Santiago de Cali, 18 de julio del 2024

Doctor
Diego Luis Linares Ospina
Director Maestría en Ciencia de Datos
Facultad de Ingeniería y Ciencias
Pontificia Universidad Javeriana de Cali

Asunto: Presentación para evaluación del proyecto aplicado

Cordial Saludo,

Con el fin de cumplir con los requisitos exigidos por la Universidad para optar por el título de Magíster en Ciencia de Datos, nos permitimos presentar a su consideración el proyecto denominado "DETECCIÓN DE ENFERMEDADES EN CULTIVOS DE BANANO CON IMÁGENES AÉREAS UTILIZANDO UN MODELO DE DEEP LEARNING" el cual fue realizado por los estudiantes Michael Rodriguez Reyes y Jorge Alberto Enriquez Polanco con códigos 8939400 y 8979547 pertenecientes a la Maestría en Ciencia de Datos, bajo la dirección del PhD. Luis Eduardo Tobón Llano.

El suscrito director del Proyecto Aplicado autoriza para que se proceda a hacer la evaluación de este proyecto, toda vez que ha revisado cuidadosamente el documento y avala que ya se encuentra listo para ser presentado y sustentado oficialmente.

Atentamente,

Michael Rodriguez Reyes
C.C. 1.192.896.799 de Palmira

Jorge Alberto Enriquez Polanco
C.C. 1.062.307.074 de Santander de Quilichao

PhD. Luis Eduardo Tobón Llano
C.C. 9726670 de Armenia

Documentación anexa:
Resumen del Proyecto Aplicado en formato digital.
Una copia digital (PDF) del documento del proyecto aplicado.

Palmira, May 31st, 2024

Gentlemen:

Pontificia Universidad Javeriana Cali

I, Dr. Michael Selvaraj, Senior Scientist at the International Center for Tropical Agriculture (CIAT), hereby formally authorize the conduct of the research and degree work entitled "DETECCIÓN DE ENFERMEDADES EN CULTIVOS DE BANANO CON IMÁGENES AÉREAS UTILIZANDO UN MODELO DE DEEP LEARNING", using the aerial images of banana crops collected by my research team.

These images, obtained using drone technology and other remote sensing tools, have been collected with the objective of studying and improving agricultural practices in banana crops. We believe that their use in your research will contribute significantly to the advancement of knowledge in this area and will allow the development of new methodologies and technologies applicable to tropical agriculture.

I remain at your disposal for any further consultation or to provide the necessary information to facilitate the development of your work.

Sincerely,

Dr. Michael Selvaraj

Senior Scientist

Centro Internacional de Agricultura Tropical (CIAT)

m.selvaraj@cgiar.org

FICHA RESUMEN
PROYECTO APLICADO – MAESTRÍA EN CIENCIA DE DATOS

TÍTULO: Detección de enfermedades en cultivos de banano con imágenes aéreas utilizando un modelo de Deep Learning.

1. **ÁREA DE TRABAJO:** Ciencia de Datos y Electrónica.
2. **TIPO DE PROYECTO (Aplicado, Innovación, Investigación):** Aplicado.
3. **ESTUDIANTE(S):** Jorge Alberto Enríquez y Michael Rodríguez.
4. **CORREO ELECTRÓNICO:** jaep@javerianacali.edu.co, michael@javerianacali.edu.co .
5. **DIRECCIÓN Y TELEFONO:** Km 17, Recta Cali-Palmira, Valle del Cauca / 3166294803.
6. **DIRECTOR:** PhD. Luis Eduardo Tobón.
7. **VINCULACIÓN DEL DIRECTOR:** Directa con la Universidad.
8. **CORREO ELECTRÓNICO DEL DIRECTOR:** letobon@javerianacali.edu.co .
9. **GRUPO O EMPRESA QUE LO AVALA:** Centro Internacional de Agricultura Tropical (CIAT).
10. **OTROS GRUPOS O EMPRESAS:** No
11. **PALABRAS CLAVE (al menos 5):** Deep Learning, Drones, Agricultura, Banano, Cultivos, Monitoreo, Ciencia de Datos.
12. **FECHA DE INICIO:** Mayo 9 del 2023.
13. **FECHA DE FINALIZACIÓN:** Junio 7 del 2024.
14. **RESUMEN:**

Este documento presenta un proyecto cuyo objetivo principal es reducir la cantidad de tiempo invertida en el monitoreo de los cultivos de banano, mediante la implementación de modelos de Aprendizaje Profundo (Deep Learning). Estos modelos se han utilizado para detectar y monitorear las enfermedades de Fusarium wilt y Xanthomonas wilt en los cultivos, empleando imágenes de alta resolución en RGB obtenidas por UAV (vehículos aéreos no tripulados). El proyecto se dirige específicamente a pequeños y medianos agricultores, con el propósito de agilizar los procesos de monitoreo de los cultivos de banano y detectar de manera temprana las enfermedades mencionadas. Para lograr este objetivo, se ha desarrollado un prototipo funcional que ha sido probado en cultivos reales. La implementación del prototipo se ha basado en los avances encontrados en el estado del arte relacionado con dispositivos y arquitecturas utilizadas en la implementación de inteligencia artificial en el monitoreo de cultivos. Además, se ha utilizado una base de datos proporcionada por el Centro Internacional de Agricultura Tropical (CIAT), que cuenta con más de 30 mil plantas de banano anotadas y etiquetadas por expertos fitopatólogos. En

cuanto a los objetivos cuantitativos, se ha logrado reducir en al menos un 25% el tiempo necesario para el monitoreo de los cultivos de banano, en comparación con los métodos tradicionales utilizados por los agricultores. Para lograr estos objetivos, se ha utilizado la metodología CDIO, que implica comprender inicialmente la situación, necesidad o problema en un contexto específico. A partir de esta comprensión, se ha diseñado una solución que se ha implementado realizando las modificaciones necesarias hasta llegar a la fase operativa del proyecto.

Pontificia Universidad
JAVERIANA
Cali

DETECCIÓN DE ENFERMEDADES EN CULTIVOS DE BANANO CON IMÁGENES AÉREAS UTILIZANDO UN MODELO DE DEEP LEARNING

Michael Rodríguez Reyes

Jorge Alberto Enríquez Polanco

Proyecto Aplicado para optar al título de
Magíster en Ciencia de Datos

Director:
PhD. Luis Eduardo Tobón

Pontificia Universidad Javeriana Cali
FACULTAD DE INGENIERÍA Y CIENCIAS
Cali, 18 de julio de 2024

Agradecimientos

En primer lugar, nos gustaría agradecer a nuestro director de trabajo de grado, el PhD. Luis Eduardo Tobón, por haber estado siempre presente durante todo el proyecto.

También, queremos agradecer al Centro Internacional de Agricultura Tropical (CIAT) y a sus consultores adjuntos por facilitarnos los datos con los cuales se desarrolló este trabajo de grado.

Agradecemos a Juan José Mora y a Daniela Gómez, Investigadores Asociados de CIAT por su ayuda a lo largo de todo este proceso.

TABLA DE CONTENIDO

Pág.

Agradecimientos I

TABLA DE CONTENIDO II

Lista de figuras VI

Lista de tablas VIII

1. Introducción 1

2. Definición del Problema 3

2.1 Planteamiento del Problema 3
2.2 Formulación del Problema 5

3. Objetivos 7

3.1 Objetivo General 7
3.2 Objetivos Específicos 7

4. Marco Teórico y Antecedentes 9

4.1 Marco Teórico 9
4.1.1 Machine Learning 9
4.1.1.1 Aprendizaje supervisado 10
4.1.1.2 Aprendizaje no supervisado 11
4.1.2 Deep Learning 12
4.1.2.1 Redes Neuronales Convolucionales 12
4.1.2.2 Visión por Computadora 15
4.1.3 Detección de Objetos 15
4.1.3.1 TensorFlow Object Detection API 17
4.1.3.2 YOLO 19
4.1.3.3 SORT 21
4.1.3.4 Evaluación de modelos de detección 21
4.1.4 Captura de imágenes 22
4.1.4.1 Cámaras RGB 23
4.1.5 Vehículos Aéreos no Tripulados (VANT) 24
4.1.5.1 Ventajas de los VANT 25

4.1.5.2 Limitaciones de los VANT 25
4.1.6 Inteligencia Artificial 26
4.1.7 Enfermedades en Cultivos de Banano 27
4.1.7.1 Fusiarium wilt of Banana 27
4.1.7.2 Xanthomonas wilt of Banana 28
4.2 Antecedentes 30
4.2.1 AI-powered banana diseases and pest detection 30
4.2.2 Image processing analysis of geospatial uav orthophotos for palm oil plantation monitoring 31
4.2.3 Detection of banana plants and their major diseases through aerial images and machine learning methods: A case study in DR Congo and Republic of Benin 32
4.2.4 Deep learning models for plant disease detection and diagnosis 33
4.2.5 Fruits and vegetables quality evaluation using computer vision 34

5. Diseño y Desarrollo del Proyecto 35

5.1 Recolección y Preprocesamiento de los Datos 35
5.1.1 Captura de Imágenes con Drones 37
5.1.2 Construcción de Ortomosaicos 37
5.1.3 Primer Análisis de Cultivos 38
5.1.4 Generación Automática de Bounding Boxes 39
5.1.5 Revisión del Dataset en Colaboración con Fitopatólogos Expertos 40
5.1.6 Prueba de Laboratorio Inicial 41
5.1.7 Conversión de Formato de Etiquetas para Entrenamiento 42
5.2 Desarrollo de Diferentes modelos de Deep Learning 43
5.2.1 Faster RCNN - Resnet 50 44
5.2.2 YOLO V8 45
5.2.3 YOLO NAS 46
5.2.4 YOLO V9 46
5.3 Conteo y Seguimiento de Objetos 47
5.3.1 Filtro de kalman 48
5.3.2 Algoritmo de Asignación Húngaro 51
5.3.2.1 Cálculo de la Distancia Euclidiana 52
5.3.3 Q_discrete_white_noise 52
5.4 Detección en Tiempo Real 53
5.5 Arquitectura Propuesta de la Solución 54

6. Resultados y Análisis 57

6.1 Conjunto de Datos Preprocesado 57
6.2 Evaluación y Desempeño de Modelos con Imágenes 58
6.2.1 Faster RCNN - Resnet 50 59
6.2.2 YOLO V8 61

6.2.3 YOLO NAS 65
6.2.4 YOLO V9 69
6.2.5 Comparación de Modelos YOLO 73
6.2.6 Selección del Mejor Modelo 77
6.3 Evaluación y Desempeño de Modelos YOLO con Videos 78
6.3.1 Evaluación Inicial de Detección en Video 78
6.3.1.1 YOLO V8 79
6.3.1.2 YOLO NAS 79
6.3.1.3 YOLO V9 80
6.3.1.4 Resultado de Evaluación Inicial en Video 80
6.3.2 Evaluación de Metricas en Video - YOLO V8 81
6.4 Implementación Innovadora del Modelo 83
6.4.1 Conteo de Plantas 83
6.4.2 Detección y Conteo en Tiempo Real 87
6.5 Comparación entre el Modelo Desarrollado con los Modelos Identificados en la Literatura Científica 90
6.5.1 Modelo 1: AI-powered banana diseases and pest detection [1] 90
6.5.2 Modelo 2: Detection of banana plants and their major diseases through aerial images and machine learning methods [2] 91
6.5.3 Modelo 3: Tea leaf disease detection and identifcation based on YOLOv7 (YOLO-T) [3] 91
6.5.4 Modelo 4: Image processing analysis of geospatial uav orthophotos for palm oil plantation monitoring [4] 92
6.5.5 Modelo 5: Deep learning models for plant disease detection and diagnosis [5] 92
6.5.6 Modelo 6: Fruits and vegetables quality evaluation using computer vision [6] 93
6.5.7 Desempeño del Modelo YOLO V8 Desarrollado en Este Proyecto de Grado 93
6.5.8 Análisis Final: 95

7. Conclusiones y Trabajos Futuros 97

7.1 Conclusiones 97
7.2 Trabajos Futuros 98
7.2.1 Ampliación del Dataset de Entrenamiento: 98
7.2.2 Optimización del Modelo: 99
7.2.3 Implementación en Dispositivos Móviles y Drones: 99
7.2.4 Mejora del Conteo de Plantas: 99
7.2.5 Integración con Sistemas de Gestión Agrícola: 99
7.2.6 Investigación en Técnicas de Transferencia de Aprendizaje: 100
7.2.7 Evaluación de Impacto Económico y Ambiental: 100

Referencias 101

Apéndice. A. **Códigos preprocesamiento de dataset, entrenamiento y evaluacion de modelos** 109

Apéndice. B. **Detección y Conteo de Plantas** 111

ÍNDICE DE FIGURAS

FIGURA Pág.

4.1 Programación tradicional Vs. Machine Learning 9

4.2 Modelo de aprendizaje Supervisado 10

4.3 Modelo de aprendizaje No Supervisado 11

4.4 Arquitectura de una CNN 14

4.5 Detección de objetos utilizando redes neuronales convolucionales y segmentación semántica 16

4.6 Comparación eficiencia de modelos YOLO 20

4.7 Planta de Banano Infectada con Fusarium [fuente propia]. 28

4.8 Planta de Banano Infectada con Xanthomonas wilt [fuente propia]. 29

5.1 Ortomosaico en el software Agisoft Metashape 38

5.2 Verificación anotaciones en LabelImg 41

5.3 Rendimiento YOLO V9 47

5.4 Diagrama de bloques Filtro de Kalman 50

5.5 Pipeline detección de enfermedades en tiempo real 54

5.6 Esquema Propuesto del Proyecto 54

6.1 Ejemplos dataset preprocesado 58

6.2 Matriz de confusión Faster RCNN 59

6.3 Métricas de test Faster RCNN 60

6.4 Predicción Faster RCNN – Resnet 50 61

6.5 Gráfica de entrenamiento YOLO V8 62

6.6 Matriz de confusión YOLO V8 63

6.7 Métricas de test YOLO V8 64
6.8 Predicción YOLO V8 65
6.9 Gráfica de entrenamiento YOLO NAS 65
6.10 Matriz de confusión YOLO NAS 66
6.11 Métricas de test YOLO NAS 67
6.12 Predicción YOLO NAS 68
6.13 Gráfica de entrenamiento YOLO V9 69
6.14 Matriz de confusión YOLO V9 70
6.15 Métricas de test YOLO V9 71
6.16 Predicción YOLO V9 72
6.17 Pérdidas totales en entrenamiento 73
6.18 Pérdidas totales en validación 75
6.19 mAP en validación 76
6.20 Predicción en video YOLO V8 79
6.21 Predicción en video YOLO V9 80
6.22 Métricas fotogramas video 1 82
6.23 Métricas fotogramas video 2 82
6.24 Predicción en fotogramas de video 83
6.25 Conteo de plantas inicial 86
6.26 Prueba en tiempo real IPAD 87
6.27 Prueba Vuelo de Dron sobre cultivos 88
6.28 Prueba de conteo en tiempo real 89

ÍNDICE DE TABLAS

TABLA Pág.

4.1 Características de cada Versión 20

6.1 Comparación de modelos propuestos 77

1. Introducción

Los bananos, conocidos científicamente como Musáceas, son cultivos frutales de gran importancia tanto a nivel de producción como en el comercio global. Su consumo es fundamental en la alimentación de diversas regiones del mundo, especialmente en África, Asia y América Latina. Aunque su demanda es alta, solo una fracción de la producción mundial se destina a la exportación, lo que resalta su relevancia en los mercados nacionales y su impacto en la seguridad alimentaria [7].

A pesar de su importancia económica y nutricional, los cultivos de banano enfrentan desafíos bióticos significativos que afectan su productividad y calidad. Entre estos desafíos se encuentran las plagas y enfermedades bacterianas, virales y fúngicas, como lo son las enfermedades de *Fusarium wilt* y *Xanthomonas wilt*, que pueden causar pérdidas parciales o totales en los cultivos [8]. Estas enfermedades pueden reducir significativamente la calidad y el rendimiento de los cultivos, causando pérdidas económicas considerables para los agricultores y afectando la seguridad alimentaria de la región [9].

En países en desarrollo con limitaciones en la infraestructura agrícola, los métodos tradicionales de identificación de plagas y enfermedades dependen en gran medida de especialistas en extensión agrícola. Sin embargo, esta dependencia de expertos presenta desafíos, como la falta de disponibilidad de personal capacitado y el tiempo requerido para llevar a cabo la identificación manual de enfermedades en los campos de cultivo. Adicionalmente, muchos pequeños agricultores se basan en el conocimiento empírico, lo cual puede resultar menos efectivo para abordar los desafíos agrícolas de manera precisa y oportuna [10].

Ante este escenario, se ha explorado el potencial de las tecnologías emergentes para optimizar la detección y seguimiento de enfermedades en los cultivos de banano. Para mejorar las deficiencias en los procesos de detección de enfermedades, se implementó una base de datos suministrada por el Centro Internacional de Agricultura Tropical (CIAT). Esta base de datos contenía imágenes ya clasificadas de enfermedades en cultivos de banano por expertos fitopatólogos que habían realizado previamente vuelos en drones y

seleccionado manualmente las plantas enfermas y saludables en varias plantaciones de banano. Estas imágenes proporcionaron un conjunto valioso de datos para el desarrollo de sistemas de detección de enfermedades basados en inteligencia artificial.

En los últimos años, se han implementado recursos tecnológicos innovadores para mejorar la gestión de plagas y enfermedades en la agricultura. Entre ellos, los vehículos aéreos no tripulados (UAV) han ganado prominencia como herramientas de vanguardia en la evaluación del rendimiento de los cultivos, la monitorización de su estado de salud y la determinación de su valor económico. Los UAV tenían la capacidad de capturar imágenes espectrales-temporales de alta calidad, lo que permitía obtener información detallada sobre los cultivos [11].

El rápido avance tecnológico en los sistemas UAV ha llevado al desarrollo de los UAV de bajo costo equipados con sensores (RGB) y multiespectrales, lo que brinda la oportunidad de capturar datos de alta resolución espacio temporal, especialmente en lo que respecta a la detección de enfermedades en los campos de producción [12]. Estas imágenes aéreas se dividen en imágenes RGB, que se basan en la captura de información visual, y en imágenes multiespectrales, que recogen datos en diferentes bandas del espectro electromagnético. Las cámaras RGB han demostrado su eficacia en la identificación de enfermedades de los cultivos [13].

El uso de la base de datos suministrada por el CIAT y la tecnología de vehículos aéreos no tripulados ofrecieron una solución prometedora para abordar los desafíos en la detección temprana de enfermedades en los cultivos de banano, con la implementación de un modelo de detección usando You Only Look Once (YOLO), un algoritmo de aprendizaje profundo que permitió detectar rápidamente múltiples objetos en imágenes. Esto se complementó con el algoritmo Simple Online and Realtime Tracking (SORT) para seguir el movimiento de las plantas enfermas durante los vuelos de drone dentro de los cultivos. Esta innovadora aplicación de inteligencia artificial y visión por computadora mejoró significativamente la gestión de enfermedades en los cultivos de banano, contribuyendo a la seguridad alimentaria y la sostenibilidad de la producción de banano a nivel global.

2. Definición del Problema

2.1 Planteamiento del Problema

Los bananos (Musáceas) son uno de los cultivos frutales más significativos en términos de producción y comercio a nivel global [7]. Esto enfatiza la importancia de los mercados nacionales y la seguridad alimentaria en relación con los bananos. En África Oriental y Central (ECA), esta fruta desempeña un papel fundamental en la producción y el consumo de banano en el continente, beneficiando a alrededor de 20 millones de personas en África Oriental y 70 millones en África Occidental y Central, al ser su fuente principal de alimentos [14].

Los bananos, tanto consumidos frescos como procesados, contienen diversas moléculas bioactivas beneficiosas para la salud humana, como fenoles, carotenoides, aminas biogénicas y fitoesteroles, y poseen niveles significativos de antioxidantes. Históricamente, los plátanos han sido utilizados en el tratamiento de diversos trastornos degenerativos crónicos.

Sin embargo, en los países en desarrollo con infraestructura agrícola limitada, los métodos tradicionales de identificación de plagas, especialmente para detectar enfermedades como *Fusarium wilt* y *Xanthomonas wilt*, dependen en gran medida de especialistas en extensión agrícola. Esta dependencia presenta limitaciones significativas en términos de infraestructura humana y puede ser insuficiente para abordar los desafíos agrícolas en su totalidad [10]. Además, muchos pequeños agricultores confían en el conocimiento empírico, lo que resulta menos efectivo para abordar los desafíos relacionados con la salud de los cultivos.

Esta situación plantea un problema crítico, ya que la falta de detección temprana de enfermedades como *Fusarium wilt* y *Xanthomonas wilt* puede resultar en pérdidas significativas en la producción de banano. Estas pérdidas impactan tanto económicamente como en términos de seguridad alimentaria, ya que los bananos son una fuente esencial de alimento para millones de personas en regiones clave. En este

contexto, se hace evidente la necesidad de encontrar soluciones efectivas para la detección temprana y el manejo de estas enfermedades en los cultivos de banano.

En los últimos años, se han implementado diversas prácticas relacionadas con la Ciencia de Datos para abordar la detección de enfermedades en cultivos, especialmente en el caso de los bananos. El uso de modelos de Deep Learning se ha destacado como una tecnología de vanguardia para evaluar el estado de salud de los cultivos. Paralelamente, el análisis con modelos de Machine Learning ha permitido estimar diversas características relacionadas con los cultivos, como proyecciones y su valor económico a futuro. Estos modelos se aplican principalmente a imágenes aéreas, ya que los vehículos aéreos no tripulados (UAV) tienen la capacidad de recolectar grandes cantidades de información en un corto período de tiempo, lo que resulta crucial en la agricultura moderna [12].

El constante desarrollo de la Ciencia de Datos en los últimos años, junto con la versatilidad en la aplicación de estas nuevas tecnologías, permite su integración en la agricultura a través de sistemas UAV de bajo costo equipados con sensores visibles (RGB) y multiespectrales. Esto posibilita la captura de datos de alta resolución espaciotemporal, especialmente en lo que respecta a la detección de enfermedades en los campos de producción de banano [12]. Además, los modelos de Deep Learning han demostrado ser exitosos en la detección temprana de enfermedades en cultivos, como es el caso de *Fusarium wilt y Xanthomonas wilt*, lo que aporta soluciones eficaces a los desafíos agrícolas [15].

En este contexto, la aplicación de la Ciencia de Datos y el uso de modelos de aprendizaje profundo y machine learning se presentan como herramientas esenciales para abordar el problema de la detección temprana de enfermedades en cultivos de banano. Estas tecnologías prometen no solo optimizar la gestión y control de las enfermedades, sino también contribuir a la seguridad alimentaria y la sostenibilidad de la producción de banano a nivel global.

2.2 Formulación del Problema

Habiendo evidenciado las pérdidas de productividad en los cultivos de banano ocasionadas por la detección tardía de *Fusarium wilt y Xanthomonas wilt* y la falta de control en estos, se formuló la siguiente pregunta de investigación:

¿Cómo desarrollar un modelo de Deep Learning utilizando imágenes RGB capturadas con drones para identificar las enfermedades de Fusarium wilt y Xanthomonas wilt en los cultivos de banano?

Del planteamiento de la problemática, surgen las siguientes preguntas de sistematización:

1. ¿Cómo garantizar que una base de datos, proporcionada por el CIAT, permita desarrollar un modelo basado en Deep Learning capaz de identificar adecuadamente las dos enfermedades?
2. ¿Cuáles son las arquitecturas de Deep Learning apropiadas para la identificación precisa de las dos enfermedades en los cultivos de banano?
3. ¿Cuáles son las métricas de evaluación más adecuadas para analizar el desempeño de los modelos de Deep Learning?
4. ¿Cómo implementar de manera innovadora el modelo de Deep Learning seleccionado?
5. ¿Cómo se compara el desempeño del modelo de Deep Learning desarrollado con las identificadas en la literatura para la detección de enfermedades en cultivos?

3. Objetivos

3.1 Objetivo General

Desarrollar un modelo de Deep Learning utilizando imágenes RGB capturadas con drones con el fin de identificar las enfermedades de *Fusarium wilt* y *Xanthomonas wilt* en los cultivos de banano.

3.2 Objetivos Específicos

1. Realizar el preprocesamiento de la base de datos existente, aplicando técnicas adecuadas para garantizar la calidad de los datos utilizados en el desarrollo del modelo de Deep Learning.
2. Desarrollar diferentes modelos de Deep Learning para determinar los más efectivos en la identificación de las enfermedades de *Fusarium wilt* y *Xanthomonas wilt* en los cultivos de banano.
3. Evaluar el desempeño de los modelos de Deep Learning desarrollados con métricas adecuadas, incluyendo precision, recall, F1-score y curvas ROC.
4. Implementar de manera innovadora el modelo de Deep Learning desarrollado, adaptándolo a las necesidades del contexto específico y optimizando su rendimiento.
5. Comparar el desempeño del modelo de Deep Learning desarrollado en este proyecto con los modelos identificados en la literatura científica para la detección de enfermedades en cultivos, evaluando su precisión, eficiencia computacional y capacidad de adaptación.

4. Marco Teórico y Antecedentes

4.1 Marco Teórico

A continuación se presenta el marco teórico que fundamenta la teoría de este proyecto. En esta sección se expondrán los conceptos básicos necesarios para iniciar; es importante establecer una base sobre qué son los vehículos aéreos no tripulados, la inteligencia artificial y las demás tecnologías que estarán presentes en la investigación.

4.1.1 Machine Learning

El Machine Learning, o aprendizaje automático, es un campo de la inteligencia artificial que ha demostrado ser altamente efectivo en diversas aplicaciones, incluida la detección de enfermedades en cultivos de banano. Mediante algoritmos y modelos específicos, el Machine Learning permite analizar grandes cantidades de datos e identificar patrones y características que pueden ser utilizados para diagnosticar y predecir enfermedades en los cultivos [16] [17].

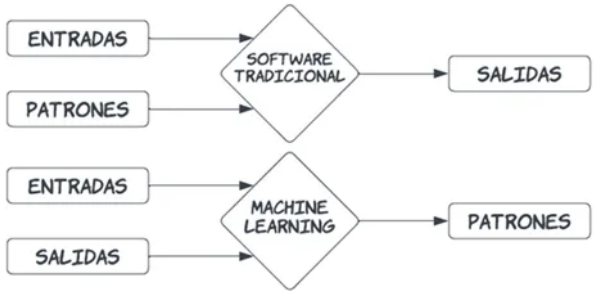


Figura 4.1: Programación tradicional Vs. Machine Learning

En el caso de la detección de enfermedades en cultivos de banano, el Machine Learning ofrece varias ventajas significativas. Primero, el uso de imágenes aéreas de los cultivos permite obtener una visión general del estado de las plantas y detectar posibles anomalías. Estas imágenes se pueden procesar y analizar mediante algoritmos de Machine Learning para identificar signos tempranos de enfermedades, como manchas, deformaciones o cambios en el color de las hojas y los frutos.

Esta capacidad de detección temprana y precisa de enfermedades en los cultivos de banano mediante el Machine Learning es fundamental para los agricultores y especialistas agrícolas [18]. Permite una respuesta rápida ante la presencia de enfermedades, lo que a su vez facilita la implementación de medidas de control y prevención adecuadas, como el tratamiento específico de las plantas afectadas o la implementación de prácticas de cultivo adecuadas para minimizar la propagación de enfermedades.

4.1.1.1 Aprendizaje supervisado

El aprendizaje supervisado se basa en utilizar conjuntos de datos etiquetados para entrenar un modelo y hacer predicciones o tomar decisiones con base en esos datos. En el aprendizaje supervisado, se proporciona al algoritmo de Machine Learning un conjunto de ejemplos de entrada junto con las salidas deseadas correspondientes, lo que se conoce como conjunto de entrenamiento [19].

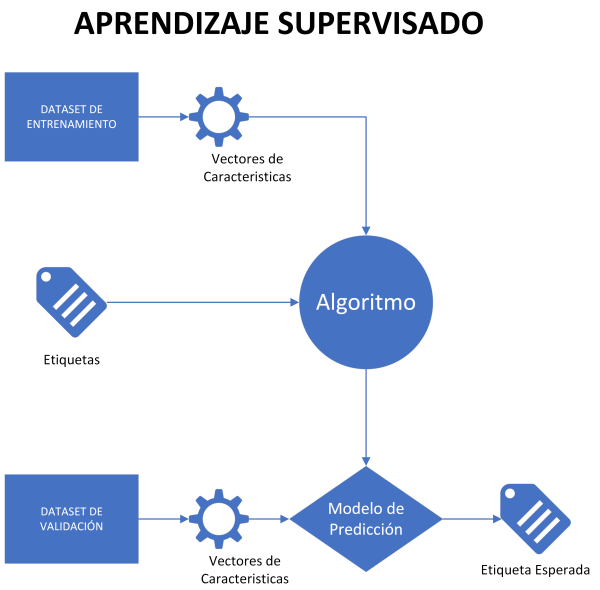


Figura 4.2: Modelo de aprendizaje Supervisado

El objetivo del aprendizaje supervisado es que el algoritmo aprenda a mapear las entradas a las salidas correctas, de modo que pueda generalizar y hacer predicciones precisas sobre nuevos ejemplos no vistos previamente.

En este trabajo de grado, el aprendizaje supervisado se utiliza para entrenar modelos que puedan distinguir entre plantas sanas y plantas afectadas por enfermedades específicas. Para ello, se recopila un conjunto de imágenes de cultivos de banano junto con las etiquetas correspondientes que indican si la planta está sana o enferma.

4.1.1.2 Aprendizaje no supervisado

El aprendizaje no supervisado se centra en el análisis y la extracción de patrones y estructuras ocultas en conjuntos de datos no etiquetados. A diferencia del aprendizaje supervisado, en el aprendizaje no supervisado no se proporcionan salidas deseadas o etiquetas para el conjunto de datos, lo que significa que el algoritmo debe descubrir patrones por sí mismo.

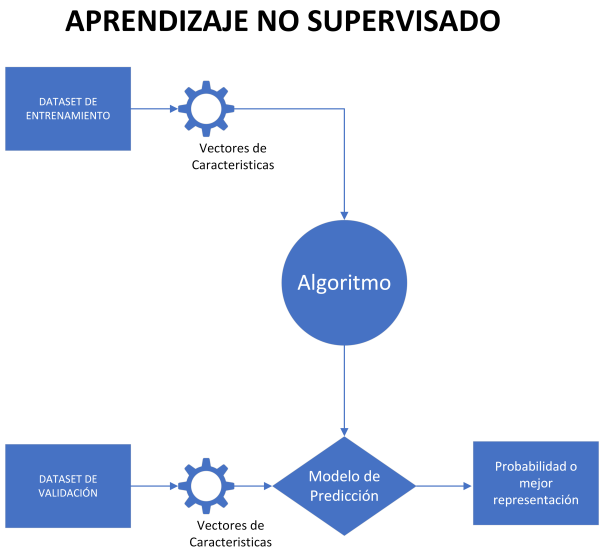


Figura 4.3: Modelo de aprendizaje No Supervisado

El objetivo principal del aprendizaje no supervisado es explorar y comprender la estructura subyacente de los datos, encontrar agrupaciones naturales, identificar relaciones y descubrir características ocultas. Este enfoque es útil cuando se tienen grandes cantidades de datos no etiquetados y se desea obtener información valiosa sin la necesidad de una supervisión o conocimiento previo sobre las categorías o clases de los datos.

En el contexto de este trabajo de grado, el aprendizaje no supervisado puede ser utilizado para identificar patrones y anomalías en los datos agrícolas. Por ejemplo, se pueden aplicar algoritmos de agrupamiento, como el clustering, para dividir los datos en grupos homogéneos basados en características similares.

4.1.2 Deep Learning

El Deep Learning es una rama avanzada del aprendizaje automático (machine learning) que utiliza modelos computacionales compuestos por múltiples capas de procesamiento para aprender representaciones de datos con diferentes niveles de abstracción. Este enfoque ha revolucionado diversas áreas, como el reconocimiento de voz, la detección de objetos visuales y el descubrimiento de fármacos, al permitir que las máquinas descubran estructuras intrincadas en grandes conjuntos de datos. [20]

El Deep Learning se basa en el algoritmo de retropropagación, el cual indica a la máquina cómo ajustar sus parámetros internos en cada capa para calcular la representación adecuada a partir de la capa anterior. Este proceso de retroalimentación permite que el modelo aprenda automáticamente características y patrones complejos.

El aprendizaje profundo, a través de su capacidad para aprender representaciones complejas y su aplicación en redes convolucionales y recurrentes, se convierte en una herramienta poderosa para analizar datos de manera efectiva en diversos dominios, incluido el monitoreo de enfermedades en cultivos de banano a partir de imágenes de alta resolución obtenidas por UAV [21].

4.1.2.1 Redes Neuronales Convolucionales

Las redes neuronales convolucionales, una clase especializada de arquitectura de redes neuronales, han revolucionado el procesamiento de imágenes y otras tareas relacionadas con la percepción. Ampliamente utilizadas en aplicaciones de visión por computadora y reconocimiento de patrones, estas redes destacan por su capacidad para aprender automáticamente características relevantes y complejas directamente de

los datos de entrada [22]. Inspiradas en el sistema visual del cerebro humano, especialmente en la corteza visual, estas redes se basan en la idea de campos receptivos, donde las células son sensibles a regiones específicas del campo visual [22].

En el contexto de las CNN, la convolución implica aplicar un filtro o kernel a una región local de la imagen de entrada y calcular el producto escalar entre los valores del filtro y los valores de píxeles en esa región. Esto se repite para todas las regiones de la imagen, y el resultado es un mapa de características que resalta la presencia de características específicas en diferentes partes de la imagen [22].

Los componentes clave de estas redes neuronales son:

1. Capas de convolución: Estas son las capas fundamentales de las CNN. Cada capa de convolución aplica un conjunto de filtros a la imagen de entrada y produce un mapa de características, donde cada mapa corresponde a una característica específica detectada por el filtro. Los filtros se entrenan para aprender patrones significativos de la imagen durante el proceso de entrenamiento [23].

2. Función de activación: Después de la convolución, se aplica una función de activación no lineal, típicamente ReLU (Unidad Lineal Rectificada), que introduce no linealidades en la red. Esto es crucial para permitir que la red aprenda relaciones complejas entre características en los datos [23].

3. Capas de agrupación (pooling): Estas capas reducen la dimensión espacial de los mapas de características, lo que disminuye la cantidad de parámetros y cálculos en la red. El método de agrupación más común es el max-pooling, donde se toma el valor máximo en una región de la característica y se descartan los demás valores [23].

4. Capas totalmente conectadas (FC): Después de varias capas de convolución y agrupación, se utilizan capas totalmente conectadas para combinar las características aprendidas y realizar la clasificación final. Estas capas son similares a las de las redes neuronales tradicionales y se utilizan para mapear las características aprendidas a las etiquetas de salida [23].

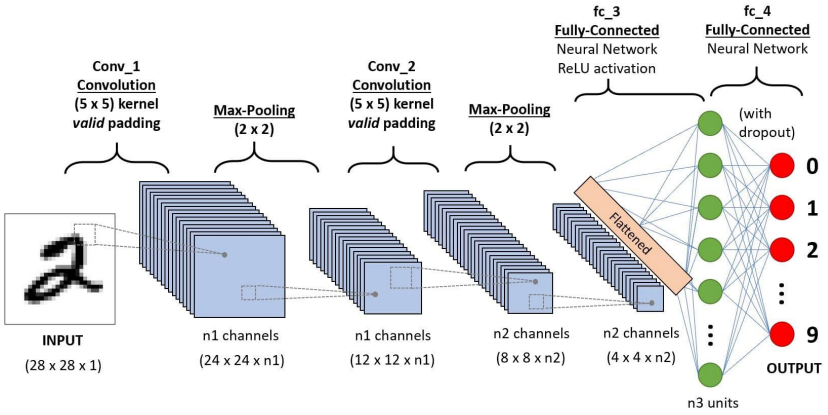


Figura 4.4: Arquitectura de una CNN
[23]

Durante el proceso de entrenamiento, las redes neuronales convolucionales ajustan los pesos de los filtros y los parámetros de las capas completamente conectadas con el fin de minimizar una función de pérdida. Esta función mide la discrepancia entre las salidas predichas por el modelo y las etiquetas reales del conjunto de entrenamiento. Por ejemplo, durante el entrenamiento de una red neuronal convolucional para reconocer dígitos escritos a mano, los pesos de los filtros se ajustan iterativamente para mejorar la precisión del modelo en la clasificación de las imágenes de los dígitos. Este ajuste se basa en la comparación entre las predicciones del modelo y las etiquetas reales de los dígitos. Utilizando el algoritmo de descenso de gradiente estocástico (SGD) o sus variantes más avanzadas [23], la red busca minimizar la función de pérdida, mejorando así su capacidad para reconocer con precisión los dígitos en el conjunto de entrenamiento.

Las redes neuronales convolucionales han demostrado ser extremadamente efectivas en una variedad de tareas, como clasificación de imágenes, detección de objetos, segmentación semántica, reconocimiento facial, entre otras. Su capacidad para aprender jerarquías de características y su arquitectura especializada para procesar datos en forma de mallas (como imágenes) las convierten en una herramienta poderosa para la resolución de problemas de visión por computadora y otras áreas relacionadas [23].

4.1.2.2 Visión por Computadora

La visión por computadora se refiere al campo interdisciplinario que se centra en el desarrollo de técnicas matemáticas y algoritmos para permitir que las máquinas perciban e interpreten la información visual de manera similar a los seres humanos. Inspirada en el sistema visual humano, la visión por computadora tiene como objetivo comprender y extraer información significativa de imágenes o datos de video [24].

Mediante el análisis de patrones de luz y sombreado, los algoritmos de visión por computadora pueden reconstruir la estructura tridimensional y apariencia de objetos dentro de imágenes. Estas técnicas permiten la creación de modelos 3D precisos de entornos utilizando múltiples fotografías superpuestas.

Mientras que los seres humanos perciben sin esfuerzo la forma, textura y segmentación de objetos en la escena visual, los investigadores de visión por computadora se esfuerzan por replicar esta capacidad en las máquinas. Aunque las ilusiones ópticas y la psicología perceptual ofrecen información sobre el funcionamiento del sistema visual humano, aún nos falta comprender de manera completa los principios subyacentes.

Más allá de la existencia de este campo desde hace varias décadas, el uso incrementado de deep learning y arquitecturas basadas en redes neuronales como las *convolutional neural networks*, *Deep Belief Networks* y *Deep Boltzmann Machines*, y los autoencoders han presentado como nuevas alternativas a los métodos de visión por computadora originales [20]. El uso de estas nuevas herramientas han dado paso a nuevas áreas dentro de la visión por computadora como el reconocimiento de rostros, reconocimiento de acciones, reconocimiento de poses humanas y de la detección de objetos.

4.1.3 Detección de Objetos

La detección de objetos se refiere al proceso de detectar instancias de objetos semánticos de una clase específica (como humanos, aviones o aves) en imágenes digitales y videos. Este proceso se realiza mediante la creación de un conjunto grande

de ventanas candidatas que posteriormente se clasifican utilizando características de redes neuronales convolucionales (CNN, por sus siglas en inglés)[20].

Por ejemplo, un enfoque común en los marcos de detección de objetos consiste en utilizar el método de búsqueda selectiva (*selective search*) para obtener propuestas de objetos, extraer características de CNN para cada propuesta y luego alimentar las características a un clasificador SVM para determinar si las ventanas contienen o no el objeto. Muchos trabajos se basan en el concepto de "Regiones con características de CNN" propuesto en [25]. Los enfoques que siguen este paradigma generalmente tienen una buena precisión de detección, pero a menudo no pueden determinar con precisión la posición exacta del objeto [20].

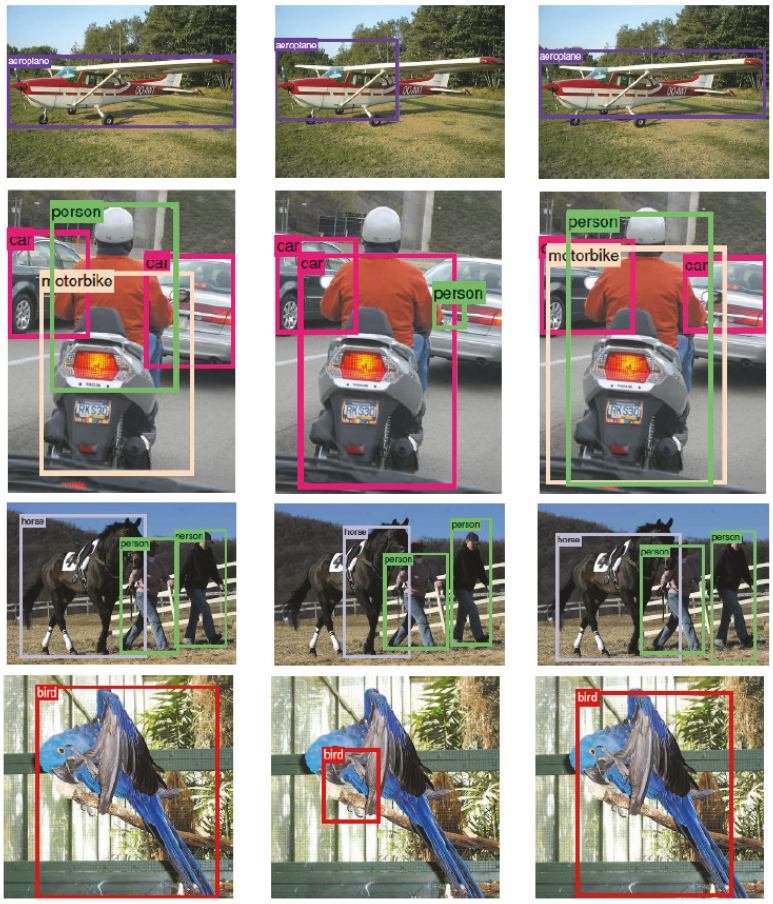


Figura 4.5: Detección de objetos utilizando redes neuronales convolucionales y segmentación semántica

[20]

Para abordar este problema, algunos métodos combinan la detección de objetos con la segmentación semántica, logrando resultados satisfactorios. Además, la mayoría de los trabajos en detección de objetos utilizan variaciones de las redes neuronales convolucionales (CNN), aunque también existen intentos de utilizar otros modelos profundos, como DBN (redes bayesianas profundas) en la detección de objetos en imágenes de teledetección o máquinas de Boltzmann de tercer orden en el reconocimiento de objetos 3D.

4.1.3.1 TensorFlow Object Detection API

Es una poderosa biblioteca de código abierto desarrollada por Google que facilita la implementación de sistemas de detección de objetos de alta precisión.

El TensorFlow Object Detection API está diseñado específicamente para abordar el problema de detección de objetos en imágenes y videos. Permite entrenar, evaluar y desplegar modelos de detección de objetos utilizando diferentes arquitecturas de redes neuronales convolucionales previamente entrenadas o personalizadas [26].

Las funciones principales del TensorFlow Object Detection API son las siguientes:

1. Implementación de arquitecturas de detección de objetos: El API proporciona implementaciones de varias arquitecturas de redes neuronales convolucionales específicas para la detección de objetos. Entre ellas, destacan Faster R-CNN, SSD (Single Shot Multibox Detector) y EfficientDet, entre otras. Estas arquitecturas se han destacado por su alta precisión y velocidad en la detección de objetos en imágenes y videos.
2. Preentrenamiento y transferencia de aprendizaje: El API permite utilizar modelos preentrenados en grandes conjuntos de datos (como COCO, Pascal VOC, etc.) y adaptarlos para tareas específicas de detección de objetos en dominios particulares. Esto se conoce como transferencia de aprendizaje, lo que acelera el proceso de entrenamiento y mejora la precisión de los modelos en conjuntos de datos con menos ejemplos.
3. Soporte para múltiples *backbones*: Las arquitecturas de detección de objetos en

el API pueden utilizar diferentes backbones, como ResNet, MobileNet, EfficientNet, etc. Estos *backbones* son redes preentrenadas utilizadas como extractores de características que capturan patrones relevantes en imágenes y facilitan el proceso de detección de objetos.

4. Evaluación de modelos: El API proporciona herramientas para evaluar el rendimiento de los modelos de detección de objetos, como la precisión, el recall y el F1-score. También permite generar métricas de evaluación como la matriz de confusión y las curvas de precisión-recall.

5. Exportación e implementación de modelos: Una vez entrenados, los modelos se pueden exportar y desplegar para su uso en aplicaciones en tiempo real o en sistemas de producción.

En cuanto a las arquitecturas presentes, algunas de las más populares y utilizadas en el TensorFlow Object Detection API son:

1. Faster R-CNN: Basada en una combinación de region proposal network (RPN) y una red de clasificación para la detección precisa de objetos.

2. SSD (Single Shot Multibox Detector): Un enfoque de detección de objetos en tiempo real que predice directamente las ubicaciones y las clases de los objetos en una sola pasada.

3. EfficientDet: Una arquitectura eficiente que utiliza un enfoque de escalado compuesto para lograr un equilibrio entre la precisión y la eficiencia computacional.

TensorFlow Object Detection API es una herramienta poderosa y versátil para implementar sistemas de detección de objetos en imágenes y videos. Su capacidad para utilizar arquitecturas de vanguardia, realizar transferencia de aprendizaje y permitir la evaluación y despliegue de modelos la convierte en una opción popular en proyectos de ciencia de datos y visión por computadora [26].

4.1.3.2 YOLO

YOLO (You Only Look Once) es un revolucionario enfoque para la detección de objetos en imágenes en tiempo real. Fue desarrollado por Joseph Redmon y su equipo en la Universidad de Washington y se ha convertido en una de las arquitecturas más influyentes en el campo de la visión por computadora.

La principal característica de YOLO es su capacidad para realizar detecciones precisas y rápidas en una sola pasada (end-to-end), lo que lo hace muy eficiente en términos computacionales. A diferencia de otros enfoques que dividen la imagen en regiones y luego clasifican y ajustan cada región por separado, YOLO propone y clasifica objetos directamente en una sola etapa. Esta simplicidad en su arquitectura le permite lograr tiempos de detección en tiempo real en dispositivos con recursos limitados, como cámaras de vigilancia y vehículos autónomos [27].

El rendimiento de YOLO es impresionante en términos de precisión y velocidad. Aunque otras arquitecturas como Faster R-CNN pueden superar a YOLO en precisión en ciertos conjuntos de datos, YOLO generalmente logra un rendimiento competitivo con una velocidad significativamente más rápida. Esto lo hace especialmente adecuado para aplicaciones en tiempo real donde se requiere una detección rápida y precisa de objetos [27].

A lo largo del tiempo, se han desarrollado varias versiones de YOLO, cada una con mejoras y refinamientos. Algunas de las versiones más destacadas incluyen (Tabla 4.1):

Tabla 4.1: Características de cada Versión

Versión	Fecha	Framework	Backbone	AP(%)
YOLO	2015	Darknet	Darknet24	63,4
YOLOV2	2016	Darknet	Darknet24	63,4
YOLOV3	2018	Darknet	Darknet53	36,2
YOLOV4	2020	Darknet	CSPDDarknet53	46,5
YOLOV5	2020	Pytorch	Modified CSP V7	55,8
YOLOV6	2022	Pytorch	EfficientRep	52,5
YOLOV7	2022	Pytorch	RepConvN	56,8
YOLOV8	2023	Pytorch	YOLOV8	53,9
YOLO-NAS	2023	Pytorch	YOLO-NAS	52,2

El éxito de YOLO ha inspirado a otros investigadores a desarrollar versiones mejoradas y adaptaciones específicas para ciertas aplicaciones. YOLO sigue siendo un referente en la detección de objetos en tiempo real y ha tenido un impacto significativo en el campo de la visión por computadora.

La siguiente generación, **YOLO NAS**, es un Modelo fundamental de detección de objetos generado por la tecnología de búsqueda de arquitectura neuronal desarrollado por **Deci-AI**, este nuevo modelo proporciona capacidades superiores de detección de objetos en tiempo real y rendimiento listo para producción.

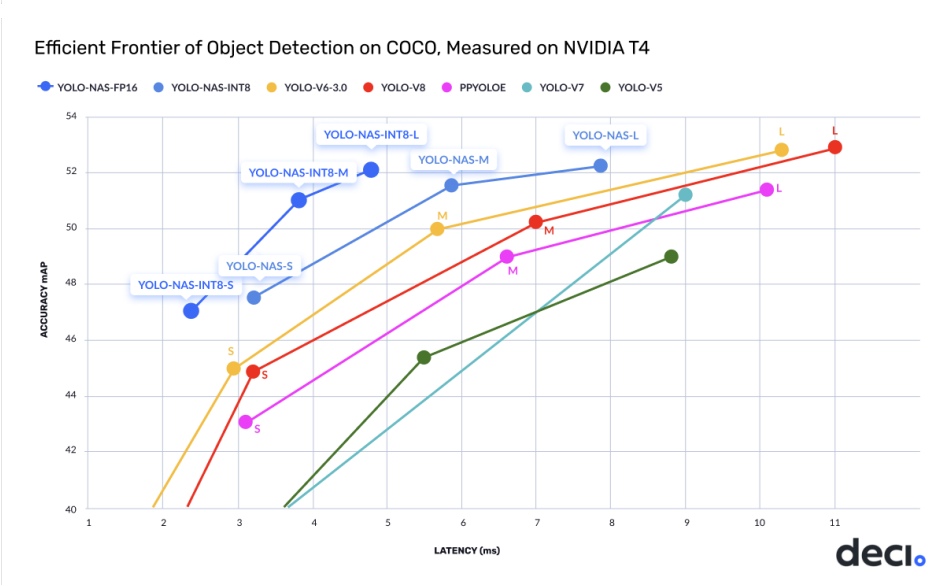


Figura 4.6: Comparación eficiencia de modelos YOLO
[28]

Su simplicidad y eficiencia lo convierten en una herramienta poderosa para aplicaciones de detección de objetos en una amplia gama de industrias, desde la seguridad hasta la automoción y más allá. YOLO NAS es 0,5 puntos, (Figura 4.6) más preciso y entre un 10 % y un 20 % más rápido que las variantes equivalentes de YOLO V8 y YOLO V7.

4.1.3.3 SORT

4.1.3.4 Evaluación de modelos de detección

La evaluación de los modelos de detección en visión por computadora enfrenta varios desafíos. Uno de ellos es el uso de métricas adecuadas para medir la precisión y el rendimiento del modelo. Dado que los modelos de detección deben clasificar objetos y proporcionar coordenadas de sus ubicaciones, las métricas tradicionales de clasificación, como la precisión y el recall, deben combinarse con métricas de regresión, como el error de la caja delimitadora (*bounding box error*).

Otro desafío es el besbalance de clases en los datos de prueba, lo que implica que algunas clases pueden estar subrepresentadas, lo que puede afectar negativamente las métricas de evaluación. Además, la evaluación de modelos en imágenes con múltiples objetos y superposiciones también agrega complejidad, ya que un modelo debe detectar objetos precisamente incluso en situaciones de alta densidad [29].

Para medir el rendimiento de los modelos de detección, se utilizan diversas métricas, como las siguientes [29]:

- Precisión (Accuracy): Mide la proporción de detecciones correctas (tanto verdaderos positivos como verdaderos negativos) en relación con todas las detecciones realizadas por el modelo.

- Recall (Sensibilidad): Representa la capacidad del modelo para detectar todos los objetos presentes en la imagen de prueba, es decir, la proporción de verdaderos positivos respecto a todos los objetos reales presentes.

- MAP (Mean Average Precision): Calcula el promedio de las precisión-Recall (P-R) obtenidas para cada clase en todas las imágenes de prueba. Es una métrica ampliamente utilizada para evaluar modelos de detección.
- F1 Score (Puntuación F1): Combina precisión y recall en una única métrica que ayuda a evaluar el equilibrio entre ambas medidas.
- Curvas ROC (Receiver Operating Characteristic): Son gráficos que muestran cómo varía el rendimiento del modelo a diferentes niveles de umbral de decisión. La curva ROC representa la relación entre la tasa de falsos positivos y la tasa de verdaderos positivos.

Para evaluar los modelos de detección, se utilizan conjuntos de datos específicos para tareas de detección de objetos, como COCO (Common Objects in Context) y Pascal VOC (Visual Object Classes). Estos conjuntos de datos contienen imágenes anotadas con etiquetas y coordenadas de cajas delimitadoras de los objetos presentes [29].

La evaluación de modelos de detección en visión por computadora es un proceso crítico para determinar su rendimiento y aplicabilidad en aplicaciones del mundo real. La selección adecuada de métricas y conjuntos de datos, junto con la comparación de arquitecturas, es esencial para tomar decisiones informadas en la implementación y desarrollo de sistemas de detección precisos y efectivos. Un enfoque riguroso y objetivo en la evaluación garantiza resultados confiables y contribuye al avance de la investigación en el campo de la visión por computadora.

4.1.4 Captura de imágenes

La teledetección agrícola es una disciplina que utiliza sensores remotos para adquirir información sobre los cultivos y el entorno agrícola. Esta técnica permite obtener datos sobre la salud de los cultivos, la calidad del suelo, el estado de la vegetación y otros parámetros importantes para el análisis y la toma de decisiones en agricultura [30].

La teledetección agrícola se basa en la captura de imágenes o datos espectrales de las superficies terrestres utilizando diferentes tipos de sensores, como cámaras aéreas, imágenes satelitales, sensores LiDAR y sensores multiespectrales. Estos sensores

recopilan información en diferentes bandas del espectro electromagnético, desde el visible hasta el infrarrojo cercano y el infrarrojo térmico.

Al combinar y analizar los datos obtenidos a través de la teledetección, los investigadores y agricultores pueden realizar diversas tareas, como:

1. Monitoreo de la salud de los cultivos: La teledetección permite detectar signos tempranos de estrés en las plantas, como deficiencias nutricionales, enfermedades, plagas o sequías. Al analizar las imágenes o datos espectrales, se pueden identificar cambios en la reflectancia de las plantas, que pueden indicar problemas de salud.

2. Evaluación del estado de la vegetación: La teledetección agrícola puede proporcionar información sobre la densidad, el crecimiento y la vitalidad de la vegetación. Los índices de vegetación, como el Índice de Vegetación de Diferencia Normalizada (NDVI), se calculan utilizando la reflectancia de las plantas en diferentes longitudes de onda, lo que permite evaluar el desarrollo y la salud de los cultivos.

3. Determinación de la calidad del suelo: Mediante la teledetección, es posible analizar características del suelo, como la textura, la humedad y la salinidad. Estos datos pueden ser utilizados para tomar decisiones sobre la gestión de riego, la aplicación de fertilizantes y otros aspectos relacionados con la productividad del suelo.

4. Seguimiento de cambios en el uso de la tierra: La teledetección agrícola también es útil para monitorear cambios en el uso de la tierra, como la expansión de áreas agrícolas, la deforestación o el crecimiento urbano. Esto ayuda a comprender el impacto de estas transformaciones en el entorno agrícola y en el medio ambiente.

4.1.4.1 Cámaras RGB

Las cámaras RGB (Red-Green-Blue) en agricultura se utilizan para capturar imágenes que representan los colores visibles del espectro electromagnético. Estas cámaras son una herramienta valiosa en el monitoreo y la gestión de cultivos, ya que permiten obtener información visual detallada sobre las plantas y su entorno [31].

Las cámaras RGB se basan en la captura de luz en tres canales principales: rojo (R), verde (G) y azul (B). Estos canales se combinan para formar imágenes a todo color, similares a las que percibe el ojo humano. Esta capacidad de capturar imágenes en color permite a los agricultores y especialistas agrícolas identificar y analizar una amplia gama de características relacionadas con los cultivos.

4.1.5 Vehículos Aéreos no Tripulados (VANT)

Los Vehículos Aéreos No Tripulados (VANT), también conocidos como drones, son una categoría de vehículos aéreos motorizados, reutilizables y deshabitados. Estos dispositivos son controlados de forma remota, semi-autónoma o autónoma, y tienen la capacidad de llevar a cabo una variedad de tareas específicas tanto dentro como fuera de la atmósfera terrestre, en función de sus misiones y las cargas útiles que pueden transportar. [32]

A diferencia de las aeronaves tripuladas, los VANT no requieren de la presencia física de un piloto a bordo. Sin embargo, esto no implica necesariamente que vuelen de manera completamente autónoma. En muchos casos, un equipo de operadores, pilotos de respaldo y otros especialistas es responsable de supervisar y controlar las operaciones de un VANT. De hecho, en algunos casos, la tripulación de un VANT puede llegar a ser más numerosa que la de una aeronave convencional, especialmente cuando se considera la necesidad de controlar múltiples drones simultáneamente. [33]

Es importante destacar que el término "VANT" es ampliamente utilizado en campos como la Ciencia de la Computación, la Robótica, la Inteligencia Artificial, la Fotogrametría y la Teledetección. No obstante, también es posible encontrar sinónimos menos frecuentes en la literatura, como Vehículo de Piloto Remoto (VPR), Aeronave Operada a Distancia (AOD) o Aeronave de Piloto Remoto (APR), así como Sistemas de Vehículos no Tripulados (SVNT). Estos términos se utilizan para describir la misma categoría de dispositivos que los VANT. [33]

4.1.5.1 Ventajas de los VANT

Los Vehículos Aéreos No Tripulados (VANT) ofrecen numerosas ventajas en una variedad de aplicaciones. En comparación con los sistemas de aeronaves tripuladas, los UAV tienen ventajas significativas. Por un lado, pueden utilizarse en situaciones de alto riesgo sin poner en peligro la vida humana, así como en áreas inaccesibles, volando a baja altitud y perfiles cercanos a los objetos donde los sistemas tripulados no pueden operar [33]. Estas áreas incluyen lugares afectados por desastres naturales como zonas montañosas, áreas volcánicas, llanuras inundables, regiones afectadas por terremotos y zonas desérticas, así como escenas de accidentes.

Además, los drones ofrecen la posibilidad de realizar tareas en condiciones climáticas desfavorables, como en días nublados o con llovizna. En estas circunstancias, la adquisición de datos con UAV sigue siendo posible al volar por debajo de las nubes, mientras que las aeronaves tripuladas con cámaras de gran formato no pueden cumplir los requisitos de vuelo a mayor altitud sobre el terreno. De esta manera, los drones permiten la obtención de datos incluso en condiciones meteorológicas adversas [33].

Los drones no están limitados por las restricciones fisiológicas ni los gastos económicos asociados con los pilotos humanos. Esto los convierte en una opción más flexible y rentable para llevar a cabo diversas tareas. Además, los UAV tienen la capacidad de adquirir datos en tiempo real y transmitir imágenes, videos y datos de orientación de manera instantánea a la estación de control en tierra. Esto permite una toma de decisiones rápida y eficiente durante las operaciones [34].

4.1.5.2 Limitaciones de los VANT

Los vehículos aéreos no tripulados (VANT) presentan una serie de limitaciones desde una perspectiva científica. En primer lugar, se encuentra la limitación de la carga útil, especialmente en los VANT de bajo costo, donde se impone una restricción en cuanto al peso y las dimensiones de los sensores y equipos adicionales que pueden ser transportados [33]. Esta restricción puede afectar significativamente la capacidad de recopilación de datos y la calidad de los mismos [34].

Otra limitación importante de los VANT radica en su autonomía. Estos vehículos dependen de sistemas de energía, como baterías o combustibles, los cuales tienen una capacidad limitada. Esto implica que los VANT, en especial los de menor tamaño o menor costo, tienen una duración de vuelo restringida en comparación con los modelos más grandes y avanzados. Además, los VANT enfrentan limitaciones en términos de altitud y alcance. Existen regulaciones y restricciones que limitan la altitud máxima a la que pueden volar, así como el rango de vuelo que pueden cubrir. Estas restricciones pueden restringir el uso de los VANT en áreas específicas o en tareas que requieren un mayor rango de vuelo, como el monitoreo extenso de grandes áreas geográficas [34].

Las limitaciones meteorológicas también juegan un papel importante en el rendimiento de los VANT. Condiciones climáticas adversas, como fuertes vientos, lluvia intensa o niebla densa, pueden afectar la capacidad de vuelo de los VANT, así como la estabilidad de la plataforma y la calidad de los datos recopilados durante el vuelo [34]. Es importante destacar que los VANT carecen de la percepción humana y la capacidad de respuesta instantánea ante situaciones imprevistas. A diferencia de los pilotos humanos, los VANT no poseen la capacidad de reaccionar rápidamente ante obstáculos o eventos inesperados durante el vuelo. Esta limitación puede comprometer su capacidad para evitar colisiones o realizar ajustes en tiempo real.

4.1.6 Inteligencia Artificial

La IA, o inteligencia artificial, es el campo de la ciencia e ingeniería que busca crear máquinas y programas informáticos inteligentes. Se ocupa de desarrollar métodos para que las máquinas muestren comportamientos inteligentes, incluso si no imitan la inteligencia humana. La inteligencia se refiere a la capacidad computacional de lograr objetivos en el mundo, que puede variar en humanos, animales y máquinas. Aunque no existe una definición sólida de inteligencia, los investigadores de IA estudian y resuelven problemas del mundo real utilizando diversos métodos, sin limitarse a la observación humana. [35]

El avance de la IA ha sido impulsado por la combinación compleja de matemáticas, computación y datos masivos. Las matemáticas proporcionan los fundamentos incuestionables en los que se basan las máquinas, mientras que los

avances en hardware e informática han permitido la puesta en práctica de estos conceptos. Sin embargo, es esencial alimentar estas herramientas con datos, un recurso valioso en la era actual impulsada por los datos. El acceso instantáneo a Internet y los dispositivos equipados con sensores han permitido recopilar datos en tiempo real sobre el comportamiento en línea y diversas transacciones [36].

En el ámbito agrícola, la IA desempeña un papel crucial al convertir los datos de los cultivos en información útil para los agricultores en la toma de decisiones [37]. Esto contribuye a reducir el impacto ambiental y aumentar la sostenibilidad en la agricultura. La combinación de tecnologías digitales con IA ofrece oportunidades para mejorar la eficiencia de las operaciones agrícolas y reducir el impacto ambiental. Además, la IA ha facilitado la interoperabilidad de datos entre sistemas de almacenamiento, la gestión de fertilizantes y el seguimiento de cultivos mediante visión artificial.

En el contexto específico de este trabajo de grado, la implementación de técnicas de Deep Learning en drones permitirá monitorear los cultivos de banano de manera más precisa y eficiente. Se espera que este enfoque contribuya a la recopilación de grandes volúmenes de datos nuevos de cultivos y a extraer información útil para mejorar el monitoreo y la detección de enfermedades.

4.1.7 Enfermedades en Cultivos de Banano

4.1.7.1 Fusiarium wilt of Banana

La enfermedad de *Fusarium wilt* del banano es causada por el hongo *Fusarium oxysporum Schlechtend.*:Fr. f.sp. cubense (Foc)[38]. Esta enfermedad, también conocida como enfermedad de Panamá [39] es de naturaleza policíclica y sus principales medios de dispersión son al entrar en contacto con fuentes de agua contaminadas, herramientas de campo, o por las oxisporum que entren en contacto por las raíces de la planta [39].

Los síntomas iniciales incluyen una decoloración marrón rojiza en las raíces más finas o más pequeñas que no son leñosas en los sitios de infección y el amarillamiento de las hojas más antiguas. El proceso de amarillamiento de las hojas comienza en el borde y avanza hacia la nervadura central. Posteriormente, el pecíolo se vuelve marrón

y se dobla. Las plantas infectadas desarrollan con frecuencia grietas longitudinales en el pseudotallo justo encima del nivel del suelo. El síntoma externo típico es la presencia de hojas muertas alrededor del pseudotallo, lo que parece una falda. Con el tiempo, la hoja central se marchita y el pseudotallo permanece en pie hasta que se retira o se derrumba [40], [41], [9].

(a) Foto desde Tierra. (b) Foto desde UAV.

Figura 4.7: Planta de Banano Infectada con Fusarium [fuente propia].

La sección transversal del cormo y el pseudotallo muestra una decoloración marrón purpúrea de los haces vasculares, mientras que, en el cormo, la decoloración aparece como una colección de diminutos puntos y vetas rojizos o parduzcos [42]. La decoloración del rizoma es grave donde el cilindro vascular se une al córtex [38]. En general, las plantas infectadas no producen racimos de frutas y, si los producen, las frutas son muy pequeñas, con pocos dedos que tienen una pulpa esponjosa y ácida [42].

4.1.7.2 Xanthomonas wilt of Banana

La enfermedad del marchitamiento por *Xanthomonas* conocida en inglés como *Xanthomonas wilt*, es una enfermedad presente en cultivos de bananos y otras variedades de musaceas causada por la bacteria *Xanthomonas campestris pv. musacearum* (Xcm). Esta enfermedad fue reportada por primera vez en 1968 en la planta del ensete [43] y posteriormente en el plátano en 1974 [43] en Etiopía. Desde

2001, se ha reportado la presencia de la enfermedad en Uganda [44], RD Congo [45], Ruanda, Tanzania [46], Kenia [47] y Burundi [47].

La infección ocurre cuando la bacteria ingresa al sistema vascular de la planta a través de cualquier orifico expuesto que entre contacto con algún insecto o herramienta de campo infectado con la bacteria. La enfermedad también puede transmitirse mediante el uso de material de siembra infectado [45]. Se ha implicado a animales que se alimentan del rizoma, como el cerdo hormiguero y el puercoespín, en la propagación local en los jardines de ensete en Etiopía [43] y, teóricamente, rumiantes en libertad, aves voladoras más grandes y murciélagos también podrían actuar como posibles vectores de la enfermedad [48].

(a) Foto desde Tierra. (b) Foto desde UAV.

Figura 4.8: Planta de Banano Infectada con Xanthomonas wilt [fuente propia].

Los síntomas suelen ser evidentes dentro de las 3 semanas después de la infección, aunque el tiempo necesario para alcanzar diferentes etapas de expresión de síntomas puede diferir según el cultivar, la etapa de crecimiento de la planta, el modo de transmisión de la enfermedad y las condiciones ambientales [49] [50]. Las plantas infectadas muestran un amarillamiento y marchitamiento progresivo de las hojas, y una maduración desigual y prematura de la fruta [44]. Los primeros síntomas en las yemas masculinas se observarán si se produce una infección a través de la parte masculina de la inflorescencia, mientras que los síntomas de marchitamiento de las hojas se observarán primero si la infección ocurrió a través de otras partes de la planta

como raíces, cormo, vainas de hojas y hojas. Las secciones transversales de los pseudotallos enfermos revelan vetas amarillas o marrones en el tejido vascular y una secreción bacteriana amarillenta [44].

4.2 Antecedentes

Se realizó una revisión de estudios previos relacionados con el tema y que han sido un aporte en la realización de este trabajo. A continuación, se muestran dichos estudios:

4.2.1 AI-powered banana diseases and pest detection

Este artículo realizado por M. Selvaraj en 2019, presenta un estudio sobre el desarrollo de un sistema de detección de enfermedades y plagas en plátanos basado en inteligencia artificial (IA) utilizando redes neuronales convolucionales profundas (DCNN) para ayudar a los agricultores de plátanos. Los investigadores recolectaron conjuntos de datos extensos de imágenes de enfermedades y plagas de plátanos preseleccionadas por expertos de diversas áreas de África e India del Sur. Para construir un modelo de detección, reentrenaron tres arquitecturas diferentes de redes neuronales convolucionales (CNN) utilizando el enfoque de transferencia de aprendizaje. Se desarrollaron un total de seis modelos diferentes a partir de 18 clases diferentes de enfermedades (según las partes de la planta) utilizando imágenes recolectadas de diferentes partes de la planta de plátano.

El estudio reveló que los modelos basados en ResNet50 e InceptionV2 tuvieron un mejor rendimiento en comparación con la arquitectura MobileNetV1. Estas arquitecturas alcanzaron una precisión de más del 90 % en la mayoría de los modelos probados, lo cual representa resultados de vanguardia en la detección de enfermedades y plagas en plátanos. Estos resultados experimentales fueron comparables con otros modelos de vanguardia encontrados en la literatura. Con miras a utilizar estas capacidades de detección en dispositivos móviles, se evaluó el rendimiento de SSD (detector de disparo único) MobileNetV1. También se calcularon métricas de rendimiento y validación para medir la precisión de los diferentes modelos en los métodos de detección automática de enfermedades.

Los resultados mostraron que las DCNN son una estrategia robusta y fácilmente desplegable para la detección digital de enfermedades y plagas en plátanos. Utilizando un modelo preentrenado de reconocimiento de enfermedades, se logró un aprendizaje profundo y transferido (DTL) para producir una red capaz de hacer predicciones precisas. Esta alta tasa de éxito del modelo lo convierte en una herramienta útil para la detección temprana de enfermedades y plagas, y esta investigación podría ampliarse para desarrollar una aplicación móvil completamente automatizada que ayude a millones de agricultores de plátanos en países en desarrollo. [1]

El proyecto se inspira en las metodologías aplicadas en la detección de enfermedades y plagas en bananos utilizando IA, como se discute en [1]. Este trabajo previo constituye una base técnica para nuestro enfoque, aunque nos diferenciamos al integrar nuevas arquitecturas de aprendizaje profundo y ajustar los modelos para manejar la variabilidad específica de las enfermedades *Fusarium wilt* y *Xanthomonas wilt* en nuestro conjunto de datos proporcionado por el CIAT.

4.2.2 Image processing analysis of geospatial uav orthophotos for palm oil plantation monitoring

En este estudio hecho por Fahmi, se investigó el uso de vehículos aéreos no tripulados (UAV, por sus siglas en inglés) como una herramienta para monitorear las plantaciones de palma de aceite de manera remota. Con fotografías geoespaciales ortográficas, es posible identificar qué parte de la plantación es fértil para los cultivos plantados, lo que significa un crecimiento óptimo. También es posible identificar áreas menos fértiles en términos de crecimiento pero no perfectas, así como áreas del campo de plantación que no están creciendo en absoluto. Esta información se puede obtener rápidamente con el uso de fotos tomadas por UAV.

En este estudio, se utilizó un algoritmo de procesamiento de imágenes para analizar de manera más precisa y rápida las imágenes ortográficas resultantes. Las imágenes ortográficas resultantes se procesaron utilizando Matlab, incluyendo la clasificación de plantas de palma de aceite fértiles, infértiles y muertas utilizando el método de Matriz de Co-ocurrencia de Niveles de Gris (GLCM, por sus siglas en inglés). El método GLCM se desarrolló en base a cuatro parámetros de dirección con grados específicos: 0°, 45°,

90° y 135°. A partir de los resultados de la investigación realizada con 30 muestras de imágenes, se encontró que la precisión del sistema se puede alcanzar utilizando las características extraídas de la matriz como parámetros Contraste, Correlación, Energía y Homogeneidad.[4]

Este antecedente [4] proporciona una perspectiva valiosa sobre cómo las imágenes geoespaciales capturadas por UAV pueden procesarse para monitorear plantaciones a gran escala. Si bien se centra en la palma aceitera, aplicamos técnicas similares para analizar la presencia y el avance de enfermedades en cultivos de banano, ajustando los algoritmos para diferenciar características fenotípicas específicas de las enfermedades de interés.

4.2.3 Detection of banana plants and their major diseases through aerial images and machine learning methods: A case study in DR Congo and Republic of Benin

Este artículo de M.Selvaraj expone el uso de herramientas de detección remota y aprendizaje automático en el monitoreo de cultivos y la vigilancia de enfermedades. Se destaca la importancia de la clasificación de tipos de cultivos y la detección temprana de enfermedades en aplicaciones de detección remota, que proporcionan información precisa y rentable a diferentes niveles de resolución espacial, temporal y espectral. Se menciona que la mayoría de los sistemas de vigilancia de enfermedades se centran en soluciones basadas en un solo sensor y se carece de la integración de múltiples fuentes de información.

Además, se señala que el monitoreo de paisajes más grandes mediante el uso de vehículos aéreos no tripulados (UAV) presenta desafíos, y se propone combinar datos de imágenes satelitales de alta resolución con modelos avanzados de aprendizaje automático a través de aplicaciones móviles para detectar y clasificar plantas de plátano y obtener información sobre su estado de salud general.

El estudio describe cómo se clasificó el plátano en paisajes africanos complejos utilizando clasificaciones basadas en píxeles y modelos de aprendizaje automático derivados de imágenes satelitales y plataformas UAV. Se menciona que el modelo de

clasificación basado en píxeles utilizando el algoritmo Random Forest (RF) con características combinadas de índices de vegetación y análisis de componentes principales logró una precisión de más del 90 % en imágenes multiespectrales de alta resolución.

Además, se presenta un sistema de modelo mixto que utiliza imágenes aéreas UAV-RGB para la detección y clasificación simultánea de plátanos y enfermedades. Se menciona que este modelo logró una alta precisión en la clasificación de plantas sanas y enfermas en diferentes clases de enfermedades, como la enfermedad de la punta del racimo de plátano (BBTD) y la marchitez bacteriana del plátano (BXW). Se destaca el potencial de estos enfoques de modelos de aprendizaje automático basados en imágenes aéreas para proporcionar un sistema de apoyo a la toma de decisiones en la detección y control de enfermedades del plátano en África.[2]

Este estudio de caso [2] es directamente relevante para nuestro trabajo, ya que aborda el problema que intentamos resolver. Sin embargo, nuestro proyecto amplía el alcance de las técnicas de aprendizaje automático empleadas y se enfoca en la implementación práctica de modelos en el contexto colombiano, proporcionando así un enfoque innovador en la identificación y clasificación de enfermedades.

4.2.4 Deep learning models for plant disease detection and diagnosis

En esta investigación llevada a cabo por Ferentinos, se desarrollaron modelos de redes neuronales convolucionales para realizar la detección y diagnóstico de enfermedades en plantas utilizando imágenes de hojas simples de plantas sanas y enfermas a través de metodologías de aprendizaje profundo. El entrenamiento de los modelos se realizó utilizando una base de datos abierta de 87,848 imágenes, que contenía 25 plantas diferentes en un conjunto de 58 clases distintas de combinaciones (planta, enfermedad), incluyendo plantas sanas. Se entrenaron varias arquitecturas de modelos, y el mejor rendimiento alcanzó una tasa de éxito del 99.53 % en la identificación de la combinación correspondiente [planta, enfermedad] (o planta sana). La tasa de éxito significativamente alta hace que el modelo sea una herramienta muy útil como asesor o sistema de alerta temprana, y es un enfoque que podría ampliarse aún más para respaldar un sistema integrado de identificación de enfermedades en

plantas que funcione en condiciones reales de cultivo [5].

Nos basamos en el análisis presentado en [5] para construir y validar nuestros modelos de Deep Learning. Nuestra investigación difiere en el desarrollo de modelos más robustos y adaptados a datos multiespectrales y en la consideración de condiciones locales para la recopilación de datos y la validación del modelo.

4.2.5 Fruits and vegetables quality evaluation using computer vision

Este artículo realizado por Bhargava se centra en el uso de la visión por computadora para la clasificación y evaluación de calidad de frutas y verduras. El objetivo es superar las limitaciones de la clasificación manual, que es inconsistente, subjetiva y costosa. La automatización de este proceso mediante algoritmos y técnicas de visión por computadora permite obtener resultados más precisos y eficientes.

El artículo proporciona una visión general detallada de los diferentes métodos utilizados en este campo, incluyendo el preprocesamiento de imágenes, la segmentación de objetos, la extracción de características y la clasificación de frutas y verduras en función de su color, textura, tamaño, forma y defectos. Estos aspectos son considerados características sensoriales clave que afectan el valor de mercado, las preferencias del consumidor y la elección de los productos.

Los investigadores han desarrollado diversos algoritmos y enfoques para abordar estos desafíos, y el artículo ofrece una comparación crítica de estos métodos propuestos. Se discuten tanto las ventajas como las limitaciones de cada enfoque, con el objetivo de mejorar la precisión y eficiencia de los sistemas de clasificación de frutas y verduras. [6]

Aunque [6] se enfoca en la calidad general de frutas y vegetales, nuestro proyecto utiliza técnicas de visión por computadora similares como punto de partida para la identificación de patologías específicas en bananos. Nuestro enfoque es único en la personalización de estas técnicas para resolver problemas específicos de enfermedades de banano en un entorno de producción.

5. Diseño y Desarrollo del Proyecto

El diseño y desarrollo de este proyecto se llevó a cabo siguiendo una serie de etapas cruciales. Inicialmente, se realizó un exhaustivo proceso de preprocesamiento de la base de datos proporcionada por el CIAT. Esto incluyó la minuciosa limpieza y organización de los datos, preparándolos para el posterior entrenamiento de modelos de Deep Learning (DL).

Posteriormente, se exploraron y desarrollaron diversos modelos de Deep Learning para la detección de enfermedades en los cultivos de banano. Se consideraron varias arquitecturas, incluyendo redes neuronales convolucionales (CNN) y modelos basados en YOLO.

Una vez entrenados, se llevó a cabo una evaluación exhaustiva del desempeño de cada modelo. Utilizamos métricas relevantes como precisión, recall y F1-score para determinar cuál de ellos era el más eficiente y preciso para la detección de enfermedades en los cultivos de banano.

Tras seleccionar el modelo óptimo, se procedió a su implementación para permitir la detección en tiempo real de enfermedades en los cultivos de banano. Aprovechando los modelos previamente entrenados, optimizamos su desempeño para su ejecución en dispositivos de procesamiento de bajo costo.

Esta implementación no solo facilita una detección eficiente y precisa de enfermedades en tiempo real, sino que también proporciona una solución escalable y económica para los agricultores.

5.1 Recolección y Preprocesamiento de los Datos

Para preparar los datos, se realizó una exhaustiva revisión y organización de las imágenes aéreas de los cultivos de banano, clasificándolas en dos categorías principales:

- Plantas saludables.

- Plantas infectadas.

Se consideraron variaciones que pudieran indicar la presencia de enfermedades como *Fusarium* o *Xanthomonas Wilt*, lo que llevó a simplificar las clases para reducir la complejidad del modelo y facilitar la diferenciación entre las dos condiciones principales, así como para permitir la detección de enfermedades con síntomas similares.

Además, se llevó a cabo un proceso semiautomático para corregir errores en las etiquetas de la base de datos. Se utilizaron el software LabelImg para realizar las anotaciones en formato XML, compatible con TensorFlow. Dado que se requería entrenar modelos con YOLO, se desarrolló un código en colaboración con expertos fitopatólogos del CIAT para transformar las anotaciones de formato PASCAL VOC (.xml) a formato YOLO (.txt). Este proceso incluyó la verificación de una muestra del conjunto de datos corregidos por parte de los expertos para garantizar la precisión de las anotaciones y la corrección de las bounding boxes para que se ajustaran adecuadamente a los límites de cada planta.

Antes de la transformación de las anotaciones, se rectificó la rotación de las imágenes para evitar errores durante el entrenamiento. Es crucial que las imágenes estén correctamente alineadas con su metadata para garantizar un entrenamiento preciso en YOLO, que es susceptible a rotaciones. Tras la limpieza y mejora de las anotaciones, el conjunto de datos se redujo de 12.000 a 5.395 imágenes más realistas y concretas.

Una vez realizadas las anotaciones y transformadas al formato YOLO, se organizó el conjunto de imágenes en carpetas de entrenamiento, validación y prueba. Se utilizaron 5.395 imágenes en total, distribuidas de la siguiente manera: 4.390 para entrenamiento, 516 para validación y 499 para pruebas, garantizando así una distribución adecuada y un entrenamiento efectivo de los modelos basados en Deep Learning.

5.1.1 Captura de Imágenes con Drones

El Centro Internacional de Agricultura Tropical (CIAT) establece contacto con aliados y contratistas en todo el mundo para ubicar zonas bananeras afectadas por enfermedades como *Fusarium wilt y Xanthomonas wilt*, principalmente en África, Vietnam, Perú y Brasil. Tras el contacto con las partes interesadas, se agenda la recolección de imágenes de dron en campos de banana, obteniendo el consentimiento de los propietarios. Sin embargo, debido a las diferentes condiciones y regulaciones de cada país, no es posible establecer un protocolo estandarizado para esta tarea.

El proceso de captura de imágenes implica programar una misión de vuelo en la aplicación utilizada por cada piloto, se corre la misión en los campos afectados y esto genera un archivo de imágenes (en este caso RGB), que se encuentran simplemente dentro de una carpeta. Posteriormente, los pilotos envían esta información a los Investigadores Asociados ubicados en el CIAT Pamira, y somos nosotros quienes nos encargamos de un primer preprocesamiento de los datos.

5.1.2 Construcción de Ortomosaicos

La construcción de ortomosaicos constituye una etapa crucial en el proceso de análisis de imágenes aéreas de cultivos. Una vez obtenidas las imágenes en alta resolución, estas son transferidas a los servidores pertinentes, donde se utilizan herramientas especializadas como el software Agisoft Metashape para llevar a cabo su procesamiento.

Durante este proceso, las imágenes individuales se combinan para crear ortomosaicos, que esencialmente representan una visión compuesta de la zona capturada durante el sobrevuelo. Este resultado se asemeja a la vista satelital proporcionada por Google Maps, con la diferencia principal de que la resolución obtenida suele oscilar entre 1 y 5 centímetros por píxel.

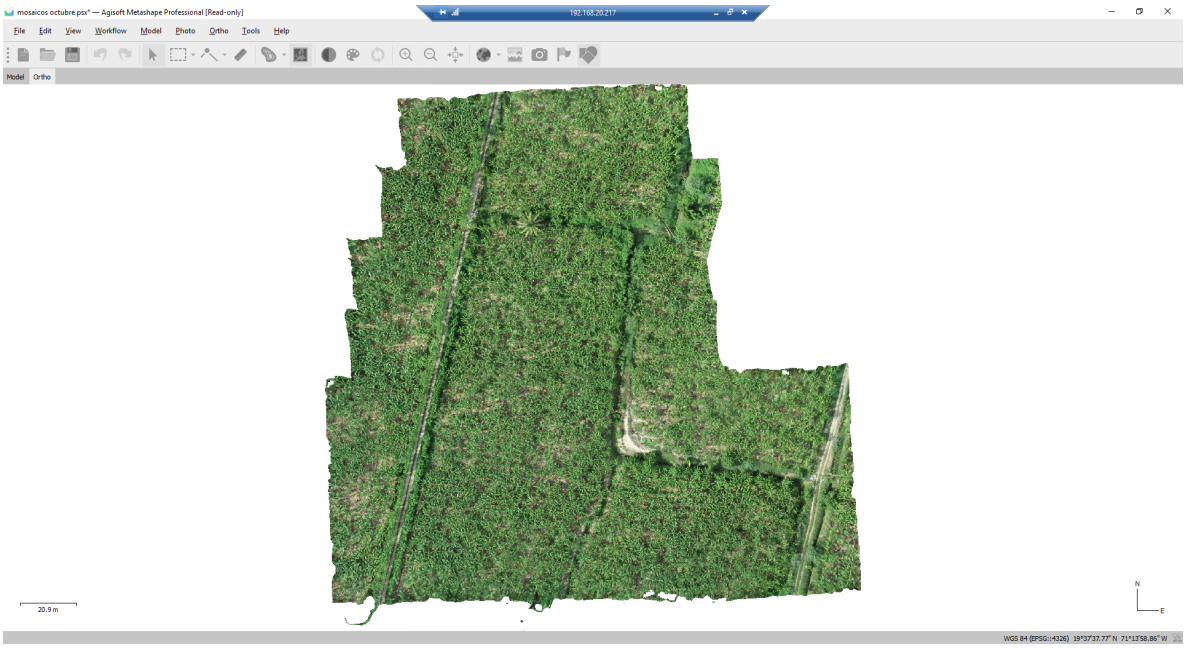


Figura 5.1: Ortomosaico en el software Agisoft Metashape

Comparativamente, esta resolución supera significativamente la ofrecida por Google Maps, que se sitúa en torno a los 30 centímetros por píxel. Este aumento en la resolución permite un nivel de detalle considerablemente mayor, lo que resulta fundamental para el análisis detallado de las imágenes y la identificación de posibles patrones o anomalías en los cultivos.

5.1.3 Primer Análisis de Cultivos

Una vez finalizada la fase de construcción de ortomosaicos en alta calidad, se inicia el proceso de envío de estos mosaicos a fitopatólogos especializados. Estos profesionales desempeñan un papel crucial al llevar a cabo la identificación y clasificación de las plantas de banano presentes en los mosaicos. Su labor consiste en determinar si las plantas están sanas, infectadas con *Xanthomonas* o afectadas por *Fusarium*.

Es importante destacar que este proceso implica un significativo consumo de tiempo debido a la necesidad de examinar meticulosamente miles de plantas en múltiples mosaicos. Los fitopatólogos deben realizar una evaluación minuciosa de cada

planta para determinar su estado de salud y registrar cualquier indicio de enfermedad presente. Este análisis exhaustivo garantiza la precisión en la identificación de las plantas infectadas y contribuye a generar datos fiables para el posterior análisis y seguimiento de la salud de los cultivos de banano.

Además del tiempo dedicado a la identificación de las plantas, los fitopatólogos también deben documentar con detalle sus observaciones, proporcionando información específica sobre la ubicación de las plantas infectadas y la gravedad de la enfermedad. Esta información adicional es crucial para la generación de informes detallados y la toma de decisiones informadas en cuanto a medidas de control y gestión de enfermedades en los cultivos de banano.

5.1.4 Generación Automática de Bounding Boxes

Una vez completada la identificación de las plantas de banano por parte de los fitopatólogos especializados, se procede al envío de los archivos .shp, que contienen la información sobre las plantas identificadas, a los Investigadores Asociados del CIAT Palmira. En esta fase, se asume la responsabilidad de ejecutar un código diseñado específicamente para fragmentar los ortomosaicos en imágenes más pequeñas y livianas, así como para generar automáticamente bounding boxes alrededor de cada una de las plantas identificadas. Estas bounding boxes tienen la finalidad de delimitar visualmente cada planta y etiquetarlas según su estado de salud.

Es importante destacar que estas anotaciones se generan en formato PASCAL VOC, el cual es compatible con TensorFlow. Sin embargo, este formato no es directamente compatible con arquitecturas YOLO, una situación que requerirá una solución más adelante en el proceso. La incompatibilidad con YOLO puede representar un obstáculo temporal, pero se abordará en etapas posteriores para garantizar la integración adecuada de las anotaciones en el flujo de trabajo general del proyecto.

Además de la generación de bounding boxes y etiquetado de las plantas, es posible que se realicen otras tareas durante este proceso, como la normalización de las imágenes fragmentadas para asegurar una consistencia adecuada en cuanto a tamaño y resolución.

Estas acciones adicionales contribuyen a optimizar el procesamiento posterior de las imágenes y facilitan la implementación de algoritmos de detección de enfermedades con mayor precisión y eficacia.

5.1.5 Revisión del Dataset en Colaboración con Fitopatólogos Expertos

En esta etapa del proyecto, disponemos de un dataset inicial que incluye las anotaciones automáticas generadas por el código previo. Estas anotaciones se basan en el archivo .shp que contiene las plantas identificadas en los campos, las cuales están etiquetadas según su estado de salud.

Uno de los primeros pasos cruciales en este punto es verificar que las imágenes no estén rotadas. A menudo, la orientación visual de las imágenes puede diferir de la orientación registrada en su metadata. Es esencial asegurar la consistencia entre ambas orientaciones, ya que las anotaciones están vinculadas a la orientación presente en la metadata de la imagen y no a su orientación visual.

Una vez completada la verificación de la orientación de las imágenes, se procede a una revisión manual exhaustiva del dataset obtenido. Este proceso implica la eliminación de imágenes que no aporten información relevante al modelo, así como la corrección del tamaño de las bounding boxes en caso de que no estén ajustadas al tamaño real de la planta. Además, se colabora con fitopatólogos expertos para identificar y anotar manualmente aquellas plantas que probablemente hayan quedado sin anotar durante la fase automática.

Los fitopatólogos involucrados en este proceso de anotación fueron el Dr. Ahn Hai Nguyen, experto en PCR, ADN, electroféresis, expresión de genes y extracción de ADN. Al igual que la Mágister Nancy Safari, licenciada en en Ciencias Agrícolas y Fototéncnica, y con un máster en geomática, arreglo y gestión de recursos.

Para llevar a cabo la revisión manual de las anotaciones, se utiliza el software LabelImg, una herramienta gratuita y de código abierto ampliamente reconocida por su eficacia en tareas de etiquetado de imágenes.

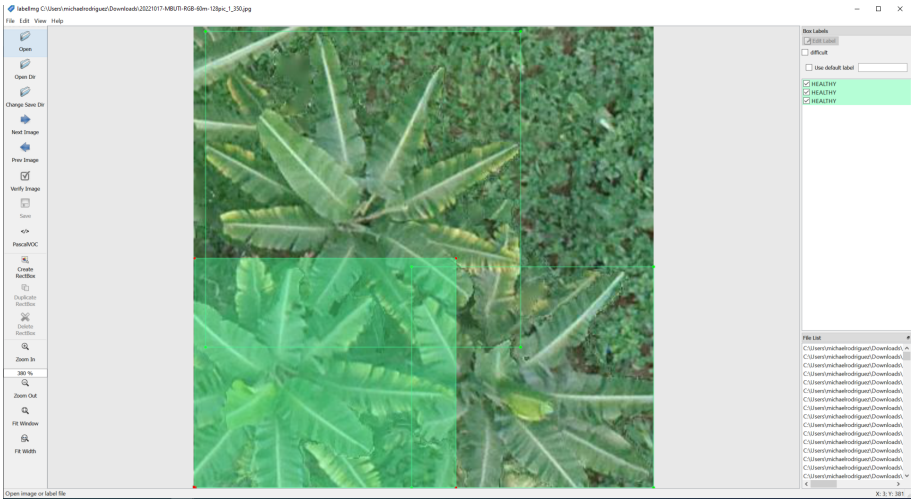


Figura 5.2: Verificación anotaciones en LabelImg

Una vez finalizados todos los procesos mencionados anteriormente, el dataset se compone aproximadamente de 12,000 imágenes. Sin embargo, aún es necesario realizar una revisión minuciosa para garantizar que el dataset esté limpio y listo para su uso en el entrenamiento de modelos de detección de enfermedades en los cultivos de banano.

5.1.6 Prueba de Laboratorio Inicial

Para validar la viabilidad del conjunto de datos obtenido, se lleva a cabo una prueba de laboratorio inicial. En esta fase, se selecciona un subconjunto de datos compuesto por 800 imágenes limpias, revisadas y corregidas meticulosamente. Este subconjunto representa una muestra representativa del dataset completo y se utiliza para realizar un pequeño proceso de entrenamiento.

El objetivo principal de esta prueba es evaluar si los modelos planeados tienen la capacidad de reconocer adecuadamente los individuos en las imágenes. Durante el proceso de entrenamiento, se monitorean métricas relevantes, como la precisión y el recall, para determinar la eficacia del modelo en la identificación de las plantas de banano en diversas condiciones de salud.

Una vez completada la prueba de laboratorio y evaluados los resultados obtenidos,

se llega a una conclusión sobre la idoneidad del conjunto de datos para su uso en la detección de enfermedades en los cultivos de banano. Si los modelos muestran un desempeño satisfactorio en la tarea de reconocimiento, se procede con la revisión y análisis del conjunto de datos completo. En caso contrario, se consideran ajustes adicionales o mejoras en el conjunto de datos antes de continuar con el proceso de revisión integral.

5.1.7 Conversión de Formato de Etiquetas para Entrenamiento

Para adecuar las etiquetas de entrenamiento, se ha desarrollado un proceso para convertir el formato de las anotaciones de PASCAL VOC a YOLO. Este ajuste es necesario para entrenar los modelos en las arquitecturas YOLO, ya que estas requieren un formato específico de etiquetas.

El proceso implica el uso de un código diseñado para realizar esta conversión, el cual transforma las anotaciones de cada imagen en formato .xml a archivos .txt, que contienen las anotaciones en el formato requerido por YOLO.

Además, es crucial crear un archivo .yaml para cada conjunto de datos, el cual debe ser incluido durante el proceso de entrenamiento y validación de los modelos de YOLO V8 y YOLO V9. Este archivo .yaml proporciona información importante sobre la estructura y características del conjunto de datos, asegurando una adecuada configuración durante el entrenamiento.

La estructura del archivo .yaml es la siguiente:

```
names:
- HEALTHY
- SICK

nc: 2

test: /home/mrodriguez/TESIS/Banana/test/images
train: /home/mrodriguez/TESIS/Banana/train/images
val: /home/mrodriguez/TESIS/Banana/val/images
```

Codigo Fuente 5.1: Creacion de rutas de archivos

Este archivo incluye la ruta de los archivos de entrenamiento y validación en formato YOLO (.txt), el número de clases y los nombres de las clases presentes en el conjunto de datos. Es esencial proporcionar esta información para garantizar una configuración adecuada y precisa durante el entrenamiento de los modelos de YOLO.

5.2 Desarrollo de Diferentes modelos de Deep Learning

En el contexto de este trabajo de grado se realizó un análisis de diversas arquitecturas basadas en redes neuronales y algoritmos de Deep Learning ya empleados en la detección de enfermedades en cultivos agrícolas y situaciones similares. El propósito fue identificar las arquitecturas más actuales y adecuadas para nuestra aplicación específica: la detección de enfermedades en cultivos de banano mediante imágenes aéreas.

Concluido este análisis, se eligieron las arquitecturas más pertinentes para la identificación de enfermedades en los cultivos de banano, incluyendo:

1. Faster RCNN - Resnet 50
2. YOLO V8
3. YOLO NAS
4. YOLO V9

Durante este proceso, se tomaron en consideración no solo las métricas de rendimiento del modelo, sino también los tiempos de latencia, aspecto crucial para la implementación innovadora que se buscaba lograr.

Para el entrenamiento de dichos modelos, se realizó la división del conjunto de datos en tres carpetas denominadas: 'train', 'test' y 'val'. En este caso, 'train' contaba con el 80 % de las imágenes, mientras que 'val' y 'test' contaban con el 10 % cada una.

La división se hace de esta manera debido a que los modelos a entrenar cuentan con la particularidad de que dentro de su entrenamiento deben de especificarse estas

tres particiones de los datos. Las imágenes de la carpeta 'train' son usadas únicamente para realizar el entrenamiento, la carpeta 'val' se usa para validar el entrenamiento en cada época y obtener métricas de entrenamiento al finalizarlo. Por otro lado, la carpeta 'test' es la que contiene los datos que nunca fueron vistos por el modelo y con la cual se hacen las diferentes validaciones y pruebas de rendimiento a la hora de evaluar el desempeño de dichos modelos.

Cabe resaltar que tanto el modelo de TensorFlow como los modelos YOLO son modelos preentrenados, en este caso todos han sido entrenados previamente con el famoso dataset COCO y las métricas de desempeño mostradas a continuación están basadas en la evaluación con dicho dataset. De igual modo, para los modelos YOLO existen diferentes variantes dentro de cada arquitectura, iniciando con los modelos S (modelos más reducidos), continuando con los modelos M (modelos de tamaño mediano), y finalizando con los modelos L (los modelos con más parámetros y mejor desempeño). Para nuestro caso en particular, usamos en las diferentes arquitecturas de YOLO los modelos L para garantizar el mejor desempeño posible.

5.2.1 Faster RCNN - Resnet 50

Es una arquitectura desarrollada entre los años 2015 y 2017, y está disponible en el repositorio de Model Zoo. Este modelo, implementado en TensorFlow, ha demostrado un rendimiento notable en la detección de enfermedades en plantas, especialmente cuando se utiliza con imágenes capturadas por teléfonos inteligentes.

Sin embargo, en nuestro contexto, trabajaremos con imágenes capturadas por drones, las cuales pueden presentar características diferentes a las obtenidas mediante teléfonos inteligentes. A pesar de esta diferencia, hemos decidido entrenar el modelo utilizando el mismo conjunto de datos preprocesado que se utilizará en los modelos siguientes.

Es importante mencionar que TensorFlow ofrece diversas formas de configurar y entrenar modelos con datos personalizados. En nuestro caso, hemos optado por utilizar el repositorio Falcon CV, que simplifica considerablemente el proceso de entrenamiento al proporcionar una interfaz intuitiva y simplificada. Con Falcon CV,

solo necesitamos especificar los hiperparámetros, el modelo a utilizar y el conjunto de datos en un solo script, y el repositorio se encarga de manejar la descarga, la configuración y la actualización de los archivos necesarios.

5.2.2 YOLO V8

Una vez definida la implementación del modelo Faster RCNN, se optó por explorar arquitecturas más recientes, que ofrecieran un desempeño más alto y tiempos de latencia menores.

En primer lugar, se decidió explorar la arquitectura YOLO V8, la cual presenta un alto rendimiento en cuanto a precisión media (mAP) y baja latencia, lo que posibilita la detección en video e incluso en tiempo real, aspectos fundamentales para el propósito de nuestro proyecto.

Para la implementación de este modelo, se llevaron a cabo los siguientes pasos:

1. Se elaboró un archivo .yaml (véase Código Fuente 5.1) para definir las rutas y las características de cada conjunto de datos (test, train, val).
2. Se procedió al entrenamiento del modelo utilizando el siguiente código (véase Código Fuente 5.2):

```
from ultralytics import YOLO
import subprocess

command = "yolo task=detect mode=train model=yolov8x.pt data=
    Drone_data.yaml epochs=200 imgsz=640 batch=1"
subprocess.run(command, shell=True)
```

Codigo Fuente 5.2: Entrenamiento de modelo

3. Se realizó la validación del modelo utilizado con datos no vistos ubicados en el subconjunto 'test'.
4. Se realizaron los ajustes requeridos dentro del modelo con el fin de mejorar sus métricas de desempeño. Dichos ajustes incluyen modificaciones en el número de épocas, el batch size y la distribución del dataset.

Este proceso de entrenamiento no sólo permitió obtener el mejor modelo, sino también evaluar las métricas de entrenamiento. Fue un proceso iterativo que involucró refinamientos en el conjunto de datos y ajustes de parámetros del modelo para asegurar su correcto funcionamiento y la obtención de resultados precisos y confiables.

5.2.3 YOLO NAS

Siguiendo con la evaluación de modelos recientes dentro de la arquitectura YOLO, se decidió explorar el modelo YOLO NAS, este modelo cual tiene características similares a las de YOLO V8, con la diferencia de que cuenta con tiempos de latencia más bajos los cuales permiten detecciones más rápidas, pero sacrifican un poco de desempeño, lo que hace que el modelo sea bueno detectando, más no tanto como YOLO V8 como puede observarse en la figura 4.6.

En este caso, el proceso de entrenamiento y métricas es similar al que se sigue en YOLO V8, con la diferencia de que el código es realmente más robusto y no tan amigable para su entrenamiento y validación. Además, la partición del dataset en 'train', 'test' y 'val' es exactamente igual que la usada en los pasos anteriores.

De igual modo, en este caso el proceso de entrenamiento es iterativo, variando parámetros de entrenamiento e intervalos de confianza dentro del modelo para garantizar el mejor desempeño posible.

5.2.4 YOLO V9

Finalmente, y con la intención de estar a la vanguardia de las últimas arquitecturas, se desarrolló un último modelo haciendo uso de la arquitectura YOLO V9. En las pruebas publicadas por Ultralytics se evidencia que este modelo presenta una ventaja significativa en su desempeño cuando se compara con YOLO V8, ya que logra un mejor rendimiento en términos de precisión media (mAP) y una latencia reducida. Sin embargo, es importante destacar que, aunque su latencia es menor que la de YOLO V8 como se muestra en la figura 5.3, no logra ser más baja que la del modelo YOLO NAS.

Se consideró importante realizar la implementación de esta arquitectura debido a que podría ser usada en la identificación dentro de video, lo cual era necesario para la implementación deseada. Por otra parte, al tratarse de un modelo con menor latencia y mayor mAP que YOLO V8 se consideró como una opción importante a la hora de comparar distintos tipos de arquitecturas.

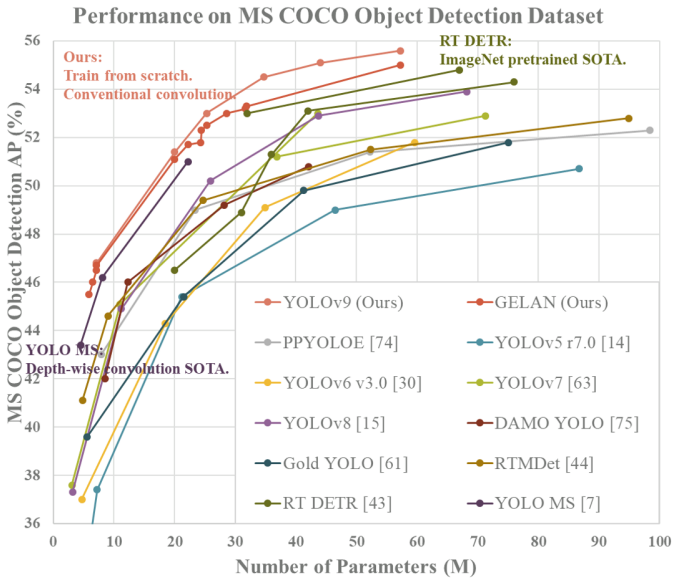


Figura 5.3: Rendimiento YOLO V9
[51]

5.3 Conteo y Seguimiento de Objetos

El conteo y seguimientos de objetos es una tarea fundamental en aplicaciones como, el monitoreo de tráfico, inventario y agricultura de precisión. Para entrar en contexto, el seguimiento de objetos es el proceso de identificar y mantener la trayectoria de uno o varios objetos a lo largo del tiempo en una secuencia, en este caso de video. Esta tarea se dificulta cuando dichos objetos se mueven, se ocultan temporalmente, cambian de apariencia o están en un ambiente de constantes cambios.

En el contexto de la agricultura, el seguimiento y conteo de objetos se convierte en una herramienta poderosa para el monitoreo de cultivos y la detección temprana de enfermedades. Utilizando drones equipados con cámaras y algoritmos avanzados de

visión por computadora, es posible obtener imágenes aéreas de los campos y procesarlas para identificar y rastrear plantas individuales. En este proyecto, se implementa una solución para el seguimiento y conteo preciso de plantas de banano enfermas y saludables utilizando YOLO V8 para la detección de objetos, filtros de Kalman para el seguimiento de las detecciones y el algoritmo de asignación húngaro para la asociación de detecciones entre frames consecutivos. El objetivo es evitar que una misma planta sea contada más de una vez, mejorando así la precisión del conteo.

5.3.1 Filtro de kalman

El filtro de Kalman es un algoritmo que proporciona estimaciones de estados de un sistema dinámico, basándose en mediciones ruidosas o incompletas. El algoritmo utiliza un conjunto de ecuaciones matemáticas para estimar el estado de un sistema dinámico a lo largo del tiempo, aun cuando las mediciones estén incompletas o contaminadas por ruido. Se utiliza comúnmente en aplicaciones de seguimiento debido a su capacidad para predecir y corregir las posiciones de objetos en movimiento.

El filtro de Kalman consta de dos fases principales:

1. **Predicción:** En esta fase, se utiliza el estado anterior para predecir el nuevo estado.

$$\mathbf{x}_k = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{B}\mathbf{u}_k + \mathbf{w}_k$$

$$\mathbf{P}_k = \mathbf{F}\mathbf{P}_{k-1}\mathbf{F}^T + \mathbf{Q}$$

   Aquí, $\mathbf{x}_k$ es el estado predicho, $\mathbf{F}$ es la matriz de transición de estado, $\mathbf{P}_k$ es la matriz de covarianza del error predicho, y $\mathbf{Q}$ es la covarianza del proceso.

2. **Actualización:** En esta fase, se corrige el estado predicho utilizando las mediciones observadas.

   Medición del estado del sistema:

$$\mathbf{y}_k = \mathbf{z}_k - \mathbf{H}\mathbf{x}_k$$

   Aquí, $\mathbf{y}_k$ es la innovación o residual, $\mathbf{z}_k$ es la medición observada, y $\mathbf{H}$ es la matriz

de observación.

Incertidumbre de medida:

$$\mathbf{S}_k = \mathbf{H}\mathbf{P}_k\mathbf{H}^T + \mathbf{R}$$

Aquí, $\mathbf{S}_k$ es la covarianza de la innovación, y $\mathbf{R}$ es la covarianza del ruido de observación.

Ganancia de Kalman:

$$\mathbf{K}_k = \mathbf{P}_k\mathbf{H}^T\mathbf{S}_k^{-1}$$

Aquí, $\mathbf{K}_k$ es la ganancia de Kalman, que determina cuánto se debe ajustar la predicción del estado con la nueva medición.

Actualización de estado:

$$\mathbf{x}_k = \mathbf{x}_k + \mathbf{K}_k\mathbf{y}_k$$

Aquí, $\mathbf{x}_k$ es el estado actualizado.

Actualización de covarianza:

$$\mathbf{P}_k = (I - \mathbf{K}_k\mathbf{H})\mathbf{P}_k$$

Aquí, $\mathbf{P}_k$ es la matriz de covarianza del error actualizado, e I es la matriz identidad.

En la figura 5.4, se proporciona un diagrama de bloques para el Filtro de Kalman

En el contexto de este proyecto, se usa para predecir y actualizar las posiciones de las detecciones de las plantas entre frames.

- **Inicialización del Filtro de Kalman**

```
def create_kalman_filter(dt):
kf = KalmanFilter(dim_x=4, dim_z=2)
kf.x = np.array([0., 0., 0., 0.])  # initial state: [x, y, dx/dt
    , dy/dt]
```

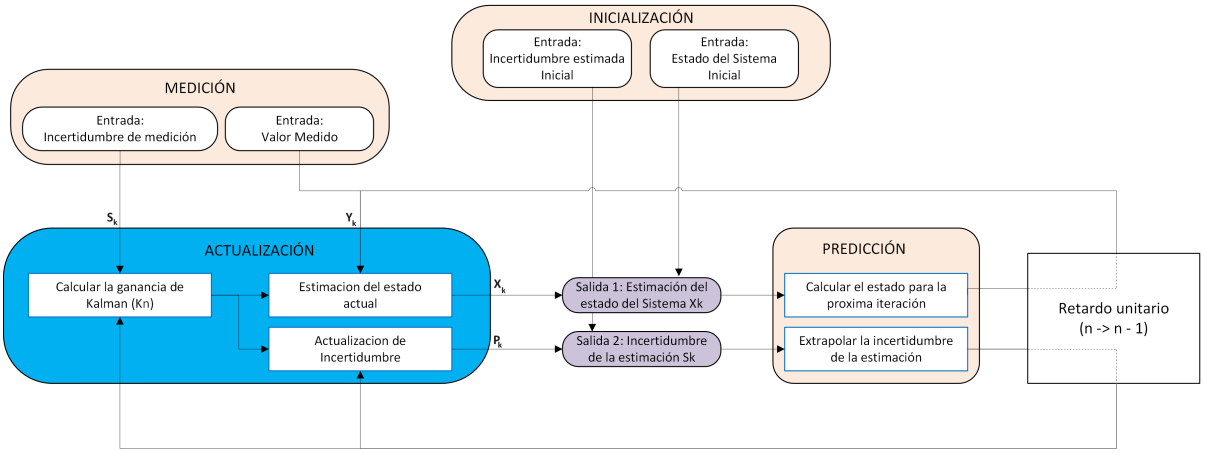


Figura 5.4: Diagrama de bloques Filtro de Kalman

```
4
5  # state transition matrix
6  kf.F = np.array([[1., 0., dt, 0.],
7                   [0., 1., 0., dt],
8                   [0., 0., 1., 0.],
9                   [0., 0., 0., 1.]])
10
11 # measurement function
12 kf.H = np.array([[1., 0., 0., 0.],
13                  [0., 1., 0., 0.]])
14
15 # measurement noise covariance
16 kf.R = np.eye(2) * 0.1
17
18 # process noise covariance
19 kf.Q = Q_discrete_white_noise(dim=2, dt=dt, var=0.01)
20
21 return kf
```

Codigo Fuente 5.3: Actualizacion del Filtro de Kalman

La función *create_kalman_filter* configura el filtro de Kalman con las matrices de transición del estado, de función de medición, de covarianza del ruido de medición y del ruido del proceso. Aquí, *Q_discrete_white_noise* se utiliza para generar la matriz de covarianza del proceso, que representa la incertidumbre en el modelo del sistema.

- **Actualización del Filtro de Kalman**

```
def update_kalman_filter(kf, measurement):
    kf.predict()
    kf.update(measurement)
    return kf.x[:2]
```

Codigo Fuente 5.4: Creación Filtro de Kalman

Esta función predice el estado actual basado en el estado anterior y actualiza el estado del filtro de Kalman con la nueva medición (centroide del bounding box).

5.3.2 Algoritmo de Asignación Húngaro

El algoritmo de asignación húngaro, también conocido como el algoritmo de Kuhn-Munkres, es un método para encontrar una asignación óptima que minimice (o maximice) la suma de los costos en un problema de asignación. En el contexto del seguimiento de objetos, se utiliza para asociar las detecciones de objetos en un frame con las detecciones en el siguiente frame, minimizando la distancia (o el costo) entre las detecciones.

El proceso de asignación incluye los siguientes pasos:

- **Construcción de la Matriz de Costos:**

  La matriz de costos se basa en la distancia euclidiana entre los centroides de las detecciones.

$$C(i, j) = \|d_i - d_j\|$$

  Donde $C(i, j)$ es el costo de asociar la detección i en el frame anterior con la detección j en el frame actual.

- **Resolución del Problema de Asignación:**

  El algoritmo encuentra una correspondencia óptima entre las detecciones, minimizando el costo total.

5.3.2.1 Cálculo de la Distancia Euclidiana

El cálculo de la distancia euclidiana es fundamental para asociar las detecciones de objetos entre frames consecutivos en el seguimiento de objetos. Este cálculo se utiliza en el algoritmo de asignación húngaro para medir la similitud entre las detecciones en diferentes frames.

La distancia euclidiana es una medida de la distancia entre dos puntos en un espacio euclidiano. Dados dos puntos $p = (p_1, p_2, \ldots, p_n)$ y $q = (q_1, q_2, \ldots, q_n)$ en un espacio n-dimensional, la distancia euclidiana $d(p, q)$ se define como:

$$d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \ldots + (p_n - q_n)^2}$$

Para el caso específico del seguimiento de objetos en un video, normalmente se consideran las coordenadas de los centroides de las detecciones en dos dimensiones (x, y). Por lo tanto, si tenemos dos detecciones $\mathbf{d}_i = (x_i, y_i)$ y $\mathbf{d}_j = (x_j, y_j)$, la distancia euclidiana entre estas detecciones es:

$$d(\mathbf{d}_i, \mathbf{d}_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

5.3.3 Q_discrete_white_noise

Esta función se utiliza para generar una matriz de covarianza de ruido de proceso para el filtro de Kalman. Esta matriz modela la incertidumbre en el sistema y ayuda a suavizar las predicciones.

La función tiene la siguiente forma:

$$Q = \text{Q\_discrete\_white\_noise}(\text{dim}, \text{dt}, \text{var})$$

Donde:

- `dim` es la dimensión del espacio de estado.
- `dt` es el intervalo de tiempo.
- `var` es la varianza del ruido.

5.4 Detección en Tiempo Real

Después de considerar diversas opciones para ejecutar el modelo, se determinó que el uso de un computador portátil sería la mejor elección. Esta decisión se fundamenta en que las arquitecturas diseñadas para la detección en tiempo real no están óptimamente adaptadas para dispositivos embebidos, lo que podría restringir su capacidad para procesar grandes volúmenes de datos y detectar múltiples objetos.

Una vez establecido este punto, se llevó a cabo una investigación detallada sobre los drones empleados en agricultura, observando que la mayoría de ellos pertenecen a la marca DJI [52]. Estos drones permiten la conexión de un iPad a su control, posibilitando al usuario el control de ciertas variables adicionales y la visualización en tiempo real de lo que el dron está registrando. Esta función brindó una visualización instantánea de los cultivos a través del iPad, planteando el desafío de transmitir esta imagen al computador encargado de procesar el modelo.

Afortunadamente, los iPads incluyen Airplay, una función estándar de los dispositivos Apple que permite la transmisión inalámbrica de video a otros dispositivos. Utilizando la aplicación Airdroid Cast en un computador con sistema operativo Windows, fue posible visualizar en tiempo real lo que el dron estaba sobrevolando. Con la imagen de la cámara del dron disponible en tiempo real, se procedió a emplear una capturadora para que el video transmitido al computador se viera como si fuera una cámara web estándar.

Finalmente, para completar la implementación, solo fue necesario ejecutar un código que realizara la detección en tiempo real a través de la entrada de la cámara web. De

esta manera, se logró una implementación innovadora que permitió la detección de enfermedades en los cultivos de manera eficiente y en tiempo real. (Ver Figura 5.5).

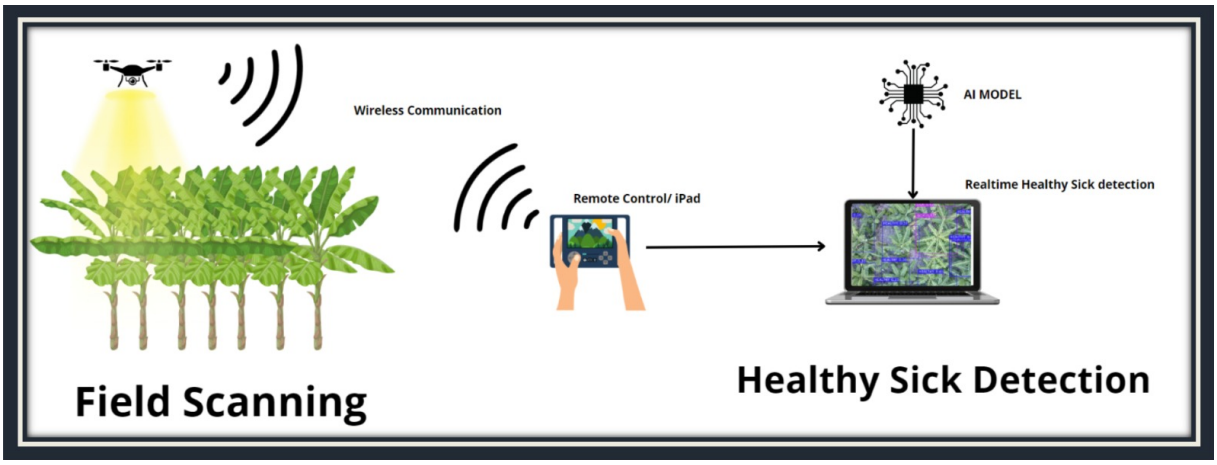


Figura 5.5: Pipeline detección de enfermedades en tiempo real

5.5 Arquitectura Propuesta de la Solución

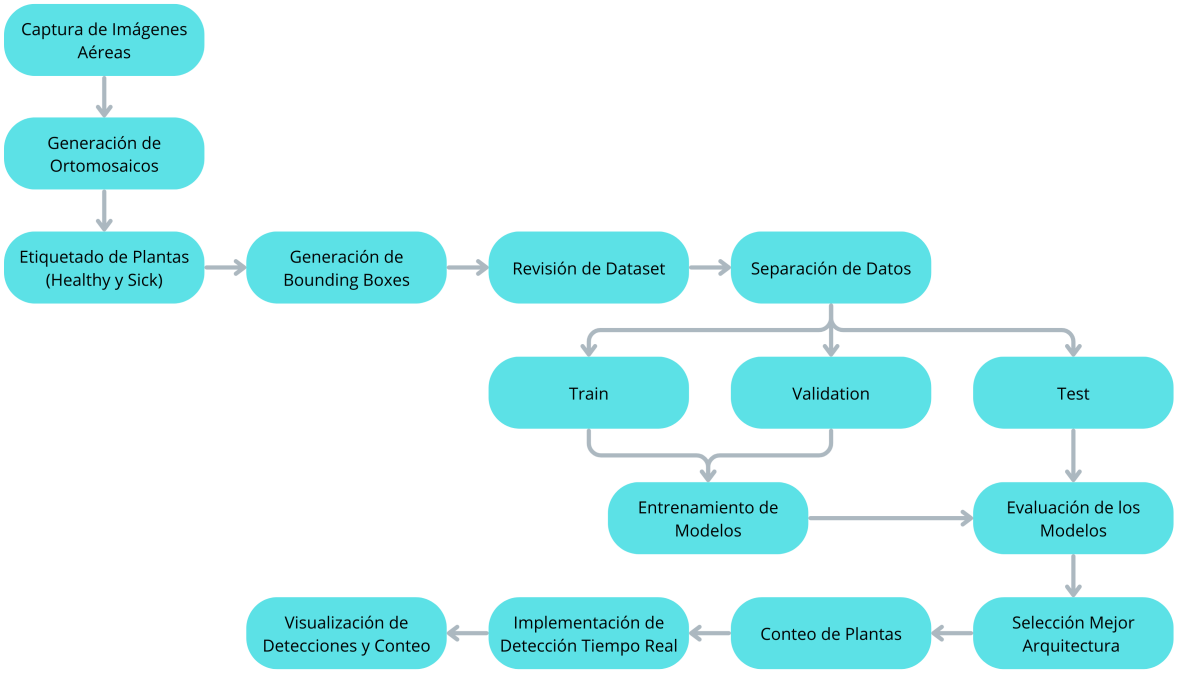


Figura 5.6: Esquema Propuesto del Proyecto

La solución propuesta para la detección y monitoreo de enfermedades en cultivos de banano se basa en una arquitectura integral que combina tecnologías avanzadas de detección de objetos, seguimiento en tiempo real y procesamiento de imágenes. A continuación, se presenta la arquitectura detallada:

1. **Captura de Imágenes Aéreas:** La arquitectura comienza con la captura de imágenes aéreas de los cultivos de banano utilizando drones equipados con cámaras RGB. Estas imágenes son fundamentales para la detección temprana de enfermedades y el monitoreo de la salud de las plantas, al igual que para la construcción de los modelos de Deep Learning.

2. **Preprocesamiento de Imágenes:** Las imágenes capturadas son preprocesadas para mejorar su calidad e identificar sus síntomas. Esto incluye la generación de ortomosaicos, la anotación de las plantas sanas y enfermas, la generación de las etiquetas y finalmente, la revisión de la calidad de las mismas.

3. **Entrenamiento de Modelos:** Después de realizar el preprocesamiento de imágenes, se procede a realizar el entrenamiento de diferentes modelos haciendo uso de dichas imágenes. Para este caso, se entrenan modelos Faster Rcnn Resnet 50, YOLO V8, YOLO NAS y YOLO V9.

4. **Evaluación de Desempeño de los Modelos:** Una vez entrenados los modelos, se realiza una evaluación de desempeño de los modelos con el fin de seleccionar el más prometedor para continuar con el proceso. En este caso se toman en cuenta las matrices de confusión, las métricas de entrenamiento y las métricas de prueba de los modelos.

5. **Detección de Enfermedades:** Utilizamos un modelo de detección de objetos basado en aprendizaje profundo, específicamente el modelo preentrenado YOLO V8 (You Only Look Once, versión 8), para identificar y localizar enfermedades en las imágenes de los cultivos de banano. Este modelo ha sido entrenado con un conjunto de datos diverso y etiquetado correctamente para reconocer enfermedades específicas.

6. **Seguimiento y Conteo de Plantas:** Se implementa una solución para el seguimiento y conteo preciso de plantas de banano enfermas y saludables. Esta

solución utiliza YOLO V8 para la detección de objetos, filtros de Kalman para el seguimiento de las detecciones y el algoritmo de asignación húngaro para la asociación de detecciones entre frames consecutivos.

7. **Integración y Procesamiento en Tiempo Real:** Toda la información capturada y procesada, incluyendo las imágenes y los resultados de detección y seguimiento, se integra en un sistema centralizado. Utilizamos un computador portátil para realizar el procesamiento en tiempo real de la información capturada por el dron y ejecutar el modelo de detección de enfermedades. Esto permite tomar decisiones rápidas y precisas sobre intervenciones agronómicas según sea necesario.

8. **Visualización y Análisis:** Los resultados de la detección y el seguimiento se visualizan en una interfaz gráfica que proporciona al usuario una representación visual de la salud de los cultivos de banano. Esta interfaz permite a los agricultores y técnicos agrícolas visualizar los cultivos y tomar decisiones informadas sobre el manejo de la plantación.

6. Resultados y Análisis

Dentro del desarrollo de este proyecto de grado, se han realizado una serie de pruebas exhaustivas con sus respectivos análisis, esto con el propósito de evaluar de manera concreta el desempeño de cada uno de los modelos realizados y la eficacia de las soluciones propuestas. Se inicia con la construcción de una base de datos preprocesada con imágenes de excelente calidad, siguiendo con el entrenamiento de cuatro modelos con diferentes arquitecturas, realizando la evaluación de estos haciendo uso de imágenes, continuando con la evaluación de videos, para posteriormente evaluar la implementación del conteo y del desempeño del modelo en su detección en tiempo real, para finalizar con la comparación del modelo obtenido con otros encontrados en la literatura. Todo esto con el fin de garantizar la precisión y la confiabilidad de nuestro sistema en condiciones reales.

6.1 Conjunto de Datos Preprocesado

Después de revisar las 12,000 imágenes con la colaboración de los fitopatólogos expertos, el Dr. Ahn Hai Nguyen y la Mágister nancy Safari, y de depurar aquellas imágenes que no son útiles, se ha generado un conjunto de datos final. Este conjunto consta de un total de 5,395 imágenes, con 31,264 plantas anotadas en total. Estas plantas se dividen en 25,999 individuos clasificados como 'healthy' y 5,265 individuos clasificados como 'sick'.

Para simplificar el proceso de análisis y considerando la similitud de los síntomas entre *Fusarium y Xanthomonas*, se ha decidido unificar estas categorías en una sola, denominada 'sick'. Del mismo modo, la categoría de plantas sanas se ha etiquetado como 'healthy'.

Siguiendo la metodología estándar para el manejo de conjuntos de entrenamiento, se ha dividido el conjunto de datos en tres subconjuntos: train, val y test. La distribución de estas divisiones es la siguiente:

- Train: 4,390 imágenes, que incluyen 21,369 individuos HEALTHY y 4,559 individuos SICK.
- Val: 516 imágenes, que comprenden 2,159 individuos HEALTHY y 319 individuos SICK.
- Test: 499 imágenes, que contienen 2,471 individuos HEALTHY y 387 individuos SICK.

Esta distribución equitativa y representativa asegura una adecuada evaluación de los modelos de detección de enfermedades en los cultivos de banano y permite una validación rigurosa de su desempeño en diferentes conjuntos de datos.

(a) Ejemplo 1 (b) Ejemplo 2 (c) Ejemplo 3

Figura 6.1: Ejemplos dataset preprocesado

6.2 Evaluación y Desempeño de Modelos con Imágenes

Como era de esperarse, para la construcción de los modelos se probaron diferentes tipos de entrenamiento para cada uno de ellos, variando en general la distribución de los conjuntos de datos y el número de imágenes usadas, por otro lado, se varió el batch size y el número de épocas para el entrenamiento de cada modelo.

Después de realizar dichas modificaciones, se seleccionó la configuración que resultó ser más óptima cada caso, contando finalmente con el conjunto de datos que se describe en la etapa de desarrollo del proyecto.

Los resultados de desempeño mostrados a continuación corresponden a los mejores resultados obtenidos para cada uno de los modelos.

6.2.1 Faster RCNN - Resnet 50

Una vez completado el entrenamiento del modelo con los conjuntos de train y val, hemos evaluado su desempeño utilizando el conjunto de test (datos no vistos), para obtener métricas sobre imágenes que no han sido vistas previamente por el modelo. Como resultado de esta evaluación, hemos obtenido una matriz de confusión que proporciona información detallada sobre la capacidad de nuestro modelo para clasificar correctamente las imágenes.

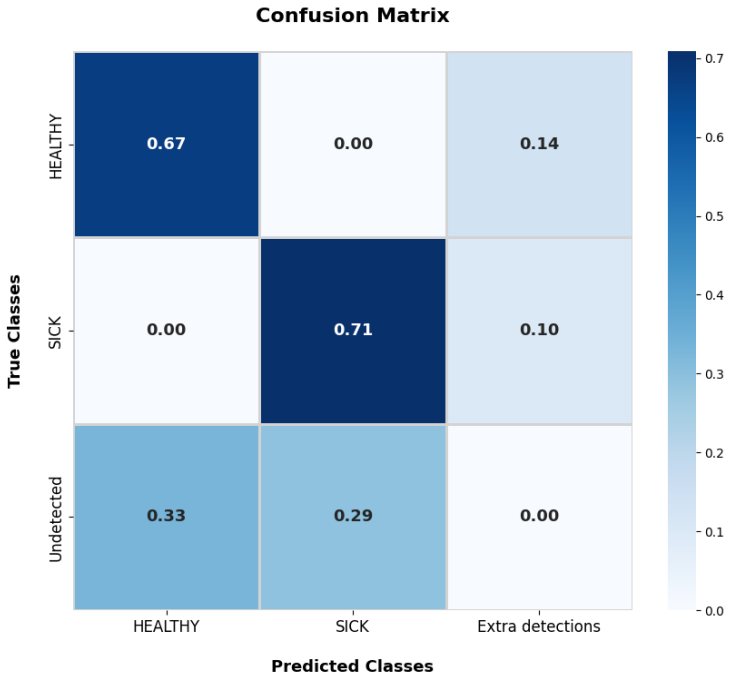


Figura 6.2: Matriz de confusión Faster RCNN

La matriz de confusión revela que, en términos generales, no se detectaron plantas de una categoría como pertenecientes a la categoría contraria, es decir que ninguna planta enferma fue detectada como sana y viceversa. Sin embargo, también se observa un número considerable de plantas que no fueron detectadas. Esto se debe a la naturaleza de

los campos de banano, donde las plantas suelen estar muy juntas, organizadas de modo aleatorio y con personas, pequeñas construcciones y vehículos dentro de las plantaciones, lo que dificulta su identificación incluso para el ojo humano.

Además, se observaron detecciones adicionales, que podrían ser plantas no anotadas en el dataset, detecciones duplicadas o erróneas en zonas sin plantas.

Por otro lado, las métricas de desempeño del modelo muestran resultados aceptables

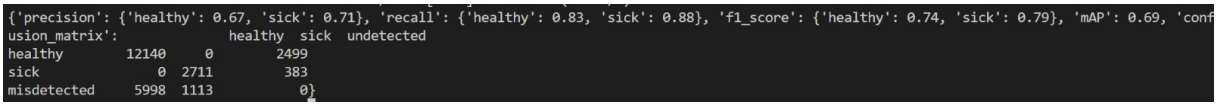


Figura 6.3: Métricas de test Faster RCNN

- **Precision:** El modelo alcanza una precisión de 0.67 para la clase 'healthy' (plantas sanas) y de 0.71 para la clase 'sick' (plantas enfermas con *Xanthomonas* o **Fusarium wilt**). Esto indica que el 67 % de las detecciones clasificadas como 'healthy' son realmente plantas sanas, mientras que el 71 % de las detecciones clasificadas como 'sick' son realmente plantas enfermas.

- **Recall:** El recall, o exhaustividad, del modelo es de 0.83 para la clase 'healthy' y de 0.88 para la clase 'sick'. Esto significa que el modelo logra identificar correctamente el 83 % de todas las plantas sanas presentes en el conjunto de datos y el 88 % de todas las plantas enfermas.

- **F1 score:** El F1 score, que combina precision y recall en una sola métrica, es de 0.74 para la clase 'healthy' y de 0.79 para la clase 'sick'. Esta métrica proporciona una medida del equilibrio entre la precisión y la exhaustividad del modelo para cada clase.

- **mAP(mean Average Precision):** El valor del mAP es 0.69. Esta métrica proporciona una medida agregada de la precisión del modelo para diferentes umbrales de confianza. Cuanto mayor sea el valor del mAP, mejor será el rendimiento general del modelo en la detección de objetos en diferentes condiciones. En este caso, un mAP de 0.69 indica un buen rendimiento del modelo en la detección de enfermedades en los cultivos de banano.

Estos valores indican un desempeño general satisfactorio de la arquitectura, lo que sugiere que es factible implementar el modelo de detección deseado. Es posible observar algunas de las predicciones realizadas por esta arquitectura en las siguientes imágenes:

(a) Predicción 1 (b) Predicción 2 (c) Predicción 3

Figura 6.4: Predicción Faster RCNN – Resnet 50

Sin embargo, es importante destacar que, aunque el modelo presenta un buen desempeño en métricas, también tiene altos tiempos de latencia. Esto impide la implementación de detección en tiempo real y dificulta la detección en video, lo cual es un requisito para el caso de uso en cuestión.

Por consiguiente, se ha decidido buscar arquitecturas con buen desempeño en métricas, pero también con bajos tiempos de latencia, con el objetivo de mejorar la eficiencia y la viabilidad de la implementación en tiempo real del modelo final.

6.2.2 YOLO V8

Una vez finalizado el entrenamiento del modelo, fue posible obtener las gráficas de entrenamiento que muestran el desempeño general del entrenamiento de dicho modelo.

Las gráficas de entrenamiento (Figura 6.7) proporcionan una visualización detallada del proceso de entrenamiento del modelo de detección de enfermedades en cultivos de banano. Cada una de las métricas y pérdidas monitoreadas ofrece una perspectiva única sobre el rendimiento y la eficacia del modelo a lo largo del tiempo.

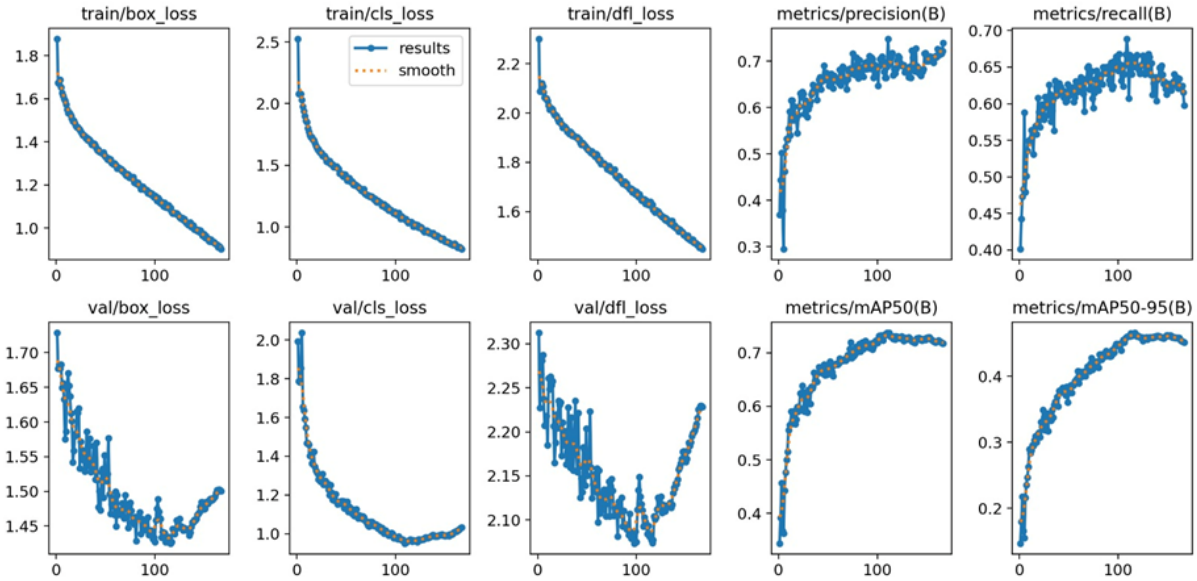


Figura 6.5: Gráfica de entrenamiento YOLO V8

Dentro de la gráfica anterior es posible encontrar funciones de pérdidas, datos de comportamiento de la precisión y el recall con el paso de las épocas, las pérdidas y el mAP.

Por un lado, se observa que las pérdidas obtienen en general el comportamiento esperado, disminuyendo con el paso de las épocas, acercándose a valores relativamente bajos. Así mismo, se observa un comportamiento creciente en las métricas de desempeño, en donde destaca un punto en especial un poco después de la época 100, pues su precisión y recall son notablemente superiores.

A continuación, se procede a realizar la validación con los datos no vistos (conjunto test), en donde se obtuvo la siguiente matriz de confusión:

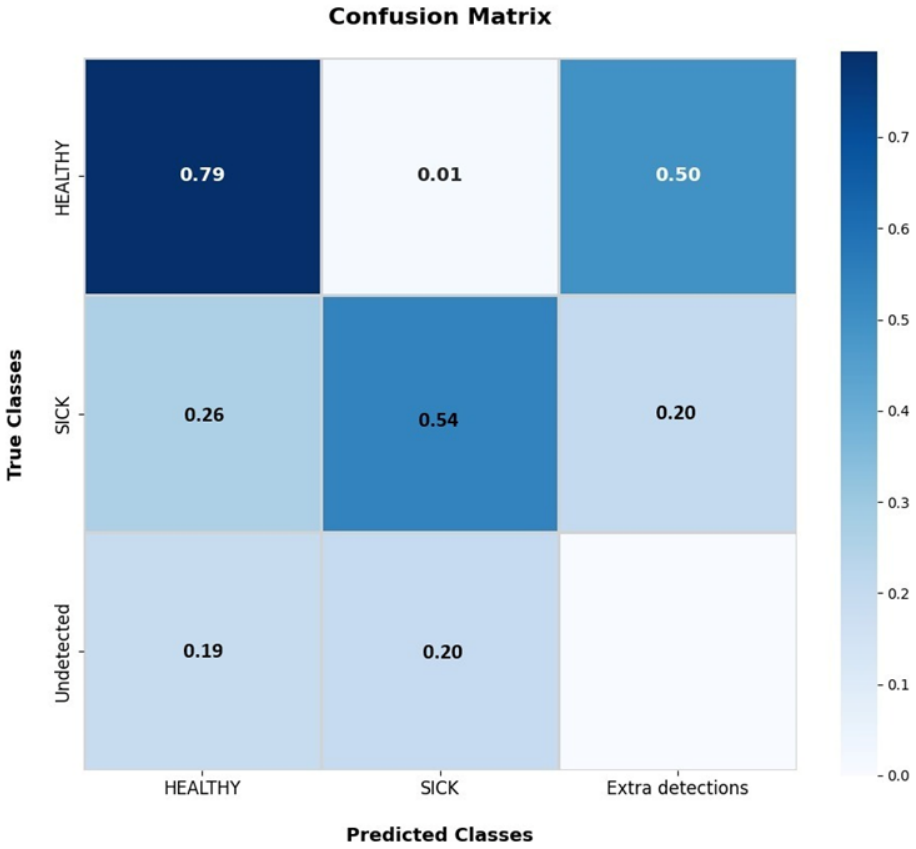


Figura 6.6: Matriz de confusión YOLO V8

La matriz de confusión (Figura 6.6) revela que, en términos generales, el modelo identifica correctamente las plantas sanas como tales en un 79 % de los casos y las plantas enfermas como enfermas en un 54 % de los casos. Sin embargo, se observa un número considerable de plantas que no fueron detectadas, representadas por la fila "undetected". Este valor es particularmente alto, con un 19 % de las plantas sanas no detectadas y un 20 % de las plantas enfermas no detectadas.

Este comportamiento es similar al que se evidencia en el modelo anterior, por lo que una vez más se concluye que este fenómeno puede atribuirse a la estructura de las plantaciones de banano analizadas, en donde las plantas suelen estar muy próximas unas de otras. La estrecha proximidad entre las plantas dificulta la detección precisa, tanto para el modelo como para el ojo humano. Esto sugiere que, a pesar de la capacidad del modelo para distinguir entre plantas sanas y enfermas en general, la detección se ve obstaculizada por la falta de claridad en la separación entre las plantas individuales.

De igual modo, para la validación de los datos no vistos del modelo YOLO V8, se obtuvieron las siguientes métricas de rendimiento:

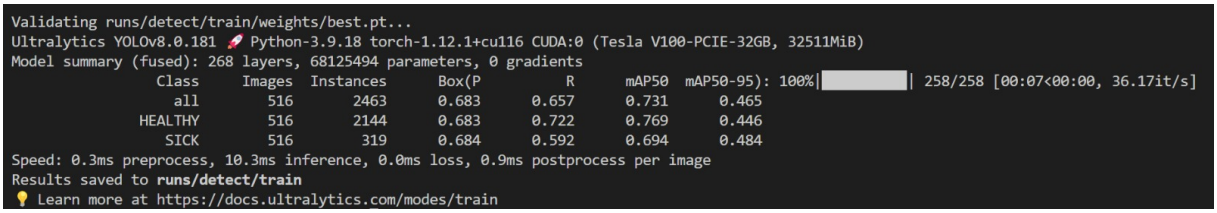


Figura 6.7: Métricas de test YOLO V8

- **Precisión:** El modelo alcanza una precisión de 0.683 para la clase 'healthy' y 0.684 para la clase 'sick'. Teniendo un comportamiento ligeramente mayor en la clase healthy respecto al modelo Faster RCNN, y ligeramente menor para la clase sick.
- **Recall:** El recall, o exhaustividad, del modelo es de 0.722 para la clase 'healthy' y 0.592 para la clase 'sick'. Siendo métricas inferiores que en el modelo Faster RCNN.
- **mAP (mean Average Precision):** El valor del mAP general es 0.731. Siendo más alta que en el caso del modelo anterior, indicando que el modelo tiene un mejor desempeño en condiciones variadas.

El modelo YOLO V8 muestra un buen rendimiento en la clasificación de plantas sanas y enfermas, con una precisión y recall razonables. Adicionalmente, se debe de tomar en cuenta que la latencia de este modelo es realmente inferior a la del modelo Faster RCNN, lo que permite hacer detección en videos donde el dron vaya a una velocidad relativamente rápido, e incluso en tiempo real que es lo que se busca para la implementación innovadora.

En total son más de 30 mil anotaciones, para garantizar la validez del modelo, se realizó una validación con imágenes aleatorias para evaluar la capacidad de detección del mismo.

(a) Predicción 1 (b) Predicción 2 (c) Predicción 3

Figura 6.8: Predicción YOLO V8

Después de validar las imágenes, se observa que el modelo YOLO V8 tiene una buena precisión en la detección de las plantas de banano en general, pues no confunde las plantas con otros elementos del entorno, de igual modo, se observa que funciona mejor cuando las imágenes muestran a las plantas más cerca, lo que podría indicar que para la detección en tiempo real será importante que el dron vuele a una altura relativamente baja.

6.2.3 YOLO NAS

Al igual que para el modelo de YOLO V8, se obtienen las siguientes gráficas (Figura 6.11), las cuales muestran el desempeño en el entrenamiento para el modelo de YOLO NAS:

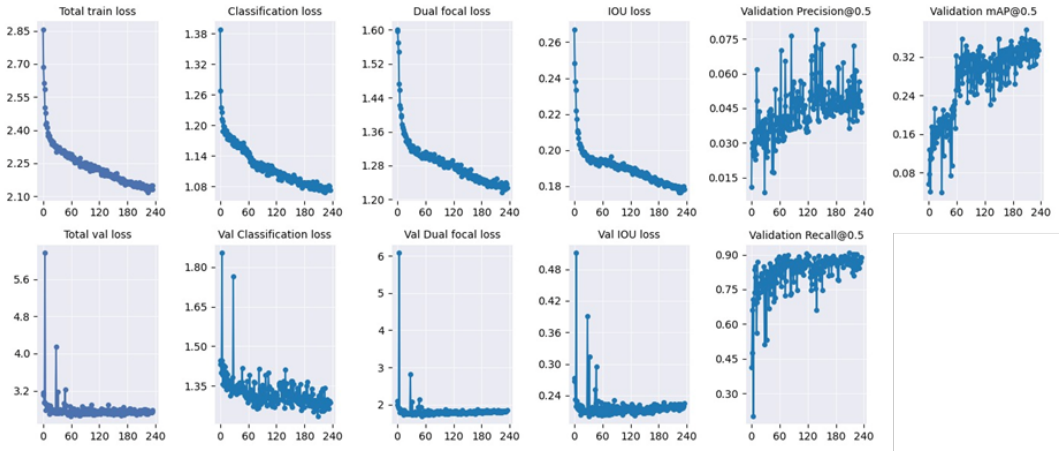


Figura 6.9: Gráfica de entrenamiento YOLO NAS

Analizando las gráficas de entrenamiento, se concluye que los resultados no son los esperados. Aunque las pérdidas en general disminuyen con el pasar de las épocas, siguen siendo bastante altas, pues ni siquiera son menores a uno, por otro lado, se observa que aunque las métricas de validación aumentan con las épocas, tienen datos muy dispersos y no muestran una mejoría notable después de la época 50.

Ahora bien, se realiza la validación con los datos del conjunto test (datos no vistos), y se obtiene la siguiente matriz de confusión (Figura 6.10):

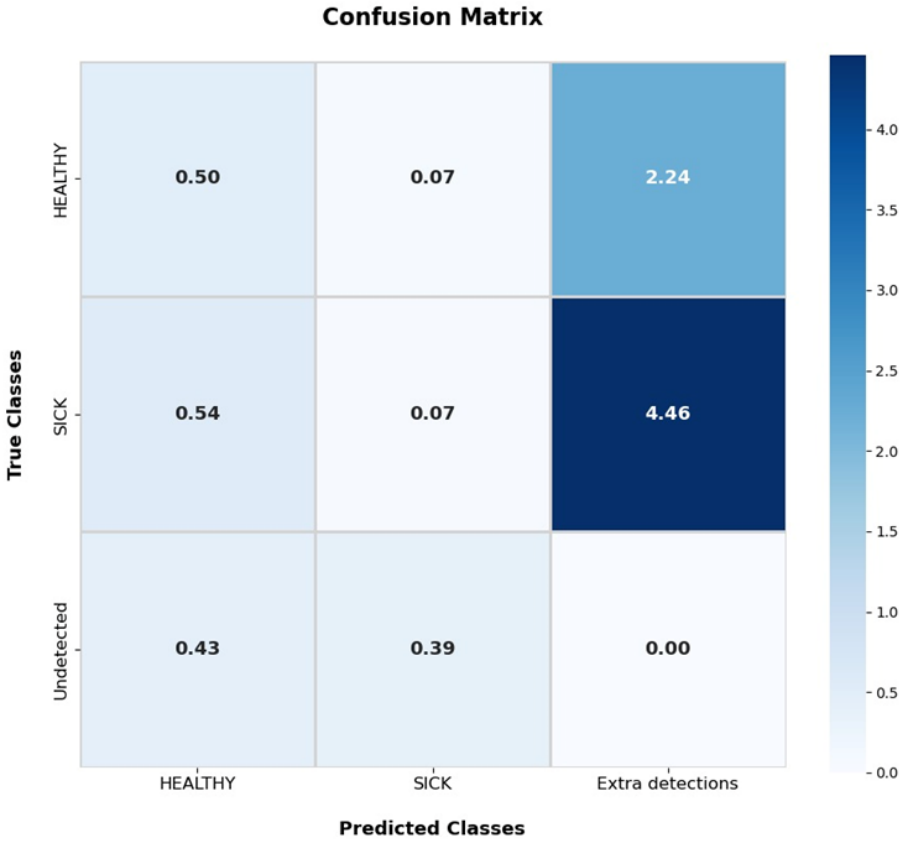


Figura 6.10: Matriz de confusión YOLO NAS

La matriz de confusión revela que, en términos generales, el modelo YOLO NAS no ha tenido el desempeño esperado y clasifica de peor manera que los modelos anteriores. Las celdas diagonales (por ejemplo, (1,1) y (2,2)) muestran los valores muy bajos, indicando que sólo el 50 % de las plantas sanas se clasifican como sanas y el 7 % las plantas enfermas se clasifican como enfermas.

Otro aspecto preocupante es el número considerable de plantas no detectadas, como se refleja en la fila ”undetected”. Específicamente, las celdas (1,3) y (2,3) tienen valores relativamente altos, lo que indica que un porcentaje significativo de plantas, tanto sanas como enfermas, no fueron detectadas por el modelo.

De igual modo, se evidencia una gran cantidad de detecciones extra, porque sugiere que el modelo está confundiendo objetos del entorno con plantas de banano, o que está detectando una misma planta múltiples veces, casi triplicando las detecciones por individuo.

La dificultad en la detección que se observa en el modelo YOLO NAS también puede atribuirse a la densidad de las plantaciones de banano, donde las plantas suelen estar agrupadas muy juntas. La estrecha proximidad entre las plantas dificulta la identificación precisa tanto para el modelo como para el ojo humano. Esta complejidad en la distribución de las plantas puede llevar a errores en la detección, lo que se refleja en el número de plantas no detectadas en la matriz de confusión.

Aunque el modelo YOLO NAS tiene la capacidad de distinguir entre plantas sanas y enfermas en general, la detección se ve obstaculizada por la falta de claridad en la separación entre las plantas individuales, un problema similar al observado en el modelo YOLO V8.

Para el modelo YOLO NAS, se obtuvieron las siguientes métricas de rendimiento (figura 6.11):

Class	Images	Instances	Box(P	R	mAP50	mAP50-95):
all	499	2886	0.663	0.559	0.582	0.317
HEALTHY	499	2499	0.712	0.695	0.68	0.378
SICK	499	387	0.614	0.424	0.485	0.255

Figura 6.11: Métricas de test YOLO NAS

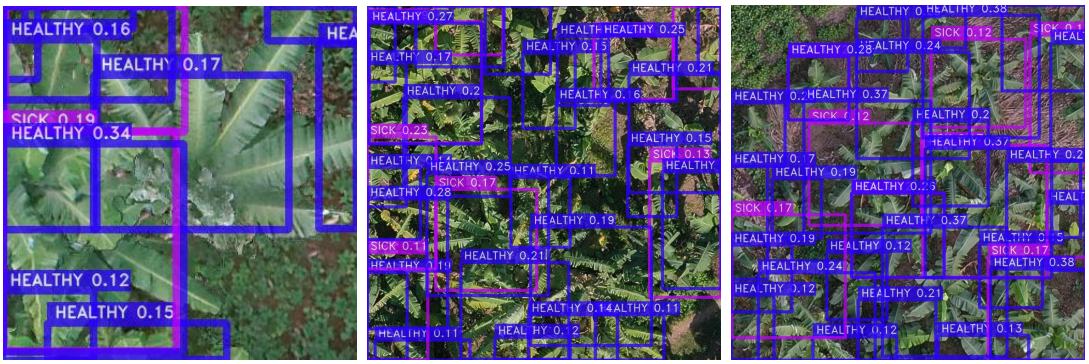
- **Precisión:** El modelo muestra una precisión de 0.712 para la clase ’healthy’ y de 0.614 para la clase ’sick’. Esto es un poco contradictorio si se observa la matriz de confusión, pero se podría explicar con la hipótesis planteada en donde se comenta

que las métricas de la matriz de confusión pueden deberse a que se detectan los mismos individuos más de una vez y con diferentes etiquetas, lo que podría ocasionar una confusión dentro de las métricas.

- **Recall:** El recall del modelo es de 0.695 para la clase 'healthy' y de 0.424 para la clase 'sick'. Esta métrica tiene un valor bastante bajo, mucho más que el de los modelos anteriores.
- **mAP (mean Average Precision):** El valor del mAP es 0.582. Un valor realmente malo, que nos indica que el modelo no funciona correctamente y que su desempeño no se acerca al desempeño esperado.

El modelo YOLO NAS muestra unas métricas bastante pobres, indicando que probablemente el desempeño del modelo no sea suficiente para evaluar de manera efectiva los cultivos de banano, por tanto, no sea recomendable usar este modelo para realizar la implementación del conteo de plantas y la detección en tiempo real.

Ahora bien, se realiza la selección de imágenes aleatorias dentro del dataset de test para observar cómo fueron las detecciones del modelo de manera visual(Figura 6.12):



(a) Predicción 1 (b) Predicción 2 (c) Predicción 3

Figura 6.12: Predicción YOLO NAS

Como resultado del análisis y observando las imágenes mostradas anteriormente, es posible identificar que cada una de las plantas tiene más de una detección, siendo entre 2 y 4 detecciones por planta. Adicionalmente, se observa que estas detecciones no son del todo acertadas, pues dentro de la misma planta se observan detecciones 'healthy' y 'sick'.

Se considera que los problemas de desempeño de este modelo pueden deberse a que YOLO NAS es un modelo de mucha menor latencia que los modelos anteriormente analizados, lo que fuerza al modelo a hacer detecciones más rápidas y con menor valor de confianza.

Según nuestra experiencia previa en la implementación de los modelos YOLO, concluimos que YOLO NAS funciona muy bien cuando se trata de diferenciar entre objetos totalmente distintos entre sí, más no cuando se trata de clasificar el mismo objeto dentro de diferentes categorías.

6.2.4 YOLO V9

En el transcurso del desarrollo de este proyecto de grado, se lanzó al público la arquitectura YOLO V9, por esta razón se decidió realizar en entrenamiento y análisis de resultados de esta arquitectura. Al igual que para los modelos YOLO analizados anteriormente, se obtienen las siguientes gráficas (Figura 6.15), las cuales muestran el desempeño en el entrenamiento para el modelo de YOLO V9:

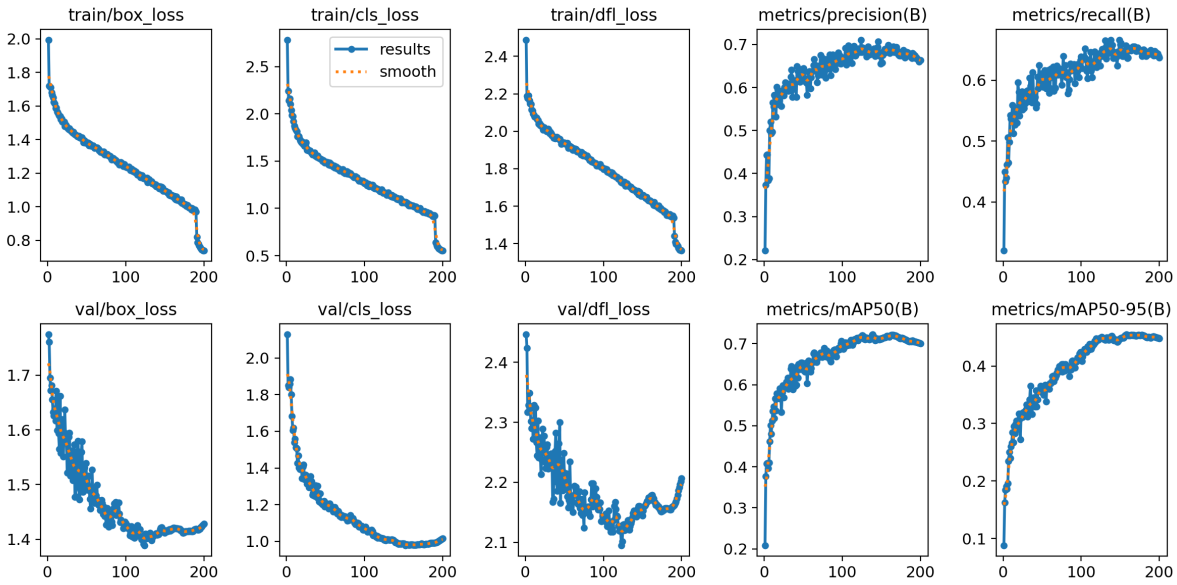


Figura 6.13: Gráfica de entrenamiento YOLO V9

Al analizar las gráficas de entrenamiento es posible observar que presenta un

comportamiento similar al modelo YOLO V8, con unas métricas aceptables en cada ámbito. En primer lugar, se observan funciones de pérdida bastante prometedoras, aunque con un comportamiento extraño en las últimas épocas en donde las pérdidas cambian drásticamente en el entrenamiento. Por otro lado, las funciones de métricas presentan el comportamiento esperando, teniendo una forma creciente que indica una mejora en las métricas a medida que aumenta el número de epocas.

Así como en los modelos YOLO anteriores, se realiza la validación con los datos del conjunto test (datos no vistos), y se obtiene la siguiente matriz de confusión (Figura 6.14):

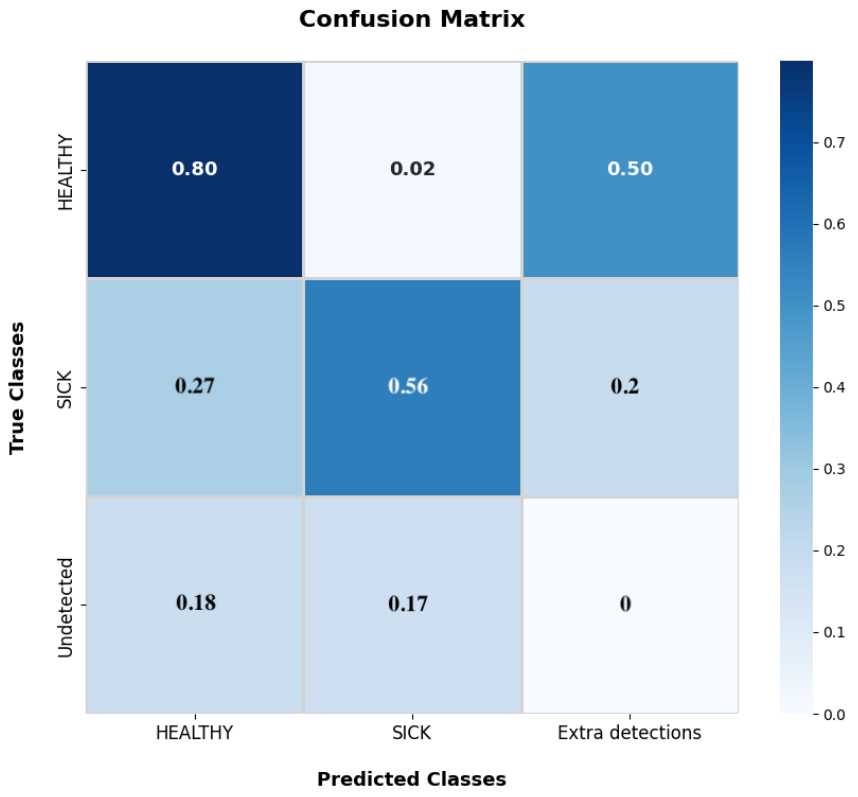


Figura 6.14: Matriz de confusión YOLO V9

Analizando la matriz de confusión se observan valores prometedores, en donde el comportamiento es bastante similar al de YOLO V8, vemos un gran desempeño al detectar plantas saludables, mientras que se observa un desempeño medio en la

detección de plantas enfermas. De igual modo, se observa que existen un gran número de detecciones extras cuando se trata de plantas saludables, caso similar al de YOLO V8 en donde las detecciones extras pueden deberse a individuos detectados más de una vez.

Para el modelo YOLO V9, se obtuvieron las siguientes métricas de rendimiento (figura 6.15):

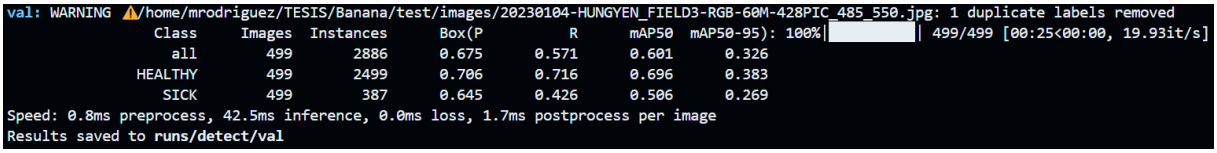


Figura 6.15: Métricas de test YOLO V9

- **Precisión:** El modelo muestra una precisión de 0.706 para la clase 'healthy' y de 0.645 para la clase 'sick'. Comparando estas métricas con el modelo YOLO V8 que es el que mejor desempeño ha mostrado hasta el momento, se ve que para la clase 'healthy' es mas alto, mientras que para la clase 'sick' es un poco más bajo.
- **Recall:** El recall del modelo es de 0.716 para la clase 'healthy' y de 0.426 para la clase 'sick'. Siendo inferiores cuando se compara con el modelo de YOLO V8.
- **mAP (mean Average Precision):** El valor del mAP es 0.601, siendo bastante más bajo que en el caso del modelo YOLO V8.

El modelo YOLO V9 muestra unas métricas aceptables, dentro de las que destaca su matriz de confusión. Se observa que en cuanto a métricas de entrenamiento y matriz de confusión, el modelo presenta un desempeño similar al de YOLO V8, pero cuando se comparan las métricas más específicas como el recall y el mAP, vemos que YOLO V8 continúa mostrando mejor desempeño en general.

Ahora bien, se realiza la selección de imágenes aleatorias dentro del dataset de test para observar cómo fueron las detecciones del modelo de manera visual(Figura 6.16):

(a) Predicción 1 (b) Predicción 2 (c) Predicción 3

Figura 6.16: Predicción YOLO V9

Haciendo el análisis de las imágenes seleccionadas para evaluar el desempeño del modelo, es posible concluir que en términos generales las predicciones están realizándose de manera correcta. Los individuos detectados e identificados como saludables son en su mayoría correctos, al igual que para el caso de las plantas enfermas. Por otro lado, se observa que en muchas ocasiones cuando se trata de plantas muy juntas entre sí, resulta bastante complejo para el modelo identificar todas las plantas de banano, tal como se ha observado en los modelos anteriores.

Un punto a destacar es el de la predicción 2, en donde se observa que la planta de banano se encuentra entre otras plantas de diferente especie, y el modelo logra detectar adecuadamente cual es la planta de banano y no confunde los puntos verdes de su alrededor con plantas de banano.

En general, se observa que el modelo de YOLO V9 tiene un desempeño aceptable dadas las condiciones en las que se encuentran los cultivos a evaluar, pues resulta complejo hasta para el ojo humano realizar la identificación de todas las plantas de banano dentro de las imágenes. También, se destaca el hecho de que este modelo tiene menor latencia que YOLO V8, lo que podría ser una limitante a la hora de analizar las imágenes y detectar en su totalidad todos lo individuos dentro de la misma.

6.2.5 Comparación de Modelos YOLO

Tal como se mencionó anteriormente, la implementación del modelo Faster RCNN fue completamente descartada, esto debido a que por sus altos tiempos de latencia no era posible realizar conteo y detección en tiempo real, que era la finalidad de este trabajo de grado. Por esta razón, el modelo basado en Tensorflow fue descartado y se optó por evaluar en conjunto las tres arquitecturas de YOLO con el fin de seleccionar la que mejor desempeño mostrara para continuar con ella en los pasos posteriores.

En primer lugar, se agruparon las gráficas de desempeño resultantes del entrenamiento de los modelos YOLO más relevantes, con el fin de comparar el desempeño de los modelos entre sí.

Se inicia dicha comparación con la gráfica de pérdidas de entrenamiento totales.

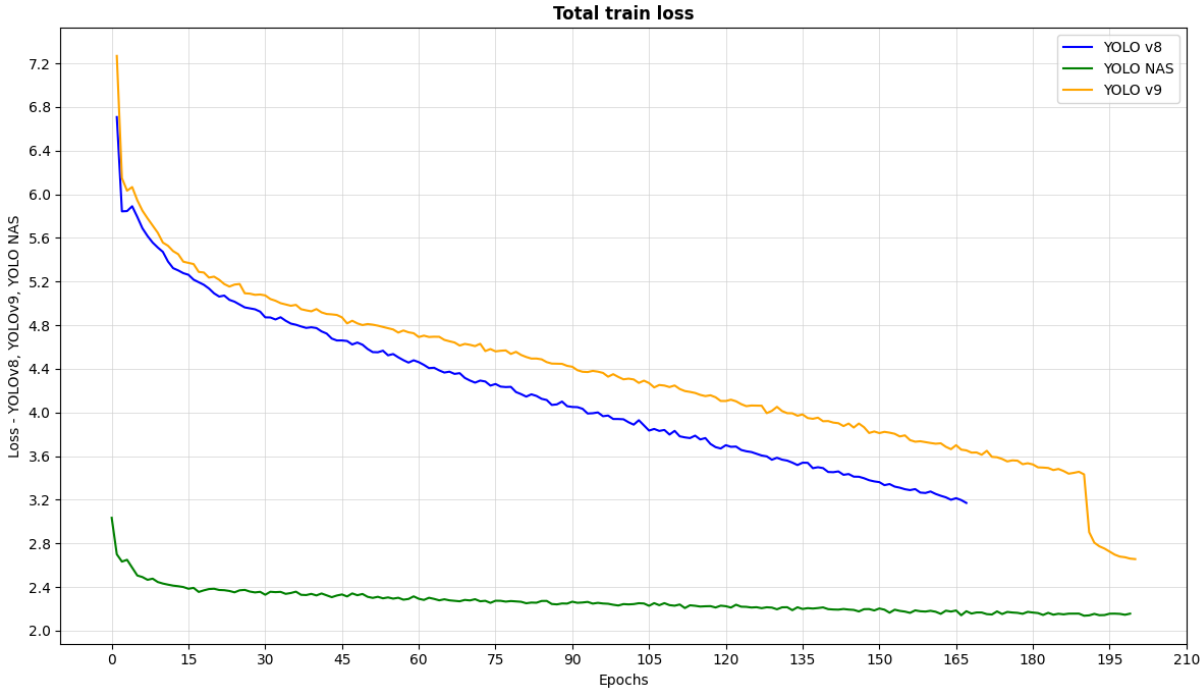


Figura 6.17: Pérdidas totales en entrenamiento

Analizando la gráfica (Figura 6.17), de pérdidas totales de entrenamiento, es posible observar que el modelo de YOLO NAS presenta unas menores pérdidas cuando se

compara con los modelos YOLO V8 y YOLO V9, los cuales presentan pérdidas similares entre sí. Por un lado, se observa que aunque YOLO NAS tiene pérdidas menores, su mejoría no es tan notoria con el pasar de las épocas, siendo muy similares desde la época 15 hasta la época 200. El hecho de que las pérdidas en YOLO NAS no bajen a un ritmo tan rápido como el que se esperaba demuestra que este modelo no tuvo una buena curva de aprendizaje, lo que podría afectar dentro del rendimiento del mismo y que podrá observarse más adelante.

Come se observa en la Figura 6.17, el comportamiento de pérdidas para los modelos YOLO V8 y YOLO V9 es similar, ambos modelos disminuyen sus pérdidas a un ritmo aceptable, indicando que el modelo está aprendiendo. Cabe aclarar que en el caso de YOLO V8 el modelo cuenta con una función que detiene su entrenamiento cuando identifica que el mAP no muestra una mejora significativa después de cierto número de épocas (Early Stopping) y es por ello por lo que el número de épocas mostradas en las gráficas para este modelo es considerablemente menor los demás.

Además, se observa que para el caso de YOLO V9 el modelo presenta un comportamiento anormal en su función de pérdidas de entrenamiento a partir de la época 190 aproximadamente, pues sus pérdidas decaen considerablemente en las últimas épocas.

Se continúa la comparación de los modelos con la gráficas de pérdidas de validación totales:

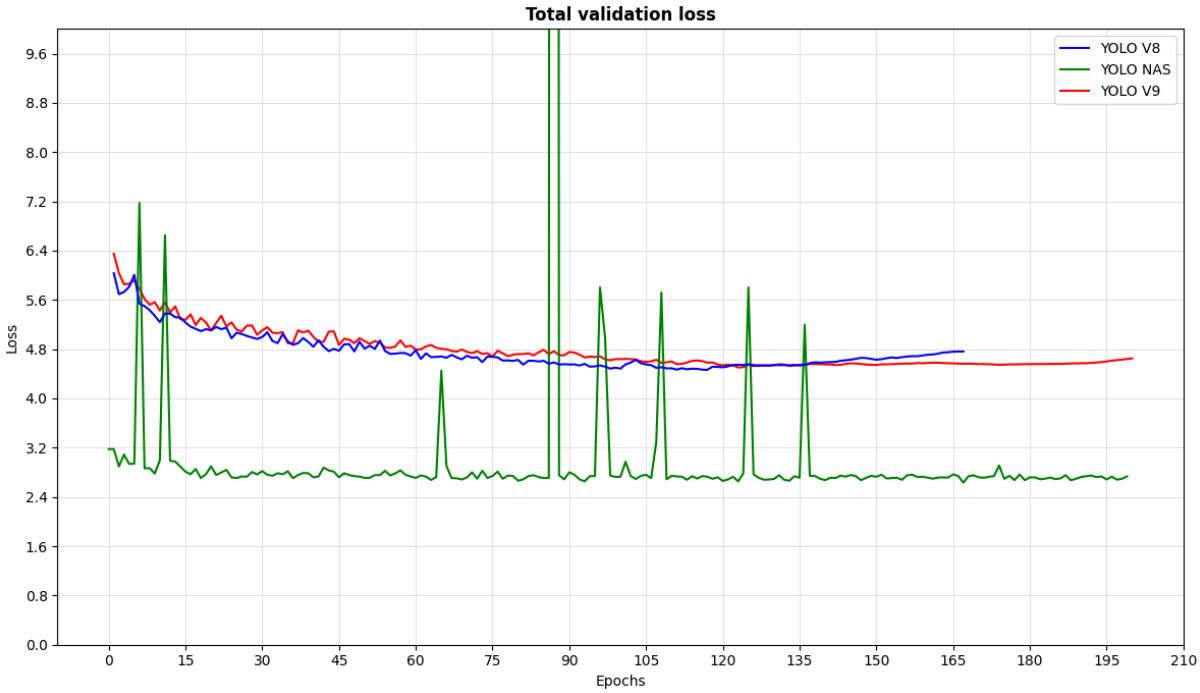


Figura 6.18: Pérdidas totales en validación

Analizando la gráfica (Figura 6.18), se observa que para el caso de la validación el YOLO V8 y YOLO V9 el comportamiento es similar, ambos valores de pérdidas son similares conforme avanzan las épocas y la mayor diferencia notable es el hecho de que YOLO V8 alcanza su punto más bajo en la época 100, siendo incluso menor que el punto más bajo del modelo YOLO V9 en todo el histórico de su entrenamiento. Se identifica que el comportamiento de las pérdidas para estos dos modelos es el esperado, pues disminuyen a medida que las épocas avanzan.

Por otro lado, para el modelo de YOLO NAS sucede algo inesperado, se observa que las pérdidas no mejoran como deberían, permaneciendo casi constantes desde el inicio. Por otro lado, se evidencian unos picos bastante altos dentro de la función de pérdidas, lo que podría significar que el modelo no aprende como debería y que el comportamiento del modelo no será el esperado.

Ahora bien, se analiza la gráfica de mAP (Figura 6.19), para evaluar el desempeño de los modelos:

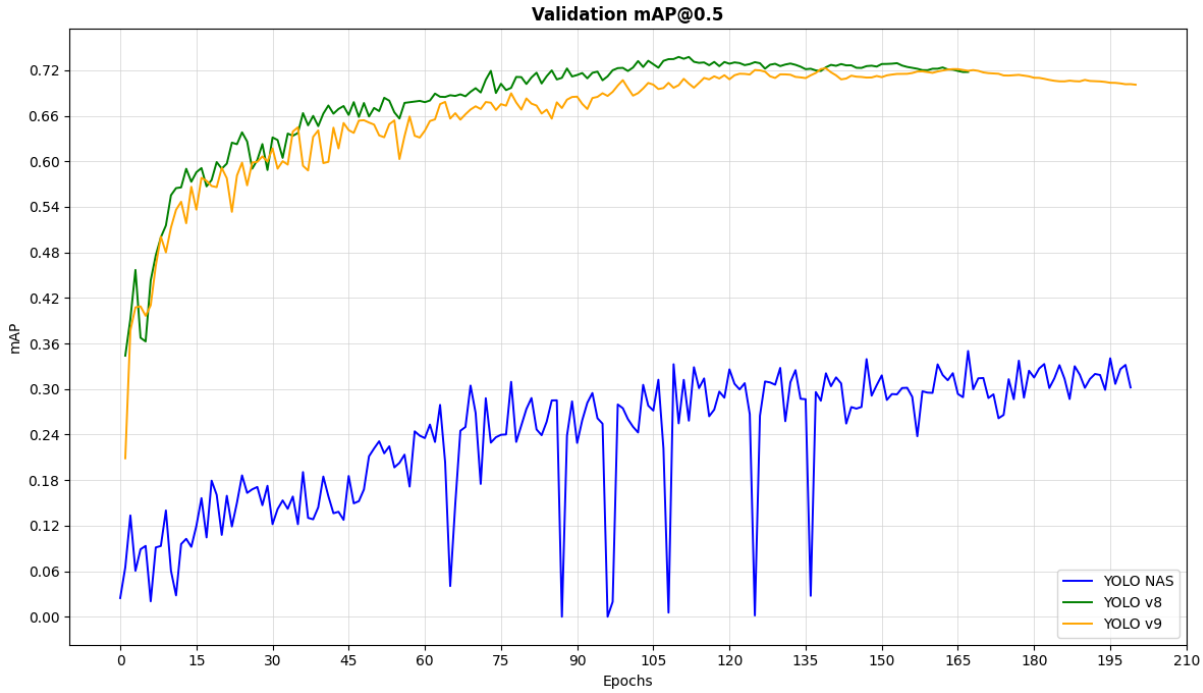


Figura 6.19: mAP en validación

Se observa que el comportamiento de los modelos en la gráfica de mAP es consecuente con lo observado en las dos gráficas anteriores, pues el desempeño para los modelos YOLO V8 y YOLO V9 es relativamente bueno, caso contrario al modelo de YOLO NAS, en donde se presentan valles en la gráfica que pueden indicar problemas con la validación dentro de este modelo.

Al igual que en la gráfica 6.18, se observa que YOLO V8 alcanza su punto máximo de mAP cerca a la época 110, y que aunque YOLO V9 presenta un comportamiento aceptable, no llega a alcanzar el desempeño de YOLO V8 en ningún punto de su entrenamiento.

Esta primera etapa de evaluación de métricas de entrenamiento da una visión bastante clara acerca del desempeño de los modelos. Si esta información se contrasta con los resultados analizados en las subsecciones anteriores, es posible observar que los modelos de YOLO V8 y YOLO V9 muestran un desempeño aceptable, mientras que el modelo YOLO NAS muestra un muy mal desempeño.

Con el fin de continuar con el análisis integral de las métricas y la comparación directa entre los modelos, se propone la siguiente tabla 6.1 comparativa en donde se muestra el desempeño por clase de cada modelo, al igual que el desempeño general del mismo. Cabe resaltar que las métricas mostradas a continuación corresponden a las obtenidas cuando se realizó la prueba de detección con los datos del subset 'test', es decir, datos no vistos.

Tabla 6.1: Comparación de modelos propuestos

Clase	Modelo	Métricas		
		Precisión	Recall	mAP 50
Healthy	YOLO V8	0.683	0.722	0.769
	YOLO NAS	0.712	0.695	0.68
	YOLO V9	0.706	0.716	0.696
Sick	YOLO V8	0.684	0.592	0.694
	YOLO NAS	0.614	0.424	0.485
	YOLO V9	0.645	0.426	0.506
General	YOLO V8	0.683	0.657	0.731
	YOLO NAS	0.663	0.559	0.582
	YOLO V9	0.675	0.571	0.601

En la tabla 6.1, es posible observar las métricas divididas por clase, al igual que el modelo con mejor desempeño, en cada métrica resaltado en color verde. Estas métricas se explicaron en de talle en las subsecciones anteriores, por esta razón en esta subsección simplemente se realiza un análisis rápido del modelo con el mejor desempeño en general.

Gracias a la tabla, es posible observar que el modelo con mejor desempeño en términos generales es YOLO V8, pues destaca en todas las métricas de cada una de las clases excepto en la precisión de la clase 'healthy'.

6.2.6 Selección del Mejor Modelo

Basándose en la evaluación conjunta de los resultados de entrenamiento y validación, así como en el análisis de las métricas de desempeño y capacidades de cada modelo, se llegó a la conclusión de que el modelo con el mejor desempeño es YOLO V8.

El modelo YOLO V8 cuenta con la capacidad de detección en video al igual que detección en tiempo real debido a su baja latencia. De igual modo, su matriz de confusión destacó entre los demás modelos mostrando una excelente capacidad en la detección de las plantas de banano y sus enfermedades. Adicionalmente, sus gráficas de entrenamiento muestran el comportamiento esperado y sus métricas de desempeño con imágenes no vistas demuestran una vez más que el modelo destaca sobre los demás.

Debe de tenerse en cuenta que el desempeño del modelo es óptimo y el esperado, pues dentro de la agricultura es complejo tener condiciones ideales y más en esta área de aplicación en donde encontramos plantaciones de banano poco industrializadas dentro de las cuales pueden observarse plantas bastante cercanas entre sí, distintos tipos de vegetación, vehículos, personas e incluso construcciones dentro de la misma plantación.

Se considera que YOLO V8 ha logrado afrontar estos retos satisfactoriamente, mostrando un desempeño excepcional en las pruebas realizadas y destacando siempre por su confiabilidad en las predicciones.

6.3 Evaluación y Desempeño de Modelos YOLO con Videos

6.3.1 Evaluación Inicial de Detección en Video

Para esta primera evaluación se seleccionaron tres videos con condiciones diferentes (altura, luminosidad, variedad del cultivo, resolución) para la evaluación visual del desempeño de los modelos de YOLO. El primer video fue tomado en Colombia, el segundo video fue tomado en Perú y el tercer video fue tomado en África .

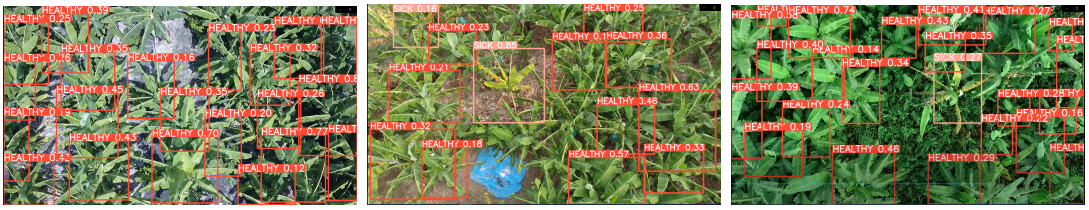
Cada uno de los videos fue procesado con los tres modelos de YOLO, esto con el fin de identificar si la conclusión obtenida en la evaluación de métricas en imágenes se aplicaba también para la detección en video y YOLO V8 seguía mostrando el mejor desempeño.

A continuación, se mostrarán fotogramas concretos de cada video, con el fin de poder explicar dentro del trabajo de grado como fue el comportamiento de los modelos bajo

las mismas condiciones de video.

6.3.1.1 YOLO V8

Incialmente, se muestran fotogramas concretos de cada video (Figura 6.20) para el modelo YOLO V8:



(a) Predicción 1 (b) Predicción 2 (c) Predicción 3

Figura 6.20: Predicción en video YOLO V8

Gracias a las detecciones analizadas, es posible observar que el modelo YOLO V8 cuenta con una buena capacidad para detectar las plantas en video, teniendo una buena identificación de los individuos en la mayoría de los casos. Por otro lado y como era de esperarse, existen casos concretos en los que el modelo no es capaz de detectar todas las plantas dentro del video, esto debido a que el cultivo no está estandarizado, no tiene un orden concreto y tiene zonas de mayor densidad de plantas donde se confunden unas con otras y resulta complejo identificarlas.

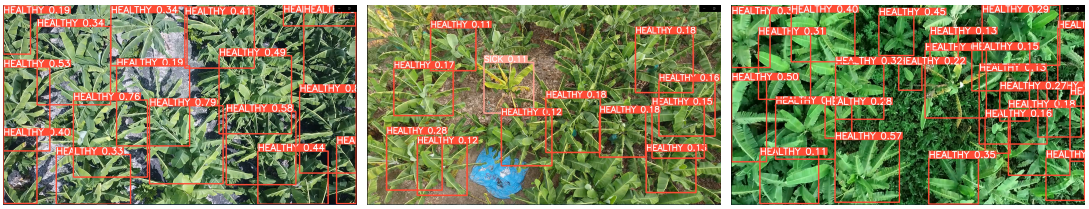
En cuanto a la clasificación de plantas entre 'healthy' y 'sick', el modelo muestra una buena capacidad para hacerlo, siendo mejor cuando se trata de plantas saludables, pero lográndolo bien cuando se muestran plantas enfermas en la mayoría de los casos.

6.3.1.2 YOLO NAS

Para el caso de YOLO NAS, el modelo no tuvo la capacidad de detectar en video haciendo uso de los mismos parámetros de IoU y confidence que se usaron en el caso de los demás modelos. Por esta razón y por su mal desempeño en las pruebas anteriores, se confirma que este modelo no será candidato para realizar la implementación de detección en tiempo real y conteo dentro de este trabajo de grado.

6.3.1.3 YOLO V9

Finalmente, se muestran fotogramas concretos de cada video (figura 6.21) para el modelo YOLO V9:



(a) Predicción 1 (b) Predicción 2 (c) Predicción 3

Figura 6.21: Predicción en video YOLO V9

Se observa un buen desempeño para el modelo YOLO V9 en los fotogramas seleccionados, detectando adecuadamente las plantas de banano sin confundirlas con su entorno y clasificando la mayoría de ellas en la categoría adecuada. Sin embargo, al compararlo con el modelo YOLO V8, se observa que para este modelo existe un mayor número de individuos que no fue posible detectar en video, tal vez debido a que este modelo tiene una latencia menor que YOLO V8 y por tanto toma un menor tiempo para procesar cada fotograma, corriendo del riesgo de ignorar ciertos detalles.

6.3.1.4 Resultado de Evaluación Inicial en Video

Con base en los resultados obtenidos en la evaluación de los modelos con imágenes y teniendo en cuenta el resultado dentro de la primera fase de evaluación en detección en video, se decide continuar con la implementación innovadora del modelo (detección en tiempo real y conteo de plantas), haciendo uso del modelo YOLO V8.

El modelo YOLO V8 ha demostrado desempeño y constancia destacables en las pruebas realizadas. Por esta razón, se decide continuar con dicho modelo para realizar una evaluación más exhaustiva en lo que concierne a las métricas de video.

6.3.2 Evaluación de Metricas en Video - YOLO V8

Con el fin de lograr obtener un valor cuantitativo en cuanto a métricas de video se refiere y teniendo en cuenta de que actualmente la detección en video no cuenta con unas métricas estándar para su evaluación, se decidió seguir la metodología descrita a continuación con el fin de evaluar el desempeño del modelo seleccionado (YOLO V8) en la detección en video:

1. **Selección de 2 videos para realizar la evaluación:** Se seleccionaron 2 videos con diferentes condiciones y de diferentes regiones (Colombia, Perú), con el fin de evaluar el desempeño del modelo cuando se enfrenta a cambios en las condiciones como la iluminación, el porcentaje de plantas sanas versus enfermas, la variedad del cultivo, la altura del dron y la calidad de la imagen con la que fue capturado el video.

2. **Extracción de un fotograma por segundo de cada video:** Se realizó la creación de un código capaz de extraer un fotograma por segundo de cada video, para posteriormente convertir ese fotograma en una imagen.

3. **Selección de fotogramas a evaluar:** Se seleccionaron los fotogramas más relevantes de cada video con el fin de tener la posibilidad de medir el desempeño del modelo en ciertos puntos del video.

4. **Anotación de los fotogramas seleccionados:** Se realizó la anotación de los fotogramas seleccionados con ayuda de los Investigadores Asociados expertos dentro de CIAT en la detección de las enfermedades de *Fusarium y Xanthomonas wilt* en cultivos de banano. Esto con el fin de tener un punto de comparación en cuanto a la detección realizada por el modelo.

5. **Obtención de métricas para los fotogramas seleccionados:** Se realiza la predicción y obtención de métricas de los fotogramas seleccionados con el fin de tener un punto comparativo dentro de la detección en video.

En primer lugar, se analiza un video capturado en Colombia, este video tiene únicamente plantas sanas, por tanto, sirve para evaluar qué tanto detecta en estos

fotogramas específicos a los individuos. Para este primer caso se obtuvieron las siguientes métricas (Figura 6.22):

Class	Images	Instances	Box(P	R	mAP50
all	6	139	0.714	0.468	0.594
HEALTHY	6	139	0.714	0.468	0.594

Figura 6.22: Métricas fotogramas video 1

Como era de esperarse, el video detecta únicamente plantas sanas, pues en esta primera evaluación no se contaba con plantas enfermas. En términos generales, las métricas de precisión, recall y mAP parecen ser óptimas. No obstante, se debe de tener en cuenta que un fotograma no es igual a una foto, ya que se extrajo de un video que corre a 60 fps, y el hecho de que el modelo detecte o no en estos fotogramas no quiere decir que en todo el video tenga el mismo comportamiento.

En segundo lugar, se analizó un video capturado en Perú, en donde sí hay presencia de plantas enfermas. Las métricas obtenidas fueron las siguientes (Figura 6.23):

Class	Images	Instances	Box(P	R	mAP50
all	6	138	0.413	0.292	0.342
HEALTHY	6	132	0.625	0.417	0.518
SICK	6	6	0.2	0.167	0.166

Figura 6.23: Métricas fotogramas video 2

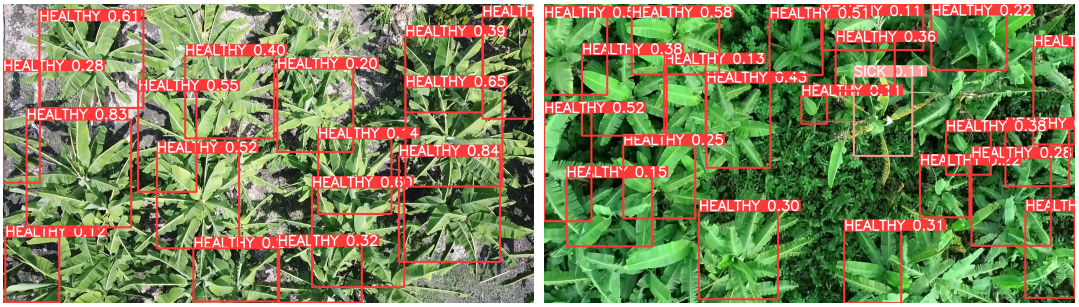
En esta segunda validación, se observa que aunque las métricas muestran una detección en las plantas enfermas, esta no es la esperada. El desempeño del modelo en estos fotogramas resulta ser menor al obtenido en la validación por medio de imágenes y en lo observado de manera visual al reproducir los videos.

Después de un análisis profundo, se considera que el fallo en el desempeño del modelo dentro de esta evaluación se da debido a que no es posible analizar la totalidad de los fotogramas del video, por lo tanto, no se puede recolectar la información necesaria para medir el desempeño real del modelo al analizar video.

Para cada uno de los videos se están analizando únicamente 6 de miles de fotogramas capturados. Se hace de esta manera debido a que para poder analizar los fotogramas en su totalidad deberían de anotarse completamente los miles de fotogramas, lo que requeriría de bastante tiempo adicional al igual que de una implementación rigurosa de un protocolo que identifique cuando cada individuo dentro del video fue detectado para no repetir detecciones.

Todo lo expuesto anteriormente, sumado al hecho de que los cultivos analizados son bastantes complejos, con diversos factores externos agregados que no se encontrarían en un ambiente uniforme y controlado, ocasionan que en métricas de fotogramas el desempeño del modelo no sea el esperado. Pero si se revisan las métricas en imágenes y los videos analizados, será posible observar que el modelo tiene un desempeño más que aceptable en cuanto a detección de plantas y clasificación de su salud se refiere.

Por último, se muestran algunas imágenes correspondientes a la detección de los fotogramas (Figura 6.24):



(a) Fotograma video 1 (b) Fotograma video 2

Figura 6.24: Predicción en fotogramas de video

6.4 Implementación Innovadora del Modelo

6.4.1 Conteo de Plantas

Para abordar el problema del seguimiento y conteo de plantas de banano enfermas y saludables, se utiliza la siguiente estrategia:

1. **Detección de Objetos:**

Se utiliza YOLO V8 para detectar plantas en cada frame del video. YOLO V8 proporciona las coordenadas de los bounding boxes y las clases de los objetos detectados.

2. **Seguimiento de Detecciones:**
   Se implementa un filtro de Kalman para predecir y actualizar las posiciones de los objetos detectados. Cada detección se asocia a un track utilizando el algoritmo de asignación húngaro para asegurar que cada planta se rastree correctamente a través de los frames consecutivos.

3. **Actualización y Conteo:**
   Los tracks se actualizan con cada nueva detección, y se mantienen contadores separados para plantas enfermas y saludables. El conteo se muestra en tiempo real en el video.

Esta combinación de técnicas garantiza un seguimiento preciso y un conteo confiable de las plantas, minimizando errores de detección múltiple de la misma planta en diferentes frames. En este proyecto, se usa para asociar de manera eficiente las detecciones actuales con los tracks existentes, minimizando la distancia euclidiana entre ellos.

```
from scipy.optimize import linear_sum_assignment

# Matriz de costos basada en la distancia euclidiana entre tracks y detecciones
cost_matrix = np.zeros((len(tracks), len(detections)))
for i, track in enumerate(tracks):
    for j, det in enumerate(detections):
        x1, y1, x2, y2 = det
        centroid = np.array([x1 + (x2 - x1) / 2, y1 + (y2 - y1) / 2])
        cost_matrix[i, j] = euclidean_distance(track['centroid'], centroid)

# Resolver el problema de asignaci n
row_indices, col_indices = linear_sum_assignment(cost_matrix)
```

Codigo Fuente 6.1: Algoritmo de Asignación Hungaro

La matriz de costos se construye calculando las distancias euclidianas entre los centroides de las detecciones y los centroides de los tracks existentes. El algoritmo de asignación húngaro encuentra la asignación óptima que minimiza el costo total.

Funciones Adicionales

- **Cálculo de la Distancia Euclidiana**

  En el código (Ver Anexo B), para el seguimiento y conteo de plantas, la distancia euclidiana se utiliza para construir la matriz de costos que el algoritmo de asignación húngaro necesita para asociar las detecciones entre frames consecutivos.

```
def euclidean_distance(point1, point2):
    return np.sqrt(np.sum((point1 - point2)  2))
```

Codigo Fuente 6.2: Calculo de Distancia Euclidiano

Esta sección de código (Código 6.2), calcula la distancia euclidiana entre dos puntos, utilizada para llenar la matriz de costos.

```
# Matriz de costos basada en la distancia euclidiana entre
   tracks y detecciones
cost_matrix = np.zeros((len(tracks), len(detections)))
for i, track in enumerate(tracks):
    for j, det in enumerate(detections):
        x1, y1, x2, y2 = det
        centroid = np.array([x1 + (x2 - x1) / 2, y1 + (y2 - y1)
   / 2])
        cost_matrix[i, j] = euclidean_distance(track['centroid'
   ], centroid)
```

Codigo Fuente 6.3: Crear Matrix de Costos

Esta función (Código 6.3), construye la matriz de costos utilizando la distancia euclidiana entre todas las posibles combinaciones de detecciones en dos frames consecutivos.

- **Drawing Results**

```python
def draw_results(frame, tracks):
    for track in tracks:
        x1, y1, x2, y2 = map(int, track['bbox'])  # Convertir las coordenadas a enteros
        cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 255, 0), 2)
        cv2.putText(frame, f"ID: {track['id']}", (x1, y1 - 5), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 2)

    cv2.imshow("Tracking", frame)
```

Codigo Fuente 6.4: Dibujar Resultados

Esta función dibuja los bounding boxes y los IDs en el frame, mostrando los resultados del seguimiento.

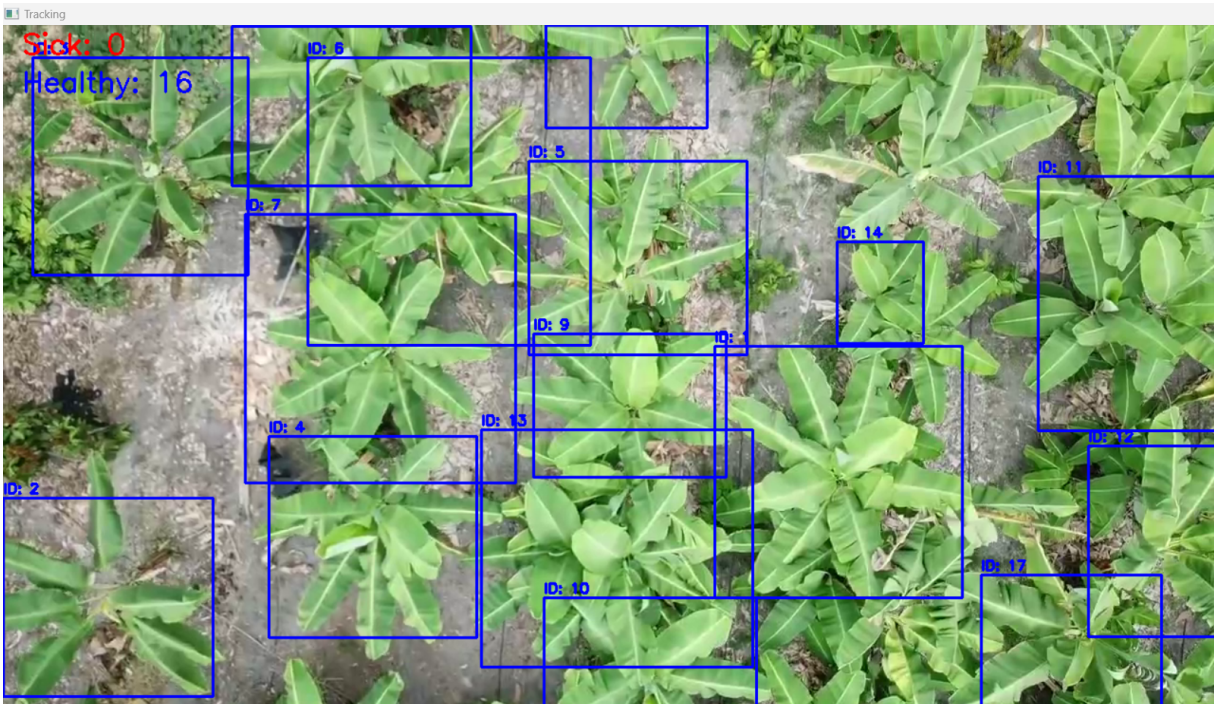


Figura 6.25: Conteo de plantas inicial

6.4.2 Detección y Conteo en Tiempo Real

La implementación del modelo de detección y seguimiento de enfermedades en plantas de banano ha demostrado ser una solución innovadora y eficiente. A través del uso de YOLO V8 para la detección de objetos, filtros de Kalman para el seguimiento de las detecciones y el algoritmo de asignación húngaro para la asociación entre frames, se logró un sistema robusto y con buena precisión. La combinación de estas técnicas permite un seguimiento continuo y un conteo de las plantas enfermas y saludables, minimizando los errores de detección múltiple y garantizando una estimación precisa del estado de los cultivos.

Una de las innovaciones clave de este proyecto fue la integración de un flujo de trabajo que utiliza drones equipados con cámaras de alta resolución para capturar imágenes en tiempo real, transmitidas a un computador portátil para su procesamiento.

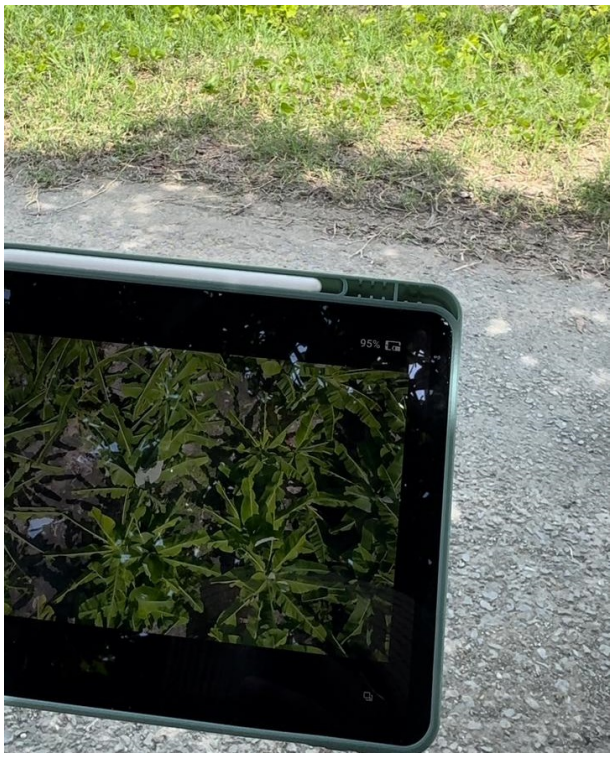


Figura 6.26: Prueba en tiempo real IPAD

La capacidad de los drones DJI para transmitir video en tiempo real a través de dispositivos como iPads (Figura 6.26), y la posterior transmisión a un computador mediante Airplay y aplicaciones como Airdroid Cast, demostró ser una solución práctica para la visualización y análisis en tiempo real.

Figura 6.27: Prueba Vuelo de Dron sobre cultivos

Este flujo de trabajo no solo facilita la detección oportuna de enfermedades, sino que también mejora la capacidad de los agricultores para tomar decisiones rápidas y basadas en datos precisos.

La implementación de la función `Q_discrete_white_noise` para generar la matriz de covarianza de ruido de proceso en el filtro de Kalman ha mejorado significativamente la robustez del sistema, permitiendo un seguimiento más estable. Además, la modularidad y escalabilidad del código desarrollado aseguran que la solución puede adaptarse a otros tipos de cultivos y enfermedades, lo que representa un valor añadido significativo para la agricultura de precisión.

Figura 6.28: Prueba de conteo en tiempo real

Esta implementación innovadora ha demostrado ser un avance notable en el monitoreo de cultivos, proporcionando una herramienta eficaz y adaptable para la detección y gestión de enfermedades en plantas, y contribuyendo al desarrollo de técnicas avanzadas para la agricultura sostenible y eficiente.

6.5 Comparación entre el Modelo Desarrollado con los Modelos Identificados en la Literatura Científica

Con el fin de comparar el modelo desarrollado y su implementación, se realiza una búsqueda en la literatura científica de modelos e implementaciones que se hayan realizado en el ámbito de la agricultura para la detección de enfermedades. Actualmente, no existen muchos documentos que describan la implementación de arquitecturas YOLO en la identificación de enfermedades de plantas de banano. Por esta razón, se decidió ampliar la búsqueda y evaluar distintos modelos, con distintas implementaciones de diferentes plantas y arquitecturas.

Adicionalmente, queremos resaltar que hasta la fecha de envío de este documento, no hemos encontrado ningún artículo científico que implemente la detección en tiempo real de enfermedades en cultivos haciendo uso de una señal de video. Por esta razón, consideramos que la implementación planteada es innovadora y va más allá de las implementaciones encontradas en los artículos científicos.

Ahora bien, realizará la comparación con 6 modelos encontrados en la literatura científica, yendo desde los más específicos a los más generales.

6.5.1 Modelo 1: AI-powered banana diseases and pest detection [1]

- Precisión: El modelo logró una precisión del 92 % en la detección de enfermedades en plátanos, demostrando ser eficaz en la identificación de enfermedades específicas.

- Eficiencia Computacional: Utiliza un enfoque basado en redes neuronales convolucionales (CNN), que requiere una GPU potente para el entrenamiento pero es eficiente en la inferencia, permitiendo su implementación en dispositivos de campo con

capacidad computacional adecuada.

- Capacidad de Adaptación: El modelo fue entrenado con un conjunto de datos limitado a plátanos, lo que limita su capacidad de adaptación a otras especies de cultivos sin realizar ajustes significativos en el entrenamiento y la arquitectura del modelo.

6.5.2 Modelo 2: Detection of banana plants and their major diseases through aerial images and machine learning methods [2]

- Precisión: Logra una precisión del 90 % en la identificación de plantas de plátano y sus principales enfermedades, demostrando ser un modelo robusto para este tipo de cultivo.

- Eficiencia Computacional: Utiliza técnicas avanzadas de procesamiento de imágenes y aprendizaje automático que son eficientes pero requieren hardware especializado para alcanzar resultados óptimos.

- Capacidad de Adaptación: Aunque es eficaz para plátanos, la adaptación a otros cultivos puede ser limitada sin modificaciones sustanciales en los datos de entrenamiento y los parámetros del modelo.

6.5.3 Modelo 3: Tea leaf disease detection and identifcation based on YOLOv7 (YOLO-T) [3]

- Precisión: Utilizando el modelo YOLO V7, la precisión en la detección de enfermedades en hojas de té es superior al 95 %, destacándose por su alta precisión en la identificación de múltiples clases de enfermedades.

- Eficiencia Computacional: YOLO V7 es extremadamente eficiente en la detección, haciendo uso óptimo de recursos computacionales y permitiendo su implementación en dispositivos de bajo costo.

- Capacidad de Adaptación: Aunque el modelo está optimizado para hojas de

té, su arquitectura permite la adaptación a otros cultivos con ajustes mínimos en el entrenamiento, lo que lo hace una opción flexible y escalable para diversas aplicaciones agrícolas.

6.5.4 Modelo 4: Image processing analysis of geospatial uav orthophotos for palm oil plantation monitoring [4]

- Precisión: La precisión varía según el método utilizado, pero generalmente alcanza alrededor del 85 % en la clasificación de áreas fértiles y no fértiles en plantaciones de palma aceitera.

- Eficiencia Computacional: El procesamiento de imágenes UAV requiere considerable poder computacional, especialmente para la segmentación y extracción de características utilizando algoritmos como la Matriz de Co-ocurrencia de Niveles de Gris (GLCM).

- Capacidad de Adaptación: El modelo es específico para el monitoreo de plantaciones de palma aceitera y podría necesitar ajustes significativos para adaptarse a otros tipos de cultivos o entornos de plantación.

6.5.5 Modelo 5: Deep learning models for plant disease detection and diagnosis [5]

- Precisión: Los modelos basados en CNN, como VGG y AlexNetOWTBn, alcanzan una precisión de hasta 99.53 % en la detección de enfermedades en diferentes plantas. Esta alta precisión se logra mediante el uso de grandes conjuntos de datos y arquitecturas profundas de redes neuronales.

- Eficiencia Computacional: Estos modelos son altamente eficientes durante la inferencia pero requieren un tiempo significativo de entrenamiento (hasta 5.5 días en una sola GPU), lo que puede ser una limitación para su implementación en entornos con recursos computacionales limitados.

- Capacidad de Adaptación: Estos modelos son altamente adaptables debido a su capacidad de generalizar en diferentes condiciones de cultivo, demostrando su eficacia tanto en entornos de laboratorio como en condiciones de campo. Sin embargo, la precisión puede disminuir si no se cuenta con datos de entrenamiento representativos de las condiciones reales de cultivo.

6.5.6 Modelo 6: Fruits and vegetables quality evaluation using computer vision [6]

- Precisión: Varía según la técnica aplicada, pero generalmente es alta en la evaluación de la calidad de frutas y verduras, proporcionando resultados confiables en la clasificación y evaluación de calidad.

- Eficiencia Computacional: Las técnicas utilizadas son eficientes pero pueden requerir optimización para diferentes tipos de productos agrícolas, especialmente cuando se trata de procesar grandes volúmenes de datos en tiempo real.

- Capacidad de Adaptación: El modelo es versátil y puede adaptarse a diferentes tipos de frutas y verduras, aunque requiere ajustes específicos para cada tipo de producto, lo que puede implicar un esfuerzo significativo en la recolección de datos y el ajuste de parámetros.

6.5.7 Desempeño del Modelo YOLO V8 Desarrollado en Este Proyecto de Grado

Precisión:

- La matriz de confusión (Figura 6.6) revela que, en términos generales, el modelo identifica correctamente las plantas sanas como tales en un 79 % de los casos y las plantas enfermas como enfermas en un 54 % de los casos.

- La precisión del modelo es de 0.683 para la clase 'healthy' y 0.684 para la clase 'sick'.

- El recall es de 0.722 para la clase 'healthy' y 0.592 para la clase 'sick'.

- El valor del mAP (mean Average Precision) general es 0.731.

Eficiencia Computacional:

Lo que distingue a nuestro modelo es su capacidad de detección en tiempo real. Utilizando una arquitectura optimizada de YOLO V8 y técnicas avanzadas de procesamiento paralelo, el modelo puede analizar imágenes y detectar enfermedades en fracciones de segundo. Esto permite su implementación en drones para monitoreo continuo y dinámico en campo. La baja latencia del modelo permite hacer detección en videos donde el dron vaya a una velocidad relativamente rápida, e incluso en tiempo real, lo que es necesario para la toma de decisiones rápidas y efectivas en el manejo de cultivos.

Capacidad de Adaptación:

El modelo desarrollado en este proyecto no solo es capaz de detectar enfermedades con alta precisión, sino que también puede realizar el conteo de plantas. Esta capacidad adicional es importante para aplicaciones en agricultura de precisión, donde el monitoreo y la gestión de los cultivos a nivel individual pueden mejorar significativamente los rendimientos y la salud general de las plantaciones. La validación con imágenes aleatorias demostró que el modelo funciona mejor cuando las imágenes muestran a las plantas más cerca, lo que sugiere que para la detección en tiempo real será importante que el dron vuele a una altura relativamente baja.

Innovación:

La integración de detección en tiempo real y conteo de plantas en un solo modelo representa un avance significativo en la tecnología agrícola. Esta dualidad de funciones permite a los agricultores no solo identificar rápidamente problemas de enfermedades, sino también gestionar eficientemente la densidad y distribución de sus cultivos. Esto no solo mejora la capacidad de respuesta ante enfermedades, sino que también optimiza el uso de recursos y la planificación de cultivos. La capacidad de implementar este modelo en dispositivos móviles y drones agrega un nivel de flexibilidad y accesibilidad que es

indispensable para su adopción en diversos entornos agrícolas.

6.5.8 Análisis Final:

El modelo desarrollado en este proyecto se compara favorablemente con los modelos identificados en la literatura científica. Su precisión, eficiencia computacional y capacidad de adaptación lo posicionan como una contribución relevante en el campo de la detección de enfermedades en cultivos. Con un enfoque basado en YOLO V8 y la inclusión de imágenes capturadas en condiciones reales de cultivo, el modelo demuestra una capacidad robusta para la detección y diagnóstico de enfermedades en diversas especies de plantas. Su capacidad de detección en tiempo real y conteo de plantas lo hace una herramienta valiosa tanto para agrónomos como para agricultores, destacándose como una innovación significativa en la agricultura de precisión.

Cabe resaltar que de los 6 modelos mencionados solo uno de ellos realizaba la detección con imágenes de drones mientras los otros lo hacían con imágenes capturadas con smartphones, la dificultad adicional de nuestra implementación radica en la poca estandarización de los campos de banano, su entorno y lo juntas que se encontraban algunas de sus plantas.

7. Conclusiones y Trabajos Futuros

7.1 Conclusiones

El desarrollo de una solución efectiva para la detección de enfermedades en plantas de banano utilizando drones y técnicas avanzadas de visión por computadora representa un avance significativo en la agricultura de precisión. A lo largo de este trabajo de grado, se han abordado varios desafíos técnicos y se ha logrado implementar una solución innovadora para mejorar el proceso de monitoreo de cultivos. A continuación, se presentan las conclusiones principales derivadas de este trabajo de grado:

- **Eficiencia en la Detección de Enfermedades:**
  La implementación de múltiples modelos avanzados de detección de objetos, como Faster R-CNN con ResNet-50, YOLO V8, YOLO NAS y YOLO V9, permitió evaluar y comparar sus rendimientos en la detección de enfermedades en plantas de banano. Tras varias pruebas y análisis de desempeño, YOLO V8 fue la opción más eficiente y precisa. Esta arquitectura de red neuronal convolucional, proporciona detecciones rápidas y precisas, superando a los otros modelos en términos de precisión y velocidad de procesamiento. La capacidad de YOLO V8 para identificar plantas enfermas y saludables en tiempo real es crucial para tomar decisiones oportunas y efectivas en el manejo de cultivos, mejorando así la productividad y la salud general de los campos de banano.

- **Precisión en el Seguimiento de Detecciones:**
  La combinación del filtro de Kalman con el algoritmo de asignación húngaro resultó en un seguimiento preciso de las detecciones a lo largo de los frames consecutivos. El filtro de Kalman permite estimaciones actualizadas de las posiciones de las plantas, mientras que el algoritmo húngaro asegura una asignación óptima de las detecciones. Esto minimiza en cierta medida el riesgo de contar la misma planta más de una vez, mejorando considerablemente la precisión del sistema.

- **Robustez en la Matriz de Covarianza:**
  La utilización de la función `Q_discrete_white_noise` para generar la matriz de covarianza de ruido de proceso en el filtro de Kalman contribuyó significativamente a la robustez del sistema. Esta matriz modela la incertidumbre en el sistema, ayudando a suavizar las predicciones y proporcionando una mayor estabilidad en el seguimiento de objetos.

- **Integración y Visualización:**
  La integración de todas las técnicas mencionadas permitió no solo detectar y seguir las plantas de banano, sino también visualizar el conteo de plantas enfermas y saludables en tiempo real en el video capturado por los drones. Esta visualización en tiempo real es esencial para los agricultores, ya que les proporciona información inmediata sobre el estado de sus cultivos, facilitando la toma de decisiones rápidas y bien informadas.

- **Escalabilidad y Aplicabilidad**
  La solución desarrollada demostró ser escalable y aplicable a otros tipos de cultivos y enfermedades. La modularidad del código y la flexibilidad de las técnicas utilizadas permiten adaptar el sistema a diferentes escenarios agrícolas, lo cual representa un valor añadido significativo para la agricultura de precisión y la gestión sostenible de los cultivos.

7.2 Trabajos Futuros

Con base a los resultados obtenidos y las limitaciones identificadas durante el desarrollo del trabajo de grado, se proponen los siguientes puntos a tratar en trabajos futuros con el fin de mejorar y expandir las capacidades del sistema desarrollado.

7.2.1 Ampliación del Dataset de Entrenamiento:

Se plantea la posibilidad de incluir imágenes con un mayor número de especies de plantas de banano, al igual que incluir más imágenes de la categoría 'sick'con el fin de balancear el dataset, aumentando la robustez del modelo. De igual modo, se sugiere capturar imágenes bajo distintas condiciones de iluminación, clima, región y distribución de plantas con el fin de aumentar la adaptabilidad del modelo.

7.2.2 Optimización del Modelo:

Se deja abierta la posibilidad de explorar nuevas y mejores arquitecturas de redes neuronales tales como YOLO V10 que fue lanzada hace un par de semanas, esto con el fin de aumentar la precisión y la eficiencia computacional del modelo. Además, de ser posible se propone optimizar los modelos para reducir su latencia dentro de las predicciones en tiempo real para de esta manera facilitar las detecciones.

7.2.3 Implementación en Dispositivos Móviles y Drones:

Desplegar aplicaciones móviles y software para drones que tengan la capacidad de implementar los modelos de detección en tiempo real, con el fin de que el uso de dichos modelos sea más generalizado y práctico estando al alcance de cualquier agricultor y agrónomo que lo requiera. Se busca realizar una evaluación más profunda en campo bajo diferentes condiciones con el fin de evaluar el desempeño del modelo y las aplicaciones desarrolladas.

7.2.4 Mejora del Conteo de Plantas:

Desarrollar e implementar algoritmos de segmentación avanzados tales como Segment Anything Model (SAM), que puedan mejorar la precisión en el conteo de plantas, esto serviría especialmente para las plantas que se encuentran muy juntas entre sí. Esto acompañado de una valiadación del modelo de segmentación que garantice su correcto funcionamiento en diferentes contextos agrícolas.

7.2.5 Integración con Sistemas de Gestión Agrícola:

Integrar el sistema de detección de enfermedades y conteo de plantas con plataformas de gestión agrícola existentes, facilitando la toma de decisiones y el manejo integral de cultivos. De esta manera se tendrá la capacidad de desarrollar herramientas analíticas que permitan a los agricultores interpretar los datos recolectados y tomar decisiones informadas sobre el manejo de sus cultivos dentro de dichos sistemas.

7.2.6 Investigación en Técnicas de Transferencia de Aprendizaje:

Explorar técnicas de transferencia de aprendizaje para adaptar el modelo entrenado en un conjunto de datos a nuevos cultivos y enfermedades con datos limitados, reduciendo así el costo y tiempo de recolección de datos.

7.2.7 Evaluación de Impacto Económico y Ambiental:

Realizar estudios para evaluar el impacto económico y ambiental del uso de la tecnología desarrollada en la agricultura, midiendo su efectividad en la reducción de enfermedades y mejora de rendimientos.

Referencias

[1] M. G. Selvaraj, A. Vergara, H. Ruiz, N. Safari, S. Elayabalan, W. Ocimati, and G. Blomme, "Ai-powered banana diseases and pest detection," *Plant Methods*, vol. 15, p. 92, 08/12 2019.

[2] M. Gomez Selvaraj, A. Vergara, F. Montenegro, H. Alonso Ruiz, N. Safari, D. Raymaekers, W. Ocimati, J. Ntamwira, L. Tits, A. B. Omondi, and G. Blomme, "Detection of banana plants and their major diseases through aerial images and machine learning methods: A case study in dr congo and republic of benin," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 169, pp. 110–124, 2020.

[3] M. Soeb, M. Jubayer, T. Tarin, M. Al Mamun, F. Ruhad, A. Parven, N. Mubarak, S. Karri, and I. Meftaul, "Tea leaf disease detection and identification based on yolov7 (yolo-t)," *Scientific reports*, vol. 13, Dec. 2023. Publisher Copyright: © 2023, The Author(s).

[4] F. Fahmi, D. Trianda, U. Andayani, and B. Siregar, "Image processing analysis of geospatial uav orthophotos for palm oil plantation monitoring," in *Journal of Physics: Conference Series*, vol. 978, p. 012064, IOP Publishing, 2018.

[5] K. P. Ferentinos, "Deep learning models for plant disease detection and diagnosis," *Computers and Electronics in Agriculture*, vol. 145, pp. 311–318, 2018.

[6] "Fruits and vegetables quality evaluation using computer vision: A review," *Journal of King Saud University - Computer and Information Sciences*, vol. 33, no. 3, pp. 243–257, 2021.

[7] fao, "Banana market review and banana statistics 2012–2013," *Market and Policy Analyses of Raw Materials, Horticulture and Tropical (RAMHOT) Products Team. Rome*, 2014.

[8] S. De Buck and R. Swennen, "Bananas, the green gold of the south," 2016.

[9] N. Y. Moore, S. Bentley, K. G. Pegg, and D. R. Jones, "Fusarium wilt of banana," *Musa Disease Fact Sheet N°5, INIBAP*, 1995.

[10] C. Hillnhuetter and A.-K. Mahlein, "Early detection and localisation of sugar beet diseases: new approaches," *Gesunde Pflanzen*, vol. 60, pp. 143–149, 2008.

[11] S. Ji, C. Zhang, A. Xu, Y. Shi, and Y. Duan, "3d convolutional neural networks for crop classification with multi-temporal remote sensing images," *Remote Sensing*, vol. 10, no. 1, p. 75, 2018.

[12] K. Neupane and F. Baysal-Gurel, "Automatic identification and monitoring of plant diseases using unmanned aerial vehicles: A review," *Remote Sensing*, vol. 13, no. 19, p. 3841, 2021.

[13] M. G. Selvaraj, M. Valderrama, D. Guzman, M. Valencia, H. Ruiz, and A. Acharjee, "Machine learning for high-throughput field phenotyping and image processing provides insight into the association of above and below-ground traits in cassava (manihot esculenta crantz)," *Plant methods*, vol. 16, no. 1, pp. 1–19, 2020.

[14] "FAOSTAT: Food and Agriculture Organization Corporate Statistical Database." `https://www.fao.org/faostat/en/#data/QCL/visualize`, 2023. Accessed: June 9, 2023.

[15] J. Boulent, M. Beaulieu, P.-L. St-Charles, J. Théau, and S. Foucher, "Deep learning for in-field image-based grapevine downy mildew identification," in *Precision agriculture'19*, p. 8689, Wageningen Academic Publishers, 2019.

[16] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[17] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[18] A. Calderon and H. D. H. Cortes, "Machine learning en la detección de enfermedades en plantas: Machine learning en la detección de enfermedades en plantas," *Tecnología Investigación y Academia*, vol. 7, no. 2, pp. 55–61, 2019.

[19] F. Osisanwo, J. Akinsola, O. Awodele, J. Hinmikaiye, O. Olakanmi, J. Akinjobi, *et al.*, "Supervised machine learning algorithms: classification and comparison," *International Journal of Computer Trends and Technology (IJCTT)*, vol. 48, no. 3, pp. 128–138, 2017.

[20] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: A brief review," *Computational Intelligence and Neuroscience*, vol. 2018, p. 7068349, 2018.

[21] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[22] C. M. Bishop, "Neural networks and their applications," *Neural Computing Research Group, Department of Computer Science and Applied Mathematics*, vol. Accepted for publication, March 1994. Received 16 August 1993.

[23] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: Analysis, applications, and prospects," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 12, pp. 6999–7019, 2022.

[24] R. Szeliski, *Computer vision: algorithms and applications*. Springer Nature, 2022.

[25] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the 27th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '14)*, (Columbus, Ohio, USA), pp. 580–587, June 2014.

[26] B. N. K. Sai and T. Sasikala, "Object detection and count of objects in image using tensor flow object detection api," in *2019 International Conference on Smart Systems and Inventive Technology (ICSSIT)*, pp. 542–546, 2019.

[27] B. Rekha, A. Marium, G. Srinivasan, and S. A. Shetty, "Literature survey on object detection using yolo," *International Research Journal of Engineering and Technology (IRJET)*, vol. 7, no. 06, 2020.

[28] Deci-AI, "Deci-AI/super-gradients: Easily train or fine-tune Sota Computer Vision models with one open source training library. the home of Yolo-Nas.." `https://github.com/Deci-AI/super-gradients`, 2023. Accessed Sep. 15, 2023.

[29] R. Padilla, S. L. Netto, and E. A. B. da Silva, "A survey on performance metrics for object-detection algorithms," in *2020 International Conference on Systems, Signals and Image Processing (IWSSIP)*, pp. 237–242, 2020.

[30] G. E. R. Ibarra, "Agricultura de precisión: La integración de las tic en la producción agrícola," *Computer and Electronic Sciences: Theory and Applications*, vol. 3, no. 1, pp. 34–38, 2022.

[31] M. d. J. Marcial-Pablo, A. Gonzalez-Sanchez, S. I. Jimenez-Jimenez, R. E. Ontiveros-Capurata, and W. Ojeda-Bustamante, "Estimation of vegetation fraction using rgb and multispectral images from uav," *International journal of remote sensing*, vol. 40, no. 2, pp. 420–438, 2019.

[32] P. van Blyenburgh, "Uavs: an overview," *Air Space Europe*, vol. 1, no. 5, pp. 43–47, 1999.

[33] H. Eisenbeiß, "Uav photogrammetry," 01 2009.

[34] I. Colomina, M. Blázquez, P. Molina, M. Parés Calaf, and M. Wis, "Towards a new paradigm for high-resolution low-cost photogrammetry and remote sensing," vol. 37, 01 2008.

[35] J. McCarthy, "What is artificial intelligence?," *Computer Science Department, Stanford University*, November 2007.

[36] S. G. Salcedo, *Artificial Intelligence in Digital Agriculture. Towards In-Field Grapevine Monitoring using Non-invasive Sensors.* PhD thesis, Universidad de La Rioja, 2019.

[37] P. P. Jayaraman, D. Palmer, A. Zaslavsky, A. Salehi, and D. Georgakopoulos, "Addressing information processing needs of digital agriculture with openiot platform," in *Interoperability and Open-Source Solutions for the Internet of Things: International Workshop, FP7 OpenIoT Project, Held in Conjunction with SoftCOM 2014, Split, Croatia, September 18, 2014, Invited Papers*, pp. 137–152, Springer, 2015.

[38] R. H. Stover, "Fusarial wilt (panama disease) of bananas and other musa species," *Commonwealth Mycological Institute*, p. 117, 1962.

[39] R. C. Ploetz, "Variability in fusarium oxysporum f.sp. cubense," *Canadian Journal of Botany*, vol. 68, pp. 1357–1363, 1990.

[40] R. Ploetz, S. Freeman, J. Konkol, A. Al-Abed, Z. Naser, K. Shalan, and et al., "Tropical race 4 of panama disease in the middle east," *Phytoparasitica*, vol. 43, pp. 283–293, 2015.

[41] E. W. Brandes, "Banana wilt," *Phytopathology*, vol. 9, pp. 339–389, 1919.

[42] H. Su, S. Hwang, and W. Ko, "Fusarium wilt of cavendish bananas in taiwan," *Plant Disease*, vol. 70, pp. 814–818, 1986.

[43] D. Yirgou and J. F. Bradbury, "A note on wilt of banana caused by the enset wilt organism xanthomonas musacearum," *East African Agricultural and Forestry Journal*, vol. 40, pp. 111–114, 1974.

[44] W. Tushemereirwe, A. Kangire, F. Ssekiwoko, L. C. Offord, J. Crozier, E. Boa, M. Rutherford, and J. J. Smith, "First report of xanthomonas campestris pv. musacearum on banana in uganda," *Plant Pathology*, vol. 53, p. 802, 2004.

[45] V. Ndungo, S. Eden-Green, G. Blomme, J. Crozier, and J. Smith, "Presence of banana xanthomonas wilt (xanthomonas campestris pv. musacearum) in the democratic republic of congo (drc)," *New Disease Reports*, vol. 11, p. 18, 2005.

[46] S. R. B. Mgenzi, D. Muchunguzi, T. Mutagwaba, F. Mkondo, and R. Mohamed, "An outbreak of banana bacterial wilt in muleba district, kagera region, tanzania," *Disease report, Maruku Agriculture Research and Development Institute Tanzania*, 2006.

[47] B. A. Carter, R. Reeder, S. R. Mgenzi, Z. M. Kinyua, J. N. Mbaka, K. Doyle, V. Nakato, M. Mwangi, F. Beed, V. Aritua, M. L. Lewis Ivey, S. A. Miller, and J. J. Smith, "Identification of xanthomonas vasicola (formerly x. campestris pv. musacearum), causative organism of banana xanthomonas wilt, in tanzania, kenya and burundi," *Plant Pathology*, vol. 59, p. 403, 2010.

[48] M. Biruma, M. Pillay, L. Tripathi, G. Blomme, S. Abele, M. Mwangi, R. Bandyopadhyay, P. Muchunguzi, S. Kassim, S. Nyine, L. Turyagenda, and S. Eden-Green, "Banana xanthomonas wilt: a review of the disease, management strategies and future research directions," *African Journal of Biotechnology*, vol. 6, pp. 953–962, 2007.

[49] W. Tinzaara, E. B. Karamura, G. Blomme, W. Jogo, W. Ocimati, A. Rietveld, J. Kubiriba, and F. Opio, "Why sustainable management of xanthomonas wilt of banana in east and central africa has been elusive," vol. 986, pp. 157–164, 2013.

[50] D. Yirgou and J. F. Bradbury, "Bacterial wilt of enset (ensete ventricosum) incited by xanthomonas musacearum sp. n.," *Phytopathology*, vol. 58, pp. 111–112, 1968.

[51] Vp-Koen, "Yolov9: El primer modelo inducido por transformador," Mar 2024.

[52] DJI, "Dji official," 2023.

[53] T. Lescot, "World plantain and banana production systems," 2013.

[54] W. H. Organization *et al.*, "Fruit and vegetables for health: report of the joint fao," 2005.

[55] P. W. Crous, X. Mourichon, *et al.*, "Mycosphaerella eumusae and its anamorph pseudocercospora eumusae spp. nov.: causal agent of eumusae leaf spot disease of banana," *SYDOWIA-HORN-*, vol. 54, no. 1, pp. 23–34, 2002.

[56] A. Surridge, A. Viljoen, R. Crous, and F. Wehner, "Identification of the pathogen associated with sigatoka disease of banana in south africa," *Australasian plant pathology*, vol. 32, pp. 27–31, 2003.

[57] S.-H. Lin, S.-L. Huang, Q.-Q. Li, C.-J. Hu, G. Fu, L.-P. Qin, Y.-F. Ma, L. Xie, Z.-L. Cen, and W.-H. Yan, "Characterization of exserohilum rostratum, a new causal agent of banana leaf spot disease in china," *Australasian Plant Pathology*, vol. 40, pp. 246–259, 2011.

[58] M. Hernandez-Restrepo, J. Z. Groenewald, and P. W. Crous, "Neocordana gen. nov., the causal organism of cordana leaf spot on banana," 2015.

[59] S. Huang, B. Yan, J. Wei, W. Yan, Z. Cen, and T. Yang, "First report of plantain zonate leaf spot caused by pestalotiopsis menezesiana in china," *Australasian plant disease notes*, vol. 2, no. 1, pp. 61–62, 2007.

[60] E. B. Villaseñor *et al.*, "Redes neuronales para la toma de decisiones en el sector agrícola análisis exploratorio.," 2018.

[61] M. Kulbacki, J. Segen, W. Knieć, R. Klempous, K. Kluwak, J. Nikodem, J. Kulbacka, and A. Serester, "Survey of drones for agriculture automation from planting to harvest," in *2018 IEEE 22nd International Conference on Intelligent Engineering Systems (INES)*, pp. 000353–000358, 2018.

[62] L. Ale, A. Sheta, L. Li, Y. Wang, and N. Zhang, "Deep learning based plant disease detection for smart agriculture," in *2019 IEEE Globecom Workshops (GC Wkshps)*, pp. 1–6, IEEE, 2019.

[63] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the 27th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '14)*, (Columbus, Ohio, USA), pp. 580–587, June 2014.

[64] C. W. Wardlaw, *Banana diseases, including plantains and abaca.* Longmans, Green & Co., Ltd., 1961.

[65] Y. Pushpavathi, S. N. Dash, N. Madhavi, and D. Deepika, "Biological control of fusarium wilt disease in banana with emphasis on trichoderma spp. and pseudomonas spp.," *Plant Archives*, vol. 16, pp. 51–59, 2016.

[66] Empresa Brasileira de Pesquisa Agropecuária (Embrapa) Mandioca e Fruticultura, Ministério da Agricultura, Pecuária e Abastecimento, Embrapa Mandioca e Fruticultura, Leandro de Souza Rocha, Zilton José Maciel Cordeiro, Harllen Sandro Alves Silva, Fernando Haddad, and Francisco Ferraz Laranjeira, "Subsídios para plano de contingência da banana xanthomonas wilt – bxw," DOCUMENTOS 242, Empresa Brasileira de Pesquisa Agropecuária (Embrapa) Mandioca e Fruticultura, Cruz das Almas, BA, 2020.

[67] V7 Labs, "Neural network architectures guide." `https://www.v7labs.com/blog/neural-network-architectures-guide`, Month Year. Accessed on Day Month Year.

[68] M. Merenda, C. Porcaro, and D. Iero, "Edge machine learning for ai-enabled iot devices: A review," *Sensors*, vol. 20, no. 9, 2020.

[69] A. Morales Carreño, "lmplementación y metodología de calibración de cámaras multiespectrales para aplicaciones en agricultura de precisión," Master's thesis, 2019.

Apéndice A

Códigos preprocesamiento de dataset, entrenamiento y evaluacion de modelos

En este anexo se encuentra la documentación relacionada con el procesamiento del dataset, entrenamiento del modelo, pruebas de detección principal de enfermedades en plantas y evaluación de los modelos:

1. **Preprocesamiento del Dataset:**
   https://javerianacaliedu-my.sharepoint.com/:f:/g/personal/michael_javerianacali_edu_co/Eo6cHJiBZQlPp_GU3v6T6ScBEW3BWrvz_Z0yj8oZCkzE0w?e=gHEqHF
2. **Entrenamientos de Modelos:**
   https://javerianacaliedu-my.sharepoint.com/:f:/g/personal/michael_javerianacali_edu_co/Ek6rqBvRx8pLs_oS9MxHPA8BmThWMnC-PeP_UV3JEljNug?e=2fI0WH
3. **Evaluación de los Modelos:**
   - **Evaluación de Imágenes:**
     https://javerianacaliedu-my.sharepoint.com/:f:/g/personal/michael_javerianacali_edu_co/EgC_IcUGQflNsYhSV8ALJmQB9naA2ornUbOYUouuuk_RyQ?e=hw3uvH
   - **Evaluación de Videos:**
     https://javerianacaliedu-my.sharepoint.com/:f:/g/personal/michael_javerianacali_edu_co/EjfwW7tt_yJOvfa6SzoB2oUBA8kdcRcKxgt7c44P0RXXUw?e=AgFVRd
4. **Detección en Tiempo Real**
   https://javerianacaliedu-my.sharepoint.com/:f:/g/personal/michael_javerianacali_edu_co/EqgegGuw1o5GvrHFhDGvIgUBCikv9IOGRTcJfSNPVZMgBA?e=S3RZ0m

5. **Comparación de Arquitecturas:**
   https://javerianacaliedu-my.sharepoint.com/:f:
   /g/personal/michael_javerianacali_edu_co/
   EqgegGuw1o5GvrHFhDGvIgUBCikv9IOGRTcJfSNPVZMgBA?e=S3RZ0m

Apéndice B

Detección y Conteo de Plantas

```
import cv2
import numpy as np
from filterpy.kalman import KalmanFilter
from filterpy.common import Q_discrete_white_noise
from ultralytics import YOLO
from scipy.optimize import linear_sum_assignment

# Inicializar la c mara
video_capture = cv2.VideoCapture(0)

# Verificar si la c mara est  abierta
if not video_capture.isOpened():
    print("Error: No se puede abrir la c mara")
    exit()

dt = 1. / video_capture.get(cv2.CAP_PROP_FPS)

model = YOLO(MODELO)
model.model_name

# Funci n para inicializar el filtro Kalman
def create_kalman_filter(dt):
    kf = KalmanFilter(dim_x=4, dim_z=2)
    kf.x = np.array([0., 0., 0., 0.])   # initial state: [x, y, dx/
   dt, dy/dt]

    # state transition matrix
    kf.F = np.array([[1., 0., dt, 0.],
                     [0., 1., 0., dt],
                     [0., 0., 1., 0.],
                     [0., 0., 0., 1.]])

    # measurement function
    kf.H = np.array([[1., 0., 0., 0.],
                     [0., 1., 0., 0.]])

```

```
    # measurement noise covariance
    kf.R = np.eye(2) * 0.1

    # process noise covariance
    kf.Q = Q_discrete_white_noise(dim=4, dt=dt, var=0.01)

    # initial covariance matrix
    kf.P = np.eye(4)

    return kf

# Funci n para predecir y actualizar el filtro Kalman
def update_kalman_filter(kf, measurement):
    kf.predict()
    kf.update(measurement)
    return kf.x[:2]

# Funci n para calcular la distancia euclidiana entre dos puntos
def euclidean_distance(point1, point2):
    return np.sqrt(np.sum((point1 - point2) ** 2))

# Funci n para dibujar los resultados
def draw_results(frame, tracks, count_sick, count_healthy):
    for track in tracks:
        x1, y1, x2, y2 = map(int, track['bbox'])  # Convertir las
    coordenadas a enteros

        # Determinar el color seg n el estado de la planta
        if track['status'] == 'sick':
            color = (0, 0, 255)  # Rojo para plantas enfermas
            count_sick += 1
        else:
            color = (255, 0, 0)  # Azul para plantas saludables
            count_healthy += 1

        cv2.rectangle(frame, (x1, y1), (x2, y2), color, 2)
        cv2.putText(frame, f"ID: {track['id']}", (x1, y1 - 5), cv2.
    FONT_HERSHEY_SIMPLEX, 0.5, color, 2)

    # Mostrar el conteo en la esquina superior izquierda
    cv2.putText(frame, f"Sick: {count_sick}", (20, 30), cv2.
    FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)
```

```
75      cv2.putText(frame, f"Healthy: {count_healthy}", (20, 70), cv2.
    FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0), 2)
76
77      cv2.imshow("Tracking", frame)
78
79  # Crear el filtro Kalman
80  kf = create_kalman_filter(dt)
81
82  # Lista para almacenar las detecciones
83  detections = []
84
85  # Diccionario para almacenar los tracks
86  tracks = []
87  next_id = 1
88
89  # Umbral de distancia para la asociaci n de detecciones
90  distance_threshold = 50
91
92  # Contadores para el conteo de plantas
93  count_sick = 0
94  count_healthy = 0
95
96  # Definir el codec y crear el objeto VideoWriter
97  fourcc = cv2.VideoWriter_fourcc(*'XVID')  # Codec para el archivo de
      video
98  out = cv2.VideoWriter('output_video.avi', fourcc, 20.0, (int(
    video_capture.get(3)), int(video_capture.get(4))))
99
100 #Conteo y Deteccion
101 while True:
102     ret, frame = video_capture.read()
103     if not ret:
104         break
105
106     # Obtener las detecciones del modelo
107     results = model(frame, stream=True)
108
109     # Convertir el generador en una lista de detecciones
110     detections = []
111     for result in results:
112         detections.extend(result.boxes.xyxy.tolist())  # Aseg rate
    de que este m todo extraiga las coordenadas correctas
```

```
113
114     if detections:
115         # Matriz de costos basada en la distancia euclidiana entre
    tracks y detecciones
116         cost_matrix = np.zeros((len(tracks), len(detections)))
117         for i, track in enumerate(tracks):
118             for j, det in enumerate(detections):
119                 x1, y1, x2, y2 = det
120                 centroid = np.array([x1 + (x2 - x1) / 2, y1 + (y2 -
    y1) / 2])
121                 cost_matrix[i, j] = euclidean_distance(track['
    centroid'], centroid)
122
123         # Resolver el problema de asignaci n
124         row_indices, col_indices = linear_sum_assignment(cost_matrix
    )
125
126         # Actualizar y rastrear las detecciones
127         tracked_objects = []
128         for row, col in zip(row_indices, col_indices):
129             if cost_matrix[row, col] < distance_threshold:
130                 x1, y1, x2, y2 = detections[col]
131                 measurement = np.array([x1 + (x2 - x1) / 2, y1 + (y2
     - y1) / 2])  # centroid as measurement
132
133                 # Aplicar el filtro Kalman
134                 pred = update_kalman_filter(tracks[row]['kf'],
    measurement)
135
136                 # Actualizar el track existente
137                 tracks[row]['bbox'] = (x1, y1, x2, y2)
138                 tracks[row]['centroid'] = pred
139                 tracks[row]['disappeared'] = 0
140                 tracked_objects.append(tracks[row])
141
142         # A adir nuevas detecciones no asignadas como nuevos tracks
143         assigned_cols = set(col_indices)
144         for j, det in enumerate(detections):
145             if j not in assigned_cols:
146                 x1, y1, x2, y2 = det
147                 centroid = np.array([x1 + (x2 - x1) / 2, y1 + (y2 -
    y1) / 2])
```

```
                new_kf = create_kalman_filter(dt)
                new_kf.x[:2] = centroid
                # Suponiendo que las detecciones tengan una clase para distinguir 'sick' y 'healthy'
                status = 'sick' if result.boxes.cls[j] == 1 else 'healthy'
                tracks.append({
                    'bbox': (x1, y1, x2, y2),
                    'centroid': centroid,
                    'kf': new_kf,
                    'id': next_id,
                    'disappeared': 0,
                    'status': status
                })
                next_id += 1

        # Incrementar el contador desaparecido para los tracks no asignados
        for track in tracks:
            if track not in tracked_objects:
                track['disappeared'] += 1

        # Eliminar tracks que han desaparecido por demasiado tiempo
        tracks = [track for track in tracks if track['disappeared'] <= 3]

    # Dibujar los resultados
    draw_results(frame, tracks, count_sick, count_healthy)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

video_capture.release()
cv2.destroyAllWindows()
```

Codigo Fuente B.1: Codigo Principal