (765.6ms) Inference, (0.0ms) NMS
video 1/1 (30/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 sports ball, Done. (765.6ms) Inference, (0.0ms) NMS
video 1/1 (31/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, Done. (765.6ms) Inference, (0.0ms) NMS
video 1/1 (32/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 frisbee, 1 potted plant, Done. (781.3ms) Inference, (0.0ms) NMS
video 1/1 (33/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 frisbee, Done. (844.2ms) Inference, (0.0ms) NMS
video 1/1 (34/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 frisbee, Done. (875.8ms) Inference, (0.0ms) NMS
video 1/1 (35/140) D:\UNIVERSIDAD\TFG\yolov7\inference\video1.mov: 1 person, 1 frisbee, Done. (859.8ms) Inference, (0.0ms) NMS
```

Figura 33: Resultado fotograma a fotograma de las detecciones con yolov7x.pt

6.1.2. Seguimiento con YOLO-v7

Aunque los resultados no captaron toda la información deseada se esperaba que haciendo el seguimiento pudiéramos evitar los fotogramas donde no había detecciones de pelota. Para ello se usó un algoritmo de DeepSort con YOLO-v7 [37], ya que YOLO-v7 no tiene implementado el seguimiento. Se preparó el entorno con las dependencias necesarias para que funcionara y se procedieron a las pruebas.

Cuando usamos el algoritmo DeepSort podemos modificar varios parámetros que varían el comportamiento de modelo a la hora de hacer el seguimiento. Estos parámetros son:

- **max_cosine_distance**, este valor es el encargado de marcar el perímetro de búsqueda

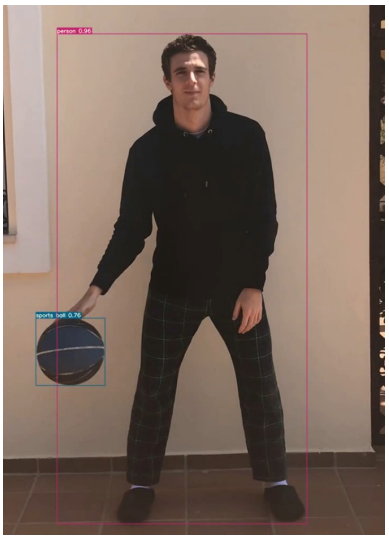

Figura 34: fotograma de un vídeo propio con yolov7x.pt

del objeto. Es decir, si se ha encontrado una pelota en unas coordenadas del fotograma y este parámetro tiene un valor de 2, como medida se toman los píxeles ya que las coordenadas se obtienen en función del tamaño de la imagen, la pelota que se encuentre en el fotograma siguiente a una distancia no superior a 2 píxeles de esas coordenadas será identificada como el mismo objeto. Este parámetro debe ser alto cuando el objeto a detectar se mueve por el vídeo a una velocidad alta, como puede ser en nuestro caso la pelota.

- **max_age**, es el tiempo de vida del objeto detectado medido en número de fotogramas. Cuántos fotogramas pueden pasar para que si nos encontramos de nuevo el mismo objeto se identifique como él mismo y no como uno nuevo. Cuando la detección de un objeto no es demasiado buena poner este valor alto pueda ayudar en el seguimiento, aunque también podría hacer identificar dos objetos distintos como el mismo.

Los valores por defecto son: *max_cosine_distance* con un valor de 0,4 y *max_age* igualado a 75. Sin modificar estos valores se decidió probar a realizar el seguimiento, Figuras 35 y 36:

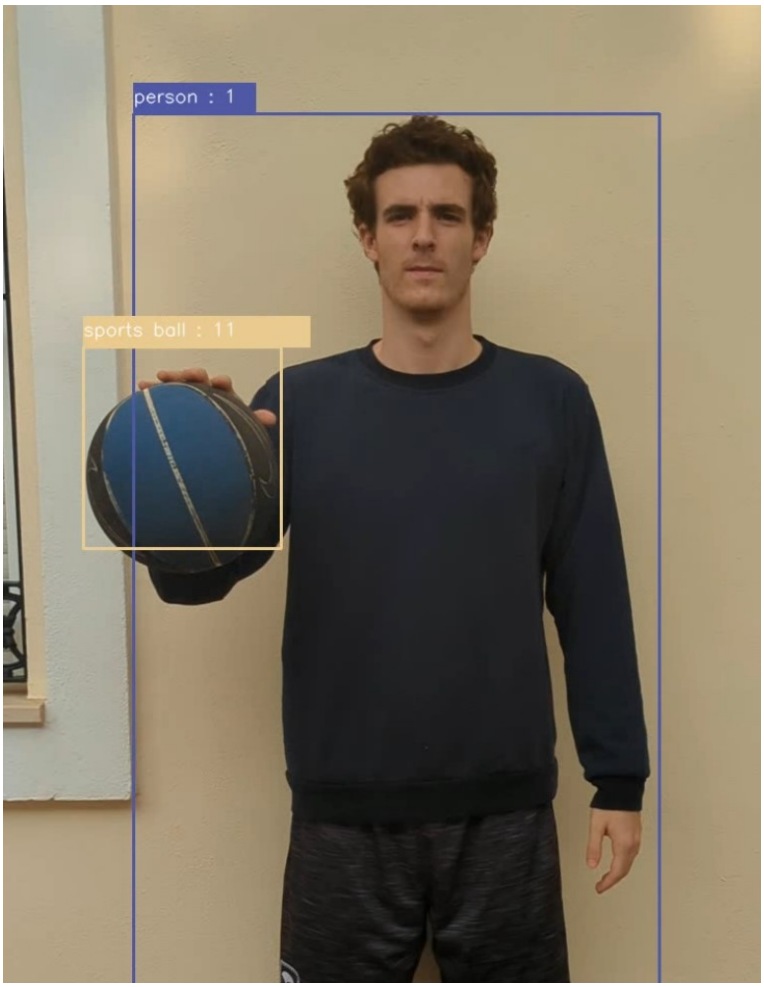

Figura 35: fotograma seguimiento YOLOv7 y DeepSort

Los resultados no son del todo buenos. Como se puede ver en 35 y 36 al objeto pelota se le atribuyen varios identificadores. Además, en la mayor parte del tiempo el objeto no está identificado por lo que los resultados han sido peores que anteriormente.

Tras esto modificamos los valores de los parámetros mencionados anteriormente. Las pruebas fueron las siguientes:

- *max_cosine_distance* con un valor de 100 y *max_age* a 500, el resultado fue practicamente idéntico al obtenido con los valores por defecto.

- *max_cosine_distance* con un valor de 1000 y *max_age* igualado a 1000, una prueba que tampoco arrojó nada distinto a lo que veníamos comprobando.

Figura 36: fotograma seguimiento YOLOv7 y DeepSort

6.2. YOLO-v8

Mientras se realizaban todas las pruebas pertinentes con YOLO-v7 salió a la luz YOLO-v8. Debido al componente académico de este trabajo podríamos haber continuado con YOLO-v7 pero la nueva versión está mejor documentada, incluye seguimiento y tiene un código mucho más legible que su predecesora entre otras mejoras de rendimiento.

Por esto volvimos a repetir las pruebas iniciales que tuvimos con YOLO-v7 pero esta vez con YOLO-v8

6.2.1. Primeras pruebas con YOLO-v8

Las primeras pruebas fueron con los vídeos propios de este trabajo, es decir un jugador y una pelota frente a una cámara de vídeo. Al igual que con YOLO-v7 realizamos varias pruebas con algunos de los modelos que ofrece YOLO-v8, Figura 37.

Las pruebas que se hicieron con estos modelos tuvieron distintos resultados. A continuación se describen distintos comportamientos que encontramos con los modelos probados.

- **yolov8n.pt**, la persona casi siempre se encuentra detectada pero la pelota la mayor parte

Model	size (pixels)	mAP^{val} 50-95	Speed CPU ONNX (ms)	Speed A100 TensorRT (ms)	params (M)	FLOPs (B)
YOLOv8n	640	37.3	80.4	0.99	3.2	8.7
YOLOv8s	640	44.9	128.4	1.20	11.2	28.6
YOLOv8m	640	50.2	234.7	1.83	25.9	78.9
YOLOv8l	640	52.9	375.2	2.39	43.7	165.2
YOLOv8x	640	53.9	479.1	3.53	68.2	257.8

Figura 37: Modelos YOLO-v8 [25]

del tiempo no.

- **yolov8l.pt**, la persona al igual que sucedía con el modelo de arriba casi siempre está detectada. Por otro lado, la pelota parece que tiene un mejor funcionamiento, aunque no siempre está detectada y hay varios fotogramas en los que se pierde.

- **yolov8x.pt**, comportamiento muy similar al modelo superior.

Con éstas no encontramos un modelo que claramente nos diera un rendimiento suficiente y pudiera ser un claro candidato a usarse como modelo principal de nuestro trabajo. Tras esto el mejor modelo, que sería yolov8x.pt según la información proporcionada por YOLO, fue el que intentamos exprimir por si sacábamos algo más de rendimiento. Como se puede ver en las figuras durante la mayor parte del tiempo yolov8n.pt no detecta la pelota, ver figura 38, sino que solo lo hace en algunos fotogramas. Sin embargo, el resultado con yolov8x.pt es mucho mejor y la pelota ahora solo se evita en algunos fotogramas, ver figura 39.

```
video 1/1 (48/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 1 sports ball, 34.9ms
video 1/1 (49/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 1 sports ball, 29.9ms
video 1/1 (50/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 1 sports ball, 32.3ms
video 1/1 (51/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 1 sports ball, 31.1ms
video 1/1 (52/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 29.8ms
video 1/1 (53/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 30.2ms
video 1/1 (54/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 40.4ms
video 1/1 (55/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 30.1ms
video 1/1 (56/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 40.0ms
video 1/1 (57/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 30.2ms
video 1/1 (58/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 31.4ms
video 1/1 (59/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 35.2ms
video 1/1 (60/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 39.9ms
video 1/1 (61/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 1 sports ball, 29.7ms
```

Figura 38: Resultado detección usando yolov8n.pt

Para seguir con las pruebas se probó con el seguimiento. A diferencias de YOLO-v7 el seguimiento en esta versión de Ultralytics sí se puede ejecutar. Como había un modelo que funcionaba de manera más o menos razonable se procedió a ello. Los resultados fueron un tanto confusos. Lo primero que llamó la atención fue que la detección que se hacía durante el seguimiento de la pelota no era tan exacta como la que se realizó cuando solamente se probó la detección. Esto llevó a que en el seguimiento la pelota se dejaba de seguir durante

```
video 1/1 (72/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 1 sports ball, 78.1ms
video 1/1 (73/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 1 sports ball, 88.4ms
video 1/1 (74/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 1 sports ball, 93.7ms
video 1/1 (75/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 1 sports ball, 78.1ms
video 1/1 (76/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 1 sports ball, 93.7ms
video 1/1 (77/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 1 sports ball, 78.1ms
video 1/1 (78/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 1 sports ball, 78.1ms
video 1/1 (79/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 78.1ms
video 1/1 (80/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 78.1ms
video 1/1 (81/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 93.7ms
video 1/1 (82/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 78.1ms
video 1/1 (83/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 1 sports ball, 78.1ms
video 1/1 (84/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 78.1ms
video 1/1 (85/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 1 sports ball, 93.7ms
```

Figura 39: Resultado detección usando yolov8x.pt

varios fotogramas, no solo unos pocos como creíamos sino varios de ellos. A continuación en la Figuras 40, 41 y 42 y 43,.

```
video 1/1 (26/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 83.7ms
video 1/1 (27/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 83.5ms
video 1/1 (28/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 82.9ms
video 1/1 (29/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 81.7ms
video 1/1 (30/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 1 sports ball, 83.7ms
video 1/1 (31/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 83.7ms
video 1/1 (32/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 1 sports ball, 83.9ms
video 1/1 (33/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 83.3ms
video 1/1 (34/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 81.8ms
video 1/1 (35/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 84.6ms
video 1/1 (36/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 82.0ms
video 1/1 (37/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 85.2ms
video 1/1 (38/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 81.0ms
video 1/1 (39/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 83.6ms
video 1/1 (40/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 83.4ms
video 1/1 (41/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 81.9ms
video 1/1 (42/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 83.6ms
video 1/1 (43/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 82.1ms
video 1/1 (44/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 1 sports ball, 84.2ms
```

Figura 40: Resultado del seguimiento con yolov8x.pt

Figura 41: Resultado del seguimiento con yolov8x.pt sin estar la pelota detectada

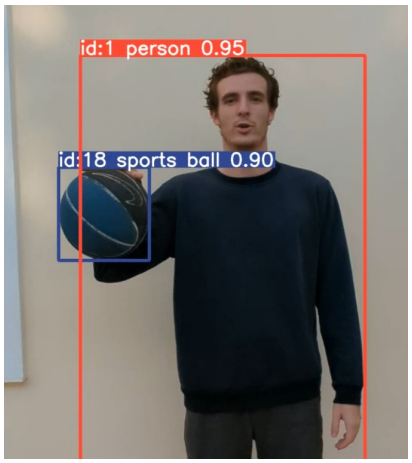

Figura 42: Resultado del seguimiento con yolov8x.pt en un momento inicial

Figura 43: Resultado del seguimiento con yolov8x.pt en un momento final

6.3. Fine-tunning

Además de elaborar un conjunto de datos propio para entrenar un modelo que pudiera encajar mejor con los requerimientos de este trabajo se entrenaron otros modelos usando imágenes de Open Images [31] y Coco [29]. Esto se llama fine-tunning, es decir, ajustar un modelo existente como pueden ser los de YOLO-v8 con imágenes de otros conjuntos de datos [38].

6.3.1. Entrenamiento inicial: conjuntos de imágenes completos

La primera idea fue introducir al ajuste del modelo, usamos yolov8n.pt, todas las imágenes de pelota y de persona de los conjuntos de Coco y Open Images. Esto resultó inviable computacionalmente para los medios a los que en este trabajo se han tenido acceso. Se necesitaba un entrenamiento de una gran cantidad de épocas y cada época suponía varias horas de cómputo. Por esto se procedió a probar con números de imágenes más pequeños.

6.3.2. Reentrenamiento 1: imágenes del objeto pelota

El primer reentrenamiento que se elaboró, dado que el mayor problema es la detección de la pelota y no de la persona, fue el siguiente:

- 5000 imágenes de pelota de Coco,
- 5000 imágenes de pelota de Open Images.

Se realizó el entrenamiento con 10 épocas. El resultado fue un modelo que no mejoró el del modelo ya existente. Se pensó que principalmente esto se pudo deber a que las imágenes que añadimos no fueron suficientes. Además, se detectó que había algunos vídeos en los que la persona se dejaba de detectar en varios momentos como se ve en la Figura 44.

```
video 1/1 (7/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 1 person, 11.5ms
video 1/1 (8/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 16.5ms
video 1/1 (9/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 1 person, 14.1ms
video 1/1 (10/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 1 person, 11.5ms
video 1/1 (11/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 11.5ms
video 1/1 (12/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 1 person, 11.5ms
video 1/1 (13/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 1 person, 11.1ms
video 1/1 (14/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 12.0ms
video 1/1 (15/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 16.7ms
video 1/1 (16/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 12.5ms
video 1/1 (17/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 12.1ms
video 1/1 (18/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 13.5ms
video 1/1 (19/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 13.5ms
video 1/1 (20/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 15.4ms
```

Figura 44: Ejemplo de la detección con el reentrenamiento 1

6.3.3. Reentrenamiento 2: pocas imágenes de todas las clases

Tras el primer entrenamiento comprobamos que había ciertos momentos en según que vídeos la persona tampoco era detectada por eso se procedió a aumentar el tamaño del conjunto de datos con imágenes de persona.

- 5000 imágenes de pelota y 5000 imágenes de persona de Coco,
- 5000 imágenes de pelota y 5000 imágenes de Open Images.

Este entrenamiento, con 10 épocas, tampoco dio bueno resultados y la persona seguía sin ser detectada, nos encontramos con vídeos en los que o bien la pelota se detectaba de manera razonable pero la persona no o viceversa.

```
video 1/1 (12/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 1 person, 14.0ms
video 1/1 (13/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 1 person, 13.2ms
video 1/1 (14/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 15.5ms
video 1/1 (15/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 14.9ms
video 1/1 (16/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 11.5ms
video 1/1 (17/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 13.5ms
video 1/1 (18/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 11.5ms
video 1/1 (19/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 13.0ms
video 1/1 (20/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 11.5ms
video 1/1 (21/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 1 person, 13.5ms
video 1/1 (22/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 11.5ms
video 1/1 (23/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 12.1ms
video 1/1 (24/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 1 person, 14.5ms
video 1/1 (25/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 1 person, 12.5ms
video 1/1 (26/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 1 person, 11.0ms
```

Figura 45: Ejemplo de la detección con el reentrenamiento 2

6.3.4. Reentrenamiento 3: aumentar imágenes de todas las clases

Dado que el reentrenamiento anterior no mejoró el modelo existente se realizó otro entrenamiento:

- 10000 imágenes de pelota y 5000 imágenes de persona de Coco,
- 10000 imágenes de pelota y 5000 imágenes de Open Images.

El resultado sigue siendo parecido, o bien se detecta un objeto o bien se detecta otro. Estos modelos no servirían para poder realizar el seguimiento de los objetos y extraer información de ellos. Se adjunta imágen del entrenamiento a 10 épocas.

```
video 1/1 (31/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 11.5ms
video 1/1 (32/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 1 person, 12.5ms
video 1/1 (33/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 1 person, 11.4ms
video 1/1 (34/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 1 person, 11.5ms
video 1/1 (35/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 11.5ms
video 1/1 (36/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 13.5ms
video 1/1 (37/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 1 person, 13.5ms
video 1/1 (38/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 14.0ms
video 1/1 (39/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 11.3ms
video 1/1 (40/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 12.5ms
video 1/1 (41/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 13.5ms
video 1/1 (42/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 16.1ms
video 1/1 (43/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 12.1ms
video 1/1 (44/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 11.1ms
video 1/1 (45/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 11.5ms
video 1/1 (46/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 person, 13.4ms
video 1/1 (47/101) D:\UNIVERSIDAD\TFG\yolov8\ultralytics\pruebas\Videos\video9.mp4: 640x384 1 sports ball, 1 person, 12.1ms
```

Figura 46: Resultado de la detección con el reentrenamiento 3

6.4. Elaboración del conjunto de imágenes propio

Como el resultado con los modelos de YOLO-v8 no eran los esperados ya que a priori necesitábamos algo más de exactitud en el seguimiento y por ende mayor precisión en las detecciones, dado el componente científico e investigador de este trabajo, se procedió a elaborar un dataset propio que pudiera aportarnos las características deseadas.

Antes de elaborar un dataset mayor, dado el esfuerzo en tiempo que requiere una tarea como ésta, se decidió elaborar un primer conjunto de imágenes más pequeño para ver qué resultados obteníamos con ello. El primer conjunto usado fue el mencionado en el apartado anterior 3.1.3 nombrado “Dataset 1”. Este conjunto de datos cuenta con 490 imágenes donde solo está etiquetado el objeto pelota, tras procesar las imágenes Roboflow nos generó 1219 (1100 de entrenamiento, 88 de validación y 35 de test). Con esto se procedió a entrenar el modelo con YOLO-v8, como este entrenamiento es de prueba y solo queríamos ver el funcionamiento del conjunto de datos se entrenó con 5 épocas, ver Figura 47.

```
Transferred 319/355 items from pretrained weights
optimizer: SGD(lr=0.01) with parameter groups 57 weight(decay=0.0), 64 weight(decay=0.0005), 63 bias
train: Scanning /home/mariomartinez/Documentos/yolov8/ultralytics/TFG.v2i.yolov8/train/labels.cache... 1096 images, 0 backgrounds, 0 corrupt: 100%|██████████| 1096/1096 [00:00<?, ?it/s]
val: Scanning /home/mariomartinez/Documentos/yolov8/ultralytics/TFG.v2i.yolov8/valid/labels.cache... 88 images, 0 backgrounds, 0 corrupt: 100%|██████████| 88/88 [00:00<?, ?it/s]
Plotting labels to runs/detect/train36/labels.jpg...
Image sizes 640 train, 640 val
Using 0 dataloader workers
Logging results to runs/detect/train36
Starting training for 5 epochs...

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
        1/5         0G       1.06      1.798      1.033          8        640: 100%|██████████| 274/274 [07:45<00:00,  1.70s/it]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 11/11 [00:13<00:00,  1.23s/it]
                   all         88         88      0.998      0.989      0.995      0.744

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
        2/5         0G     0.9832      0.855      1.008          4        640: 100%|██████████| 274/274 [07:42<00:00,  1.69s/it]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 11/11 [00:12<00:00,  1.18s/it]
                   all         88         88      0.998      0.989      0.995      0.779

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
        3/5         0G       0.97     0.7739       1.01         10        640: 100%|██████████| 274/274 [08:29<00:00,  1.86s/it]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 11/11 [00:13<00:00,  1.26s/it]
                   all         88         88      0.999          1      0.995      0.745

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
        4/5         0G      0.933     0.7161      1.008         10        640: 100%|██████████| 274/274 [23:46<00:00,  5.21s/it]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 11/11 [00:14<00:00,  1.32s/it]
                   all         88         88      0.999          1      0.995      0.773

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
        5/5         0G     0.9098      0.675     0.9947          5        640: 100%|██████████| 274/274 [07:44<00:00,  1.69s/it]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 11/11 [00:15<00:00,  1.37s/it]
                   all         88         88      0.999          1      0.995      0.785

5 epochs completed in 0.949 hours.
Optimizer stripped from runs/detect/train36/weights/last.pt, 6.2MB
Optimizer stripped from runs/detect/train36/weights/best.pt, 6.2MB
```

Figura 47: Ejemplo entrenamiento del conjunto Dataset 1

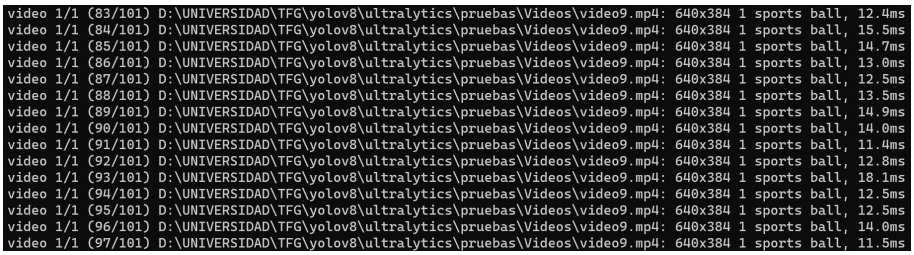

Figura 48: Resultado detect con el entrenamiento del conjunto Dataset 1

Como se ve en la Figura 48 los resultados no pudieron ser mejores, la pelota estaba detectada en todo momento. La persona no es detectada porque en nuestro conjunto de datos no está etiquetado este objeto.

El siguiente paso es elaborar un conjunto de imágenes más grande y etiquetar los objetos persona y pelota. Este conjunto es el que definimos en el apartado 3.1.3 como “Dataset 2”, está compuesto por 2020 imágenes donde 1800 son de entrenamiento, 121 de validación y 88 de test. Este conjunto de imágenes fue entrenado con YOLO a 75 épocas dado que ya no buscaba ser un entrenamiento de prueba sino el definitivo. El resultado volvió a ser positivo, tal y como esperábamos dado los resultados con el entrenamiento que hicimos de prueba. En distintos vídeos el resultado fue bueno como se ve en la Figura 49.

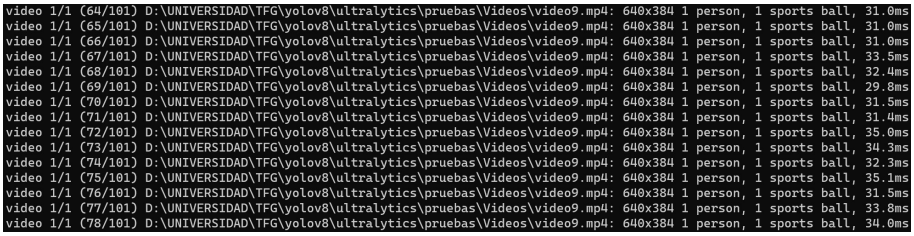

Figura 49: Resultado detect con el entrenamiento del conjunto Dataset 2

6.5. Resultados

Tras las pruebas realizadas para tener el modelo que mejor se adapte se decidió continuar trabajando con el modelo resultado del entrenamiento con el “Dataset 2” del apartado 3.1.3. Reentrenar un modelo de los que ofrece YOLO-v8 es una tarea costosa computacionalmente. Además, se optó por crear un conjunto de datos propio para también trabajar e investigar esa parte de la rama de detección de imágenes que en este trabajo se trata. Como se explica arriba, el “Dataset 2” es el que mejor resultado ha obtenido ya que en todos lo vídeos la pelota está identificada durante casi todos los fotogramas, por lo que la información a extraer del

movimiento de los objetos en el futuro será precisa.

Haber encontrado un modelo de los entrenados por YOLO-v8 hubiera acortado mucho el trabajo y nos hubiera liberado más tiempo para tareas futuras del mismo. Sin embargo, esta fase del trabajo ha sido con diferencia la más costosa. Primero por la ocupación de recursos. En las pruebas anteriores una época tardaba algo más de una hora.

6.5.1. Problemas de compatibilidad con las tarjetas gráficas

Durante los entrenamientos se intentó reforzar el proceso con la utilización de la tarjeta gráfica dedicada que contaba el equipo con el que se ha hecho este trabajo. En concreto el PC que se ha usado para la mayoría de las pruebas ha sido el siguiente:

- Intel(R) Core(TM) i7-9750H,
- NVIDIA GeForce GTX 1650.

Sin embargo, el problema que se encontró investigando en la comunidad de usuarios de YOLO-v8 fue un problema de compatibilidad [39] [40] y algunas librerías necesarias para ejecutar los distintos comandos. Cuando realizamos el entrenamiento en algunos parámetros se mostraba un valor nulo o desconocido, ver figura 50. Para resolver el problema se optó por poner valores de *batch* muy bajos, ver Figura 51 para que el entrenamiento sí tuviera efecto. De otro modo el entrenamiento terminaba pero no se alcanzaba ningún resultado. Estoy influía directamente en la velocidad que podría llegar a tener el entrenamiento ya que limitábamos en cierto modo la utilización de los recursos de la gráfica.

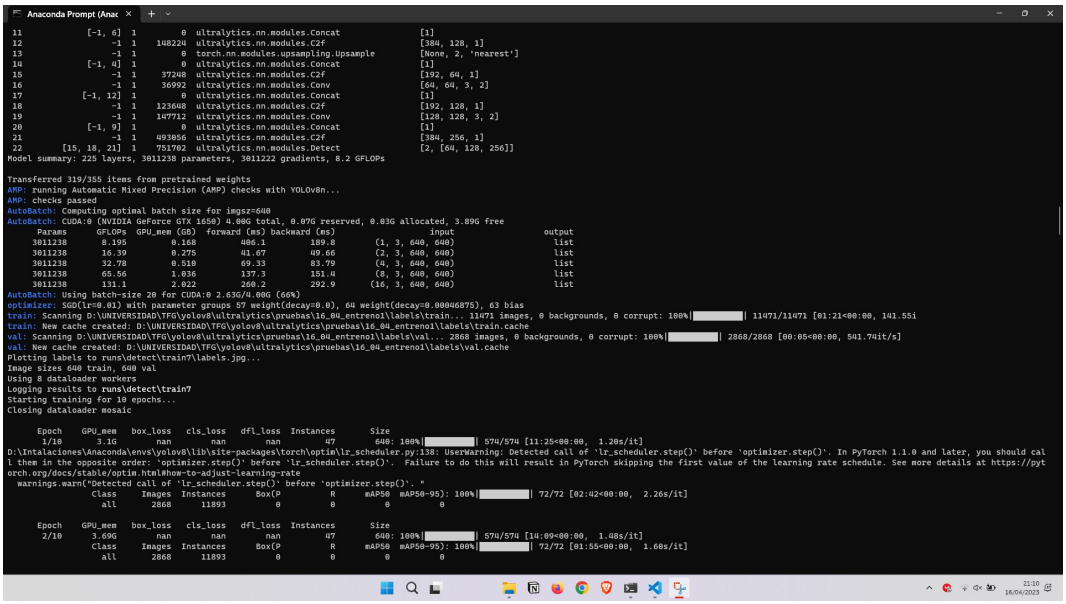

Figura 50: Entrenamiento con valores “nan” debido a la gráfica

```
yolo task=detect mode=train model=yolov8n.pt data=30_04entreno1/dataset.yaml epochs=10 imgsz=640 batch=2
```

Figura 51: comando con un batch bajo para realizar el entrenamiento

7. Valoración ejecución de ejercicios

7.1. Algoritmo de seguimiento

Una vez se ha encontrado el modelo de detección que mejor rendimiento tiene con los objetos persona y pelota se procedió a implementar junto con YOLO-v8 el algoritmo de Deep SORT (Simple Online Real-time Tracking). Este algoritmo se basa en 4 componentes clave [41].

- Detección,
- Estimación,
- Asociación,
- Creación y destrucción de las identidades.

7.1.1. Detección

Como se ha contado en secciones anteriores de este documento, la detección de los objetos ha ocupado gran parte del esfuerzo de este trabajo ya que, aunque la detección por sí no sea suficiente para realizar el seguimiento es fundamental, y cuanto mejor sea esta mejor será el seguimiento.

7.1.2. Estimación

Una vez tenemos las detecciones de los objetos en un instante determinado del vídeo tenemos que ser capaces de propagar las detecciones al instante justamente posterior. Para ello se hace uso del Filtro de Kalman [42] y un modelo de velocidad lineal. Es decir, en el momento de detección de un objeto se aplica el Filtro de Kalman para estimar y corregir la posición posterior, si no hay detección en un instante solamente se predice.

7.1.3. Asociación

La asociación se realiza calculando la intersección que hay entre una detección y los *bounding box*. Así cuanto mayor coincidencia mayor probabilidad de asociación.

7.1.4. Creación y destrucción de las identidades

Cuando los objetos aparecen o desaparecen de la imagen o sencillamente dejan de detectarse durante unos fotogramas hay que otorgar a las detecciones de ese objeto un identificador, si por ejemplo se ha dejado de detectar un objeto durante más de una cierta cantidad de fotogramas a la siguiente detección que ocurra se le otorga un nuevo identificador. Además, al detectar el objeto por primera se le otorga una incertidumbre muy grande en cuanto a cuál será la siguiente posición que ocupe, así cuando se detecte varias veces y se componga el vector velocidad la incertidumbre de la siguiente posición bajará.

7.1.5. Parámetros de nuestro modelo

Para la configuración de este modelo se ha optado por configurar los siguientes parámetros:

- *max_age*, hace referencia al tiempo que podrá dejar de detectarse un objeto para que cuando se vuelva a hacerlo se le asigne el mismo identificador.
- *max_distance*, hace referencia a la distancia que puede haber entre la detección de un objeto en un instante y en el posterior a este.

7.2. Ejercicios a evaluar

En el ámbito del bote, o manejo de balón, en el baloncesto hay prácticamente infinidad de ejercicios que se pueden realizar y por ende evaluar para otorgar un *feedback*. La idea principal que se pretende seguir es comenzar por los movimientos más sencillos para poder avanzar hacía algunos más complejos.

En principio se pretende comprobar que el jugador realiza el movimiento en cuestión correctamente. Es decir, si tiene que botar un determinado número de veces, ser capaz de controlar esto y en caso de que lo consiga sencillamente no otorgar ninguna retroalimentación, en caso contrario entonces marcar que el ejercicio no se ha ejecutado correctamente.

El bote en el baloncesto tiene particularidades propias: controlar la altura del mismo, botar fuera del cuerpo (no delante de él), etc. Con esto, el requisito principal para contabilizar un bote será controlar que el balón supera la mitad del cuerpo de la persona y siempre cuando lleve una dirección ascendente.

Otro de los gestos que contabilizaremos será el cambio de mano. Dado que si la pelota está en el lado derecho de un jugador este debe botar con la mano derecha y cuando la pelota se

encuentra en el lado izquierdo este debe botar con la mano izquierda, jamás podremos tener el balón en el lado derecho mientras se bota con la mano contraria. Por esto cuando el balón cruce la línea vertical que estableceremos en la mitad del ancho de la persona se considerará que ha cambiado de mano. Cuando esto ocurra el balón ahora podrá subir y contabilizar un bote (subir ascendentemente por encima de la mita de una persona).

Dada la explicación de qué gestos propios del bote son tenidos en cuenta y cuando se dan por realizados se decidieron contabilizar los siguientes ejercicios:

- **Ejercicio 1 contador de botes**, contabilizar el número de botes que se realizan indistintamente de la mano que se use para ello.
- **Ejercicio 2 contador botes con la misma mano**, botar siempre con la misma mano, es decir no se permiten cambios de mano. El jugador debe mantener una posición correcta para la ejecución del ejercicio, botar alto y siempre en el lado que toque.
- **Ejercicio 3 contador cambios de mano**, realizar cambios de manos continuamente sin poder realizar botes intermedios. El jugador deberá realizar sucesivos cambios de mano por delante sin poder dar ningún bote entre ellos.

Estos ejercicios, aunque son sencillos pretenden abarcar el inicio de dos de los movimientos básicos del bote en el baloncesto: el bote con una mano y el cambio de mano por delante. A partir de estos se puede controlar el bote a diferentes alturas, bote de protección, cambios de mano entre piernas, por la espalda, etc.

A continuación se muestran los resultados de las ejecuciones con el modelo seleccionado y siempre con vídeos distintos a los utilizados para entrenar, con una resolución de 1920x1080 píxeles y a 30 fotogramas por segundo.

7.2.1. Ejercicio 1 contador de botes

Para este "Ejercicio 1 contador de botes" solo se contabiliza el número de botes que realice el jugador independientemente de la mano que use para ello. El procedimiento que se sigue es relativamente sencillo dado un vídeo en el que el jugador ejecuta el movimiento se procesó con el modelo que hemos seleccionado anteriormente. Tras esto fotograma a fotograma se irán detectando los objetos de la imagen. En el código se almacenan las detecciones que hemos obtenido de cada fotograma. Con esto podemos conocer la dirección por ejemplo del objeto

pelota, puesto que, si la coordenada y va aumentado, se entiende que va en dirección ascendente. Además, tomaremos como posición de la pelota el centro del lado inferior del cuadrado que delimita la posición de este, es decir el lado inferior del *bounding box*. Por otro lado, se toma la altura de la mitad de la persona como umbral para contabilizar el bote.

Así con toda esta información podemos realizar un razonamiento simple: si la pelota ha intersectado con la línea horizontal que hemos definido como umbral y la pelota está siguiendo una trayectoria ascendente entonces contabilizamos un bote. De esta manera cuando el jugador impulse el balón hacia el suelo para realizar el bote no se contabiliza porque en este caso la dirección es descendente. A continuación, en las Figura 52, 53, 54 y 55 se muestra una secuencia de este ejercicio:

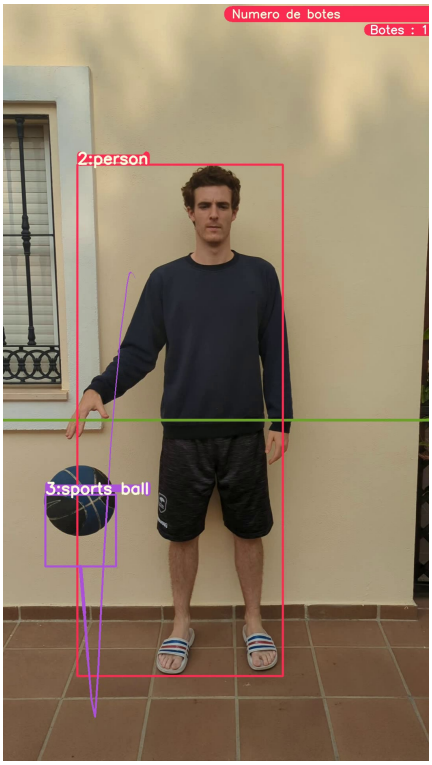

Figura 52: Ejecución Ejercicio 1 contador de botes fotograma 37

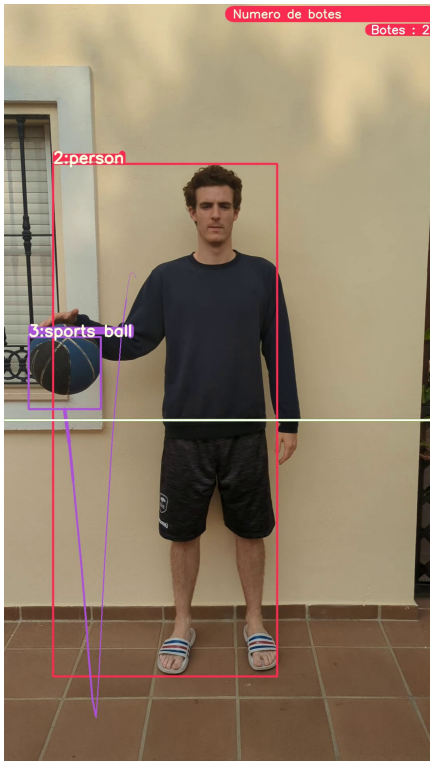

Figura 53: Ejecución Ejercicio 1 contador de botes fotograma 41

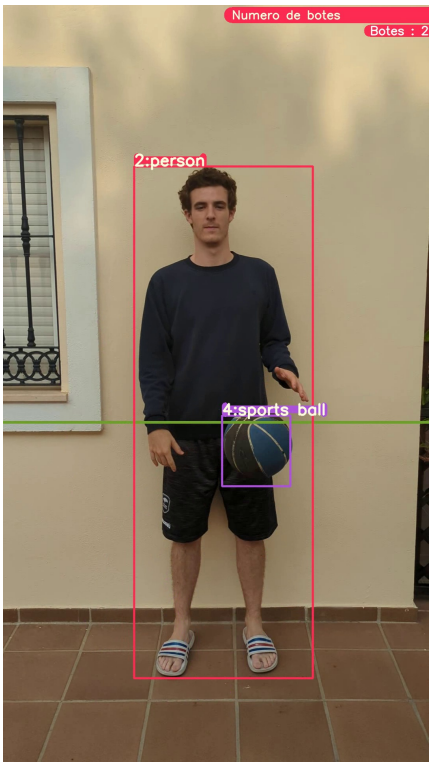

Figura 54: Ejecución Ejercicio 1 contador de botes fotograma 70

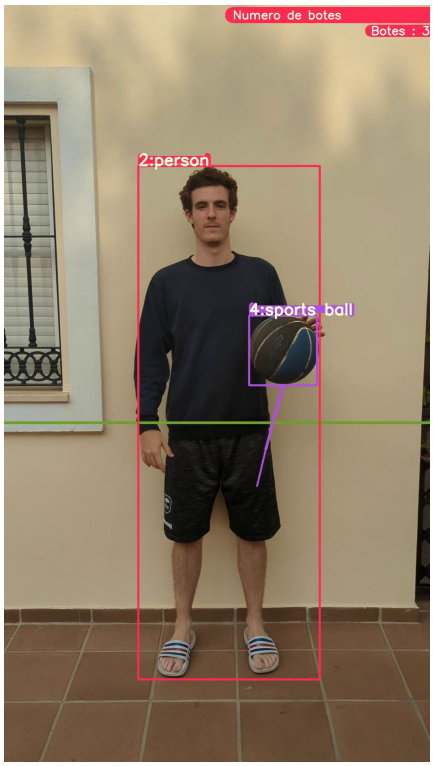

Figura 55: Ejecución Ejercicio 1 contador de botes fotograma 73

En estas imágenes de la ejecución se puede observar en la esquina superior derecha como se van contabilizando los botes cuando se dan las condiciones mencionadas anteriormente. Así podemos obtener algunas métricas, en todos los fotogramas de este vídeo y los siguientes se encuentran los objetos pelota y persona:

Número de fotogramas	76
Número de fotogramas con objeto pelota detectado	70
Número de fotogramas con objeto persona detectado	67
Número de botes reales	2
Número de botes detectados	2

7.2.2. Ejercicio 2 contador botes con la misma mano

El siguiente ejercicio tiene por objetivo que el jugador siempre bote con la misma mano y no realice ningún cambio de mano. Para ello la dinámica es la misma que en el ejercicio anterior pero, controlando que el balón no cambie del lado derecho al izquierdo o viceversa.

Así, si el jugador empieza botando con la mano derecha, en el lado derecho, se contabilizan los botes como en el ejercicio anterior pero ahora se controla que no pase la línea vertical situada en el centro de la anchura de la persona, si lo hace se paraliza la ejecución del ejercicio.

A continuación, en las Figura 56, 57, 58 y 59, se muestra una ejecución correcta del ejercicio junto con las métricas obtenidas:

Número de fotogramas	122
Número de fotogramas con objeto pelota detectado	120
Número de fotogramas con objeto persona detectado	120
Número de botes reales correctos	4
Número de botes detectados correctos	4

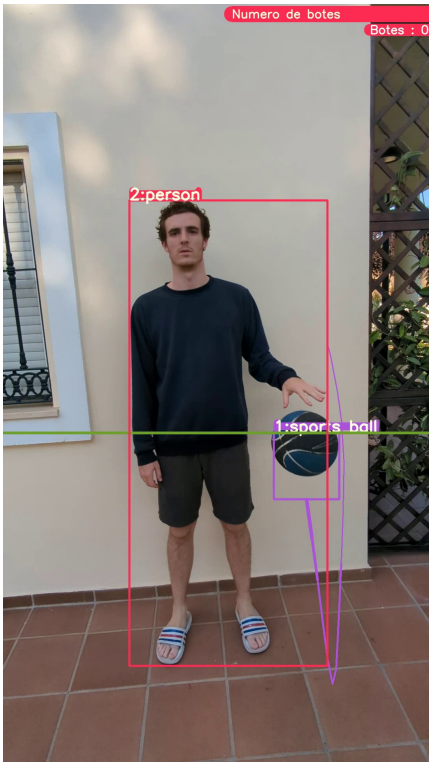

Figura 56: Ejecución Ejercicio 2 contador botes con la misma mano fotograma 17

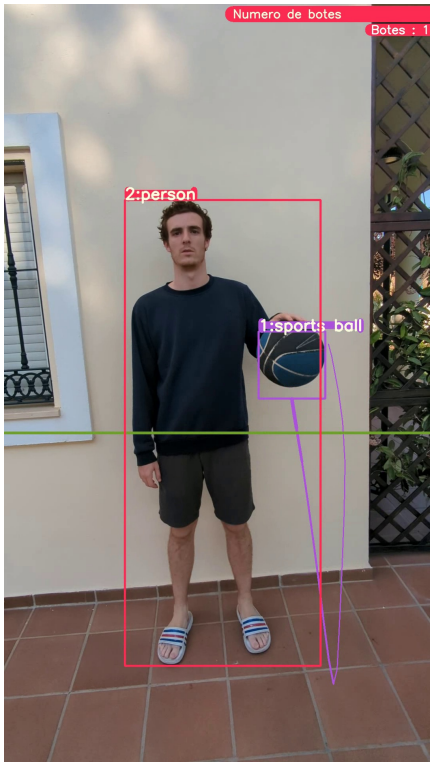

Figura 57: Ejecución Ejercicio 2 contador botes con la misma mano fotograma 20

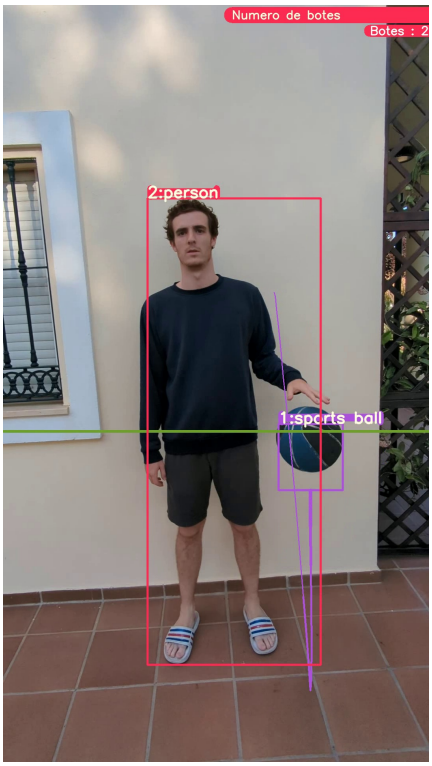

Figura 58: Ejecución Ejercicio 2 contador botes con la misma mano fotograma 110

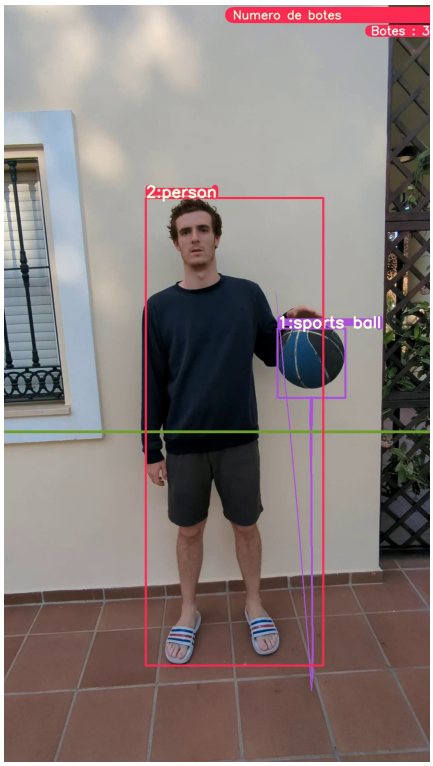

Figura 59: Ejecución Ejercicio 2 contador botes con la misma mano fotograma 113

También se adjunta una ejecución incorrecta del ejercicio, Figuras 60, y la tabla con los resultados obtenidos61.

Número de fotogramas	103
Número de fotogramas con el objeto pelota detectado	71
Número de fotogramas con el objeto persona detectado	99
Número de botes reales correctos	0
Número de botes detectados correctos	0

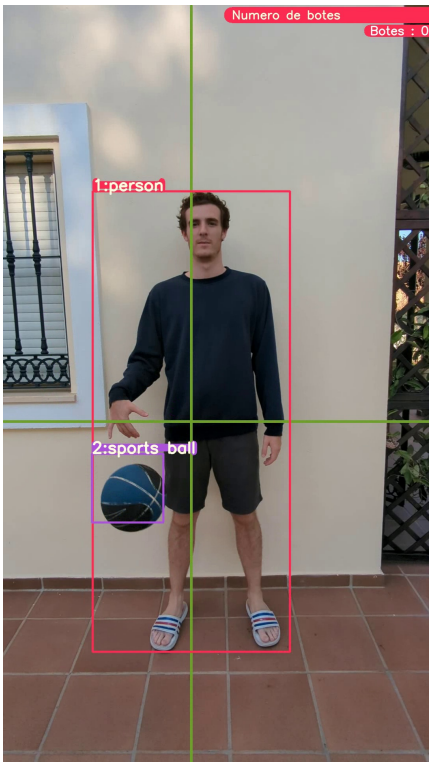

Figura 60: Ejecución incorrecta ejercicio 2 fotograma 7

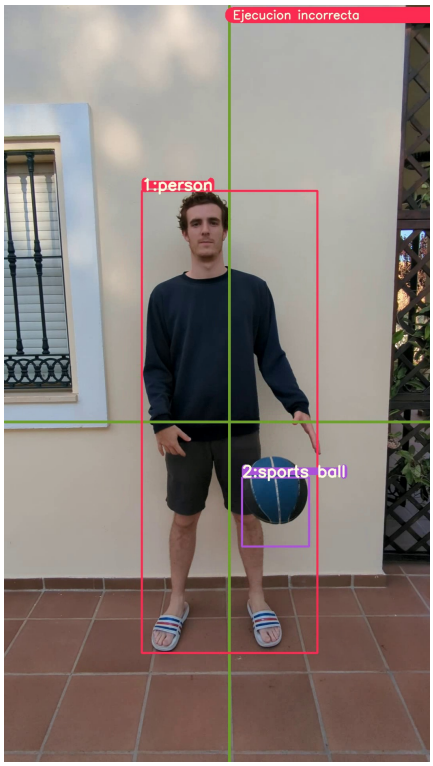

Figura 61: Ejecución incorrecta ejercicio 2 fotograma 16

7.2.3. Ejercicio 3 contador cambios de mano

El Ejercicio 3 contador cambios de mano tiene por objeto controlar que se hagan sucesivos cambios de mano sin poder dar ningún bote intermedio. Es decir, si hay un cambio de mano de la izquierda a la derecha a continuación no puedo dar un bote con la mano derecha. Así se controla que cuando hay un cambio de mano el balón solo puede superar la mitad de la altura de la persona una vez, si lo hace dos veces sin un cambio de mano entre medio entonces la ejecución es incorrecta. Se muestra a continuación una secuencia de imágenes con la ejecución de este ejercicio, Figura 62, 64, 63 y 65

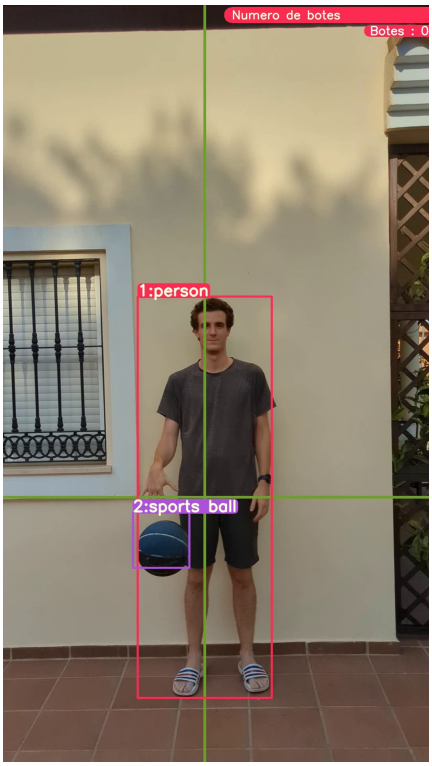

Figura 62: Ejecución Ejercicio 3 contador cambios de mano fotograma 4

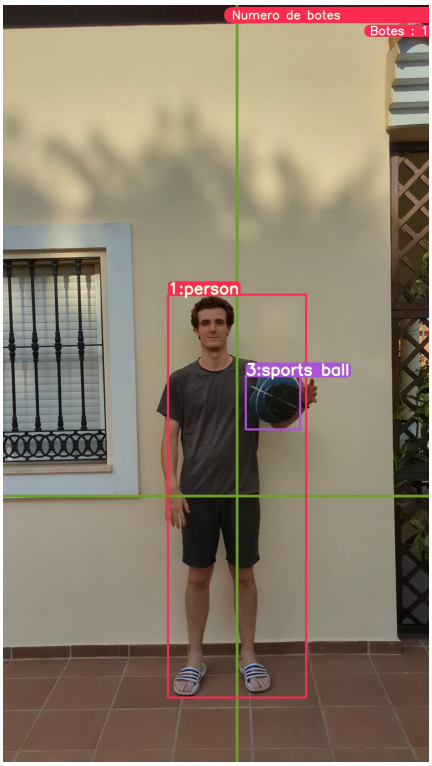

Figura 64: Ejecución Ejercicio 3 contador cambios de mano fotograma 18

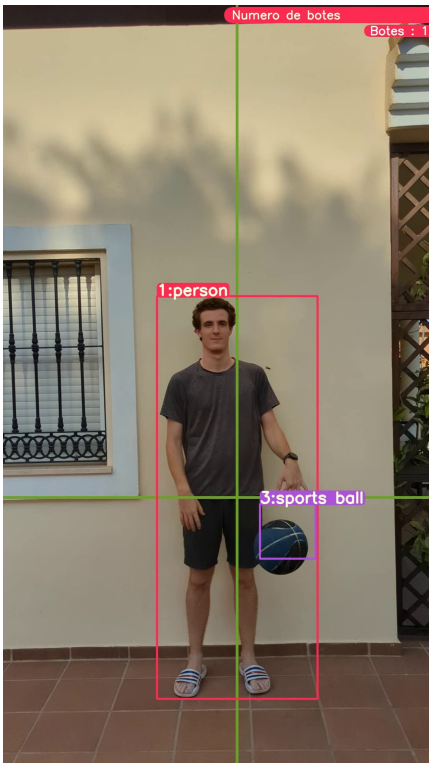

Figura 63: Ejecución Ejercicio 3 contador cambios de mano fotograma 38

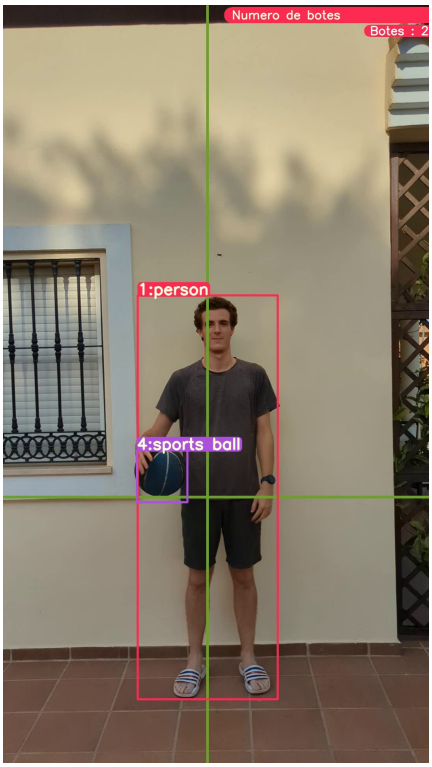

Figura 65: Ejecución Ejercicio 3 contador cambios de mano fotograma 48

Número de fotogramas	114
Número de fotogramas con el objeto pelota detectado	104
Número de fotogramas con el objeto persona detectado	112
Número de cambios de mano reales	3
Número de cambios de mano detectados	3

7.3. Limitaciones

Este modelo tiene una limitación principal, dado que el modelo está entrenado en unas circunstancias y características muy concretas no es extrapolable a fondos o balones muy diferentes al usado. Al comienzo de las pruebas, en los ejercicios llamó la atención del autor como utilizar otro balón distinto disminuía el rendimiento del programa. Lo mismo que con la pelota ocurre con el fondo del vídeo donde se realice el ejercicio. Esto se debe al uso de un modelo entrenado con un número de imágenes limitadas y no muy numerosas ya que fue una tarea lenta y repetitiva en la que no se quiso invertir gran parte del tiempo dado que el resultado no era garantizado.

Otro punto importante que limita el programa es añadir en la ejecución de los ejercicios varias pelotas y personas ya que la detección deja de funcionar correctamente y además no es capaz de realizar un seguimiento suficiente. Al igual que ocurría anteriormente el modelo está entrenado para detectar a una persona con un balón frente a una cámara y cuando añades objetos el modelo comienza a tener un comportamiento algo más aletorio.

8. Conclusiones y Líneas Futuras

8.1. Conclusiones

La elaboración de este trabajo ha llevado a la investigación de varios modelos actuales de detección de objetos en imágenes, como YOLO-v7 y YOLO-v8, y su comportamiento en un ámbito específico como es el baloncesto y el entrenamiento individual. Como resultado se ha visto que los modelos ya preentrenados, aunque muy avanzados, no funcionaban en todas las situaciones y ajustarlos con imágenes de conjuntos de datos públicos, es computacionalmente demasiado costoso y no está al alcance de los medios de este trabajo. Por eso para centrarse en el objeto principal del trabajo se optó por entrenar un modelo propio que si permitiera avanzar debido a su buen funcionamiento con los vídeos que en este trabajo se iban a tratar.

Tras concluir con el modelo que se iba a usar se procedió a adaptar un algoritmo que aunara YOLO-v8 y el algoritmo Deep Sort que permitiera identificar los objetos persona y pelota dentro de un vídeo, realizarles el seguimiento y en función de la posición durante el vídeo poder otorgar al usuario un *feedback* de si el ejercicio está bien ejecutado o no.

Para encontrar un modelo que se ajustara correctamente se realizaron multitud de pruebas. Primero, con la red neuronal YOLO-v7 se testearon los modelos ya preentrenados. Los resultados no fueron del todo malos, pero cuando se procedió a realizar el tracking bajó el rendimiento sustancialmente. Por ello y dado que en el transcurso de las pruebas se lanzó la nueva versión YOLOv-8 se decidió cambiar de red a esta última. Los cambios a priori fueron mejoras, por ejemplo, por la facilidad de uso y calidad del código. Sin embargo, los modelos preentrenados de esta nueva red no proporcionaron una detección suficiente en los objetos, sobre todo en la pelota.

Tras esto se pasó a realizar entrenamientos con conjuntos de datos libres y otros propios. Con los conjuntos de imágenes de COCO y Open Images se realizaron numerosos entrenamientos variando número de épocas y número de imágenes, intentando aumentar este último en beneficio de mejorar el rendimiento. Se llegó a la conclusión de que para entrenar un modelo con estos conjuntos se necesita una capacidad de cómputo y tiempo por encima de los que nos podíamos permitir. Por otro lado, el entrenamiento que realizamos con el conjunto de datos propio dio resultados muy buenos ya que permitía en vídeos que se tomaban en parecidas condiciones detectar la pelota y por ende seguirla durante casi todo el tiempo del vídeo,

permitiendo así evaluar los ejercicios con mayor precisión.

Por consiguiente, una de las conclusiones que hemos obtenido tras la elaboración de este trabajo es la complicada tarea de encontrar un modelo que pueda funcionar en cualquier circunstancia para la detección de objetos en imágenes. La tarea de entrenar un conjunto de datos es una tarea que requiere altos recursos de cómputo. Así, aunque el software y las redes neuronales que se enfocan en la detección de imágenes han realizado muchos avances, el hardware y sus capacidades siguen teniendo un peso importante en el desarrollo que estas tecnologías puedan tener en el futuro.

Con todo ello, tras el desarrollo de este trabajo, se ha conseguido un software que dado un vídeo de entrada permite la evaluación de una serie de ejercicios de baloncesto. El software consiste en la utilización de una Red Neuronal Convolucional, como YOLOv-8, diseñada para la detección de imágenes y de la aplicación del algoritmo Deep Sort para poder realizar el seguimiento de los objetos. Para ello el vídeo de entrada será el de un jugador de baloncesto frente a una cámara con una pelota. Los ejercicios a evaluar son ejercicios relacionados con la técnica individual del baloncesto: contabilizar el número de botes que se realizan, el número de botes siempre que sean con la misma mano y el número de cambios de mano siempre que sean seguidos y no haya botes intermedios.

Por último, esta rama de la informática tiene un potencial asombroso. Son numerosas las noticias que surgen casi diariamente de avances que la Inteligencia Artificial está realizando para mejorar la vida de las personas y en conjunto de la sociedad, desde términos de salud para la detección precoz de distintos cánceres hasta aumentar la productividad de empresas y de tareas cotidianas para poner el foco en la creatividad y las habilidades sociales.

8.2. Líneas Futuras

Dado el poco recorrido de esta rama de investigación, el futuro, además de ser imprevisible, probablemente sea más cercano de los que nos podemos imaginar todos. Por ello algunas líneas futuras que podrían mejorar este trabajo son las siguientes.

8.2.1. Mejorar la detección de objetos

Actualmente el modelo registra un buen rendimiento con los vídeos que en este trabajo se utilizan como prueba. Una de las mejoras sustanciales sería ajustar mejor el modelo para que sea capaz de reconocer la pelota y la persona en un número mayor de situaciones, y así el

modelo pueda ser extrapolable al uso de distintas pelotas de baloncesto, distintas ubicaciones, etc. Esto aumentaría la utilidad práctica de este trabajo.

El mejorar el modelo en cuanto a detección también debe implicar un buen rendimiento en el seguimiento de la pelota. Hemos visto como algunos modelos tenían un resultado distinto a la hora de realizar la detección y el seguimiento por lo que es una mejora que debe ir en sintonía.

8.2.2. Introducir nuevos ejercicios de técnica individual

Actualmente el sistema es capaz de contabilizar los botes que se producen en un vídeo, corregir un ejercicio en el que se bote siempre con la misma mano y evaluar un ejercicio en el que se realizan continuamente cambios de mano. Sin embargo, el número de ejercicios posibles es mucho más amplio y aumentar el abanico de ejercicios podría ser una posible mejora importante para este trabajo.

Por otro lado, aplicar esta técnica a otros gestos técnicos como el pase o el tiro puede ser también una posible futura línea de actuación. Introducir ejercicios que evalúen estos gestos técnicos enriquecería mucho el sistema.

8.2.3. Reconocer acciones

Otro punto de mejora podría ser el de reconocer acciones específicas, por ejemplo, si se ha realizado alguna violación de las contempladas en el reglamento y poder dar el *feedback* necesario al jugador. Otro ejemplo podría ser reconocer los tipos de cambios de mano que se pueden dar: cambio de mano por delante, entre piernas y por la espalda, entre otros.

8.2.4. Ampliar el rango de uso

Este tipo de tecnología puede ser usada también para la evaluación y el seguimiento de la táctica colectiva y la táctica individual y así extraer estadísticas, que sistemas funcionan mejor en el partido o métricas más difíciles de medir para un humano y que las máquinas procesan instantáneamente. Sobre esto ya hay mucho trabajo e investigaciones en los departamentos de las distintas ligas del mundo. Es por esto que podría ser un ámbito en el que trabajar y mejorar el sistema.

8.2.5. Convertirlo en una red social

Las redes sociales a día de hoy son probablemente las aplicaciones a la que más tiempo durante el día le dedica el usuario. Puede ser una mejora obtener métricas de las distintas ejecuciones de los ejercicios y compartirlas con los usuarios para que puedan competir, etc.

Apéndice A
Manual de Instalación

A continuación, se indican los pasos para preparar el entorno en el que posteriormente se ejecutará el código pertinente.

A.1. Preparación del entorno

Antes de comenzar necesitamos, para no tener conflictos con librerías que puedan estar ya instaladas en la máquina, instalar un gestor de entornos. En este caso se usará Conda [27]. Para ello se seguirán los siguientes pasos:

1. Abrir en el navegador [43] y en la sección de *Regular Installation* seguir los pasos de su Sistema Operativo.

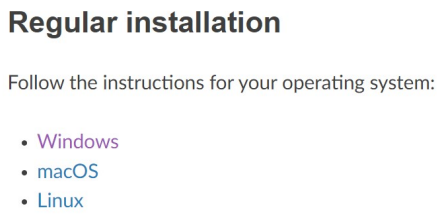

Figura 66: Sección de la web de Conda para instalarlo [43]

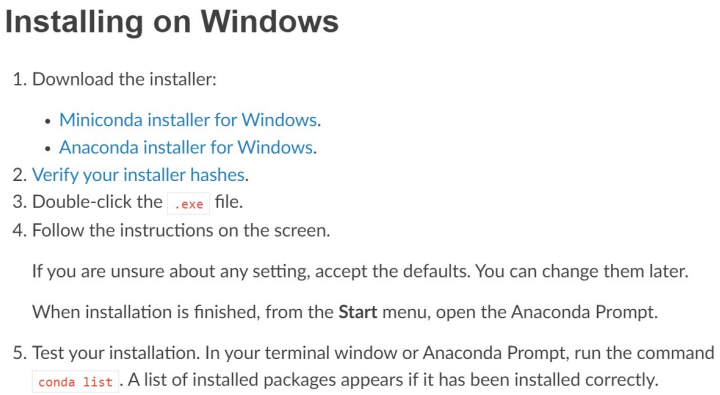

Figura 67: Pasos para la instalación de Conda en Windows

2. A continuación, desde la terminal de "Anaconda prompt" instalada anteriormente navegar hasta la carpeta donde desee copiar el repositorio, en nuestro lo haremos en la carpeta Documentos, para ello ejecutaremos el siguiente comando:

- cd Documents

```
(base) C:\Users\Mario Martínez>cd Documents

(base) C:\Users\Mario Martínez\Documents>
```

Figura 68: Navegar a la carpeta deseada

3. Crear un entorno de trabajo y activarlo:

   - conda create -n NOMBRE_ENTORNO python=3.9

   - conda activate NOMBRE_ENTORNO

```
(base) C:\Users\Mario Martínez\Documents>conda create -n NOMBRE
Collecting package metadata (current_repodata.json): done
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 22.9.0
  latest version: 23.5.2

Please update conda by running

    $ conda update -n base -c defaults conda



## Package Plan ##

  environment location: D:\Intalaciones\Anaconda\envs\NOMBRE



Proceed ([y]/n)? y

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate NOMBRE
#
# To deactivate an active environment, use
#
#     $ conda deactivate

Retrieving notices: ...working... done

(base) C:\Users\Mario Martínez\Documents>conda activate NOMBRE

(NOMBRE) C:\Users\Mario Martínez\Documents>
```

Figura 69: Navegar a la carpeta deseada

4. Clonar el repositorio:

   - git clone https://github.com/Mario31y/YOLOv8DeepSort-Basketball-Training.git

5. Instalar las librerias necesarias:

- pip install -r requirements.txt

- pip install ultralytics

Tras estos pasos ya tenemos creado el entorno de trabajo en el que vamos a instalar las librerías necesarias y que no haya conflictos con las que tengamos en nuestra máquina local. Además, hemos clonado el repositorio donde se encuentran todos los archivos python necesarios para poder ejecutar el resultado de este trabajo.

Apéndice B
Manual de Ejecución

B.1. Ejecución

Se recomienda haber completado con éxito los pasos del "Manual de Instalación" A.

1. Abrir "Anaconda Prompt"
2. Navegar hasta la carpeta donde hayamos clonado el repositorio, en nuestro caso Docuemtos:

   - cd Documents

   - cd YOLOv8DeepSort-Basketball-Training
3. Navegar hasta la carpeta donde están los scripts de python

   - cd yolov\v8\detect
4. Ejecutar los scripts con el modelo y los vídeos que queramos usar

   Encontramos 3 modelos para utilizar el que mejor se ajuste:

   - yolov8l.pt
   - train001.pt
   - train001l.pt

   Además en la carpeta Videos encontramos varios vídeos para usar de ejemplo

   Si solo queremos contabilizar los botes:

```
YOLOv8-Object-Detection-with-DeepSORT-Tracking\yolo\v8>python tracking_ball_person_counting.py model=yolov8l.pt source="Videos\video8.mp4" show=True
```

Figura 70: Comando para la ejecución en Windows

   Si queremos contabilizar solo los botes que se dan con la misma mano:

```
YOLOv8-Object-Detection-with-DeepSORT-Tracking\yolo\v8>python tracking_ball_person_counting1.py model=yolov8l.pt source="Videos\video9.mp4" show=True
```

Figura 71: Comando para la ejecución en Windows

   Si queremos contabilizar solo los cambios de mano

Figura 72: Comando para la ejecución en Windows

Referencias

[1] Ehran Khan. *Advanced NBA Stats for Dummies: How to Understand the New Hoops Math.* en. URL: https://bleacherreport.com/articles/1813902-advanced-nba-stats-for-dummies-how-to-understand-the-new-hoops-math (visitado 08-07-2023).

[2] *Calculating PER.* en. URL: https://www.basketball-reference.com/about/per.html (visitado 08-07-2023).

[3] *LaLiga y Microsoft presentan Beyond Stats, un proyecto de análisis futbolístico avanzado que profundiza en el juego de cada equipo – Centro de noticias.* URL: https://news.microsoft.com/es-es/2021/10/06/laliga-y-microsoft-presentan-beyond-stats-un-proyecto-de-analisis-futbolistico-avanzado-que-profundiza-en-el-juego-de-cada-equipo/ (visitado 08-07-2023).

[4] *Mediacoach | La solución visual completa para profesionales del fútbol.* URL: https://www.mediacoach.es/ (visitado 08-07-2023).

[5] The Info Show. *Revolutionizing Sports NBA Launches New AI Simulator Scan Yourself and Play as Any Player in the Gam.* Feb. de 2023. URL: https://www.youtube.com/watch?v=ObzdIYSdPRE (visitado 08-07-2023).

[6] Cristóbal Fuentes Barassi. "1. El juego de la imitación." es. En: ().

[7] https://www.innovaciondigital360.com/periodista/equipo-editorial. *Test de Turing: qué es, cómo funciona y qué robots lo superan |.* Ene. de 2023. URL: https://www.innovaciondigital360.com/i-a/test-de-turing-que-es-como-funciona-y-que-robots-lo-superan/ (visitado 19-07-2023).

[8] *Historia de la inteligencia artificial.* es. Page Version ID: 152350889. Jul. de 2023. URL: https://es.wikipedia.org/w/index.php?title=Historia_de_la_inteligencia_artificial&oldid=152350889 (visitado 09-07-2023).

[9] *El ordenador 'Watson' gana la partida al cerebro humano.* es. Section: Aplicaciones. Feb. de 2011. URL: https://www.lavanguardia.com/tecnologia/aplicaciones/20110217/54116621022/el-ordenador-watson-gana-la-partida-al-cerebro-humano.html (visitado 09-07-2023).

[10] *Un ordenador logra superar por primera vez el test de Turing*. es. Section: ciencia. Jun. de 2014. URL: https://www.elmundo.es/ciencia/2014/06/09/539589ee268e3e096c8b4584.html (visitado 09-07-2023).

[11] El País. "La inteligencia artificial gana al campeón humano de Go". es. En: *El País* (mar. de 2016). ISSN: 1134-6582. URL: https://elpais.com/elpais/2016/03/15/ciencia/1458034897_194344.html (visitado 09-07-2023).

[12] Team rédac. *Inteligencia artificial : definición, historia, usos, peligros*. es. Ago. de 2022. URL: https://datascientest.com/es/inteligencia-artificial-definicion (visitado 09-07-2023).

[13] Team rédac. *Machine Learning: definición, funcionamiento, usos*. es. Dic. de 2021. URL: https://datascientest.com/es/machine-learning-definicion-funcionamiento-usos (visitado 09-07-2023).

[14] *¿Qué es Deep Learning? | IBM*. es-es. URL: https://www.ibm.com/es-es/topics/deep-learning (visitado 09-07-2023).

[15] *Introducción a las Redes Neuronales Pt. I*. es. Jun. de 2019. URL: https://futurelab.mx/redes%20neuronales/inteligencia%20artificial/2019/06/25/intro-a-redes-neuronales-pt-1/futurelab.mx/redes%20neuronales/inteligencia%20artificial/2019/06/25/intro-a-redes-neuronales-pt-1/ (visitado 09-07-2023).

[16] ROCÍO GIL GRANDE (@pepi_rocio). *Primer mapa en 3D de la red neuronal del cerebro*. es. Section: NOTICIAS. Sep. de 2016. URL: https://www.rtve.es/noticias/20160923/primer-mapa-3d-red-neuronal-del-cerebro-revela-estructuras-nunca-vistas/1408922.shtml (visitado 09-07-2023).

[17] Ligdi González. *¿Qué es el Perceptrón? Perceptrón Simple y Multicapa*. es. Oct. de 2021. URL: https://aprendeia.com/que-es-el-perceptron-simple-y-multicapa/ (visitado 09-07-2023).

[18] *¿Qué es una red neuronal?* es. URL: https://www.tibco.com/es/reference-center/what-is-a-neural-network (visitado 09-07-2023).

[19] *¿Qué son las redes neuronales convolucionales? | IBM*. es-es. URL: https://www.ibm.com/es-es/topics/convolutional-neural-networks (visitado 09-07-2023).

[20] Pascual Parada Torralba. “Qué son las Redes Neuronales Convolucionales”. es. En: *Thinking for Innovation* (sep. de 2022). URL: https://www.iebschool.com/blog/redes-neuronales-convolucionales-big-data/ (visitado 10-07-2023).

[21] Anh H. Reynolds. *Anh H. Reynolds*. en-us. URL: https://anhreynolds.com/ (visitado 10-07-2023).

[22] Ross Girshick et al. *Rich feature hierarchies for accurate object detection and semantic segmentation*. 2014. arXiv: 1311.2524 [cs.CV].

[23] Joseph Redmon et al. “You Only Look Once: Unified, Real-Time Object Detection”. en. En: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, jun. de 2016, págs. 779-788. ISBN: 978-1-4673-8851-1. DOI: 10.1109/CVPR.2016.91. URL: http://ieeexplore.ieee.org/document/7780460/ (visitado 10-07-2023).

[24] *Figure 1. Timeline of You Only Look Once (YOLO) variants*. en. URL: https://www.researchgate.net/figure/Timeline-of-You-Only-Look-Once-YOLO-variants_fig1_370153499 (visitado 10-07-2023).

[25] Glenn Jocher, Ayush Chaurasia y Jing Qiu. *Ultralytics YOLOv8*. Ver. 8.0.0. 2023. URL: https://github.com/ultralytics/ultralytics.

[26] *Qué es Python: Características, evolución y futuro*. es. Sep. de 2019. URL: https://openwebinars.net/blog/que-es-python/ (visitado 30-05-2022).

[27] *Conda — conda documentation*. URL: https://docs.conda.io/en/latest/ (visitado 10-07-2023).

[28] *Roboflow: Give your software the power to see objects in images and video*. en. URL: https://roboflow.com/ (visitado 10-07-2023).

[29] *COCO - Common Objects in Context*. URL: https://cocodataset.org/#home (visitado 11-07-2023).

[30] *CVDF - Common Visual Data Foundation*. URL: http://www.cvdfoundation.org/ (visitado 11-07-2023).

[31] *Open Images V7*. URL: https://storage.googleapis.com/openimages/web/index.html (visitado 11-07-2023).

[32] *¿Qué es la metodología ágil?* es. URL: https://www.redhat.com/es/devops/what-is-agile-methodology (visitado 17-07-2023).

[33] Sandra Garrido Sotomayor. "Las metodologías ágiles más utilizadas y sus ventajas dentro de la empresa". es. En: *Thinking for Innovation* (dic. de 2021). URL: https://www.iebschool.com/blog/que-son-metodologias-agiles-agile-scrum/ (visitado 17-07-2023).

[34] *scientific method | Definition of scientific method in English by Oxford Dictionaries*. Mar. de 2019. URL: https://web.archive.org/web/20190321224044/https://en.oxforddictionaries.com/definition/scientific_method (visitado 17-07-2023).

[35] *Método científico*. es. Page Version ID: 152484691. Jul. de 2023. URL: https://es.wikipedia.org/w/index.php?title=M%C3%A9todo_cient%C3%ADfico&oldid=152484691 (visitado 17-07-2023).

[36] Kin-Yiu Wong. *Official YOLOv7*. original-date: 2022-07-06T15:14:06Z. Jul. de 2023. URL: https://github.com/WongKinYiu/yolov7 (visitado 11-07-2023).

[37] Mahesh Deshwal. *Welcome!* original-date: 2022-07-13T10:30:51Z. Jul. de 2023. URL: https://github.com/deshwalmahesh/yolov7-deepsort-tracking (visitado 11-07-2023).

[38] Jacob Marks Ph.D. *Giving YOLOv8 a Second Look (Part 3)*. en. Mar. de 2023. URL: https://medium.com/voxel51/giving-yolov8-a-second-look-part-3-bf9b7e4e8e99 (visitado 12-07-2023).

[39] *fix nan/inf loss by Laughing-q · Pull Request #490 · ultralytics/ultralytics*. en. URL: https://github.com/ultralytics/ultralytics/pull/490 (visitado 13-07-2023).

[40] *nan report in box_class cls_class and dfl_loss when train custom dataset · Issue #280 · ultralytics/ultralytics*. en. URL: https://github.com/ultralytics/ultralytics/issues/280 (visitado 13-07-2023).

[41] Ritesh Kanjee. *DeepSORT — Deep Learning applied to Object Tracking*. en. Mayo de 2022. URL: https://medium.com/augmented-startups/deepsort-deep-learning-applied-to-object-tracking-924f59f99104 (visitado 13-07-2023).

[42] *Filtro de Kalman*. es. Page Version ID: 151174191. Mayo de 2023. URL: https://es.wikipedia.org/w/index.php?title=Filtro_de_Kalman&oldid=151174191 (visitado 13-07-2023).

[43] *Installation — conda 23.5.3.dev77 documentation.* URL: https://conda.io/projects/conda/en/latest/user-guide/install/ (visitado 15-07-2023).

UNIVERSIDAD DE MÁLAGA | uma.es

E.T.S de Ingeniería Informática
Bulevar Louis Pasteur, 35
Campus de Teatinos
29071 Málaga

E.T.S. DE INGENIERÍA INFORMÁTICA