UNIVERSIDAD DEL BÍO-BÍO
FACULTAD DE CIENCIAS EMPRESARIALES
DEPARTAMENTO DE SISTEMAS DE INFORMACIÓN

DETECCIÓN DE MADUREZ EN ARÁNDANOS A PARTIR DE MODELOS DE APRENDIZAJE PROFUNDO ENTRENADOS CON IMÁGENES AÉREAS DE CULTIVOS

PROYECTO DE TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN CIENCIAS DE LA COMPUTACIÓN

SEBASTIÁN ESPINOZA SILVA

PROFESOR GUÍA:
Cristhian Aguilera
PROFESOR CO-GUÍA:
Pedro Campos

CONCEPCIÓN, CHILE
2024

RESUMEN

El sector agropecuario se está modernizando, al igual que otros sectores industriales, se implementan cada vez más tecnologías a sus procesos para acelerar la producción y mejorar la calidad de sus productos. Específicamente el subsector frutícola tiene muchas tareas que aún se realizan manualmente, como las evaluaciones de madurez o enfermedades, por lo que cada vez se publican más trabajos y herramientas para realizar estas tareas de forma más eficiente y optimizar la producción del cultivo, ahorrando tiempo y asegurando la calidad del mismo. En los últimos años, los modelos de Aprendizaje Profundo han ganado mucha popularidad en distintos ámbitos. Uno de ellos es la Visión Computacional, donde tecnologías como los modelos de Redes Neuronales Convolucionales y Visual Transformers permiten analizar imágenes digitales con un gran nivel de precisión. Con esas herramientas, algunas tareas que antes se realizaban exclusivamente manual, ahora se están evaluando para realizarse automática o semiautomáticamente. Una de estas tareas es el análisis y monitorización de frutos, que es especialmente importante para lograr una buena calidad del producto final, más aún en Chile, que exporta una gran cantidad de fruta al año. En este trabajo se presenta un caso de estudio de Visión Computacional con modelos de Aprendizaje Profundo sobre frutos arándanos. Para ello se realiza una detallada revisión sistemática de literatura, en busca de las principales tendencias de la tarea de clasificación en frutos, describiendo modelos frecuentes, métricas más utilizadas y detalles de los conjuntos de datos. Además, se realiza el entrenamiento sistemático de los modelos más relevantes actualmente en el área. De este proceso de experimentación se obtiene que el modelo YOLOv8 es el mejor modelo para detectar arándanos y clasificar su madurez con un mAP del 86 % que está en línea con otros estudios en la literatura incluso superior. Además, se repasan otras herramientas novedosas como los Visual Transformers, comparando su eficiencia y uso con los modelos CNN clásicos.

Tabla de Contenido

Índice de Tablas

Índice de Ilustraciones

Capítulo 1

Introducción

El sector agropecuario en Chile es una actividad antigua que ha acompañado al país desde sus inicios y ha sido un pilar importante en el desarrollo del mismo. Sin ir más lejos, las exportaciones del sector acumulan unos 6428 millones de dólares. Los principales subsectores que conforman estas exportaciones son las Cerezas frescas (US$ millones), Uvas frescas (US$ 796 millones), Arándanos Frescos (US$ 332 millones) entre otros [1]. Gracias a ello, Chile se ha posicionado como uno de los principales productores y exportadores frutas del hemisferio sur, con más de 50 especies frutales diferentes, destacando principalmente las cerezas, uvas, nueces y arándanos [2]. Este último es considerado un súper alimento por sus nutrientes y capacidad antioxidante debido a la gran concentración de antocianinas en el fruto [3], lo que le da además ese característico color azul que lo protege de los rayos ultravioletas del sol.

Gracias a sus propiedades, el arándano ha acrecentado su demanda, principalmente en el mercado estadounidense. Según el reporte anual de la IBO (International Blueberry Organization) de 2022, Chile es el segundo exportador más grande de arándanos de Sudamérica con 106 MT (miles de toneladas métricas) de arándanos entre 2021 y 2022, siendo Estados Unidos (56.89 MT), Países Bajos (24.28 MT) y Reino Unido (7.43 MT) sus principales importadores [4].

Por otro lado, el reporte voluntario publicado por el Departamento de Agricultura de Estados Unidos [5], indica que el área plantada de arándanos paso de 7000 hectáreas (ha) en 2011 a 18.000 ha, siendo las regiones del Maule, Ñuble, La Araucanía y Biobío las que acumulan más hectáreas plantadas del país. También se indica que la producción y exportación es una opción rentable para los fruticultores del centro-sur del país, pero que en los últimos tres años ha tenido ciertas dificultades debido a, entre otros factores, la competencia de países vecinos, incrementos en el costo del transporte, algunos requerimientos de fumigación impuestos por Estados Unidos y las desfavorables condiciones climáticas que ha tenido el país en el último periodo, principalmente la sequía y las altas temperaturas.

Aun así, bajo circunstancias desfavorables la exportación de arándanos solo disminuyó un 9 % en 2022 respecto al periodo anterior, esto gracias a la logística que existe en los envíos a largas distancias. Estos desafíos tampoco han impedido la plantación de variedades de arándanos de nueva generación, que hasta el momento se estima que constituyen un 10 % de la producción del país. La actualización completa a estas nuevas variedades podría tomar 10 años, pero la calidad del suelo nacional y la integración de operaciones I+D podría mejorar esa estimación [4].

En este aspecto, existen varias líneas de integración de las nuevas tecnologías para mejorar los procesos de cultivo. Uno de ellos es la monitorización mediante sensores, como sensores

de humedad en suelo, temperatura o salinidad que entregan datos en tiempo real y permiten mejorar los procesos de riego y control de suelo [6]. Por otro lado, está la monitorización directa del fruto en cuestión mediante imágenes o vídeos. Estos datos son procesados para automatizar el análisis de aspectos más cualitativos del fruto, como el tamaño, color y nivel de madurez.

En línea con lo mencionado anteriormente, es que se hace necesario utilizar todas las tecnologías disponibles para mejorar la producción y calidad del fruto. En este aspecto, la clasificación de imágenes tiene un gran potencial de automatización y control del cultivo. En la última década se han desarrollado y probado una gran variedad de modelos de Aprendizaje Profundo especializados en la clasificación de imágenes de todo tipo, entre ellas de frutos. Las publicaciones en esta área han crecido en los últimos años, y los modelos más utilizados para clasificación de frutos, como VGG16, Resnet50 y otros más especializados para la detección de frutos, como Fast R-CNN, Mask R-CNN y YOLO han demostrado tener un buen rendimiento en las respectivas investigaciones donde se implementan.

A pesar de su gran potencial, estos modelos deben ser adaptados específicamente para cada contexto de clasificación, por lo que se hace necesario realizar una investigación de la efectividad de implementar estas tecnologías a las plantaciones de arándanos, en términos del rendimiento de los modelos.

Específicamente para este proyecto de tesis, se presenta un caso de análisis de madurez de arándanos, esto mediante imágenes digitales utilizando un ángulo de captura aéreo lateral del fruto, por lo que se plantea evaluar los modelos adecuados revisando la literatura al respecto y realizando las correspondientes modificaciones y ajustes para el contexto del problema. Esto podría contribuir al conocimiento científico y técnico en el campo de la detección de madurez en arándanos usando técnicas de Aprendizaje Profundo y análisis de imágenes. Por otro lado, esta investigación se enmarca en un proyecto Fondef por lo que resalta una necesidad de la industria frutícola chilena.

En el siguiente capítulo se plantan en detalle todos los aspectos de este Proyecto de Tesis, desde el problema y el contexto donde se enmarca, las hipótesis planteadas y la metodología de trabajo. En el Capítulo 3 se presentan los conceptos preliminares que se abordan en esta propuesta de tesis, seguido de eso, en el Capítulo 4 del Estado del Arte, se presenta en detalle los principales hallazgos de una Revisión Sistemática de Literatura realizada para conocer las tendencias actuales de las tareas de detección. En el Capítulo 5 se presenta el proceso realizado para la preparación de materiales, más específicamente de los conjuntos de datos y modelos. En el Capítulo 6 se presenta en detalle todo el proceso de experimentación realizada para abordar los objetivos e hipótesis. Finalmente, en las Conclusiones se resalta nuevamente la importancia de la realización de esta investigación y se verifica el cumplimiento de los objetivos propuestos.

Capítulo 2

Planteamiento del Proyecto de Tesis

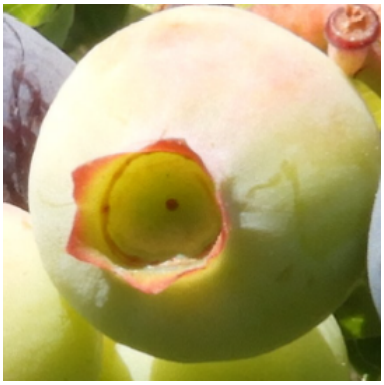
2.1. Problema

El análisis de cultivos de frutos es una tarea común en la agricultura y define en gran medida la calidad de producto final en términos de crecimiento, estado y maduración. En este proceso se analizan varios aspectos como el tamaño, el color, sabor, textura, etc. Este análisis por lo general es manual y muchos de estos aspectos se miden directamente probando la fruta [7], algo que se vuelve complicado realizarlo sobre todos los cultivos, por lo que siempre existe algún grado de error o baja de calidad no medida una vez recolectado el fruto. Además, se debe considerar el especial grado de supervisión que requieren algunos frutos o variantes del mismo.

A esto, se pueden agregar las condiciones climáticas desfavorables que ha tenido Chile en los últimos años, relacionado con los bruscos cambios de temperaturas y la sequía [8], lo que acrecienta la necesidad de monitorizar el crecimiento del cultivo.

Para el caso particular de este proyecto, el fruto se encuentra en su estado natural, aún en la planta, como se muestra en la Figura 2.1. Este puede encontrarse al descubierto u obstruido por hojas y otros arándanos. Además, posee distintos colores dependiendo del nivel de maduración, verde claro translúcido para un fruto inmaduro (Figura 2.2.a), rojo rosado en la etapa anterior a la madurez (Figura 2.2.b) y luego un azul plateado completo (Figura 2.2.c), para el fruto maduro [9].

Figura 2.1: Arándanos en estado natural



(a) Fruto inmaduro, verde claro

(b) Fruto pre-maduración, rojo rosado

(c) Fruto maduro, azul plateado

Figura 2.2: Etapas de maduración del arándano

Las muestras recolectadas como parte del proyecto Fondef considera imágenes digitales, tomadas con un ángulo de captura aéreo lateral, con el fin de adaptar un modelo de inteligencia artificial que pueda ser utilizado por un dron que navega por el campo con el mismo ángulo de captura. Una ilustración de lo mencionado se puede ver en la Figura 2.3.

Figura 2.3: Acción esperada del dron en la plantación de arándanos

Para este caso, el análisis de frutas automático basado en imágenes parece ser una buena opción para mejorar la monitorización del fruto. Sin embargo, no está libre de ciertas dificultades.

1. Los modelos más utilizados para este tipo de tareas son las Redes Neuronales Convolucionales, las cuales han demostrado en los últimos años ser bastante eficientes, pero que requieren ser adaptadas al contexto de análisis. Esto debido a que un modelo entrenado

para analizar un fruto en específico muy probablemente no funcione para otra especie. Además, estos modelos tienden a tener un alto costo computacional en el entrenamiento.

2. Las imágenes del fruto, en este caso particular el arándano, influyen en el rendimiento de los modelos. Características como el color, ángulo de captura, resolución e iluminación afectan en mayor o menor medida a la precisión del modelo, por lo que se deben tener en cuenta a la hora de seleccionar un conjunto de modelos con los que experimentar y las modificaciones que se deban realizar.

Aun así, existe una gran variedad de herramientas disponibles para experimentar con una gran variedad de modelos, algunos de los cuales están preentrenados, lo que aporta considerablemente a reducir los costos computacionales del entrenamiento y el conocimiento de los mismos.

2.2. Hipótesis

Las hipótesis para este proyecto se definen como:

- **Hipótesis H1:** Los modelos de Aprendizaje Profundo entrenados específicamente para el reconocimiento de arándanos mediante imágenes aéreas de plantaciones demostrarán ser eficientes en la identificación de madurez del arándano, mejorando en comparación con los métodos de identificación tradicionales, medido por la tasa de aciertos en la clasificación de los arándanos en categorías de madurez.
- **Hipótesis H2:** La implementación de modelos de Aprendizaje Profundo entrenados para detectar madurez en arándanos usando imágenes de plantaciones permitirá reducir el tiempo requerido para el proceso de monitorización del cultivo y ampliar la extensión del mismo en comparación con los métodos de monitoreo tradicionales utilizados actualmente. Se medirá el tiempo de predicción de una muestra de arándanos contra una estimación de tiempo de identificación tradicional para la misma muestra.

2.3. Objetivos

Objetivo General

Permitir la monitorización automática de la madurez de los arándanos empleando modelos viables y eficaces de aprendizaje profundo sobre imágenes de cultivos con ángulo de captura aéreo lateral.

Objetivos Específicos

- **OE1:** Identificar modelos potencialmente efectivos en la literatura, además de herramientas y técnicas útiles para mejorar el rendimiento
- **OE2:** Implementar y adaptar los modelos identificados para la detección de maduración de imágenes de arándanos
- **OE3:** Evaluar y comparar los modelos con métricas de evaluación correspondientes al caso de estudio

2.4. Alcance de la Investigación

Esta tesis se enmarca en un proyecto Fondef (ID21|10256) el cual genera y provee los datos que se utilizarán para la etapa de experimentación. Los datos consisten en imágenes de plantaciones capturadas desde un plano aéreo lateral. Los conjuntos de datos públicos que se utilicen para mejorar el entrenamiento deben tener características similares, principalmente el ángulo de captura y el mismo fruto.

Los modelos implementados son evaluados principalmente con métricas de tareas de detección, para identificar los que mejor se comportan sobre el contexto de los datos.

2.5. Metodología de Trabajo

La metodología de trabajo de esta tesis consta de cinco etapas. El detalle de la metodología y el trabajo a realizar de cada etapa se presenta a continuación.

2.5.1. Revisión Sistemática de Literatura

En línea con el objetivo específico OE1, se realizará una revisión exhaustiva de la literatura científica relacionada con la detección de madurez de arándanos y el uso de modelos de Deep Learning en el análisis de imágenes agrícolas con el fin de responder a las preguntas presentadas en la Sección 4.1. La metodología a utilizar para la revisión se basa en las directrices definidas en las pautas para realizar revisiones sistemáticas de literatura en Ingeniería de Software [10] y los procedimientos para realizar revisiones sistemáticas de literatura [11]. En términos generales la revisión se separa en las siguientes tres etapas:

- **Planificación de la revisión:** Definición de preguntas y objetivos de la revisión
- **Realización de la revisión:** Identificación, filtración y selección de los estudios
- **Reporte de la revisión:** Síntesis y presentación de los datos obtenidos en la lectura.

Además, debido a la gran cantidad de artículos a revisar, se utilizará una base de datos diseñada para esta actividad. Con esto, se puede almacenar información de los artículos de forma útil para recoger estadísticas y otros datos interesantes. La estructura de la base de datos sigue el diseño del modelo relacionar mostrado en la Figura 2.4.

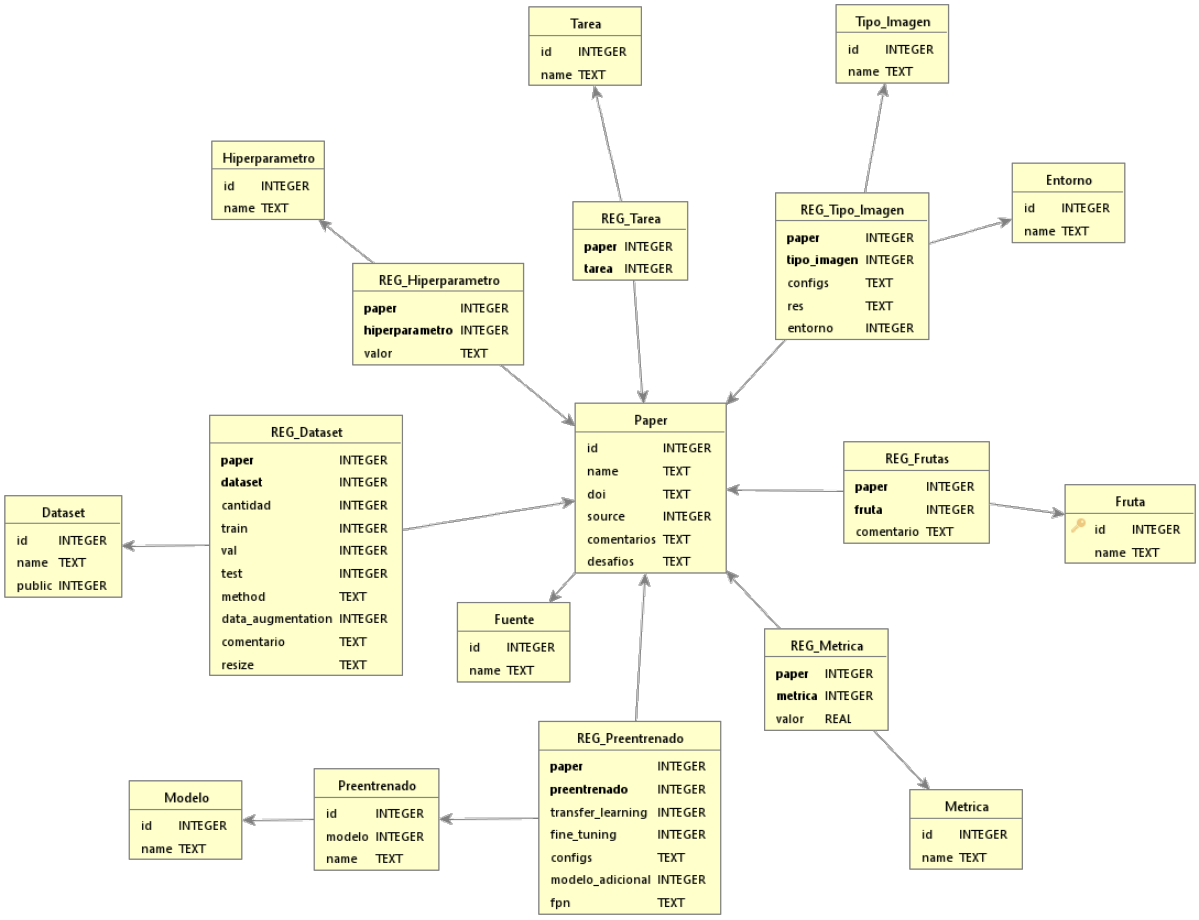


Figura 2.4: Modelo Relacional de la base de datos a utilizar en la Revisión Sistemática de Literatura

Para realizar las estadísticas, se utiliza el Lenguaje R junto a librerías como “ggplot2”, “tidyverse”, “tibble” y “rpart” que hacen más fácil la representación visual de los datos.

2.5.2. Preparación de materiales

Preparar un conjunto de datos suficiente para realizar los experimentos. Las muestras entregadas por el proyecto Fondef tienen un plano de captura aéreo. En caso de necesitar reetiquetar algunas imágenes, se utilizará el software Roboflow [12] que también permite exportar distintos formatos de entrada para los modelos a entrenar.

Por otro lado, se seleccionarán los modelos de Aprendizaje Profundo más adecuados para el análisis de imágenes y la detección de arándanos basados en los resultados de la Revisión Sistemática de Literatura. Además, se prepararán los modelos con sus correspondientes modificaciones e hiperparametrización con el objetivo de obtener el mayor rendimiento posible en el contexto de análisis.

2.5.3. Implementación y evaluación de modelos de Aprendizaje Profundo

En línea con los objetivos específicos OE2 y OE3, para todo el proceso de experimentación con los modelos de Aprendizaje Profundo, se utilizará la metodología de prototipado

evolutivo. Los modelos de Aprendizaje Profundo requieren de mucha experimentación del tipo prueba y error, especialmente para identificar los hiperparámetros ideales, por lo que el constante ciclo de creación de prototipos y retroalimentación de los profesores guía es indispensable.

El flujo de trabajo común en este tipo de implementaciones es explicado a continuación:

1. Separación del conjunto de datos original en el conjunto de entrenamiento, validación y prueba. Comúnmente se le asigna un mayor porcentaje de muestras al conjunto de entrenamiento, de esta forma el modelo tiene una mayor base para aprender patrones.

2. El conjunto de entrenamiento entra en el modelo y comienza la etapa de entrenamiento/aprendizaje del modelo.

3. En el transcurso del entrenamiento el modelo es evaluado para verificar que el proceso se está dando de forma óptima. El modelo ajusta sus parámetros para reducir la perdida.

4. Terminado el entrenamiento, el modelo final es evaluado una última ves con el conjunto de validación o de pruebas.

Para el desarrollo de algoritmos se contempla principalmente el uso del lenguaje Python [13], junto a las librerías más utilizadas para la Ciencia de Datos, como Pandas [14], Numpy [15] y Matplotlib [16]. Para la implementación de los modelos de Aprendizaje Profundo se utilizarán tanto Tensorflow 2 (junto a Keras) [17] [18] como PyTorch [19], que son librearías ampliamente utilizadas para este contexto.

En el caso de los modelos de Aprendizaje Profundo que se contemplan utilizar están:

- Convolutional Neural Networks (CNN) para detección: Los modelos clásicos de CNN con las mejoras de detección de regiones como Fast R-CNN y Faster R-CNN. También se contempla el uso YOLO en sus versiones YOLOv3, YOLOv4, YOLOv5, etc. Otros modelos que serán útiles en conjunto con los anteriores mencionados son VGG16 y ResNet.

- Visual Transformers: Con el fin de utilizar también las tecnologías más recientes se contempla el uso de los Visual Transformers, que últimamente han llamado la atención gracias a sus buenos resultados, equivalentes o mejores a las CNN en algunos contextos.

Para el caso de las métricas de evaluación se contempla el uso tanto métricas generales, como Exactitud, Precisión, Exhaustividad y F1, como métricas más específicas como AP y IoU (más detalles acerca de cada una en la Sección 2.3.2). En línea con esto último, la metodología de evaluación será mediante la comparación entre los modelos implementados y los presentados en la literatura para contextos similares.

2.5.4. Comparación con otros enfoques

Realizar una comparación de desempeño de los modelos de Aprendizaje Profundo implementados con otros enfoques tradicionales de Aprendizaje Automático para la detección de madurez de arándanos analizando las ventajas y desventajas de cada enfoque. Se contempla el uso de las mismas herramientas de la etapa anterior, añadiendo SciKit Learn [20], que permite implementar modelos tradicionales.

2.5.5. Reporte y conclusiones

Analizar los resultados obtenidos en la evaluación y comparación de modelos. Identificar ventajas y desventajas de los modelos de Aprendizaje Profundo y el potencial para mejorar la monitorización de arándanos. Resumir y reportar los hallazgos, proponer desafíos, recomendaciones de futuras investigaciones y posibles aplicaciones prácticas de la tecnología de detección de frutos basada en modelos de Aprendizaje profundo.

Capítulo 3

Conceptos Preliminares

3.1. Campos Involucrados

3.1.1. Visión Computacional

La Visión Computacional es un campo científico interdisciplinario que tiene como principal objetivo extraer información útil de las imágenes digitales. Estas últimas, desde un punto de vista informático son una matriz de números (píxeles), donde cada número representa la intensidad del color en ese punto en específico. En el caso de las imágenes en blanco y negro, es una sola matriz y los números indican el nivel de oscuridad de cada píxel. Para el caso del formato de imágenes a color, comúnmente el formato RGB, se componen de tres matrices, cada una representando un espectro de color diferente, una para el rojo (Red), otra para el verde (Green) y otra para el azul (Blue), donde los números de cada matriz presentan la intensidad de sus respectivos colores.

Considerando la composición de este tipo de datos, es difícil extraer cualquier tipo de información de las imágenes, ya que los píxeles solo otorgan información de la intensidad en un punto en concreto y no algo más útil, como si forma parte de un objeto dentro de la imagen o es parte del fondo. Es ahí donde entran los avances en el campo de la Visión Computacional, donde, por lo general, se busca agrupar conjuntos de píxeles reconociendo formas y patrones para extraer información de más alto nivel. Si bien existen muchas técnicas, las que mejores resultados han tenido el último tiempo son los modelos de Machine Learning y Aprendizaje Profundo.

3.1.2. Inteligencia Artificial

La Inteligencia Artificial (IA) puede definirse como la habilidad de una máquina de realizar funciones cognitivas propias del ser humano, como percibir, razonar, aprender e interactuar en un contexto especifico. Estas funciones son programadas por los seres humanos para crear una máquina "inteligente" en una tarea específica. Existen al menos cuatro enfoques para crear "sistemas inteligentes" [21].

- **Sistemas que piensan como humanos:** automatizar tomas de decisiones en distintas tareas, resolución de problemas y aprendizaje,
- **Sistemas que actúan como humanos:** ejecutar tareas que cuando son realizadas por

los seres humanos, requieren inteligencia.

- **Sistemas que piensan racionalmente:** intentar reproducir el pensamiento racional del ser humano, como el estudio de los cálculos o modelos.
- **Sistemas que actúan racionalmente:** intentan reproducir las conductas racionales que tendría un ser humano en un ambiente.

3.1.3. Aprendizaje Automático

El Aprendizaje Automático del inglés *Machine Learning* (ML), es un subcampo de la IA. Se puede definir como la ciencia de programar computadores para aprender de los datos, sin necesidad de programar explícitamente tal aprendizaje [22].

Existen al menos tres categorías en las que suelen caer los distintos tipos de ML.

- **De Aprendizaje Supervisado/Semisupervisado/ No Supervisado:** Que es entrenado con o sin "supervisión" (o una mezcla de ambos), es decir, con o sin conocer los resultados esperados o etiquetas.
- **De Aprendizaje por lotes y en línea:** Que pueden o no aprender incrementalmente sobre la marcha.
- **De Aprendizaje basado en instancias v/s basado en modelos:** Si funcionan comparando nuevos datos con datos conocidos o detectando patrones y creando modelos predictivos.

3.1.4. Aprendizaje Profundo

El Aprendizaje Profundo del inglés *Deep Learning* es un subconjunto del campo de Aprendizaje Automático que comúnmente está asociado a unas estructuras matemáticas inspiradas en el sistema nervioso de los seres vivos llamadas Redes Neuronales Artificiales o *Neural Networks* (NN). La diferencia entre Aprendizaje Automático y Aprendizaje Profundo, es que los modelos de este último son capaces de aprender representaciones automáticamente de los datos de entrada como texto, imágenes, o vídeos, sin necesidad de agregar reglas definidas por el humano, como es el caso de la mayoría de modelos de Machine Learning. Esto gracias al uso de sucesivas capas de representación de los datos (de ahí la denominación de "Deep" o "Profundo"). El Aprendizaje Profundo se utiliza comúnmente en aplicaciones de Visión Computacional, Análisis de Sentimientos y Sistemas de Recomendación [23].

3.2. Redes Neuronales Artificiales

Las redes neuronales son una de las principales y más populares técnicas de Aprendizaje Profundo en la actualidad. Su desarrollo lleva bastante tiempo, tiene sus orígenes en 1957 por Frank Rosenblatt con el Perceptrón [24]. A pesar de su innovación, las investigaciones futuras fueron congeladas un tiempo, debido a las críticas de Minsky y Papert en 1969 [25]. No fue hasta la década de 1980, con trabajos como el de Paul Werbos y David E. Rumelhart que popularizaron el uso del algoritmo de Backpropagation en las RNN, que se retomó el interés

en el área. Desde entonces, se han realizado variados avances, como también modificaciones y adiciones que permiten utilizar las redes neuronales para tareas específicas, desde análisis de texto, a lo que atañe esta tesis, análisis y clasificación de imágenes con las Redes Neuronales Convolucionales.

En sí, una red neuronal artificial hace cálculos sobre un dato de entrada, ya sea una imagen, texto, u otro, que se van propagando desde las neuronas de entradas hasta las neuronas de salida usando unos parámetros intermedios llamados pesos para ajustar los cálculos finales. El aprendizaje se produce cuándo dada una entrada y el posterior procesamiento de los pesos, el modelo de redes neuronales es capaz de asociar la entrada con una salida o etiqueta. Se puede decir que el modelo reconoce una entrada porque tiene "memoria" de los datos con los que fue previamente entrenado siendo capaz de reconocer conjuntos de datos similares. Existen muchos tipos de redes neuronales artificiales, orientadas a distintos tipos de tareas, pero todas mantienen una estructura y forma de trabajar similar [26].

3.2.1. Elementos de una Red Neuronal Artificial

3.2.1.1. Entradas y salidas

Las entradas comúnmente llamadas *inputs* son los datos que van a ser evaluados en la red neuronal. Por lo general, estas entradas son preprocesadas antes de entrar a la red neuronal y son transformadas en estructuras numéricas y vectores más simples de interpretar para el modelo. Por otro lado, las salidas, comúnmente llamadas *outputs*, son los resultados, o predicciones del modelo en función de las entradas. Generalmente, son comparados con los valores reales o etiquetas de los datos, y en razón de ello se calcula el error o pérdida.

3.2.1.2. Capas (Layers)

Las capas, del inglés *layers*, llevan la información desde la entrada hasta la salida trasformándola en el proceso y obteniendo representaciones convenientes para clasificar y analizar los datos. El perceptrón de una sola capa (*Single-Layer Perceptron*) es uno de los primeros ejemplos de una red neuronal [24].

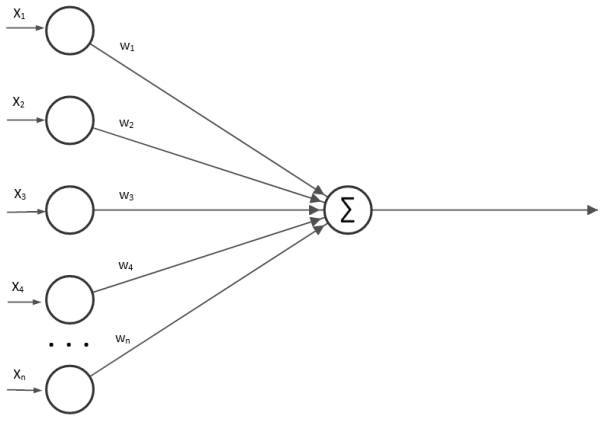


Figura 3.1: Estructura básica: perceptrón (Inspirado en ilustración de Chollet, 2021)

Por otro lado, a las estructuras con más de una capa se le llama redes neuronales multicapas (*Multilayer Neural Network*). Por lo general, las capas visibles al usuario son la capa de entrada y la capa de salida ya que las capas intermedias contienen representaciones de los datos que no siempre son legibles. A estas estructuras se les conoce además por ser redes de alimentación hacia adelante (*Feed-Forward Networks*) ya que las capas sucesivas se alimentan entre sí desde la capa de entrada a la capa de salida (Figura 3.2). Se asume además que todos los nodos de cada capa están conectados con los nodos de la capa sucesiva (Capas densamente conectadas o *Dense Layers*) [26].

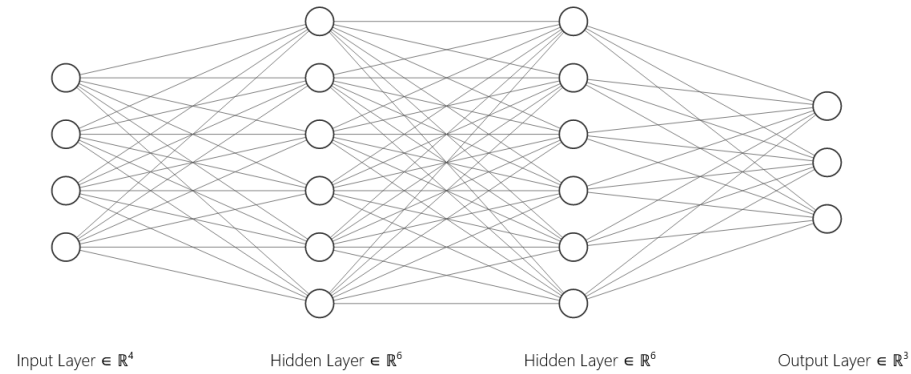


Figura 3.2: Redes Neuronales Multicapa (Inspirado en ilustración de Chollet, 2021)

3.2.1.3. Pesos (Weigths)

Los pesos en una Red Neuronal son un parámetro muy importante para el cálculo de la salida de cada capa. En términos prácticos las capas son parametrizadas por los pesos. Encontrar los pesos "ideales" es fundamental para obtener estimaciones efectivas.

3.2.1.4. Función de activación

La función de activación trabaja como un filtro o limitador que transforma los valores de la salida de cada neurona. Estas transformaciones por lo general son funciones no lineales, que permiten a la red neuronal resolver problemas cada vez más complejos. Entre las funciones de activación más relevantes se encuentran:

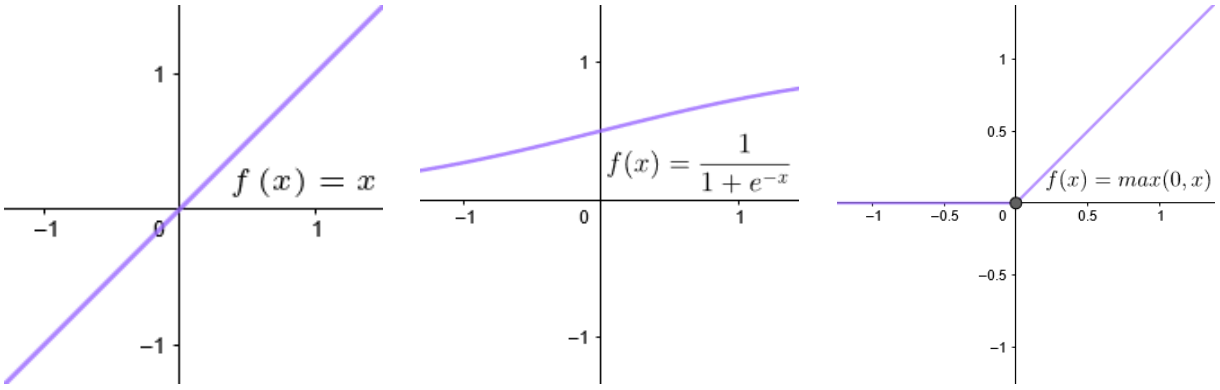
- **Función lineal:** también conocida como "función identidad", es una función que va desde $(-\infty, +\infty)$ lo que permite que las entradas sean iguales a las salidas. Por lo general se utiliza en modelos de Regresión Lineal.

- **Función logística:** también conocida como función Sigmoidal, tiene unos valores de salida que varían de cero a uno. Si los valores de entrada son negativos la función será igual a cero, en cambio, si son muy positivos se acercarán a uno. Por lo general esta función se utiliza en la última capa para tareas de clasificación binaria.

- **Función ReLU:** es una de las funciones más utilizadas ya que permite que el aprendizaje de las redes neuronales sea mucho más rápido. Sus valores de salida varían de cero

al $+\infty$ donde si los valores de entrada son negativos la salida será cero, en cambio, si son positivos estos se mantienen.

En la Figura 3.3 se muestran ejemplos de la función lineal (3.3.a), logística (3.3.b) y ReLU (3.3.c).



(a) Función Lineal (b) Función logística (c) Función ReLU

Figura 3.3: Funciones de activación.

3.2.1.5. Función de pérdida (Loss Function) y Optimizador (Optimizer)

La función de pérdida o función objetivo es la encargada de calcular la medida de éxito de las predicciones del modelo en cuestión. Esta medida, conocida como pérdida o *loss* representa lo alejadas que están las estimaciones obtenidas del modelo, de los valores reales o valores que se esperan obtener (valores objetivo o *target*). La función de pérdida es muy importante para la correcta parametrización de las capas del modelo, ya que, en base a la pérdida obtenida, se pueden actualizar los pesos del modelo, de forma que se adapten de mejor forma a los datos que van ingresando a la red. El objetivo es encontrar la mejor configuración de pesos de la red, ósea, la que obtenga mejores resultados. Es ahí donde entra en juego el optimizador (*optimizer*). El optimizador determina como se actualizarán los pesos del modelo. En general implementa alguna variante de descenso por gradiente, algoritmo de optimización de la función objetivo basado en su gradiente o derivadas parciales con respecto a los pesos.

3.2.2. Métricas de evaluación

Para evaluar el desempeño de los distintos modelos se utilizan las métricas de evaluación, estas permiten cuantificar diferentes aspectos de los modelos, centrándose principalmente en el desempeño general de las predicciones del mismo. Entre las métricas más utilizadas [27], se encuentran:

- **Matriz de confusión:** Si bien no es una métrica como tal, si es una herramienta muy útil que visualizar el rendimiento de un modelo en varios aspectos. Esta consiste en una matriz donde cada columna representa el número de predicciones del modelo y las filas representan el número de instancias reales. La cantidad de filas y columnas dependerá de la cantidad de clases de los datos. En un modelo con buen rendimiento, la mayoría de los números tenderán a centrarse en la diagonal principal de la matriz, o sea, las

predicciones coinciden con los valores reales. Para el caso contrario, la matriz puede indicar en qué clases el modelo se equivoca. Por ejemplo, en una tarea de clasificación binaria se tiene la siguiente matriz de confusión:

Predicción / Reales	Positivo (**P**)	Negativo (**N**)
Positivo (**T**)	Verdadero Positivo (**TP**)	Falso Negativo (**FN**)
Negativo (**N**)	Falso Positivo (**FP**)	Verdadero Negativo (**TN**)

- **Exactitud:** También conocida como *Accuracy*, es una métrica general que mide la proporción de predicciones correctas del modelo con respecto al total de predicciones.

$$Exactitud = \frac{TP + TN}{TP + FP + TN + FN} \tag{3.1}$$

- **Sensibilidad**: Mide la fracción de predicciones positivas que se clasifican correctamente.

$$Sensibilidad = \frac{TP}{TP + FN} \tag{3.2}$$

- **Especificidad** : Mide la fracción de predicciones negativas que se clasifican correctamente.

$$Especificidad = \frac{TN}{TN + FP} \tag{3.3}$$

- **Precisión:** Métrica que se utiliza para medir la cantidad de veces que el modelo predice correctamente una clase **(VP)** con respecto al total de las predicciones de dicha clase.

$$Exhaustividad = \frac{TP}{(TP + FP)} \tag{3.4}$$

- **Exhaustividad:** Del inglés *Recall*, es una métrica que se utiliza para medir la fracción de predicciones positivas **(VP)** que se clasifican correctamente.

$$Exhaustividad = \frac{TP}{(TP + FN)} \tag{3.5}$$

- **F1** (Métrica F): Métrica que representa la media armónica entre la Precisión y la Exhaustividad

$$F1 = \frac{2 \cdot Precision \cdot Exhaustividad}{(Precision + Exhaustividad)} \tag{3.6}$$

- **IoU**: Intersección sobre la Unión, métrica que mide la distancia entre el cuadro delimitador real y el cuadro delimitador que predice el modelo [28].

$$IoU = \frac{|B \cap B^{gt}|}{|B \cup B^{gt}|} \tag{3.7}$$

donde $B^{gt} = (x^{gt}, y^{gt}, w^{gt}, h^{gt})$ es el cuadro real, y $B = (x, y, w, h)$ el cuadro predicho por el modelo.

- **Precisión Promedio y Exhaustividad Promedio - AP y AR**: El promedio de la Precisión Y Exhaustividad por clase.

$$AP = \sum_{i}^{l} \frac{Precision_i}{l} \tag{3.8}$$

$$AR = \sum_{i}^{l} \frac{Exhaustividad_i}{l} \tag{3.9}$$

donde $C = (i_1, i_2, i_3 ... i_{l-1}, i_l)$ corresponde al conjunto de clases a evaluar en la tarea.

Generalmente, en la literatura se puede encontrar como **mAP** seguido de un valor. Esto para modelos de detección significa que se calcula el AP con una intersección sobre la unión específica. Por ejemplo, mAP50 indica que se calcula la precisión media coincidiendo en un 50 % el cuadro delimitador real de uno predecido.

3.2.3. Técnicas de entrenamiento y adaptación de redes neuronales

Una de las desventajas de los modelos de Aprendizaje Profundo es su alto costo, tanto en termino de recursos de cómputo, como en la cantidad de datos requeridos para conseguir buenas predicciones. Este último aspecto es uno de los más difíciles de resolver, ya que la ausencia de datos muy probablemente provocara que un modelo de Aprendizaje Profundo no se viable de implementar. Para este problema existen técnicas que, si bien no quitan por completo la necesidad de un conjunto de datos los suficientemente grande, generalmente a la obtención de mejores resultados finales e incluso a reducir los recursos de cómputo.

3.2.3.1. Data Augmentation

Esta técnica, para el caso específico de datos tipo imágenes, consiste en realizar trasformaciones a las imágenes que entraran a la red. Esto para simular una cantidad de muestras más variada. Las transformaciones consisten generalmente en rotaciones en distintos ángulos, giros en horizontal y vertical, zooms, entre otras. Estas modificaciones deben estar acorde al dominio de los datos, ya que podrían modificar en exceso el contexto de la imagen original y perder las características que permiten su clasificación o análisis.

3.2.3.2. Aprendizaje por Transferencia

La técnica del Transfer Learning o Aprendizaje por Transferencia, nace de la idea de reutilizar los conocimientos adquiridos en el entrenamiento de un modelo en otro de similares características y arquitectura. Además de eso, se debe asegurar que los dominios del problema a resolver de ambos modelos sean similares, ya que, por el contrario, el conocimiento del modelo entrenado no aportará en nada al modelo por entrenar, e incluso, podrían empeorar

los resultados que se hubieran obtenido sin aplicar Transfer Learning. Cuando se utiliza correctamente, el aporte de los conocimientos transferidos ayuda a mejorar el desempeño inicial del entrenamiento, la cantidad de tiempo necesario para entrenar y a los resultados finales de las métricas de evaluación [29].

3.2.3.3. Ajuste Fino

El principal problema de la técnica mencionada en la sección anterior es el dominio del problema, en este caso específico, las imágenes. A pesar de lo similares que sean los contextos de entrenamiento, diferencias como la forma o textura de lo que se quiere analizar afectarán en menor o mayor medida a los resultados finales. El Ajuste Fino o *Fine Tuning* es una técnica que consiste en modificar el conocimiento transferido de un modelo previamente entrenado y adaptarlo a los nuevos datos del modelo que se quiere entrenar. Principalmente se modifica el conocimiento de las últimas capas del modelo, ya que es ahí donde concentran los patrones más significativos de las imágenes que se quieren analizar, dejando a las primeras capas el aprendizaje de patrones locales básicos, como los bordes [30].

3.3. Redes Neuronales Convolucionales

Los inicios de las Redes Neuronales Convolucionales (CNN) se remontan hacia 1980 con la presentación de parte de Fukushima del Neocognitron [31]. Inspirado en cómo se creía que funcionaba la corteza visual de los gatos, Fukushima plantea la utilización de células simples y complejas con el fin de reconocer patrones importantes en una imagen. De esta forma, las células simples recogen información simple de la imagen, como líneas, colores, etc. Mientras que las células más complejas, toman la información de las células simples, para recoger información más detallada, como objetos, texturas, etc. Una representación visual de esto se puede observar en la Figura 3.4.

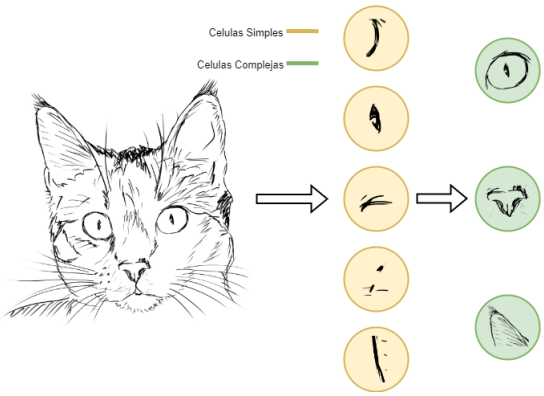


Figura 3.4: Células simples y complejas propuestas por Fukushima, 1980

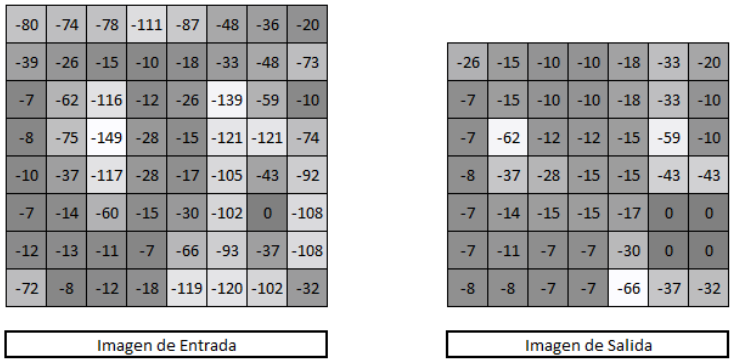
Fue más tarde en 1681 que se empezó a hablar de redes neuronales convolucionales con el método propuesto por LeCun et al. en [32] donde inspirado por el trabajo de Fukushima, implementó el algoritmo de Backpropagation, terminando de hacer viables la gran cantidad de cálculos realizados por estas redes. Aun así, no fue si hasta la década actual, gracias al avance de las tecnologías, principalmente en tarjetas gráficas, que las CNN terminaron transformándose en la opción por defecto para clasificar imágenes.

Las redes neuronales convolucionales se basan en un proceso llamado convolución, esta

realiza transformaciones sobre una imagen para crear un mapa de características. La capa de convolución, toma una imagen, o sea, una matriz de píxeles, y aplica un filtro que genera una salida, de menor tamaño, pero con las características transformadas, generalmente los filtros permiten encontrar diferentes patrones, como texturas, colores, etc. Un ejemplo se puede ver en la Figura 3.5.a donde el filtro, en ese caso para transformar una imagen a negativo, pasa como un escáner por toda la imagen, transformando el valor de sus píxeles y dando además como resultado una nueva imagen más pequeña y en negativo. Por lo general se aplican varios filtros a la imagen, obteniendo así, múltiples imágenes más pequeñas, con filtros diferentes.

Luego, las imágenes generadas pasan por la capa de agrupación. Esta reduce aún más el tamaño de las imágenes, conservando solo las zonas más importantes. Al igual que en el caso anterior un filtro de un tamaño determinado, pasa por toda la imagen, recogiendo solo el píxel con el valor más alto, creando una nueva imagen más pequeña, como se muestra en la Figura 3.5.b.

(a) Convolución



(b) Capa de Agrupación

Figura 3.5: Capas de Convolución (Inspirado en descripciones de Chollet, 2021)

Este proceso de convolución y agrupación se repite varias veces, obteniendo cada vez más imágenes, más pequeñas y con información más codificada. Esta información, pasa finalmente a una red neuronal "clásica" o densamente conectada para clasificar esas representaciones. Una representación visual de este proceso se puede ver en la Figura 3.6.

Esta forma de analizar las imágenes, tiene dos grandes propiedades identificables. En primer lugar, los patrones aprendidos son independientes de la localización, por lo que, si la CNN aprende un patrón existente en la esquina de una imagen, esta podrá reconocer ese

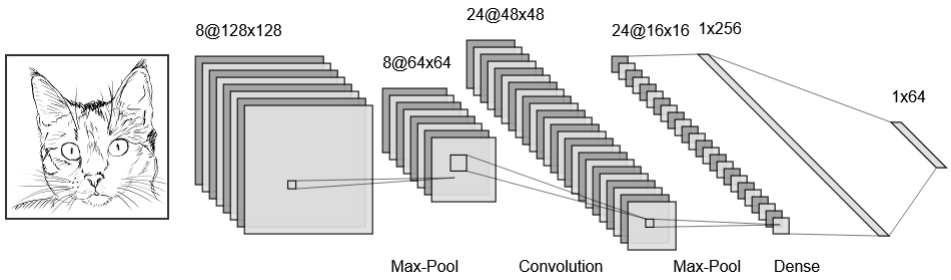


Figura 3.6: Representación de una Red Neuronal Convolucional (Inspirado en descripciones de Chollet, 2021)

patrón en cualquier parte de otra imagen. En segundo lugar, las CNNs pueden aprender jerarquías espaciales de patrones. Por ejemplo, las primeras capas de una CNN, estas pueden aprender bordes, en cambio, las capas más profundas, aprenden conceptos cada vez más complejos, como ojos, pelaje, entre otros [30]. Similar a lo presentado por Fukushima en 1980 (Figura 3.4).

A pesar de ser uno de los métodos más populares, tiene ciertas desventajas. En primer lugar, y por lo general, esta técnica funciona peor que otras técnicas de Aprendizaje Automático cuando no abundan los datos de entrenamiento. En segundo lugar, ya solventado el problema de los datos de entrenamiento, el costo en términos de cómputo que se requiere para entrenar es alto, necesitando por lo general tarjetas gráficas de alta gama o equipo especializado para acelerar el proceso.

3.3.1. VGG16

En base a los buenos resultados obtenidos con las CNN, existen algunas propuestas para mejorar la precisión de estos modelos, Así es el caso de VGG16, presentada el año 2015 en [33], donde se propone una estructura definida de una CNN, caracterizada por tener filtros de convolución muy pequeños y 16 capas de profundidad, obteniendo así una representación de las características más profunda. Esta CNN ha demostrado ser bastante solvente, especialmente debido a su estructura simple. Por contraparte, la profundidad de sus capas provoca que la cantidad de parámetros a entrenar sea alta. Una forma de solventar esta desventaja es la utilización de un modelo preentrenado, esto es, un modelo que fue previamente entrenado con muchas imágenes y es capaz de realizar la tarea de clasificación con suficiente precisión. Por lo general, estos modelos están entrenados con imágenes diferentes al dominio del problema que se desea resolver, es por ello que, se acude a técnicas de Aprendizaje por transferencia y Ajuste Fino. En la mayoría de los casos, la utilización de estos modelos preentrenados mejora en gran medida la precisión y el tiempo de entrenamiento.

3.3.2. ResNet

Otra arquitectura de CNN popular es ResNet, presentada en el año 2015 en [34]. Cuando se aumenta la cantidad de capas de una CNN la cantidad de parámetros a entrenar aumenta considerablemente, con ResNet, se propone un método para construir CNNs de gran profundidad sin generar un coste excesivo de parámetros entrenables, haciendo uso de las denominadas Redes Residuales, que, en palabras simples, permiten a las salidas de una capa, saltar una cantidad de capas hacia adelante, y no alimentar directamente la siguiente. Estas redes han demostrado ser solventes en las tareas de clasificación de imágenes, especialmente

cuando se requiere un análisis en profundidad. El modelo ResNet se encuentra en versiones de 50, 101, 152 capas. También, al igual que VGG16 se puede encontrar como modelos preentrenados.

3.4. Redes Neuronales Convolucionales para Detección

Los modelos presentados en la sección anterior, generalmente se centran en una tarea, la clasificación, pero existen otros más especializados para las tareas de detección de objetos. El objetivo en esta tarea es encontrar rectángulos, comúnmente denominadas cajas delimitadoras o *bounding boxes*, alrededor de objetos de interés en una imagen y asociar tal rectángulo con una clase [30]. Para ello, en los últimos años se han creado estructuras de redes neuronales especializadas en esta tarea. Tal es el caso de las R-CNN y sus variantes.

3.4.1. Redes Neuronales Convolucionales basadas en Regiones

Las *Region Based Convolutional Neural Networks* (R-CNN) fueron presentadas el año 2014 en [35]. Es una red neuronal que implementa módulos específicos para la búsqueda de regiones o cajas delimitadoras, que luego son analizadas por una CNN común, generalmente llamada *backbone* o columna. Más específicamente, se realiza una búsqueda selectiva de regiones. En primer lugar, se genera una subsegmentación inicial de la imagen, alrededor de 2000 regiones candidatas o regions of interest (ROI). Luego, mediante un algoritmo voraz se combinan regiones similares en regiones más grandes de forma recursiva. Finalmente, con las regiones más grandes generadas, se crean propuestas finales de regiones candidatas que pasarán al análisis de características con una CNN normal seguidos de una SVM para clasificar la presencia de un objeto.

3.4.2. Fast R-CNN

Un año después, de parte del mismo autor que propuso el modelo anterior, se presenta en [36], Fast R-CNN, un modelo que mejora en gran medida el tiempo de entrenamiento de R-CNN. Esto lo logra gracias a la definición más eficiente de las regiones, para ello, utiliza la capa de agrupación de las CNNs para la subsegmentación inicial y realizar la búsqueda selectiva. Gracias a eso, no es necesario realizar la subsegmentación de 2000 regiones por cada imagen.

3.4.3. Faster R-CNN

Los dos métodos de detección anteriores utilizan búsqueda selectiva, un proceso que es costoso y afecta a toda la red. En el año 2016 en [37] se presenta una nueva arquitectura de detección de imágenes que se inspira en Fast R-CNN, pero cambia el proceso de búsqueda selectiva de regiones por una *Region Proposal Network* o RPN, una red neuronal convolucional completamente conectada. De esta forma, la red aprende propuestas de región, logrando acelerar aún más el proceso. Luego de la predicción de las regiones propuestas, el proceso continúa similar a Fast-CNN, pasando por una capa de agrupación ROI y prediciendo la clase de las regiones detectadas.

3.4.4. Mask R-CNN

Uno de los últimos métodos basado en regiones fue presentado el 2018 en [38]. Es un método de segmentación de imagen, que, a diferencia de la detección, donde se busca una caja delimitadora, en la segmentación se busca detectar con precisión los píxeles correspondientes al objeto que se busca, definiendo sus límites, tamaños, etc. Para ello, Mask R-CNN se extiende de Faster R-CNN y agrega un módulo para enmascarar objetos, en paralelo al reconocimiento de la caja delimitadora, añadiendo solo un costo extra de cómputo.

3.4.5. YOLO

Del inglés *You Only Look Once*, YOLO es una propuesta diferente a las R-CNN, que fue presentada el 2016 en [39]. Plantea la detección de objetos como un único problema de regresión, desde los píxeles de la imagen al cuadro delimitador del objeto y las probabilidades de clase. YOLO solo mira una vez la imagen para predecir que objetos están presentes y en dónde. Esto lo logra gracias a la división de la imagen en una cuadrícula, donde cada cuadricula predice cuadros delimitadores. Para la clasificación se utiliza una CNN clásica llamada DarkNet. Luego de esta predicción, se procede a eliminar los cuadros que no tengan un mínimo de precisión junto con el filtrado de cajas que se pudieron haber detectado dos veces. Gracias a su simple funcionamiento, YOLO es extremadamente rápido, tanto que puede procesar imágenes a 45 cuadros por segundo, haciéndolas especialmente útiles para para la monitorización en tiempo real.

3.5. Visual Transformers

Los Visual Transformers (ViT) han aparecido recientemente como una gran alternativa a las CNNs. Están basados en los transformers que se utilizan principalmente para tareas de Procesamiento del Lenguaje Natural. Estos se caracterizan por trabajar con mecanismos de atención, lo que les permite identificar qué partes de una oración, representado por vectores de números, son "importantes" y que deberían tenerse en cuenta para el análisis de las palabras, evitando olvidarlas como pasaba con Redes Neuronales Recurrentes, el anterior modelo que se utilizaba para esta tarea [40]. Para las imágenes el caso es similar. Esta vez los vectores de números, que antes representaban una oración, ahora representan "parches" de tamaño fijo de la imagen, a los que se agregan incrustaciones de posición. Luego el funcionamiento es relativamente similar, los mecanismos de atención de los Transformers, identifican las zonas importantes y pasan al módulo que finalmente las clasifica. En muchos casos, Transformers iguala o mejora los resultados en comparación con un CNNs y según sus autores su entrenamiento es relativamente más barato[41].

Capítulo 4

Estado del Arte

El campo de la visión computacional, más específicamente el área de los modelos de Aprendizaje Profundo es muy amplia, cada año se desarrollan más herramientas para el análisis de imágenes, para distintos contextos y objetivos. Es por esto que se hace necesario conocer los últimos avances en esta área, ajustando específicamente al contexto de análisis de frutos, para encontrar el camino ideal para realizar los experimentos, en términos de mejores modelos, herramientas y técnicas.

4.1. Preparación de la revisión

A continuación, se presentan las preguntas de investigación definidas para encontrar información acerca de las formas y los materiales que se utilizan normalmente en área de la clasificación de imágenes de frutos.

- **PI1:** ¿Cuáles son las principales tareas ejecutadas en las investigaciones?
- **PI2:** ¿Qué modelos de Aprendizaje Profundo son utilizados actualmente en el reconocimiento de imágenes de frutos y cuáles son sus particularidades? (Patrones de uso, Disponibilidad, Facilidad de replicación, configuraciones, etc.)
- **PI3:** ¿Cuáles son las métricas que se utilizan para evaluar el comportamiento de los modelos?
- **PI4:** ¿Cuál es el contexto de la investigación?, ¿Cuáles fueron los recursos analizados?

A partir de las preguntas de investigación se identifican algunos términos clave con los que se construye la siguiente cadena de búsqueda:

(“*Convolutional Neural Network*” OR “*Visual Transformer*”) AND (“*Fruit*” OR “*Blueberry*”) AND (“*Recognition*” OR “*Classification*” OR “*Detection*” OR “*Maturity Detection*”)

Los siguientes motores de búsqueda son elegidos por su reputación y facilidad de uso, ya que tienen la capacidad de buscar por cadena de búsqueda, filtrar datos y descargar los documentos.

- Web of Science (WOS)
- ScienceDirect (SDI)

- Springer (SPI)
- IEEE
- ACM

También se definen los siguientes criterios de inclusión y exclusión que tiene como principal objetivo, centrar la temática de las investigaciones y acotar los resultados de búsqueda.

Tabla 4.1: Criterios de inclusión y exclusión para seleccionar artículos

Inclusión / Exclusión	**Criterio**
Criterios de Inclusión	**CI1:** Artículos que se refieran principalmente al análisis de cultivo de frutos mediante imágenes y Aprendizaje Profundo **CI2:** Artículos que demuestren la experiencia de la investigación. **CI3:** Artículos publicados entre los últimos tres años (2020-2023).
Criterios de Exclusión	**CE1:** Artículos con título y abstract no relacionados con el tema principal de la investigación, aún teniendo las palabras clave de la cadena de búsqueda. **CE2:** Artículos relacionados exclusivamente con las enfermedades de frutas. **CE3:** Artículos relacionados exclusivamente con las pestes de la fruta. **CE4:** Artículos que no analicen principalmente la fruta, como las plantas o el campo de cultivo. **CE5:** Artículos que no utilicen imágenes como dato de entrada principal. **CE6:** Artículos que no utilicen Redes Neuronales Convoluciones o sus derivados.

4.2. Resultados de búsqueda y selección

Luego de utilizar la cadena de búsqueda en todos los motores, se aplicaron los criterios de inclusión y exclusión. El detalle del proceso de selección puede ser revisado en Diagrama de Flujo PRISMA, presentado en la Figura 4.1

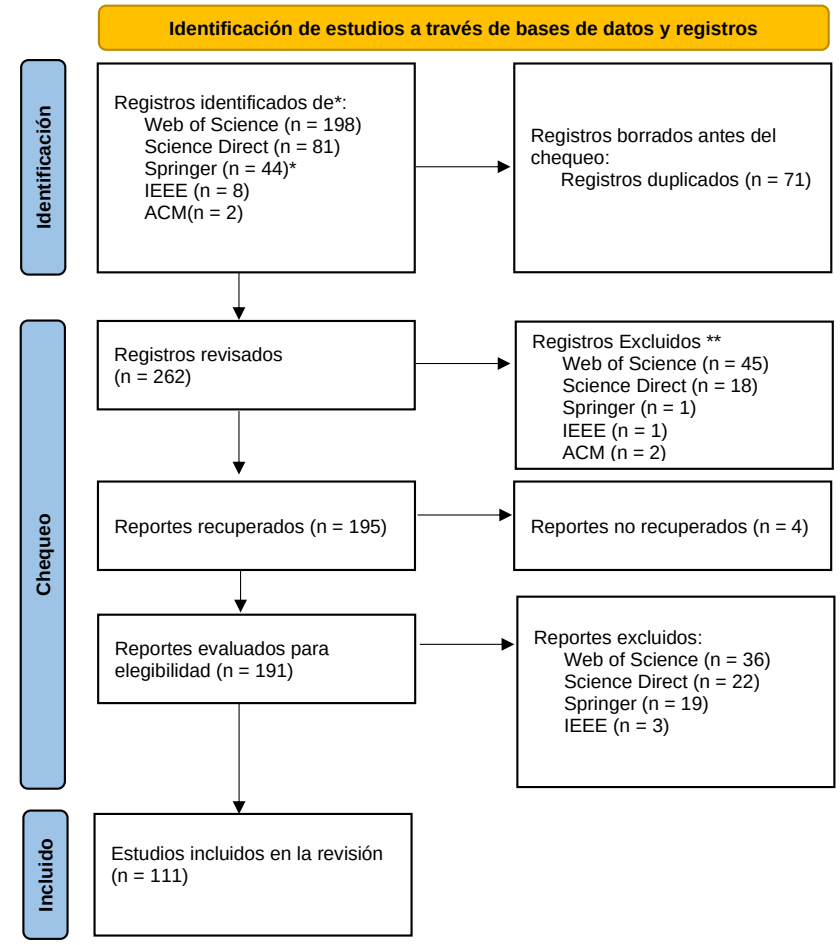


Figura 4.1: Diagrama de flujo PRISMA

4.3. Análisis y síntesis de la revisión

En línea con la pregunta de investigación **PI1**, sobre las principales tareas ejecutadas en las investigaciones, se pueden identificar dos grandes tareas generales, *clasificación* y *detección*, lo cual influye tanto en la forma de abordar el problema como en la selección de las tecnologías necesarias. En el caso de la clasificación se tiende utilizar los modelos clásicos de CNN, tanto modelos construidos desde cero como con la utilización de modelos preentrenados. Es también en estos casos donde la cantidad de muestras del conjunto de datos es mayor, precisamente por los requerimientos de estos modelos, además de contener imágenes más simples, capturadas en ambientes controlados o seleccionadas de internet, probablemente por la dificultad que tienen los modelos clásicos de CNN para identificar objetos ocluidos.

Por otro lado, están los trabajos centrados en la detección. En este caso, se tiende a utilizar los modelos basados en R-CNN, como YOLO o Faster R-CNN. Es también aquí donde las tareas se centran en la monitorización, utilizando robots móviles, cámaras en tiempo real o drones. Las imágenes son más complejas, capturadas directamente de un entorno natural, con la obstrucción de hojas y ramas, ya que el objetivo es interferir lo menor posible en los cultivos, siendo el análisis de regiones de las R-CNN bastante útiles.

Las evaluaciones de los modelos se basan principalmente en la comparación de métricas de evaluación entre distintos modelos que realizan las mismas tareas. Cerca de un 80 % de las investigaciones utilizan este método. Otras investigaciones agregan la comparación con modelos de otras investigaciones, generalmente con investigaciones que analicen el mismo fruto, buscando demostrar una mejora frente a tales investigaciones. Cerca de un 20 % utiliza esta comparación.

4.3.1. Modelos de Deep Learning

En línea con la pregunta de investigación **PI2**, con respecto a los modelos más utilizados, como se menciona en la sección anterior, se puede ver una diferencia entre los modelos de clasificación (CNN clásicas y diferentes arquitecturas como VGG16, ResNet50, etc.) que son implementados por el 58 % de las investigaciones y los modelos de detección (modelos basados en R-CNN, YOLO y sus respectivas versiones y variaciones) que son implementados por un 41 % de las investigaciones. Además, muchos de ellos utilizan más de un modelo, como es el caso de los modelos múltiples, comparaciones entre modelos o los basados en R-CNN, que pueden utilizar diferentes modelos CNN como backbone. Cabe destacar que solo el 1 % de los estudios revisados utilizan los nuevos modelos Visual Transformers (para tareas de clasificación), lo que indica la limitada adopción de estos modelos en el campo del análisis de frutas.

Si bien en la mayoría de las investigaciones modifican en mayor o menor medida los modelos, podemos diferenciar algunos que parten desde cero, construyendo estructuras propias. Estos últimos fueron etiquetados como Propuestas “Propias” y que por lo general corresponden a modelos de CNN clásica. Un resumen de la frecuencia de los modelos utilizados se muestra en la Figura 4.2. Se destaca el uso más frecuente de los modelos VGG16, ResNet50, ResNet101, siendo en algunos casos sus versiones preentrenadas y otras desde cero, también son frecuentes los modelos de detección como Faster R-CNN, Mask R-CNN y las diferentes versiones de YOLO (ninguna de ellas teniendo una mayor frecuencia apreciable frente a otras.)

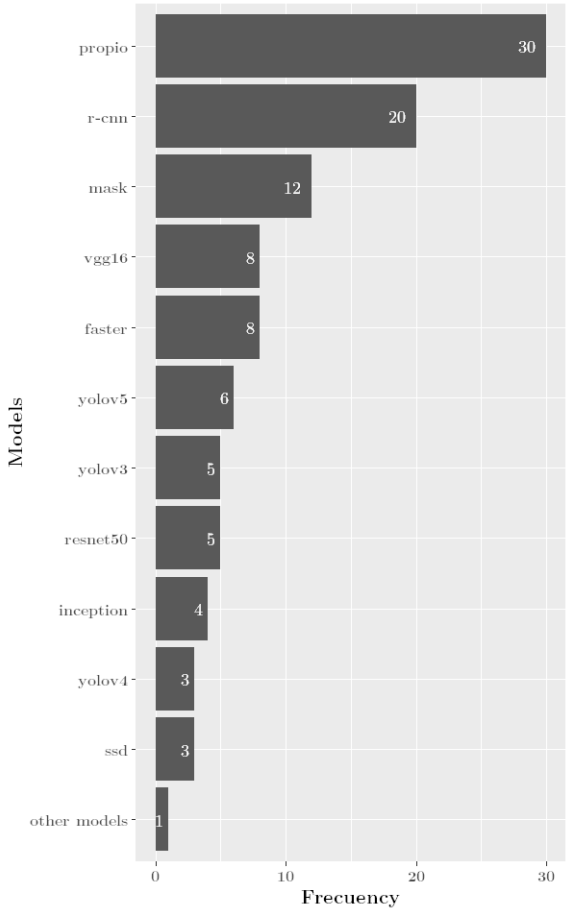


Figura 4.2: Frecuencia de modelos

Entrando en más detalle respecto a los modelos de detección, también existe una frecuencia de modelos utilizados como backbone, en el caso de los modelos basados en R-CNN, los modelos más utilizados son ResNet, VGG16 e InceptionV3. Para YOLO se utilizan más los modelos DarkNet53 y CSPDarkNet53, probablemente por ser los modelos por defecto de esta arquitectura y que han demostrado un mejor rendimiento. Aun así, existen propuestas con otros modelos con DenseNet.

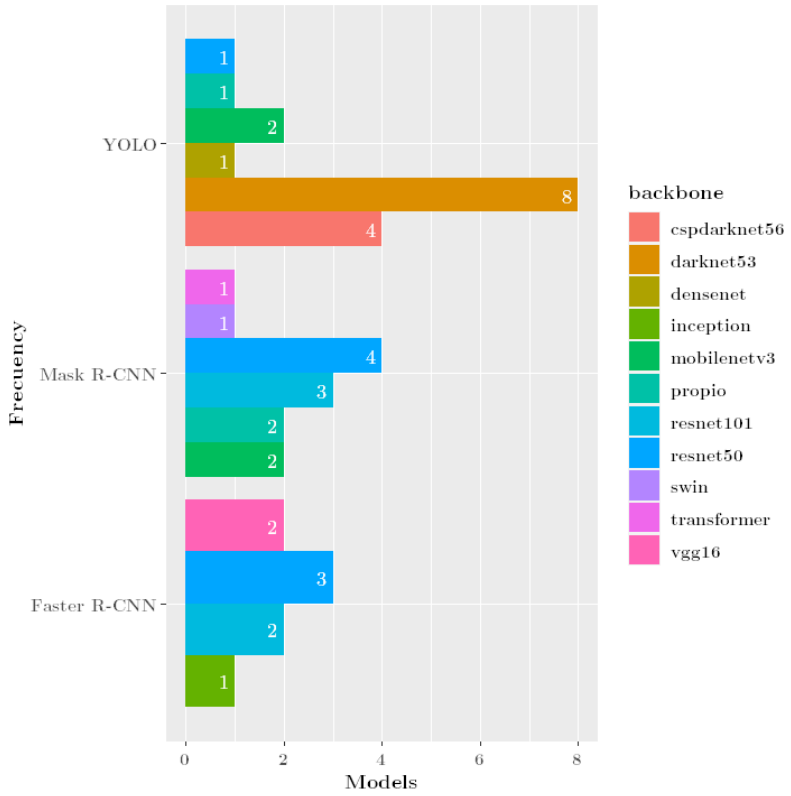


Figura 4.3: Frecuencia de backbones en R-CNNs y YOLO

Con respecto a las configuraciones de los modelos utilizados, se destaca el irregular nivel de detalle en los diferentes trabajos, probablemente por las diferencias existentes tanto en la tarea a realizar, como en los modelos y técnicas implementadas. Aun así se pueden apreciar ciertas tendencias en las configuraciones e hiperparámetros de los modelos. En casi todos los casos, independiente de la tarea y el modelo, se utiliza el optimizador "Adam". En el caso de los modelos "Propios" la función de activación más utilizada es "ReLU". En el caso del Ratio de aprendizaje, el valor más frecuente es "0.001" aunque en algunos casos se prefiere variar el valor durante el entrenamiento, como por ejemplo "0.001 a 0.0001". Por otro lado, un 70 % de las investigaciones describe explícita o implícitamente el uso de técnicas de Aprendizaje por transferencia.

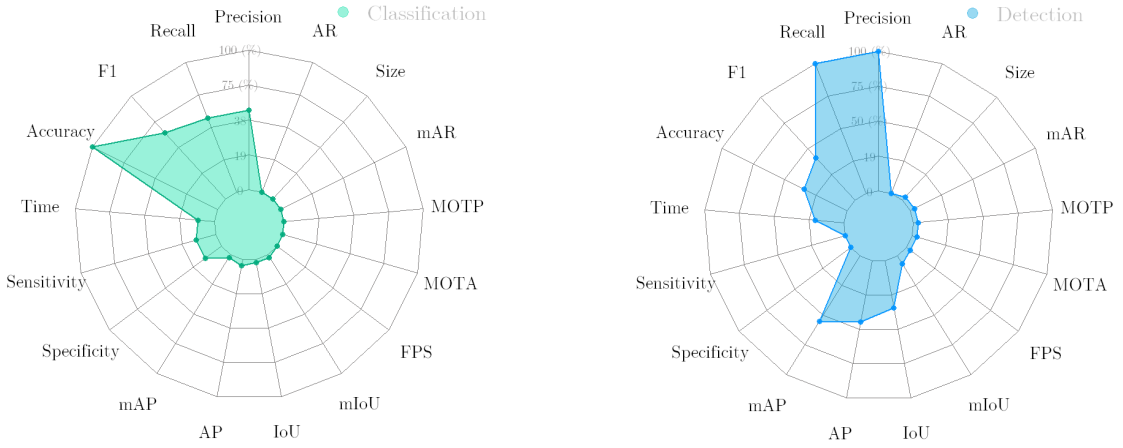
4.3.2. Evaluaciones y métricas

En línea con la pregunta de investigación **PI3**, se puede apreciar una marcada diferencia en las métricas de evaluación utilizadas entre tareas de clasificación y detección. En la Figura 4.4 se presenta gráfico de radar que resume las métricas utilizadas en las investigaciones, construido en R en base listado de métricas registradas en la lectura. Cabe destacar que se priorizan las métricas utilizadas para la evaluación final de los modelos. Las comparaciones entre modelos se centran principalmente en el nivel de mejora de un modelo específico, principalmente el construido en la investigación con los modelos frecuentemente utilizados en la literatura.

Entre las cinco métricas más utilizadas destaca la Precisión, presente en el 20,2 % de las investigaciones. Le siguen de cerca otras métricas habituales como Exactitud y Exhaustividad, ambas con un 19,6 %, y F1 con un 16 %. Finalmente, destaca la métrica mAP, que aparece

en un 6,3 % de los estudios.

Al analizar las métricas utilizadas para las tareas de clasificación, se hace evidente una prevalencia de la métrica Precisión (Figura 4.4.a). En cambio, para las tareas de detección se observa un aumento en la adopción de métricas “IoU” y “mAP” (Figura 4.4.b). Estas métricas evalúan la precisión de los cuadros delimitadores de detección, a diferencia de la métrica más general de Precisión. En ambos casos, “Precisión” y “Exhaustividad” se emplean con similar frecuencia, siendo estos esenciales para el cálculo de F1. Además, las variables que se miden con frecuencia en los modelos de detección son “Tiempo” y “FPS”, particularmente en escenarios donde dichos modelos se implementan en robots móviles, drones o cámaras en tiempo real, lo que requiere capacidades de captura y análisis rápidas.



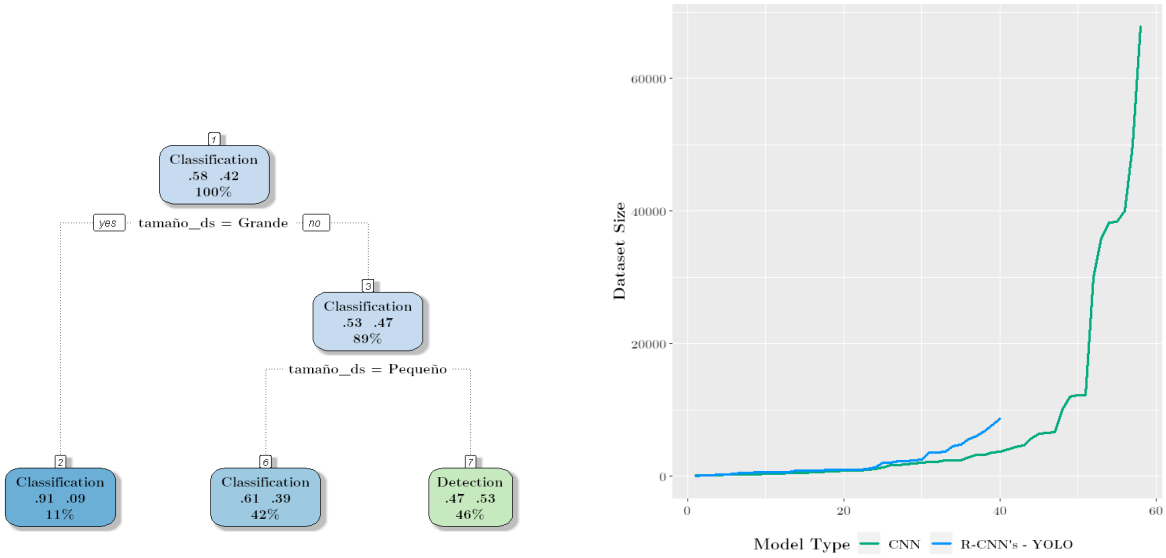
(a) Uso de métricas en tareas de Clasificación

(b) Uso de métricas en tareas de Detección

Figura 4.4: Uso de métricas por tipo de tareas

4.3.3. Contexto y recursos de la investigación

En línea con la pregunta de investigación **PI4** y con respecto a los conjuntos de datos, se encontró que los tamaños de estos son muy variados, aunque se puede apreciar una diferencia en la cantidad de datos utilizados dependiendo del tipo de modelo. En CNN clásico se tiende a usar muchos más datos, mientras que en R-CNNs se ocupa una menor cantidad. Esto último no indica que R-CNN necesita menos datos, sino que probablemente es debido a la dificultad de etiquetar cada cuadro limitador o píxel, en el caso de la segmentación, que remarque la fruta que se desea analizar. En la Figura 4.5.b, se puede apreciar un gráfico que muestra esta diferencia. Además, en la Figura 4.5.a se presenta un árbol de decisión construido con los datos recolectados en la revisión, donde se puede identificar que las investigaciones que realizan detección, tienden a utilizar conjuntos de datos ”Medianos"(entre 1.000 y 10.000 imágenes), mientras que los de clasificación acumulan más conjuntos de datos ”Pequeños"(menores a 1000 imágenes) y ”Grandes"(mayores a 10.000 imágenes)



(a) Árbol de decisión: Tamaño del conjunto de datos por tipo de tarea

(b) Comparación de tamaños de conjuntos de datos CNN y R-CNN

Figura 4.5: Tamaños de los conjuntos de datos

Sobre las proporciones de los conjuntos normalmente utilizados en la literatura, si bien también los valores son variados, se pueden observar ciertas tendencias. El conjunto de entrenamiento, por lo general, es el que tiene la mayor cantidad de muestras, repartiendo el restante entre validación y pruebas. En algunos casos no se considera el conjunto de validación, separando el conjuntos de datos en solo entrenamiento y prueba. En la Figura 4.6 el gráfico de cajas y bigotes muestra las proporciones observadas, siendo la proporción 7:1:2 para entrenamiento, validación y pruebas respectivamente la más frecuente.

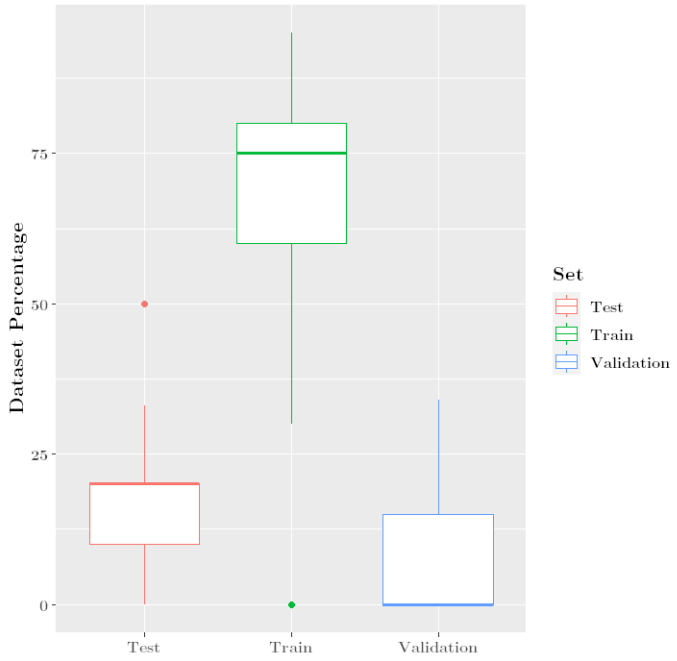


Figura 4.6: Distribución de los conjuntos de datos

El segundo aspecto en lo que respecta a las imágenes es la resolución, que es muy variada, dependiendo de las necesidades de la tarea y la capacidad de los dispositivos utilizados para capturar las imágenes. Algo que sí se puede apreciar, es que la mayoría de las capturas son en resoluciones altas (muchas mayores a 1920×1080 píxeles) pero que al ser procesadas por los modelos CNN estas son reescaladas a una resolución más manejable (por lo general, menor a 512x512 píxeles). También son los modelos clásicos de CNN los que utilizan las resoluciones más bajas. Aun así, agrupando las imágenes por su cercanía a los estándares de resolución y generando un árbol de decisión que toma como variables las resoluciones y el tipo de tareas (Figura 4.7), se puede observar que las investigaciones que realizan detección suelen utilizar resoluciones cercanas al FullHD, 4K Y 8K.

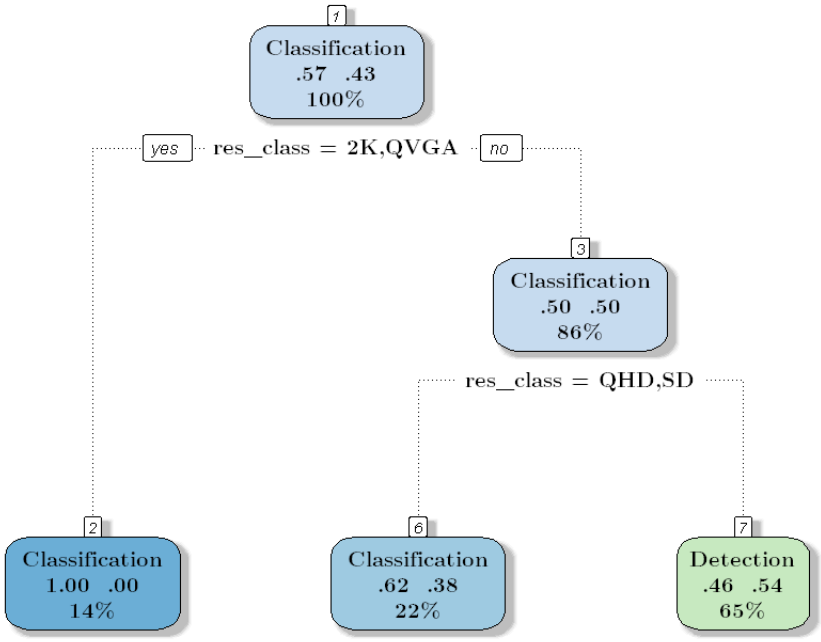


Figura 4.7: Árbol de decisión: Tipo de tarea por resolución de imagen

El tercer aspecto es el entorno de captura de las imágenes. En la Figura 4.8 se presenta un árbol de decisión que fue construido usando los registros obtenidos de la lectura, referentes a los entornos de captura de imágenes y los enfoques de los modelos donde son utilizadas como datos de entrada dependiendo del tipo de tarea. De los registros se puede apreciar que cerca del 56 % son imágenes capturadas en un entorno controlado o de laboratorio, con una distancia fija, objetos centrados y bien iluminados, generalmente con focos. Las imágenes capturadas en este entorno son más utilizadas para el entrenamiento de modelos clásicos de CNN, probablemente para reducir al máximo el ruido y la oclusión, que afecta mucho a estos modelos. Por otro lado, el otro 44 %, son imágenes captadas en un entorno natural, buscando, incluso, condiciones desfavorables para reforzar el desempeño de los modelos. Estas imágenes, por lo general, son capturadas con el fruto en la planta, evitando despejar el entorno para una captura limpia del fruto. Son tomadas de distintos ángulos y en distintas condiciones de luz solar, ya sea con la presencia o ausencia de nubes o en distintas horas del día. Las imágenes capturadas en este entorno son, por lo general, utilizadas en los modelos de R-CNN, ya que buscan reforzar, mediante la clasificación y búsqueda selectiva de regiones, la creación

de modelos robustos para estos entornos.

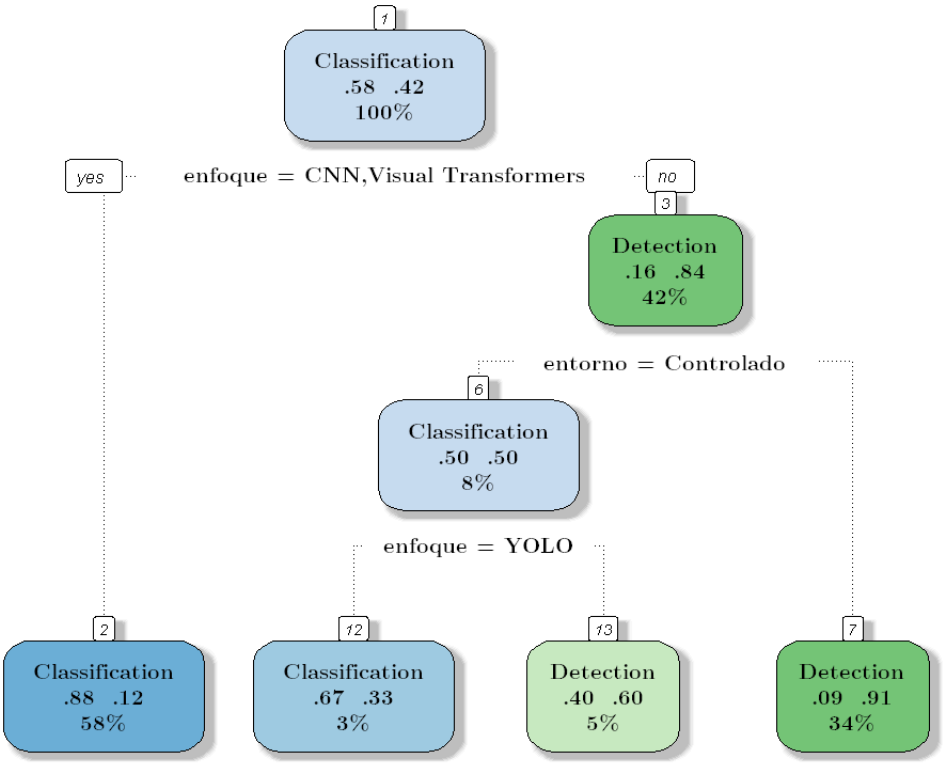


Figura 4.8: Árbol de decisión: Entorno de captura por tipo de tarea

De esta revisión, se logra identificar tendencias en varios aspectos de los procesos de experimentación de las tareas de análisis de imágenes con Deep Learning. Estas tendencias son cruciales para construir el entorno de experimentación y la preparación de materiales de este proyecto de tesis. Los modelos a utilizar basados en los hallazgos de esta revisión son presentados en la Sección 5.1.2.

Capítulo 5

Implementación y Resultados

5.1. Preparación de materiales

5.1.1. Conjuntos de datos

Para la realización de los experimentos se cuenta con un conjunto de datos etiquetado provisto por el proyecto Fondef. Este consiste en imágenes de alta resolución de plantas de arándanos y sus frutos en diferentes estados de madurez. Además, las imágenes están tomadas desde distintos ángulos (principalmente aéreos laterales) e iluminación. El detalle de este conjunto de datos se presenta en la Tabla 5.1. En la Figura 5.1 se describen las etiquetas del conjunto de datos y en la Figura 5.2 una presenta muestra con los cuadros delimitadores remarcados. Cabe destacar que el conjunto de validación se utilizará para medir el rendimiento de los modelos, mientras que el de pruebas, que contiene arándanos en otros contextos, se utilizará para pruebas con usuarios reales y para medir velocidad de inferencia.

Tabla 5.1: Descripción del conjunto de datos original

	Arándanos Inmaduros	**Arándanos Medio Maduro**	**Arándanos Maduro**	**Total Imágenes**
Conjunto de entrenamiento	2825	431	3271	190
Conjunto de validación	539	66	556	33
Conjunto de pruebas	628	170	680	42
Total	3992	667	4507	265

(a) Fruto inmaduro: "**Inmaduro**"

(b) Fruto pre-maduración: "**MMaduro**"

(c) Fruto maduro: "**Maduro**"

Figura 5.1: Etiquetado: Muestras del conjunto de datos

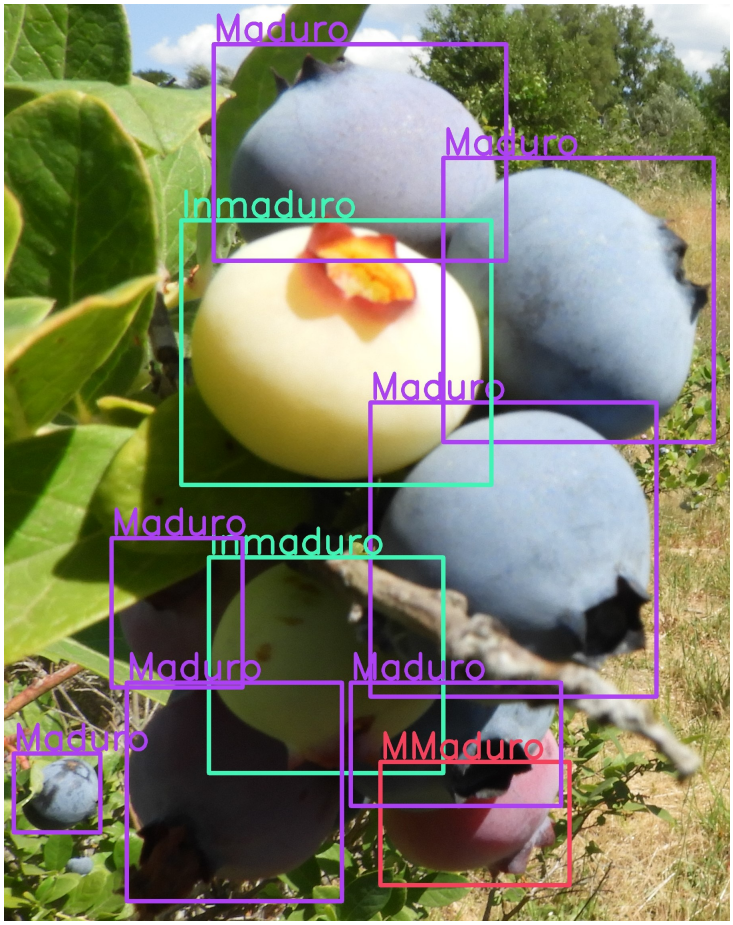


Figura 5.2: Muestra del conjunto de datos original con los cuadros delimitadores

En línea con lo revisado en el Estado del Arte, con una mayor cantidad de datos probablemente se obtengan mejores resultados. Es por esto que se realiza un proceso de *Data Augmentation* sobre el conjunto de datos original. Afortunadamente, para realizar la detección de arándanos no se requiere de una orientación ni posición específica del fruto, ya que su forma redonda es simple y no cambiará su significado si se realizan estas modificaciones.

Al contrario, esto puede resultar beneficioso para detectar el fruto en posiciones que no están originalmente en el conjunto de datos. Para realizar *Data Augmentation* se aplicaron las modificaciones descritas en la Tabla 5.2 donde la columna "Nivel" indica si la modificación se realizó a la imagen completa o a los cuadros delimitadores de los arándanos.

Tabla 5.2: Modificaciones aplicadas para *Data Aumentation*

Nivel	**Aumentación**	**Rep.**	**Detalle**
Imagen	Voltear		Horizontal y Vertical
Imagen	Rotación de 90º		Sentido horario, antihorario y al revés
Imagen	Recorte		0 % zoom mínimo, 20 % zoom máximo
Imagen	Rotación		Entre -15º y +15º
Imagen	Perspectiva		±19º horizontal y ±16º vertical
Cuadro delimitador	Voltear		Horizontal
Cuadro delimitador	Rotación de 90º		Sentido horario, antihorario y al revés
Cuadro delimitador	Rotación		Entre -20º y +20º

Estas modificaciones son aplicadas solamente sobre el conjunto de entrenamiento. El conjunto de datos aumentado se describe en la Tabla 5.3. Una muestra de este conjunto se puede observar en la Figura 5.3

Tabla 5.3: Descripción del conjunto de datos aumentado

	Arándanos Inmaduros	**Arándanos Medio Maduro**	**Arándanos Maduro**	**Total Imágenes**
Conjunto de entrenamiento	13518	2062	15534	947
Conjunto de validación	539	66	556	33
Conjunto de pruebas	628	170	680	42
Total	14685	2298	16770	1022

(a) Imagen original

(b) Imagen aumentada

Figura 5.3: Muestra del conjunto de datos aumentado

Algo a considerar desde ya es el desbalance evidente de las clases de arándanos, siendo "MMaduro" la más baja. Generalmente para evitar esto, se realiza un aumento de la clase de desbalanceada. El problema, en particular de las tareas de detección, es la dificultad de generar nueva información y posicionarla en la imagen. Esto puede ser negativo, pues es muy probable que la representación original de la imagen cambie demasiado y no se acerque lo suficiente a los valores reales que se buscan predecir. Es por esto que se continúa igualmente con la clase desbalanceada, aunque se espera un rendimiento menor de los modelos en esta etiqueta.

También se crearon distintas versiones del conjunto de datos aumentado, donde sé varía la resolución de imagen y el formato en entrada para los distintos modelos a probar. Además de estos, se construye un conjunto de datos de clasificación, recortando todos los cuadros delimitadores de las imágenes. El resumen de todos los conjuntos de datos construidos se describe en la Tabla 5.4. Los nombres indicados en la tabla serán utilizados para identificar cada conjunto de datos en las siguientes secciones del documento.

Tabla 5.4: Resumen de los conjuntos de datos generados

Nombre	**Resolución**	**Formatos**	**Tipo**
dataset_aug	640x640	YOLO, COCO, VOC	Detección
dataset_aug_800	800x800	YOLO	Detección
dataset_aug_1240	1280x1280	YOLO, COCO, VOC	Detección
dataset_aug_2048	2048x2048	YOLO, COCO, VOC	Detección
dataset_aug_class	224x224	Directory	Clasificación

5.1.2. Modelos y algoritmos

Apoyado en la información obtenida del Estado del Arte, se define el siguiente conjunto de modelos de detección y clasificación para la realización de los experimentos (un resumen

se presenta en la Tabla 5.5):

- **YOLO**: Modelo de detección altamente utilizado en la literatura, actualmente disponible en su versión v8. Para este proyecto, se utilizarán las versiones "v3", "v4tiny", "v5", "v6", "v7" y "v8". Adicionalmente se utilizan las versiones "v5-cls" y "v8-cls", para tareas de clasificación.
- **Faster R-CNN**: Modelo de detección utilizado en menor medida que YOLO, pero que sigue teniendo buenos resultados en comparativas de rendimiento. Para este proyecto, se utilizarán las versiones "X101 FPN" y "R101 FPN".
- **Retinanet**: Modelo de detección menos popular que los anteriores pero que está presente en la revisión del estado del arte. Para este proyecto, se utilizará la versión "X101 FPN"
- **EfficientDetD1**: Modelo de detección preentrenado en ImageNet, utilizado con discreción en la literatura. Para este proyecto, se utilizará la versión "D1"
- **Transformers de detección**: También llamados DETR son modelos de detección que utilizan como algoritmo principal un Visual Transformer, reemplazando a la clásica CNN. Para este proyecto, se utilizarán las versiones "DETR" Y "RT-DETR"
- **SVM**: La máquina de vectores de soporte (*Support Vector Machine*), es un modelo clásico de Aprendizaje Automático que puede ser utilizado para clasificar imágenes. En este proyecto se utilizará como base para comparar su rendimiento con modelos de Aprendizaje Profundo.

Tabla 5.5: Resumen de modelos a utilizar

Modelo	**Backbone**	**Tarea**
YOLOv3	Darknet-53	Detección
YOLOv4tiny	Darknet-53-tiny	Detección
YOLOv5	CSP-Darknet53	Detección
YOLOv6	EfficientRep	Detección
YOLOv7	E-ELAN	Detección
YOLOv8	C2f	Detección
Faster-RCNN-FPN	X101	Detección
Faster-RCNN-FPN	R101	Detección
Faster-RCNN-C4	R101	Detección
Retinanet-FPN	R101	Detección
EfficientDet	D1	Detección
DETR	CNN+ViT	Detección
RT-DETR	CNN+ViT	Detección
YOLOv5-CLS	-	Clasificación
YOLOv8-CLS	-	Clasificación
SVM	-	Clasificación

5.1.3. Preparación de entorno y configuraciones

Para la ejecución de las rutinas de entrenamiento se utiliza el equipo provisto por el Laboratorio CIM, de la Universidad del Bío-Bío. El equipo cuenta con las siguientes características de Hardware:

- Procesador: Intel I9 4Ghz
- Ram: 64 GB
- GPU: Nvidia RTX 3090

Respecto al software se instaló una distribución Linux junto a la herramienta de contenedores Docker, que permite una alta flexibilidad a la hora de realizar experimentos. En detalle se utiliza:

- SO: Linux Mint 21.2 - Ubuntu Jammy
- Docker: Docker Engine 24.0
- Imagen Base: Google Colab

De esta forma, se pueden crear distintos contenedores Docker para ejecutar los modelos con sus respectivas dependencias y versiones. Dado que estos contenedores son desechables, también se crea un almacenamiento compartido para los conjuntos de datos y resultados. Un resumen de la arquitectura del entorno de experimentación se puede observar en la Figura 5.4.

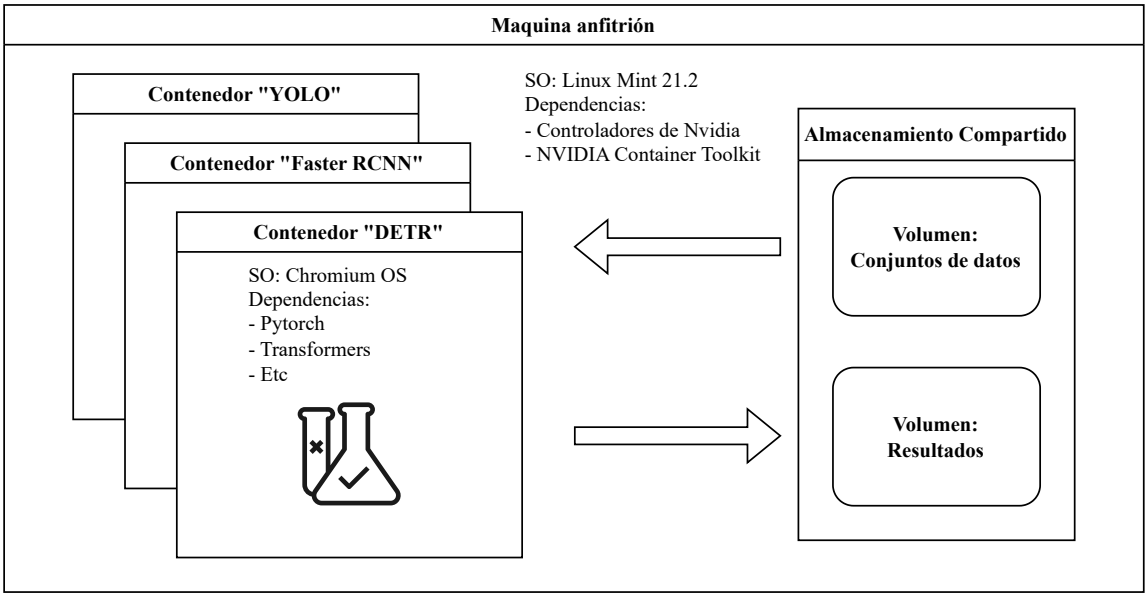


Figura 5.4: Arquitectura del entorno de experimentación

5.2. Diseño de experimentos de detección

Los modelos de detección, tienen dos factores importantes que definirán su rendimiento. Estos son su capacidad de detectar el objeto y su capacidad de clasificar correctamente ese

objeto. La mayoría de los modelos de detección realizan en primer lugar la detección, esto es, encontrar el cuadro delimitador de un posible objeto de interés, luego de realizada esa etapa (generalmente definida por un umbral de confianza) se procede a clasificar el objeto detectado. Esto implica que un modelo puede clasificar correctamente un objeto, pero no detectarlo, y viceversa. Para una correcta interpretación de resultados se busca descartar alguno de estos factores. El caso más sencillo de probar es la clasificación. De esta forma, el obtener buenos resultados en la clasificación de arándanos da paso a que se prueben modelos de detección, centrando exclusivamente el análisis en la capacidad de detección de los modelos. En la Figura 5.5 se muestra un diagrama de actividad de los experimentos a realizar. Donde en una primera etapa se evalúa si los modelos de clasificación pueden clasificar correctamente los arándanos. Si los resultados son favorables, o sea, que los valores de las métricas de rendimiento sean altas, se continúa con la etapa de experimentación con modelos de detección. En esta última etapa, se busca entrenar y evaluar una gran cantidad de modelos para obtener con certeza el mejor. Se considerará suficiente cuando se aborden la mayoría de los modelos encontrados en la literatura o se perciba un aumento mínimo en métricas de rendimiento.

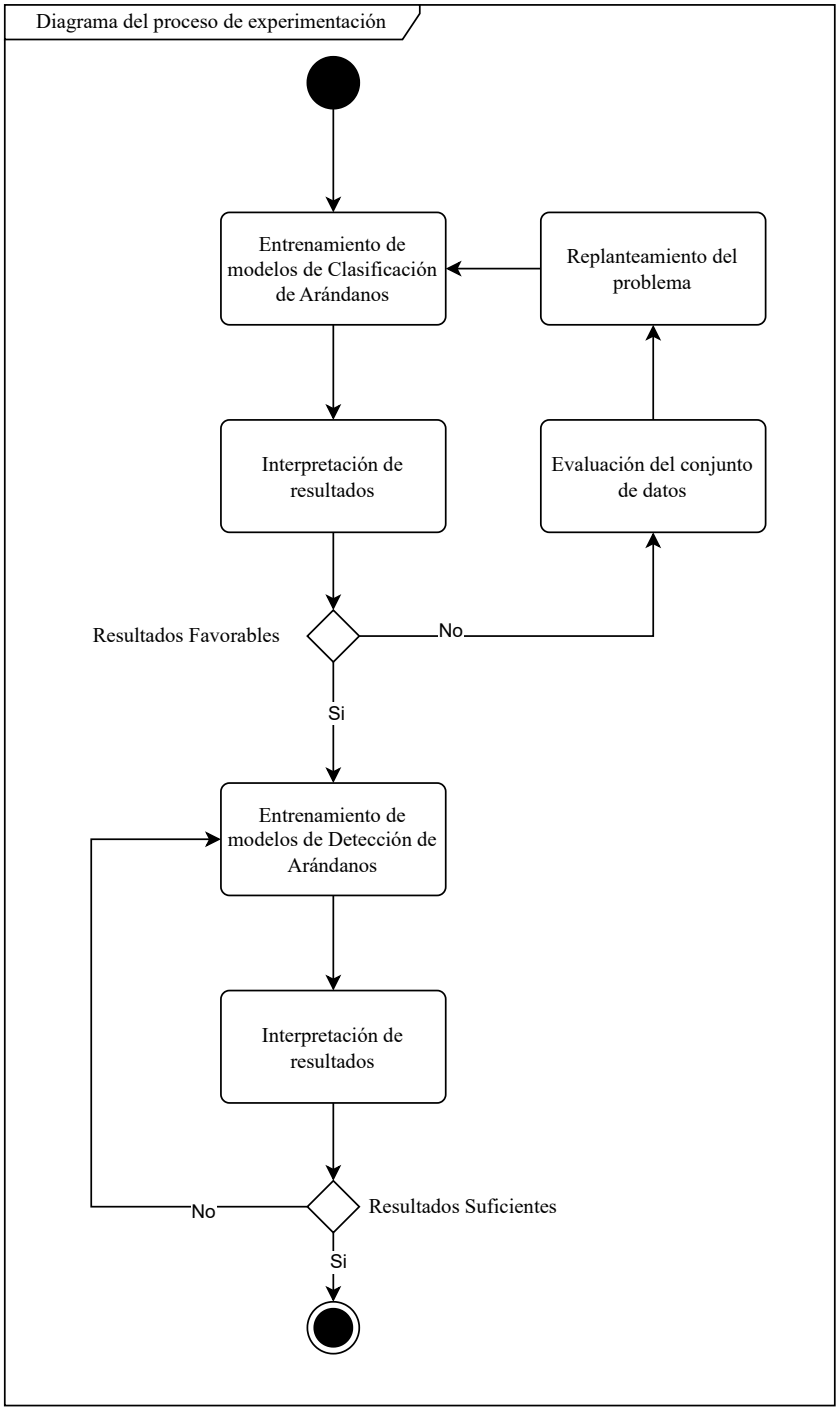


Figura 5.5: Proceso de experimentación

5.3. Implementación y resultados

5.3.1. Modelos de clasificación

En línea con la primera tarea planteada en la sección anterior, se implementaron y entrenaron modelos de clasificación. Con esto se busca confirmar que las redes neuronales son capaces de clasificar la madurez en arándanos. Para esto se utilizará el conjunto de datos

"dataset_aug_class" que consiste en todos los cuadros delimitadores de las imágenes del conjunto de datos "dataset_aug"

Los resultados (Tabla 5.6) indican que los modelos Aprendizaje Profundo pueden detectar con facilidad las distintas fases de maduración de los arándanos. Como se esperaba, la clase con un mayor error es "MMaduro", debido a su baja cantidad de muestras. Aun así, se obtienen buenos resultados. En la Figura 5.6, se puede observar la matriz de confusión del mejor modelo "YOLOv8-cls" donde se puede apreciar de mejor forma el bajo nivel de error del modelo. De la matriz se pueden calcular otras métricas como la Precisión, Exhaustividad y F1, los valores se describen en la Tabla 5.7. La Figura 5.7 presenta una pequeña muestra de una de las iteraciones de entrenamiento del modelo "YOLOv8-cls".

Por otro lado, como se evidencia en la Tabla 5.6, los modelos clásicos de Aprendizaje Automático como SVM, no son capaces de clasificar correctamente los arándanos, quedando muy por detrás de las otras técnicas de Aprendizaje Profundo.

Tabla 5.6: Resultados: Modelos de clasificación

	Exactitud/Acuraccy			
Modelo	**Inmaduro**	**Mmaduro**	**Maduro**	**Total**
SVM	0.08	0.97	0.08	0.38
YOLOV5-CLS	0,97	0,81	0,94	0,94
YOLOV8-CLS	0,95	0,83	0,99	**0,96**

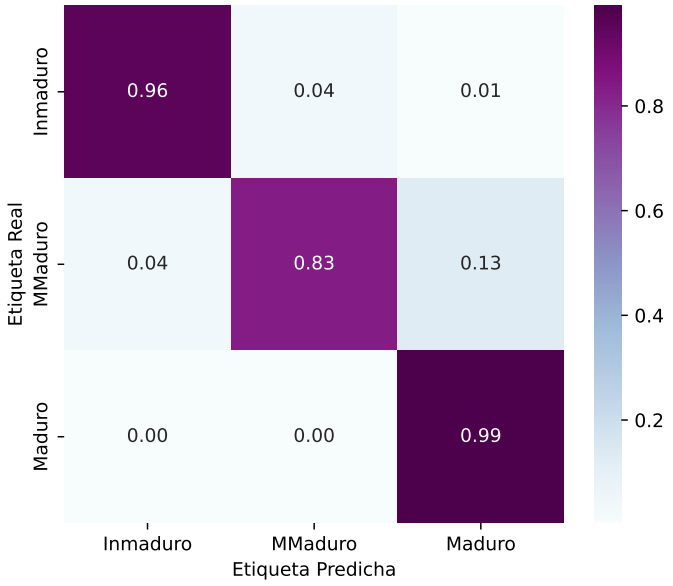


Figura 5.6: Matriz de confusión: Modelo de clasificación "YOLOv8-CLS"

Tabla 5.7: Resultados: Métricas del modelo “YOLOv8-CLS”

Métrica / Clases	Precisión	Exhaustividad	F1	Soporte
Inmaduro	0.9559	0.9882	0.9718	680
Mmaduro	0.8343	0.8294	0.8319	170
Maduro	0.9917	0.9570	0.9741	628

Los resultados indican que los modelos de redes neuronales tienen la capacidad suficiente de clasificar la madurez del fruto de arándano, por lo que, las dificultades que tengan los modelos de detección para analizar las imágenes, probablemente serán inherentes a la tarea de detección en sí, como el tamaño de los objetos, oclusión, entre otras variables.

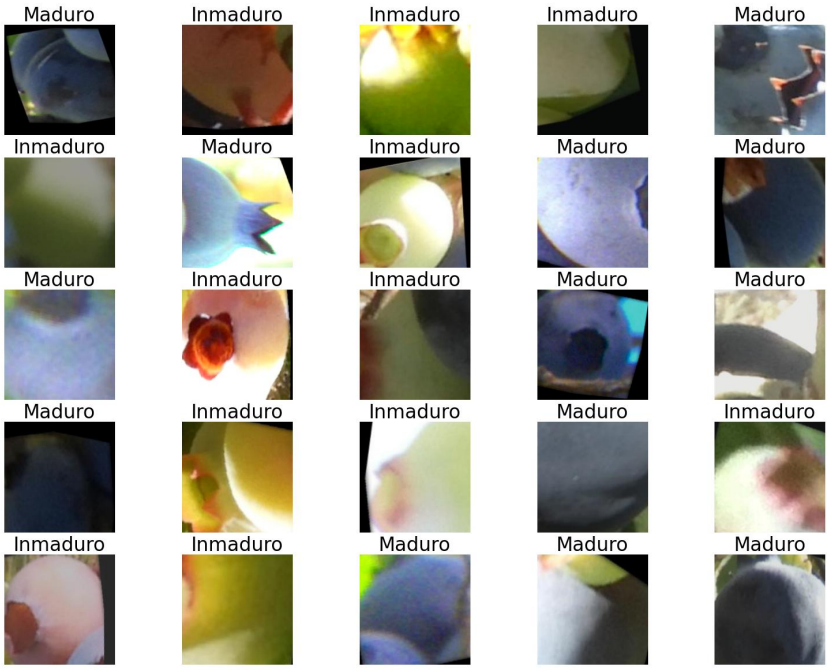


Figura 5.7: Resultados: Modelo de clasificación “YOLOv8-CLS”

5.3.2. Modelos de detección

Ya verificada la capacidad de clasificación de los modelos de redes neuronales, se procede a la implementación de los modelos de detección. En esta etapa se busca verificar la capacidad de detección de distintos modelos vistos en la literatura. Estos abarcan tecnologías desde 2015 (Faster R-CNN) hasta 2023 (YOLOv8) revisando además la capacidad de los nuevos Visual Transformers para este caso de estudio particular.

Etapas de implementación de modelos

La implementación de modelos se puede dividir en al menos 4 etapas, donde se experimenta con una gran cantidad de dependencias y librerías. Es por esto que se hace necesario la separación de entornos vista en la Figura 5.4 para evitar conflictos de versiones. Un resumen de las etapas aquí presentadas se puede ver en la Figura 5.8.

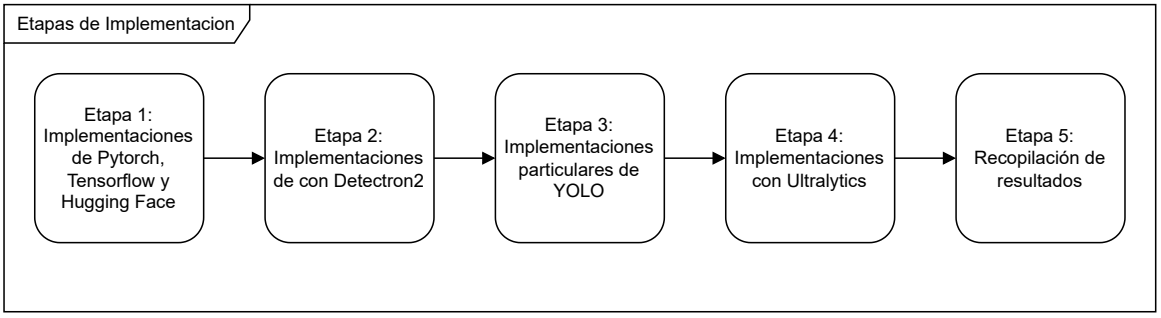


Figura 5.8: Etapas de Implementación

Se comienza realizando experimentos con las funciones y librerías que ofrece oficialmente Pytorch, más específicamente de la librería torchvision. Los modelos implementados en esta iteración fueron distintas versiones preentrenadas de Faster RCNN, con los cuales no se obtuvo buenos resultados. El formato del conjunto de datos usado y compatible con esta librería es el formato PASCAL VOC. Aun así, se continúa utilizando el entrono Pytorch gracias a las utilidades que ofrece. Por otro lado, también se utilizan las herramientas para la detección de objetos que proporciona Tensorflow. Estas requieren la instalación de dependencias desde su repositorio oficial. Al igual que Pytorch, Tensorflow ofrece múltiples modelos preentrenados que pueden ser revisados en [42]. Para este caso se implementa la versión más ligera de EfficientDet, más específicamente la versión D0 entrenado con el conjunto de datos ImageNet. Para ello, se necesitan descargar los modelos preentrenados y crear un archivo de configuración para realizar el proceso de Ajuste Fino. El formato del conjunto de datos necesario para entrenar el modelo es "tfrecord". Algunos hiperparámetros que se deben configurar para entrenar el modelo son:

- Pesos del modelo preentrenado.
- Cantidad de iteraciones de entrenamiento
- Tamaño del lote del conjunto de datos
- Número de clases

Cabe destacar el modelo entrenado en este caso tuvo un tiempo de entrenamiento bastante alto, consumiendo todos los recursos de hardware disponibles el 100 % del tiempo. Esto limitó la experimentación con otras versiones de EfficientDet como D7, ya que el tamaño del modelo, como la cantidad de parámetros entrenables escala rápidamente entre versiones.

Dentro de esta etapa también se implementó un modelo de Visual Transformers de Hugging Face [43], el modelo en particular es DETR, un modelo de Visual Transformers adaptado para la detección de objetos, ya que por defecto estos modelos son de clasificación. Para su uso requiere la instalación de la librería "transformers" en conjunto con algunas dependencias de Pytorch. Como en los casos anteriores, Hugging Face ofrece distintos modelos preentrenados para su descargar y posterior reentrenamiento. El formato necesario del conjunto de datos es COCO. En particular, este modelo no obtiene buenos resultados, pero destaca la incapacidad del modelo de detectar arándanos inmaduros.

En una segunda etapa, para el caso de los modelos de Faster RCNN se utiliza la librería Detectron2 [44], librería oficial de Facebook, quienes en un inicio presentaron estos modelos. Esta librería tiene una gran cantidad de modelos preentrenados, para diferentes casos de uso.

Para este caso particular, se descartan los modelos R-CNN y Fast R-CNN, dado que en las tablas de rendimiento presentadas en la página oficial de Detectron2 se indica que Faster tiene un mejor rendimiento que los modelos antes mencionados. De esta forma se hace mejor uso del tiempo de cómputo disponible.

La implementación de estos modelos requiere de la instalación de dependencias alojadas en el repositorio oficial de Facebook Research. De esta forma, con las herramientas disponibles se pueden descargar las arquitecturas y modelos preentrenados. Cabe destacar que todos los modelos de esta librería están preentrenados en el conjunto de datos COCO. Por consiguiente, para entrenar los modelos, se requiere que el formato de conjunto de datos sea el formato COCO. Entre los hiperparámetros que se deben configurar para entrenar el modelo se encuentran:

- Arquitectura/Modelo preentrenado a utilizar.
- Cantidad de iteraciones de entrenamiento
- Ratio de aprendizaje
- Número de clases

En la tercera etapa, se utilizan las implementaciones particulares de las distintas versiones de YOLO de sus respectivos autores. En detalle los modelos son:

- **YOLOv4 Tiny:** Obtenido de [45], requiere la descarga del código fuente del repositorio y luego compilar la versión del *backbone* Darknet utilizado por YOLOv4 Tiny. Esto es necesario ya que este modelo se destaca por ser bastante ligero, debido a la compactación de las capas convolucionales, a costa de una reducción general en el rendimiento del modelo. El formato necesario para el conjunto de datos es YOLO.
- **YOLOv5:** Obtenido de [46], requiere la instalación de las dependencias del repositorio. Para entrenar, se necesita construir un archivo de configuración donde se especifica la arquitectura del modelo. Luego, utilizando un script de entrenamiento, se definen hiperparámetros como el tamaño de imagen de entrada, número de *epoch's* y pesos preentrenados. El formato necesario para el conjunto de datos es YOLO.
- **YOLOv6:** Obtenido de [47], requiere la descarga de las dependencias del repositorio. El entrenamiento es bastante sencillo, siendo necesario solo la configuración de hiperparámetros básicos (pesos, tamaño de imagen, epoch, etc.). El formato necesario para el conjunto de datos es YOLO.
- **YOLOv7:** Obtenido de [48], requiere la descarga de las dependencias del repositorio. El entrenamiento es bastante similar a YOLOv6, siendo necesario solo la configuración de hiperparámetros. El formato necesario para el conjunto de datos es YOLO.

Luego del entrenamiento de estos modelos, se aprecia que cada vez es más sencilla la implementación de los modelos, esto puede deberse a que, por defecto, las arquitecturas han demostrado ser bastante eficientes, por lo que, el espacio de mejora y personalización recae completamente en tener un conjunto de datos de calidad. En detalle, en [49] se puede apreciar como las diferencias de rendimiento entre modelos YOLO sobre el conjunto de datos COCO, son pequeñas, superándose en algunos casos por pocos puntos de mAP. En estas pruebas de

rendimiento se puede ver como YOLOv4, YOLOv5, YOLOv6 y YOLOv7 obtienen 43.5 %, 55.8 %, 52.5 %, 56.8 % de mAP.

En la cuarta etapa, se completan las implementaciones y entrenamientos con los modelos ofrecidos por la librería Ultralytics, estos solamente requieren que se instalen las dependencias de la librería y configurar hiperparámetros como los *epoch* de entrenamiento, tamaño de imagen y optimizador. En este último caso, se pueden configurar algunas variables del optimizador, como el comportamiento del *Learning Rate*. Aun así, gracias a la información obtenida de la revisión sistemática de literatura, se decide usar en todos los entrenamientos el optimizador "Adam", que determina automáticamente el comportamiento del parámetro antes mencionado.

Los modelos entrenados con esta librería son, YOLOv3, YOLOv8 y RT-DERT. Esta última es una implementación reciente de los Visual Transformers para tareas de detección que ha demostrado un alto rendimiento, especialmente en detecciones en tiempo real.

Debido a las limitaciones de tiempo de cómputo, los experimentos a realizar dependerán completamente del rendimiento de los mismos sobre un caso base. En detalle, todos los modelos serán probados sobre el conjunto de datos "dataset_aug". Experimentos más detallados con los demás conjuntos de datos serán realizados solo sobre los modelos con mejores resultados en la etapa anterior.

Resultados de los modelos de detección

En la última etapa de implementaciones, se recogen todos los resultados del volumen configurado para aquello, además, se almacenan los modelos entrenados para posteriores pruebas. Los resultados obtenidos en esta etapa se presentan en la Tabla 5.8.

Tabla 5.8: Resultados: Modelos de detección

Modelo	**Tamaño de imagen**	**mAP50**	**mAP50-95**
Faster RCNN R101 C4	640x640	0,33	0,18
Faster RCNN X101 FPN	640x640	0,35	0,20
Retinanet R101 FPN	640x640	0,38	0,21
EfficientDetD1	640x640	0,57	0,32
DETR	640x640	0,38	0,18
RT-DERT	640x640	0,82	0,53
YOLOv3	640x640	0,81	0,53
YOLOv4Tiny	640x640	0,71	-
YOLOv5	640x640	0,78	0,47
YOLOv6	640x640	0,80	0,53
YOLOv7	640x640	0,80	0,52
YOLOv8	640x640	***0,83**	0,53

El tiempo de entrenamiento varía entre modelos y porcentaje de utilización del hardware, pero en general, los modelos Faster RCNN y YOLO con el conjunto de datos "dataset_aug" ($\approx$ 1000 imágenes a 640x640) demoran de 3 a 4 horas cada uno. El modelo más "pesado" de entrenar es EfficientDetD1 por su gran cantidad de parámetros y el peso del modelo preentrenado, demorando aproximadamente 7 horas en completar.

Los resultados indican que los modelos de Faster R-CNN no son capaces de detectar correctamente los objetos de las imágenes, obteniendo valores inferiores a 38 % de mAP . Por otro lado, EfficientDetD1 obtiene mejores resultados en su versión más simple (D1), lo que indica que sus versiones más altas, como por ejemplo D7, podría funcionar mucho mejor, sin embargo, estos modelos tienen la desventaja de ser pesados y difíciles de entrenar. Por otro lado, todas las versiones de YOLO demostraron mayor solvencia al detectar arándanos. Todos obteniendo un mAP50 mayor al 70 %. Entre estos modelos, destaca YOLOv8 con un 83 % de mAP50 utilizando imágenes de 640x640. Esto da paso realizar nuevamente el entrenamiento de YOLOv8 con imágenes de mayor resolución. Los resultados de estos entrenamientos se presentan en la Tabla 5.9.

También se destaca el rendimiento de los modelos "YOLOv3" y "RT-DERT". El primero, por ser un modelo relativamente antiguo que logra superar a algunas de las versiones posteriores de YOLO con un 81 % de mAP. El segundo, por ser una implementación de Visual Transformers, Lo que indica que estos modelos son capaces de clasificar correctamente el arándano, casi igualando al mejor modelo de la lista, algo que se había descubierto en la Revisión Sistemática de Literatura.

Tabla 5.9: Resultados: Modelos de detección

Modelo	**Tamaño de imagen**	**mAP50**	**mAP50-95**
YOLOv8	800x800	0,83	0,55
YOLOv8	1280x1280	***0,86**	0,56
YOLOv8	2048x2048	0,83	0,53

Al aumentar la resolución de las imágenes, los entrenamientos se vuelven más complejos y por ende, tardan más en entrenar. En este caso particular de YOLOv8 sobre imágenes de 800x800 el entrenamiento tardo algo más de 5 horas, mientras que sobre imágenes de 1280x1280 tardó cerca de 12 horas, para el caso del modelo de sobre imágenes de 2048x2048 el tiempo se elevó a alrededor de 16 horas.

Los resultados indican que el modelo mejora su rendimiento, especialmente sobre imágenes de 1280x1280 con un 86 % de mAP, pero bajando un poco (83 %) para imágenes de 2048x2048, lo que podría indicar un límite en resolución para entrenar.

En detalle, se pueden ver los resultados de entrenamiento del modelo de YOLOv8 sobre imágenes de tamaño 800x800 (que es el tamaño estándar recomendado en esta última versión de YOLO) se puede ver en la Tabla 5.10. La Figura 5.9 muestra las gráficas que genera automáticamente el modelo YOLO luego de terminado el entrenamiento, donde se pueden apreciar las curvas de ajuste tanto del entrenamiento como de validación y las curvas de aprendizaje de las métricas de Precisión, Exhaustividad (Recall) y mAP. Se puede apreciar que no existe sobreajuste, pues, las curvas de entrenamiento y validación generalmente siguen la misma tendencia a disminuir. Para el caso de las métricas se aprecia una gran subida en los primeros *epoch* de entrenamiento para luego subir de forma discreta.

Respecto a la matriz de confusión (Figura 5.10) podemos ver que las tendencias vistas en los modelos de clasificación también están presentes en los modelos de detección, eso sí, bajando la precisión debido a los procesos extra inherentes a la detección. Se aprecia una mayor confusión con los arándanos maduros, que en algunos casos son confundidos con el fondo, también llamado *background* en inglés. Esta nueva columna en la matriz tiene información esencial sobre el entrenamiento que será discutida en la sección de discusión de

resultados (Sección 6). En la Figura 5.11 se puede ver la matriz de confusión, del modelo YOLOv8 entrenado en imágenes de 1280x1280 que obtiene mejores resultados que el anterior.

Tabla 5.10: Curvas de entrenamiento del modelo "YOLOv8" en imágenes de 800x800

Clase	**Precisión**	**Exhaustividad**	**mAP50**	**mAP50-95**
All	0,832	0,751	0,83	0,547
Inmaduro	0,887	0,72	0,839	0,542
Mmaduro	0,8	0,726	0,78	0,531
Maduro	0,809	0,808	0,872	0,569

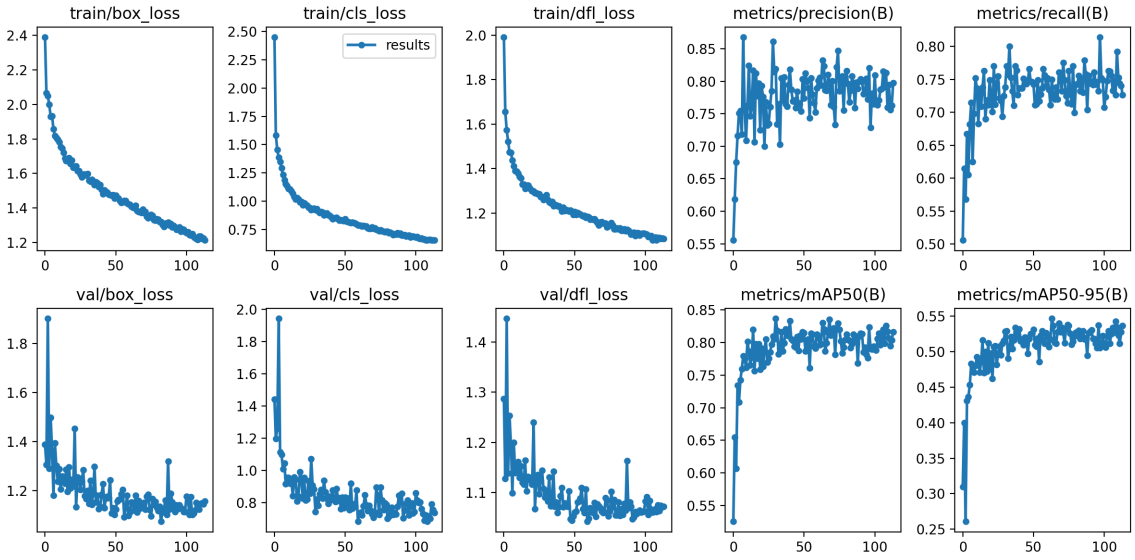


Figura 5.9: Resultados del modelo "YOLOv8" en imágenes de 800x800

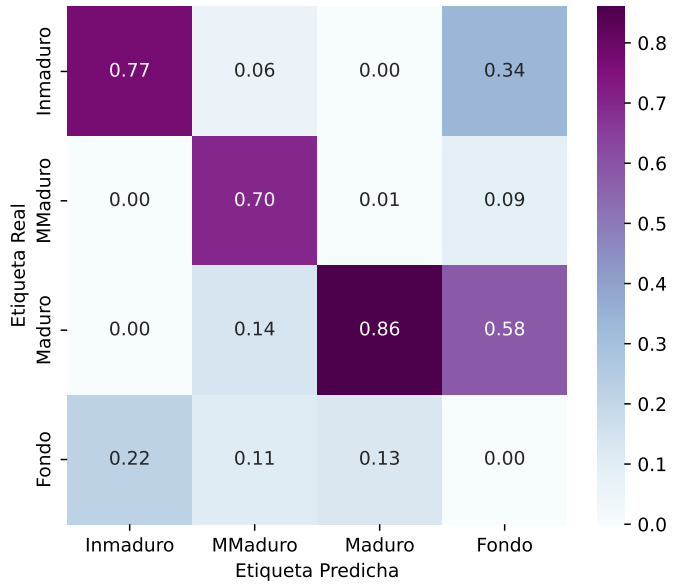


Figura 5.10: Matriz de confusión: Modelo de detección "YOLOv8" en imágenes de 800x800

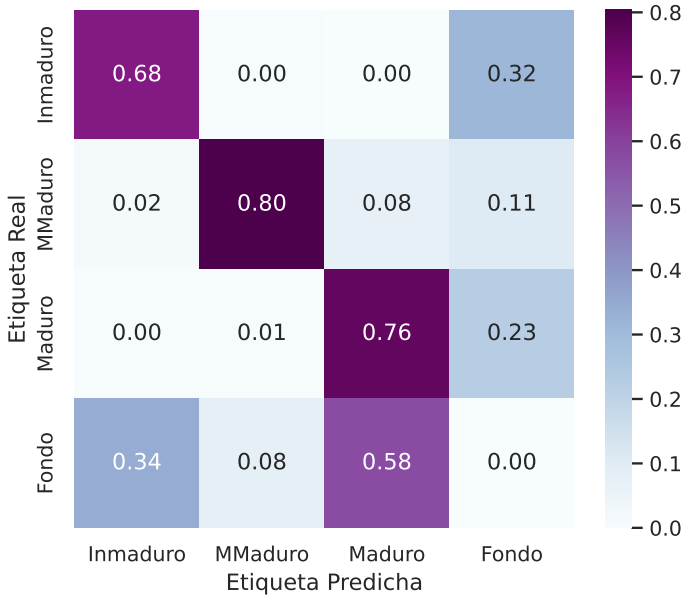


Figura 5.11: Matriz de confusión: Modelo de detección "YOLOv8" en imágenes de 1280x1280

Capítulo 6

Discusión de resultados

6.1. Respecto al rendimiento general y comparativas

Con los resultados de los modelos presentados en la sección anterior, se puede apreciar que los modelos de Aprendizaje Profundo son capaces de clasificar la madurez en arándanos. Si bien existe un desbalance en las clases del conjunto de datos, las métricas de evaluación son lo suficientemente buenas, incluso para la clase desbalanceada, "MMaduro". En detalle, los mejores tres modelos son YOLOv8 seguido de cerca de RT-DERT y YOLOv3. De estos modelos se puede destacar como YOLOv3 se mantiene entre los mejores rendimientos a pesar de ser un modelo considerablemente más antiguo. YOLOv8 obtiene los mejores resultados debido probablemente a ser el más reciente y depurado de los modelos YOLO. Por último, RT-DETR también obtiene buenos resultados, destacando por ser el único de los tres que es un Visual Transformers.

En la revisión sistemática de literatura se identificaron cuatro artículos que realizaban análisis de frutos de arándanos. Específicamente en [50],[51], [52] y [53]. De estos, solo uno de ellos ([51]) realiza detección de arándanos de forma similar a lo presentado en este proyecto. Las diferencias radican en que los arándanos, denominados "Arándanos Salvajes", están etiquetados en tres clases, "Blue Fruit", "Red Fruit" e "Green Fruit". que coinciden con las etiquetas utilizadas en este proyecto "Maduro", "Medio Maduro" e "Inmaduro" respectivamente. Otra diferencia es que el ángulo de captura es aérea vertical, ya que el campo parece no tener espacio para tomar capturas laterales.

En este estudio el modelo que obtuvo mejores resultados es YOLOv4 que en comparación con el mejor modelo de este proyecto tiene un menor rendimiento. En detalle, la corporación entre ambos modelo se presenta en la tabla 6.1

Tabla 6.1: Comparación con estudios

Modelo / Estudio	**Inmaduro / Green Fruit**	**Medio Maduro / Red Fruit**	**Maduro / Blue Fruit**	**mAP Total**
YOLOv4 en [51]	0.80	0.69	0.90	0.80
YOLOv8 en este proyecto	0,83	0,78	0,87	0,83

Considerando que los casos de estudio son ligeramente distintos en esta comparación, se puede inferir que los resultados obtenidos están en línea con lo revisado en la literatura,

inclusive algunos puntos mejor. Además, se puede observar la misma tendencia de precisión, donde los arándanos maduros tienen mejores porcentajes de mAP, seguido de los arándanos inmaduros y medio maduros, siendo estos últimos los que más se alejan en ambos casos. En particular, para solventar el problema del bajo rendimiento de los arándanos medio maduros, en el artículo se decidió construir un nuevo conjunto de datos sobrescribiendo las etiquetas, de esta forma el entrenamiento se realiza sobre dos clases, arándanos maduros ("Blue Fruit") y arándanos inmaduros ("Red Fruit" y "Green Fruit").

En el caso particular de este proyecto, los arándanos etiquetados como "MMaduros" también tienen un rendimiento menor, principalmente debido al desbalance de esa clase. Es por esto que adicionalmente, se realiza un experimento para comprobar el rendimiento de YOLOv8 sobre dos clases, siguiendo la misma técnica presentada en el artículo. Esto es, sobrescribiendo "MMaduro" por "Inmaduro". En concreto, los resultados obtenidos en este experimento son los presentados en la Tabla 6.2.

Se observa un leve incremento en el mAP sobre todas las clases e "Inmaduro" y casi nulo sobre "Maduro". Esto indica que la omisión de la etiqueta desbalanceada no resulta en mayores diferencias a la hora de evaluar los modelos.

Tabla 6.2: Resultados del entrenamiento sobre dos clases

Clase	**Precisión**	**Exhaustividad**	**mAP50**	**mAP50-95**
Todos	0,85	0,76	0,86	0,57
Inmaduro	0,88	0,74	0,85	0,55
Maduro	0,83	0,79	0,87	0,58

Respecto a la velocidad de inferencia general, como se puede intuir, es más lenta en modelos antiguos. Esto debido a la constante optimización que reciben versión tras versión. En concreto, los tiempos de inferencia para el modelo YOLOv3, YOLOv5 y YOLOv8 realizadas sobre el conjunto de pruebas con imágenes de 640x640, puede ser revisada en la Tabla 6.3. Estas mediciones se realizaron tanto con CPU (Intel Xeon 2.20GHz) como con GPU (Nvidia T4). Además, se muestra el tamaño que ocupan en memoria estos modelos. Estas mediciones son estimativas ya que dependen completamente del hardware utilizado y el tamaño de imagen.

Si bien YOLOv5 es un poco más rápido que YOLOv8, la diferencia de precisión que hay entre estos dos modelos (0.78 de mAP en YOLOv5 y 0.83 de mAP en YOLOv8) hace a YOLOv8 un mejor modelo en relación velocidad/precisión. Además, las librerías utilizadas para entrenar YOLOv8 permite también su exportación a distintivas opciones que pueden ser utilizadas para el despliegue de aplicaciones.

Tabla 6.3: Tiempo de inferencia sobre imágenes de 640x640

Modelo	**Velocidad con GPU**	**Velocidad con CPU**	**Tamaño**
YOLOv8	16,3 ms	675,6 ms	21,5 mb
YOLOv5	12,6 ms	644,3 ms	14,2 mb
YOLOv3	55,4 ms	4856,1 ms	198,1 mb

6.2. Respecto a las causas que afectan a la detección

En general, se observa una disminución en la capacidad de análisis de los arándanos entre los modelos de clasificación y detección. Esto debido a las múltiples variables extras que afectan a la tarea de detección. Algunas causas que se intuyen son las siguientes:

Tamaño

En el caso de la clasificación, todas las imágenes que entran al modelo tienen el mismo tamaño (224x224). Esto puede ser beneficioso, ya que el modelo puede centrarse en las características particulares del arándano. En cambio, en el modelo de detección, la diferencia de profundidad de los distintos arándanos presentes en la imagen, resultan en una variedad de tamaños, que, a valores más pequeños, pueden ser difíciles de clasificar.

Inexistencias

Como se ve en las matrices de confusión de las Figuras 5.10 y 5.11 hay una gran cantidad de arándanos que se identifican en la imagen, que no están en el conjunto de datos original, dando como resultado que el modelo las defina como "Fondo". En particular, cuando las detecciones se centran en la columna "Fondo" de las etiquetas predichas, se consideran detecciones de Falsos Negativos, es decir, el modelo detecta como fondo, arándanos que se encuentran etiquetados en el conjunto de datos. Esto puede deberse a dos dificultades. En primer lugar, los arándanos inmaduros pueden confundirse con hojas debido a su color, y en segundo lugar, los arándanos maduros pueden no ser detectados debido a las diferentes iluminaciones presentes en el conjunto de datos, pudiendo confundir arándanos con sombras o lugares más oscuros. Por otro lado, cuando las detecciones se centran en la columna "Fondo" de las etiquetas reales, se consideran detecciones de Falsos Positivos, es decir, el modelo detecta un arándano en cualquier categoría, pero este no está presente en el conjunto de datos. Esto provoca que las métricas de evaluación consideren esto como un error. Una ilustración de lo explicado anteriormente se encuentra en la Figura 6.1. Al revisar las matrices de confusión del modelo YOLOv8 tanto en imágenes de 800x800 como de 1280x1280 se puede apreciar como la cantidad de Falsos Negativos baja y los Falsos Positivos sube, lo que indica que a mayor resolución el modelo es capaz de encontrar de mejor forma los arándanos originalmente etiquetados y además encontrar otros que no lo estaban.

Esto a su vez está mezclado con la capacidad real de los modelos de detección para encontrar objetos, demostrando que en esta tarea existen muchas variables que afectan al rendimiento en papel. En la Figura 6.2 se pueden dos versiones de la misma imagen, una con las etiquetas reales (Figura 6.2.a) y la otra con las etiquetas predichas por el modelo (Figura 6.2.b).

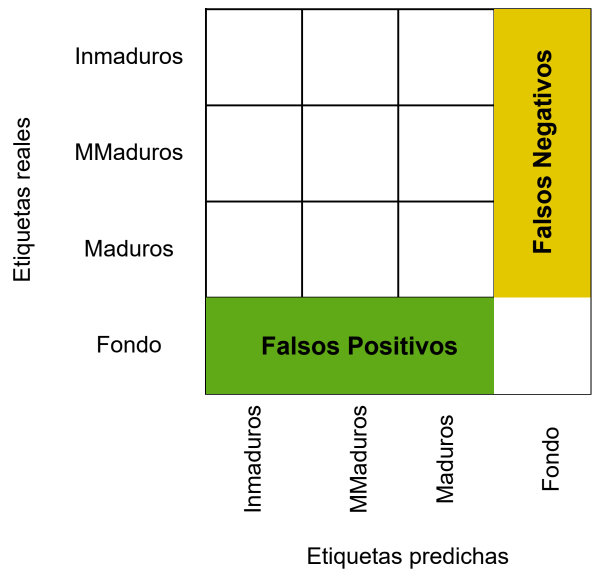
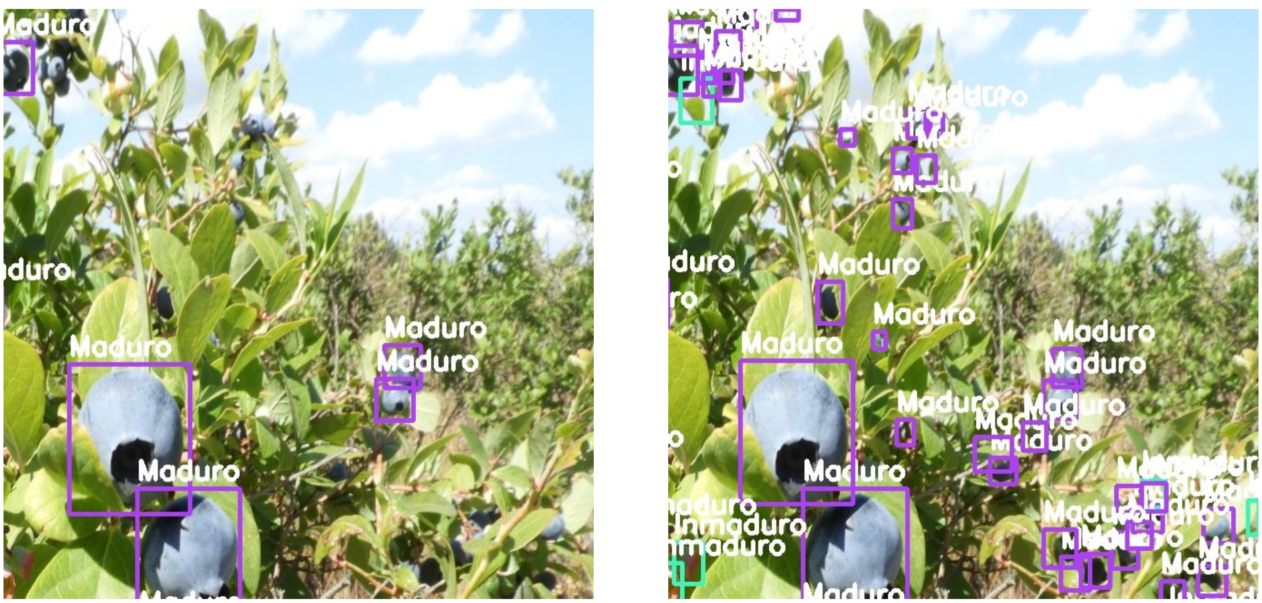


Figura 6.1: Interpretación de la matriz de confusión para la detección



(a) Imagen con etiquetas reales

(b) Imagen con etiquetas predichas

Figura 6.2: Diferencias entre las etiquetas reales y las predichas: Arándanos no etiquetados

Calidad del etiquetado

Otro resultado interesante de los entrenamientos fue la diferencia entre mAP50 y mAP50-95, esto es debido a que, las predicciones, en términos del tamaño del cuadro delimitador, no coinciden perfectamente con la etiqueta real. Un ejemplo de esto se puede ver en la Figura 6.3. En detalle mAP50 mide cuantos arándanos son clasificados correctamente coincidiendo el cuadro delimitador predicho con el real en al menos un 50 %, por otro lado, mAP50-95 va aumentando el umbral de coincidencia, por lo que generalmente se espera que el valor de la

métrica sea menor. Esta diferencia puede deberse a las representaciones erróneas del modelo, pero si se analiza más a fondo (Como por ejemplo la Figura 6.4) se puede apreciar como el cuadro delimitador predicho, (Figura 6.4.a) es más representativo que el cuadro delimitador real (Figura 6.4.b). Esto porque se ajusta de mejor forma al objeto, obteniendo un cuadro delimitador mínimo, mientras que, en la etiqueta real, tiene una etiqueta más holgada, con más espacio entre el cuadro y el objeto.

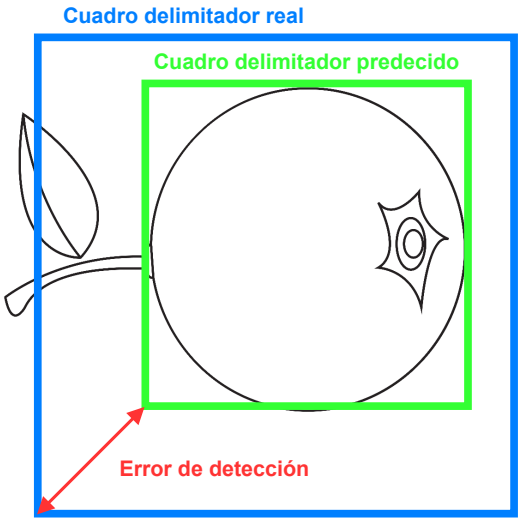
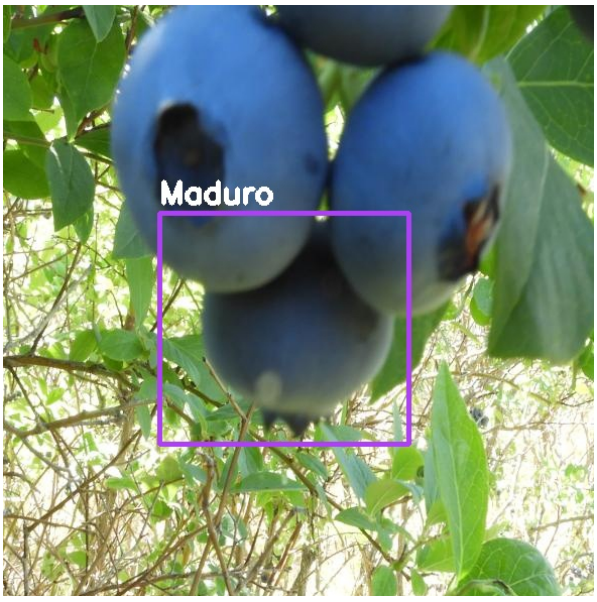
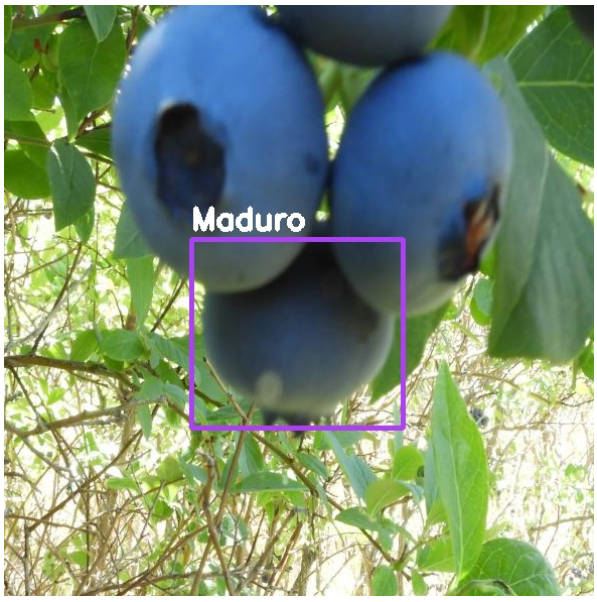


Figura 6.3: Ilustración de un error de detección encontrado



(a) Imagen con etiqueta real



(b) Imagen con etiqueta predicha

Figura 6.4: Diferencias entre las etiquetas reales y las predichas: Cuadro delimitador

6.3. Visual Transformers en arándanos

A pesar de no obtener el mejor resultado para este caso de estudio, los Visual Transformers tienen otras aristas que pueden ser de gran utilidad para esta tarea. En particular, sé probo el modelo SAM (Segment Anything Model) [54], un modelo que usa el codificador de los Visual Transformers y otros elementos para segmentar imágenes de forma no supervisada, esto es, que no necesita de entrenamiento previo para realizar la tarea. Esto puede ser de bastante utilidad para esta tarea en particular, ya que podría detallar aún más la detección de los arándanos. Para revisar la efectividad de esa segmentación sé probó el modelo con una muestra del conjunto de pruebas, dando como resultado lo presentado en la Figura 6.5

(a) Imagen original

(b) Imagen segmentada por SAM

Figura 6.5: SAM: Segmentación no supervisada

6.4. Comparación con usuarios reales

Un análisis interesante es evaluar la capacidad de los modelos frente a personas reales. Es por ello por lo que se construye una pequeña encuesta que consiste etiquetar un pequeño conjunto de imágenes. En esta encuesta se mide principalmente la velocidad del encuestado para encontrar todos los arándanos presentes en la imagen e identificar su nivel de madurez. De esta forma se puede comparar la velocidad del modelo con la del encuestado. En la Figura 6.6 se muestra la ejecución de la encuesta. La encuesta puede ser revisada en el siguiente enlace: https://colab.research.google.com/drive/1_JfA9J48mxshopmRRGr0kGEJBP9sZHlG.

Cabe destacar que se espera que el modelo sea mucho más rápido, ya que, no es directamente comparable la velocidad de identificación del ojo humano, con el tiempo que demora el encuestado en etiquetar una imagen, considerando el arrastre del ratón en cada etiquetado. Sin embargo, se pueden utilizar como referencia, además de encontrar otras aristas de comparación, como la cantidad de frutos encontrados.

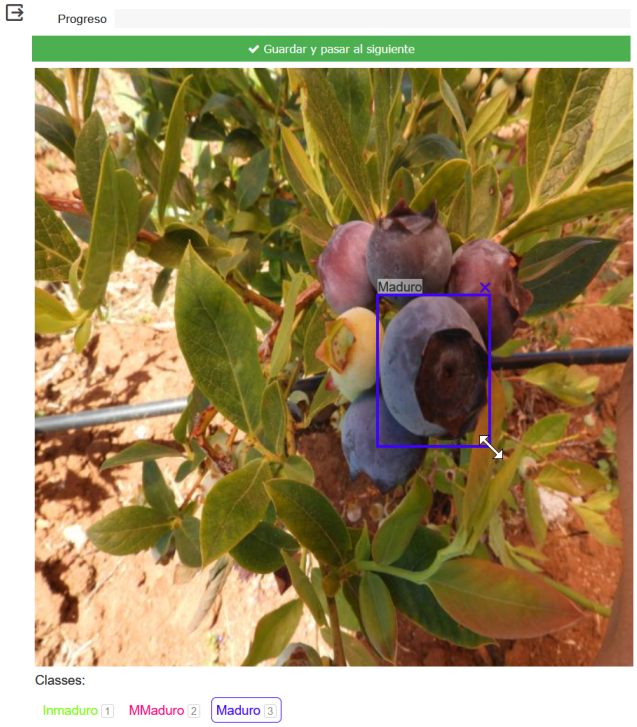


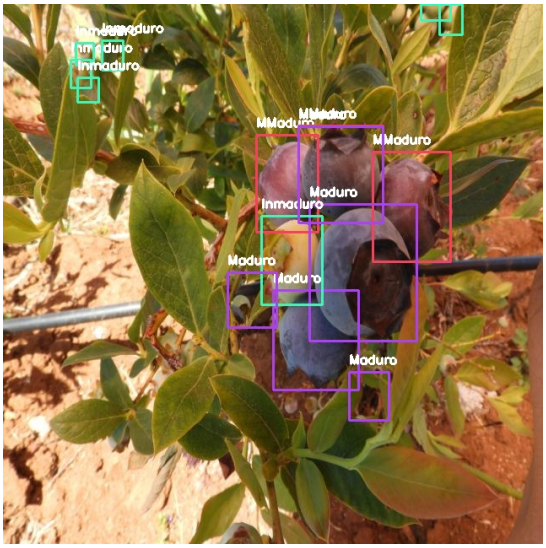
Figura 6.6: Encuesta de etiquetado de arándanos

Los resultados preliminares de la encuesta se indican en la Tabla 6.4. Como se esperaba, el modelo de detección obtiene un tiempo mucho menor a la hora de detectar arándanos.

Tabla 6.4: Tiempo de respuesta medido en segundos de los encuestados y el modelo "YOLOv8" en imágenes de 640x640

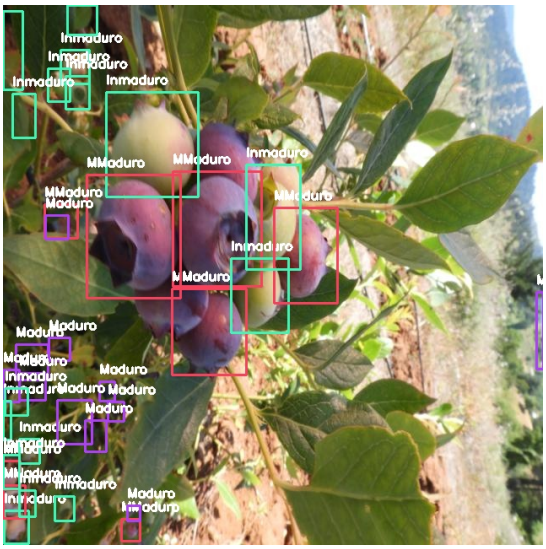
Imagen	**Modelo**	**Experto 1**	**Experto 2**	**Experto 3**	**Experto 4**	**Experto 5**	**Experto 6**	**Experto 7**
Imagen 1	0,0077	48,19	52,31	37,31	71,73	63,10	67,00	23,38
Imagen 2	0,0074	39,33	33,57	15,18	35,76	26,71	52,52	12,93
Imagen 3	0,0074	29,52	33,28	12,38	23,56	23,30	21,89	25,83
Imagen 4	0,0074	43,67	82,95	29,78	42,28	39,72	30,61	23,31
Imagen 5	0,0104	48,40	71,23	15,75	20,30	21,24	24,56	17,05

Si bien como se mencionó antes, el tiempo es referencial, un análisis interesante es comparar la cantidad de arándanos que los expertos encontraron en las imágenes con la cantidad que puede encontrar el modelo. En la Figura 6.7 se muestran tres imágenes evaluadas en la encuesta, en la primera columna, se puede ver las imágenes con los cuadros delimitadores predichos por el modelo YOLOv8 entrenado con imágenes de 640x640, mientras en la segunda columna están las imágenes con el etiquetado de los expertos.

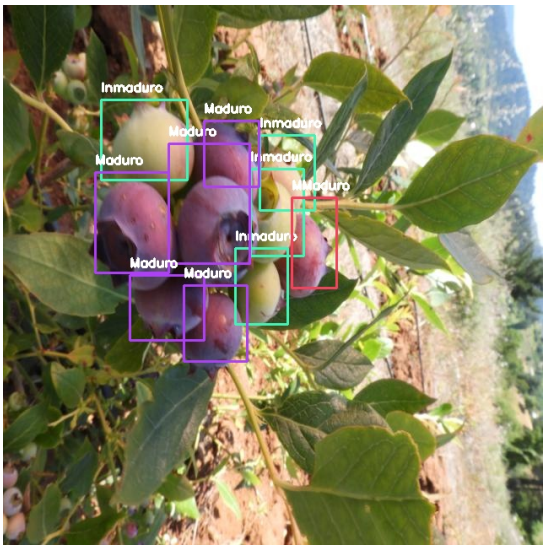


(a) Imagen con etiquetas predecidas

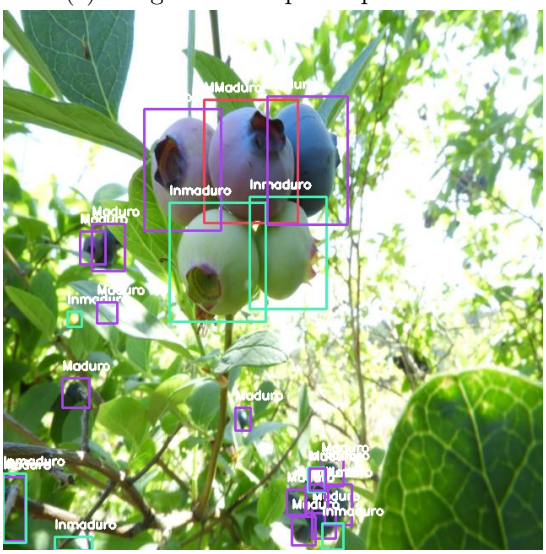
(b) Imagen con etiquetas del Experto 1



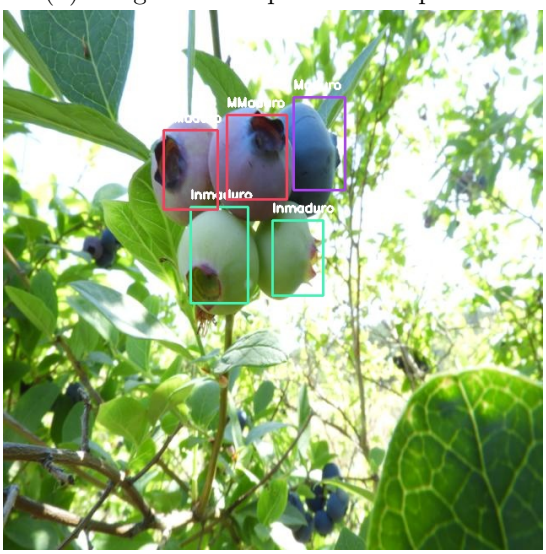
(c) Imagen con etiquetas predecidas



(d) Imagen con etiquetas del Experto 2



(e) Imagen con etiquetas predecidas



(f) Imagen con etiquetas del Experto 3

Figura 6.7: Modelo vs Expertos: Cuadros Delimitadores

Tabla 6.5: Cantidad de cuadros delimitadores etiquetados por expertos (en promedio), el modelo y las etiquetas reales

Etiquetas / **Imagen**	**Promedio**	**Predichos**	**Real**
Imagen 1	6	10	15
Imagen 2	5	5	9
Imagen 3	4	6	6
Imagen 5	9	20	17
Imagen 5	6	16	13

.

En la Tabla 6.5 se puede observar la cantidad de cuadros delimitadores etiquetados en promedio por los expertos junto a la cantidad de etiquetas predichas y las reales. Cabe destacar que el modelo utilizado en este caso es YOLOv8 en imágenes de 640x640, ya que en este tamaño de imagen se realizó la encuesta. De esta tabla se puede inferir que en general el modelo se acerca más a la cantidad de arándanos presentes en la imagen que el promedio de expertos, esto puede deberse especialmente a los arándanos más pequeños que pueden ser más difíciles de identificar.

En resumen, los modelos de Aprendizaje Profundo evaluados, tanto de clasificación como de detección, demuestran que es posible analizar el fruto del arándano y determinar su madurez. Aun así, existe un margen de mejora, que, si bien puede ser alcanzada con avances en los modelos, la principal variable a tener en cuenta a futuro es la calidad de conjunto de datos.

Conclusiones y trabajo futuro

Con base en los resultados obtenidos a lo largo del proyecto se puede concluir lo siguiente:

Respecto a los objetivos de proyecto, estos fueron correctamente alcanzados y documentados. En detalle:

1. Gracias a la revisión sistemática de literatura, se identificaron múltiples tendencias en el campo de la clasificación de frutos con técnicas de Aprendizaje Profundo. Entre estos, la identificación de los modelos más utilizados actualmente y sus correspondientes métricas de evaluación. También se identificaron las características de los conjuntos de datos para cada tarea en particular, además de herramientas y técnicas utilizadas en esta área. El trabajo realizado en este proceso culminó en la realización de un artículo de Revisión Sistemática de literatura que actualmente se encuentra aceptado y publicado en la revista IEEE Access. En concreto, el artículo tiene por nombre *Analysis of Fruit Images with Deep Learning: A Systematic Literature Review and Future Directions* y puede ser encontrado en el siguiente enlace: https://ieeexplore.ieee.org/document/10368014.

2. En cuanto a la evaluación de modelos, estos fueron correctamente implementados y entrenados. Se destaca la gran cantidad de modelos implementados que fueron seleccionados gracias a las tendencias vistas en la revisión sistemática de literatura. Con esto se puede asegurar tanto la rigurosidad de la experimentación como el hallazgo del mejor modelo para esta tarea y caso de estudio particular.

3. En cuanto a la evaluación de modelos, se evaluó el rendimiento de cada uno de ellos basado las métricas de evaluación correspondientes que también fueron encontradas en la revisión sistemática de literatura. Además, se compararon los resultados del mejor modelo implementado en este proyecto con otros presentes en la literatura, concluyendo que el rendimiento obtenido están en línea con lo esperado en este caso de estudio. Se realizó una última comparación con los resultados de una encuesta realizada a personas reales, resultando en un modelo que es más rápido y en ciertas situaciones, más preciso.

Con el cumplimiento de este último objetivo, se pudieron confirmar las hipótesis planteadas para este proyecto, donde los modelos de Aprendizaje Profundo entrenados para el reconocimiento de arándanos son capaces de clasificar correctamente los arándanos y sus distintas fases de madurez, al menos en un 86 % de mAP. Además, basándose en encuestas a personas reales y valores de tiempo estimativos, se puede comprobar que los modelos de detección son más rápidos y detallados al analizar imágenes de arándanos.

Respecto a la lineas de investigación futura se puede decir los siguiente:

En base a lo discutido en la Sección 6 se puede decir que el principal problema de este proyecto, al igual que el común de otros con objetivos similares, es el conjunto de datos. Si bien se pueden obtener buenos resultados con el conjunto de datos disponibles, un aumento en el tamaño y principiante un mejor etiquetado de los mismos, mejoraría en gran medida los resultados obtenidos. Existe una capacidad de detección de los modelos aquí implementados que no está siendo considerada en las métricas de evaluación, dado que, realizando pruebas en detalle, el modelo es capaz de reconocer arándanos que no están inicialmente etiquetados.

Por otro lado, la limitación de los recursos de cómputo limita la cantidad de experimentos que se pueden realizar. En particular, el entrenamiento de modelos más pesados, como las versiones más grandes de EfficentNet y entrenamientos con resoluciones más altas, podrían arrojar más resultados favorables que los indicados en este documento. En este proyecto se demostró, como a resoluciones más altas el rendimiento, tanto en términos cuantitativos basados en métricas y cualitativos basados en pruebas detalladas, aumenta en gran medida, especialmente desde imágenes de 800x800 a 1280x1280.

Por otro lado, el uso de Visual Transformes demostró que el rendimiento de estos es casi tan bueno como el rendimiento de las “clasicas” CNN, y que, si bien no superaron a estos últimos, sí pueden ser utilizados para otros tipos de análisis, como la segmentación de imagen. Lo que podría cambiar el enfoque de la tarea y en cómo se está abordando actualmente.

Bibliografía

[1] Gac, O., Gyllen, O., y Olate, S., "Informe Exportaciones No Cobre Junio 2023," rep. tec., ProChile, 2023.

[2] "Frutas frescas y procesadas.", https://www.odepa.gob.cl/rubros/frutas-frescas-y-procesadas (visitado el 2023-06-23).

[3] Kalt, W., Cassidy, A., Howard, L. R., Krikorian, R., Stull, A. J., Tremblay, F., y Zamora-Ros, R., "Recent Research on the Health Benefits of Blueberries and Their Anthocyanins," Advances in Nutrition, vol. 11, pp. 224–236, 2020, doi:10.1093/advances/nmz065.

[4] IBO, "Global state of the Blueberry - Industry Report 2022," rep. tec., International Blueberry Organization, 2022.

[5] Gonzalez, S., "Blueberry Annual Voluntary," rep. tec., United States Department of Agriculture, 2022. titleTranslation:.

[6] Ullo, S. L. y Sinha, G. R., "Advances in IoT and Smart Sensors for Remote Sensing and Agriculture Applications," Remote Sensing, vol. 13, p. 2585, 2021, doi:10.3390/rs13132585.

[7] Moreda, G., Ortiz-Cañavate, J., García-Ramos, F., y Ruiz-Altisent, M., "Non-destructive technologies for fruit and vegetable size determination – A review," Journal of Food Engineering, vol. 92, pp. 119–136, 2009, doi:10.1016/j.jfoodeng.2008.11.004.

[8] https://magnet.cl, "Plan Sequía - Gob.cl.", https://www.gob.cl/plansequia/ (visitado el 2023-06-24).

[9] Council, U. H. B., "How Blueberries Grow," 2023, https://blueberry.org/about-blueberries/how-blueberries-grow/ (visitado el 2023-07-28).

[10] Keele, S. y others, "Guidelines for performing systematic literature reviews in software engineering," rep. tec., Technical report, ver. 2.3 ebse technical report. ebse, 2007.

[11] Kitchenham, B., "Procedures for performing systematic reviews," Keele, UK, Keele University, vol. 33, no. 2004, pp. 1–26, 2004.

[12] Dwyer, J. y Nelson, B., "Roboflow: Give your software the power to see objects in images and video," 2023, https://roboflow.com/ (visitado el 2023-05-26). titleTranslation:.

[13] Van Rossum, G. y Drake, F. L., Python 3 Reference Manual. Scotts Valley, CA: CreateSpace, 2009.

[14] McKinney, W. y others, "Data structures for statistical computing in python," en Proceedings of the 9th Python in Science Conference, vol. 445, pp. 51–56, Austin, TX, 2010.

[15] Harris, C. R., Millman, K. J., Walt, S. J. v. d., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk,

M. H. v., Brett, M., Haldane, A., Río, J. F. d., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., y Oliphant, T. E., "Array programming with NumPy," Nature, vol. 585, pp. 357–362, 2020, doi:10.1038/s41586-020-2649-2. Publisher: Springer Science and Business Media LLC.

[16] Hunter, J. D., "Matplotlib: A 2D graphics environment," Computing in Science & Engineering, vol. 9, no. 3, pp. 90–95, 2007, doi:10.1109/MCSE.2007.55. Publisher: IEEE COMPUTER SOC.

[17] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Jia, Y., Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, y Xiaoqiang Zheng, "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," 2015, https://www.tensorflow.org/.

[18] Chollet, F. y others, "Keras," 2015, https://keras.io.

[19] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., y Chintala, S., "PyTorch: An Imperative Style, High-Performance Deep Learning Library," en Advances in Neural Information Processing Systems 32, pp. 8024–8035, Curran Associates, Inc., 2019, http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[20] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., y Duchesnay, E., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[21] Russell, S. y Norvig, P., Artificial Intelligence A Modern Approach. Pearson Education, 3ra ed., 2016.

[22] Farnham, B., Tokyo, S., Boston, B., Sebastopol, F., y Beijing, T., Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow Concepts, Tools, and Techniques to Build Intelligent Systems. O'Reilly Media, 2nd ed., 2019.

[23] NVIDIA Developer, N. C., "Deep Learning | NVIDIA Developer," 2020, https://developer.nvidia.com/deep-learning.

[24] Rosenblatt, F., "The perceptron: a probabilistic model for information storage and organization in the brain.," Psychological review, vol. 65, no. 6, p. 386, 1958. Publisher: American Psychological Association.

[25] Minsky, M. y Papert, S., "An introduction to computational geometry," Cambridge tiass., HIT, vol. 479, p. 480, 1969.

[26] Aggarwal, C. C., An Introduction to Neural Networks. Springer International Publishing, 2018, doi:10.1007/978-3-319-94463-0_1. Publication Title: Neural Networks and Deep Learning: A Textbook.

[27] M, H. y M.N, S., "A Review on Evaluation Metrics for Data Classification Evaluations," International Journal of Data Mining & Knowledge Management Process, vol. 5, pp. 01–11, 2015, doi:10.5121/ijdkp.2015.5201.

[28] Zheng, Z., Wang, P., Liu, W., Li, J., Ye, R., y Ren, D., "Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression," Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 12993–13000, 2020, doi:10.1609/aaai.v34i07.6999.

[29] Olivas, E. S., Guerrero, J. D. M., Martinez-Sober, M., Magdalena-Benedito, J. R., Serrano, L., y others, Handbook of research on machine learning applications and trends: Algorithms, methods, and techniques. IGI global, 2009.

[30] Chollet, F., Deep learning with Python. Shelter Island: Manning Publications, second edition ed., 2021.

[31] Fukushima, K., "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," Biological Cybernetics, vol. 36, pp. 193–202, 1980, doi:10.1007/BF00344251.

[32] LeCun, Y., Haffner, P., Bottou, L., y Bengio, Y., "Object Recognition with Gradient-Based Learning," en Shape, Contour and Grouping in Computer Vision, vol. 1681, pp. 319–345, Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, doi:10.1007/3-540-46805-6-19.

[33] Simonyan, K. y Zisserman, A., "Very Deep Convolutional Networks for Large-Scale Image Recognition," 2015, http://arxiv.org/abs/1409.1556 (visitado el 2022-07-05).

[34] He, K., Zhang, X., Ren, S., y Sun, J., "Deep Residual Learning for Image Recognition," 2015, http://arxiv.org/abs/1512.03385 (visitado el 2022-07-05).

[35] Girshick, R., Donahue, J., Darrell, T., y Malik, J., "Rich feature hierarchies for accurate object detection and semantic segmentation," 2014, http://arxiv.org/abs/1311.2524 (visitado el 2022-07-05).

[36] Girshick, R., "Fast R-CNN," 2015, http://arxiv.org/abs/1504.08083 (visitado el 2022-07-05).

[37] Ren, S., He, K., Girshick, R., y Sun, J., "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," 2016, http://arxiv.org/abs/1506.01497 (visitado el 2022-07-05).

[38] He, K., Gkioxari, G., Dollár, P., y Girshick, R., "Mask R-CNN," 2018, http://arxiv.org/abs/1703.06870 (visitado el 2022-07-05).

[39] Redmon, J., Divvala, S., Girshick, R., y Farhadi, A., "You Only Look Once: Unified, Real-Time Object Detection," 2016, http://arxiv.org/abs/1506.02640 (visitado el 2023-07-03). arXiv:1506.02640 [cs].

[40] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., y Polosukhin, I., "Attention Is All You Need," 2017, doi:10.48550/ARXIV.1706.03762. Publisher: arXiv Version Number: 6.

[41] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., y Houlsby, N., "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," 2021, http://arxiv.org/abs/2010.11929 (visitado el 2023-07-03). arXiv:2010.11929 [cs].

[42] "TF2 Detection Zoo.", https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md (visitado el 2024-01-06).

[43] "Transformers.", https://huggingface.co/docs/transformers/index (visitado el 2024-01-06).

[44] Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., y Girshick, R., "Detectron2," 2019, https://github.com/facebookresearch/detectron2.

[45] "Yolo-v4 and Yolo-v3/v2 for Windows and Linux," 2023, https://github.com/roboflow/darknet (visitado el 2024-01-06). original-date: 2020-04-28T22:42:52Z.

[46] Jocher, G., "YOLOv5 by Ultralytics," 2020, doi:10.5281/zenodo.3908559.

[47] "meituan/YOLOv6," 2024, https://github.com/meituan/YOLOv6 (visitado el 2024-01-06). original-date: 2022-06-08T02:44:22Z.

[48] Skalski, P., "Official YOLOv7," 2023, https://github.com/SkalskiP/yolov7 (visitado el 2024-01-06). original-date: 2022-12-27T22:02:47Z.

[49] Terven, J. y Cordova-Esparza, D., "A Comprehensive Review of YOLO: From YOLOv1 and Beyond," 2023, http://arxiv.org/abs/2304.00501 (visitado el 2024-01-06). arXiv:2304.00501 [cs].

[50] Das, A. K., Esau, T. J., Zaman, Q. U., Farooque, A. A., Schumann, A. W., y Hennessy, P. J., "Machine vision system for real-time debris detection on mechanical wild blueberry harvesters," Smart Agricultural Technology, vol. 4, p. 100166, 2023, doi:10.1016/j.atech.2022.100166.

[51] MacEachern, C. B., Esau, T. J., Schumann, A. W., Hennessy, P. J., y Zaman, Q. U., "Detection of fruit maturity stage and yield estimation in wild blueberry using deep learning convolutional neural networks," Smart Agricultural Technology, vol. 3, p. 100099, 2023, doi:10.1016/j.atech.2022.100099.

[52] Ni, X., Takeda, F., Jiang, H., Yang, W. Q., Saito, S., y Li, C., "A deep learning-based web application for segmentation and quantification of blueberry internal bruising," Computers and Electronics in Agriculture, vol. 201, p. 107200, 2022, doi:10.1016/j.compag.2022.107200.

[53] Qiao, S., Wang, Q., Zhang, J., y Pei, Z., "Detection and Classification of Early Decay on Blueberry Based on Improved Deep Residual 3D Convolutional Neural Network in Hyperspectral Images," SCIENTIFIC PROGRAMMING, vol. 2020, 2020, doi:10.1155/2020/8895875.

[54] Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., Dollár, P., y Girshick, R., "Segment Anything," 2023, http://arxiv.org/abs/2304.02643 (visitado el 2024-01-06). arXiv:2304.02643 [cs].