.UBAfiuba
FACULTAD DE INGENIERÍA

Sistema de monitoreo de *Rhynchophorus Ferrugineus* en palmeras de la ciudad de Montevideo

Ing. Bruno Masoller

Carrera de Especialización en Inteligencia Artificial

Director: Ing. Juan Ignacio Cavalieri (FIUBA)

Jurados:

Jurado 1 (pertenencia)
Jurado 2 (pertenencia)
Jurado 3 (pertenencia)

Ciudad de Montevideo, junio de 2025

Resumen

Esta memoria describe la implementación de una plataforma para la intendencia de Montevideo que utiliza visión por computadora para detectar la plaga del picudo rojo en las palmeras esta ciudad. El sistema se desarrolló con el objetivo de reducir costos operativos y mejorar la toma de decisiones para el servicio de arbolado de la institución.

Para el desarrollo del trabajo fueron necesarios conocimientos de visión por computadora, análisis de datos y aprendizaje profundo, así como de despliegue y operaciones de modelos de aprendizaje de máquinas e infraestructura de soporte.

Agradecimientos

Esta sección es para agradecimientos personales y es totalmente **OPCIONAL**.

Índice general

Índice de figuras

Índice de tablas

Dedicado a... [OPCIONAL]

Capítulo 1

Introducción general

En este capítulo se presenta una introducción general al trabajo realizado. Se describe el problema que el *Rhynchophorus Ferrugineus* (picudo rojo) presenta en las palmeras de la ciudad de Montevideo, el estado del arte en cuanto a trabajos similares y los objetivos planteados por la Intendencia de Montevideo (IM) y por el equipo de trabajo.

1.1. Descripción del problema

El picudo rojo, que puede observarse en la figura 1.1, es un insecto que afecta a las palmeras, especialmente a la *Phoenix Canariensis*, que es la especie más común en Montevideo. Este insecto ha causado un daño significativo en la flora de la ciudad, lo que ha llevado a la IM a enfocarse en su control y erradicación.

FIGURA 1.1. Picudo rojo [1].

Este escarabajo supone una amenaza grave para las palmeras infectadas ya que las larvas de este insecto se alimentan del tejido interno de la planta, causando su colapso estructural en un período de meses (dado el corto corto ciclo de vida del picudo rojo [1]), como puede verse en la figura 1.2a. Esta amenaza aumenta según la época del año, puesto que el insecto tiene diferentes tasas de dispersión y reproducción dependiendo de la temperatura y la humedad. Existen varios tipos de picudos [2], algunos autóctonos y otros introducidos.

La plaga del picudo rojo llegó a Uruguay en 2022 [3], y se esparció rápidamente por la ciudad de Montevideo. De las 25 000 palmeras que forman una parte

[1] Imagen tomada de https://es.wikipedia.org/wiki/Rhynchophorus_ferrugineus

esencial de la ciudad, muchas ya han sucumbido a la plaga, donde se estima que para el año 2030 el ecosistema se verá ampliamente afectado [4] si no se toman medidas de control adecuadas. Últimamente también se ha esparcido por el interior del país, como puede observarse en la figura 1.2b, perteneciente a la ruta 5 de Uruguay, donde se han encontrado palmeras muertas por la plaga.

(A) Palmera infectada por el picudo rojo. (B) Palmeras muertas en la ruta 5 de Uruguay.

FIGURA 1.2. Infección y muerte de palmeras por el picudo rojo.

Entre los métodos de control fitosanitarios que se utilizan para combatir la plaga se encuentran la endoterapia [5], el baño, la cirugía [6] y la remoción. Sin embargo, estos métodos son costosos y requieren de un monitoreo constante para detectar la presencia del picudo rojo. Para la IM no es solamente una cuestión ecológica sino también económica. En este sentido, el servicio de arbolado realiza un seguimiento de las palmeras afectadas por la plaga. Para ello, se registra su ubicación y estado de salud mediante campañas de detecciones a pie. Este proceso manual requiere de mucho tiempo y recursos humanos, por lo que resulta imprescindible un sistema automatizado que permita detectar la presencia de la plaga en lugares específicos de Montevideo.

Uno de los recursos aún no explotados por la IM para el monitoreo de la plaga es el uso de imágenes aéreas obtenidas mediante drones, disponibles por el servicio de geomática de ésta institución. Estos ortomosaicos, que pueden observarse en la figura 1.3, permiten obtener información detallada sobre el estado de las palmeras y su entorno.

FIGURA 1.3. Ejemplo de ortomosaico [2].

El análisis de estas imágenes es un proceso complejo que requiere de técnicas avanzadas de procesamiento de imágenes y aprendizaje automático, así como también herramientas que brinden soporte a estas actividades.

1.2. Estado del arte

En los últimos años, el campo de la visión por computadora (VPC) ha experimentado un crecimiento exponencial, impulsado principalmente por los avances en el aprendizaje profundo y las redes neuronales convolucionales (CNN, por su sigla en inglés de *Convolutional Neural Networks*). Estas tecnologías han permitido el desarrollo de soluciones innovadoras para tareas de detección, clasificación y segmentación en imágenes, lo que ha abierto nuevas posibilidades para aplicaciones en diversos ámbitos, tales como la monitorización ambiental y la agricultura de precisión. La detección de plagas mediante imágenes aéreas, en particular la del picudo rojo, ha cobrado relevancia debido al creciente impacto que este insecto tiene en la salud de las palmeras urbanas, principalmente en países del Mediterráneo y en el Medio Oriente [2]. En esta sección se revisan las principales técnicas y enfoques que han marcado el estado del arte en la detección de plagas a partir de imágenes aéreas. Se aborda tanto arquitecturas tradicionales como las más recientes, donde se destacan sus fortalezas, limitaciones y la evolución de sus aplicaciones en escenarios reales.

1.2.1. Redes convolucionales

Visión por computadora se define como el proceso en donde se extrae, analiza y comprende información significativa a partir de imágenes o secuencias de imágenes [7]. Este proceso abarca una amplia gama de tareas, tales como la detección de objetos, la segmentación de imágenes, el reconocimiento de patrones y la clasificación de imágenes. Dentro del ámbito de la inteligencia artificial (IA), la VPC se reconoce como un subcampo destinado a dotar a las máquinas de la capacidad de interpretar y comprender la información visual del entorno de manera análoga a

[2] Imagen tomada del sistema de información geográfica de la IM (SIG): https://sig.montevideo.gub.uy/

la percepción humana [8]. Las CNN constituyen una clase de modelos de aprendizaje profundo diseñados para el procesamiento y análisis de datos visuales. Tendencias recientes en el ámbito de las CNN evidencian avances significativos en el diseño de sus arquitecturas y en el procesamiento de datos, orientados a la solución de problemáticas emergentes. En particular, se ha puesto énfasis en el tratamiento de imágenes de alta resolución capturadas mediante vehículos aéreos no tripulados (UAV, por su sigla en inglés de *Unmanned Aerial Vehicles*), lo que representa un área de aplicación de creciente interés académico y privado [9] [10].

1.2.2. Detección del picudo rojo en imágenes aéreas

La relevancia de la detección de la plaga del picudo rojo a nivel global ha propiciado el surgimiento de diversas líneas de investigación. En este sentido, se han desarrollado enfoques basados en VPC a partir de imágenes aéreas, haciendo especial uso de arquitecturas de CNN tales como YOLO [11].

Entre los estudios analizados, la implementación de la detección mediante imágenes aéreas de forma exclusiva representa una metodología relativamente novedosa. En este contexto, en el estudio *Automatic large scale detection of red palm weevil infestation using street view images* [12], se emplea un enfoque combinado. Inicialmente, se localizan las palmeras a través de una CNN (Faster R-CNN [13]) aplicada a imágenes aéreas. Posteriormente, utilizando la misma arquitectura, se extrae la corona de la palmera a partir de imágenes capturadas en *street view*, lo que permite clasificar a la planta en función de la presencia o ausencia de infección. Un resumen de este trabajo se presenta en la figura 1.4.

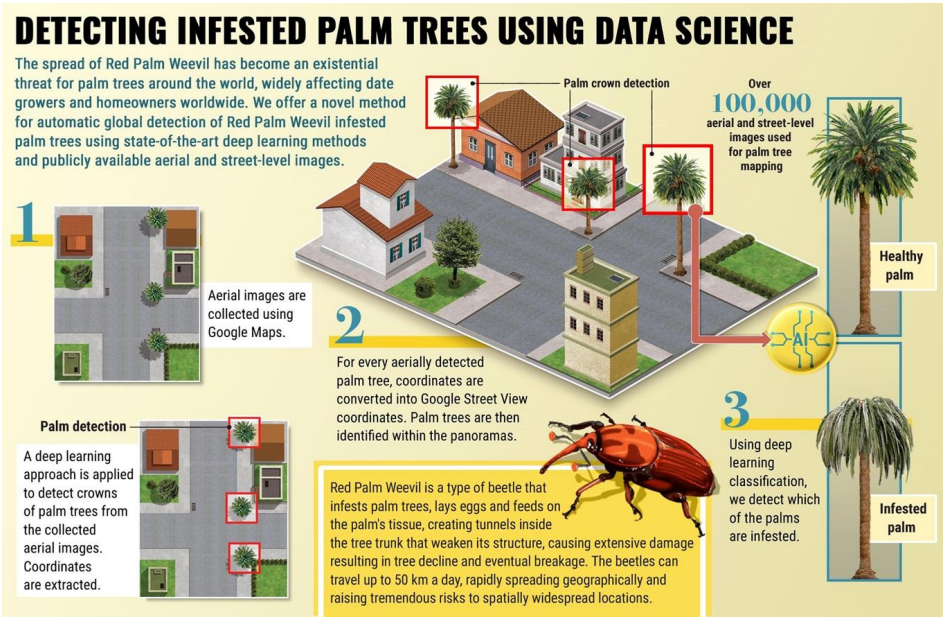

FIGURA 1.4. Proceso de detección realizado por Kagan et al [3].

Por otro lado, la detección de palmeras sí es ampliamente estudiada en la literatura, siendo un tema recurrente en la investigación de VPC. En este sentido, se han desarrollado diversos estudios. En *Implementation of slicing aided hyper inference*

[3]Imagen obtenida de: http://arxiv.org/abs/1506.01497

(SAHI) in YOLOv8 to counting oul palm trees using high-resolution aereal imagery data [14] se presenta un enfoque que utiliza la arquitectura YOLOv8 para detectar palmeras en imágenes aéreas de alta resolución. Además, utiliza *Slicing Aided Hyper Inference* para mejorar la precisión de la detección. Este enfoque se basa en la idea de dividir las imágenes en regiones más pequeñas y realizar inferencias en cada una de ellas, lo que permite una detección más precisa de los objetos de interés, algo esencial en el caso de objetos pequeños como las palmeras. Para el estudio, se utilizaron imágenes RGB capturadas con un dron Trinity F90+ a 200 metros de altura y una implementación de segmentación (SAHI) de 3000×3000 px. Si bien el estudio se centra en la detección de palmeras de aceite, su metodología puede ser adaptada para la detección del picudo rojo. La tabla 1.1 resume los métodos utilizados en la detección de palmeras en diferentes estudios anteriores.

TABLA 1.1. Comparación entre diferentes estudios de detección de palmeras [4].

Autor y año	Modelo	Métricas
Mukhles Sir Monea et al, 2022 [15]	YOLOv3	5.76 % (MAPE)
Hery Wibowo et al, 2022 [16]	YOLOv3, YOLOv4, YOLOv5m	97.28 % (v3), 97.74 % (v4), 94.94 % (v5m). (F1-Score)
Adel Ammar et al, 2022 [17]	Faster R-CNN	94.99 % (Precision), 84 % (Recall), 83 % (AP IoU)
Wardana et al, 2023 [18]	YOLOv8	98.50 % (Overall Accuracy)
Deta Sandya Prasitha et al, 2022 [19]	YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x	0.82 (v5s), 0.84 (v5m), 0.85 (v5l), 0.86 (v5x). (Average F1-Score)
Nuwar et al, 2022 [20]	YOLOv5	0.895 (F1-Score)
Junos et al, 2022 [21]	YOLOv3n	97.20 % (mAP), 0.91 (F1-Score)

También, se han realizado estudios controlados como en *Red Palm Weevil Detection in Date Palm Using Temporal UAV Imagery* [22], en donde se concluye que la detección del picudo rojo es posible mediante imágenes aéreas en el rango infrarrojo.

[4] Tabla obtenida de *Implementation of slicing aided hyper inference (SAHI) in YOLOv8 to counting oul palm trees using high-resolution aereal imagery data* [14]

1.3. Objetivos y alcance

La finalidad principal del presente trabajo consistió en validar una prueba de concepto para la detección del picudo rojo en palmeras de Montevideo, utilizando imágenes aéreas capturadas mediante drones. Con este objetivo, se desarrolló un sistema fundamentado en técnicas de VPC y aprendizaje profundo, que permite identificar la presencia de la plaga. Para ello, se utilizó la información centralizada en el SIG [23], complementado con herramientas adicionales que facilitaron la generación del conjunto de datos de entrenamiento.

Adicionalmente, se planteó ante la IM la necesidad de optimizar los procesos de detección de la plaga, con miras a una mejora en la asignación de recursos humanos y económicos. El alcance inicial se centró exclusivamente en la detección del picudo rojo. En los capítulos siguientes se detalla tanto la descripción de las herramientas utilizadas como el proceso metodológico implementado.

Capítulo 2

Introducción específica

En este capítulo se presentan en detalle las tecnologías y conceptos utilizados para el trabajo. Se parte de una introducción a las imágenes generadas por vehículos aéreos, especialmente aquellas generadas por UAV. Luego, se mencionan herramientas adicionales utilizadas en los procesos de soporte de proyectos orientados a IA, como lo son el proceso de etiquetado y la gestión de los datos. Finalmente, se puntualiza en modelos y métricas utilizados en VPC para este tipo de trabajos.

2.1. Imágenes generadas por vehículos aéreos

Los UAV han emergido como herramientas fundamentales en la obtención de imágenes aéreas de alta resolución. Su empleo ha permitido avances significativos en diversos campos, entre los que se destacan la agricultura de precisión, la vigilancia ambiental y la inspección de infraestructuras, entre otros [24].

La utilización de UAV en la adquisición de imágenes se fundamenta en varias ventajas inherentes a estos sistemas. En primer lugar, la flexibilidad operativa que ofrecen permite la planificación de vuelos adaptados a necesidades específicas de cada proyecto, garantizando la cobertura de zonas amplias o de difícil acceso (por ejemplo, palmeras en ubicaciones privadas de Montevideo o zonas en donde el acceso a pie es muy costoso). Asimismo, la capacidad de llevar equipamiento especializado (como cámaras multiespectrales, sensores térmicos y dispositivos LiDAR) posibilita la obtención de datos con distintos niveles de resolución y en diversas bandas del espectro electromagnético, que es esencial para aplicaciones que requieren análisis detallados y precisos [25].

Otro aspecto relevante es la eficiencia en términos de tiempo y costos. Comparados con métodos tradicionales de monitoreo, los UAV reducen considerablemente los recursos necesarios para este tipo de tareas, al tiempo que minimizan la exposición de personal a ambientes potencialmente peligrosos. Esto, sumado a la posibilidad de repetir vuelos en condiciones controladas, favorece la generación de series temporales de datos que permiten, en conjunción con el monitoreo, el análisis evolutivo de fenómenos específicos (como lo son las plagas).

Los UAV suelen cubrir un área específica; sin embargo, otros tipos de vehículos aéreos, como los vuelos de aviones tripulados, suelen cubren un área mucho mayor [26]. El servicio de geomática de la IM utiliza estas dos tecnologías para generar imágenes de todo Montevideo. Cada dos o tres años se realiza un vuelo tripulado que cubre toda la ciudad. En contraste, los vuelos de drones pueden ser solicitados a demanda, dado que su costo es mucho menor.

La diferencia entre estas tecnologías no solo radica en el costo y área cubierta, sino también en el tipo de cámara que llevan estos vehículos. La resolución espacial de las imágenes generadas dependen de la cámara y la altura a la que vuelan estos vehículos (estas imágenes son procesadas por varias aplicaciones informáticas, como Agisoft [27], que permite generar ortomosaicos).

2.1.1. Imágenes de drones

Las imágenes generadas por drones se destacan por su alta resolución y capacidad para capturar detalles finos del terreno. Estas imágenes suelen tener una mayor resolución (medida en $\mathrm{cm/px}$) en comparación con las generadas por aviones. El DJI Phantom 4 PRO [28] es ampliamente utilizado en aplicaciones de mapeo y fotogrametría debido a su capacidad para generar ortomosaicos con una resolución de hasta $5\,\mathrm{cm/px}$ (volando a 100 metros de altura). Otro dron ampliamente utilizado es el DJI MAVIC 3 Enterprise RTK [29]. La integración de tecnología RTK [30] en este dron permite una mayor precisión en la georreferenciación de las imágenes capturadas, a diferencia del DJI Phantom 4 PRO, que necesita puntos de apoyo relevados en el campo tomados con un GPS RTK (en las ejecuciones de vuelos con el DJI Phantom de la IM, los puntos de apoyo fueron tomados con GNSS SinoGNSS T300 Plus [31], utilizando corrección en tiempo real por NTRIP [32] con la base UYMO del Instituto Geográfico Militar uruguayo).

2.2. Herramientas de soporte en proyectos de IA

La cara visible de los proyectos de inteligencia artificial se fundamenta, en general, en los resultados y aplicaciones derivados de modelos de aprendizaje automático. No obstante, al igual que en cualquier proyecto de software, resulta fundamental integrar buenas prácticas de ingeniería de software, lo que implica focalizarse en atributos específicos del sistema, tales como la escalabilidad, la seguridad y la recuperabilidad, entre otros [33]. Para alcanzar estos objetivos, es crucial considerar disciplinas que faciliten su consecución, como la gestión de la configuración o la gestión de datos. A continuación, se exponen algunas herramientas de apoyo a varios de estos procesos.

2.2.1. Gestión de código

El control de versiones es fundamental para asegurar la trazabilidad en los proyectos. Git permite mantener un historial detallado de los cambios, lo que facilita la integración de nuevas funcionalidades y la resolución de conflictos mediante estrategias de *branching* y *merging*. Esta práctica no solo mejora la trazabilidad, sino que brinda transparencia y contribuye a la reproducibilidad del proyecto a lo largo del tiempo. Una de las implementaciones más conocidas es GitHub [34], que se ha consolidado como una plataforma de colaboración que posibilita la gestión distribuida de proyectos de software en entornos tanto académicos como empresariales.

2.2.2. Gestión de infraestructura

La correcta administración de la infraestructura es esencial para el despliegue y mantenimiento de aplicaciones de inteligencia artificial, tanto en entornos de desarrollo como en producción. Para ello, en la industria se suelen utilizar diversas

herramientas, entre que se encuentran:

- Docker y Docker Compose [35] [36]: estas herramientas permiten la creación de contenedores que encapsulan una aplicación (o varias aplicaciones) y sus dependencias, lo que asegura un entorno de ejecución uniforme y reproducible en diferentes sistemas. Docker Compose facilita la orquestación de múltiples contenedores y posibilita la simulación de entornos complejos en un ambiente local.
- Vagrant [37] [38]: se utiliza para la configuración y despliegue de máquinas virtuales. Esta herramienta ofrece un entorno virtualizado consistente y de fácil replicación, lo que resulta útil para la estandarización del entorno de desarrollo simplemente con un archivo de configuración (Vagrantfile).
- OpenShift [39] [40] y Helm [41] [42]: en entornos de producción, OpenShift basado en Kubernetes [43], permite una gestión centralizada y escalable de las aplicaciones, mientras que Helm simplifica la instalación, configuración y actualización de éstas mediante el uso de paquetes predefinidos.

2.2.3. Gestión de usuarios y permisos

La seguridad y la privacidad son aspectos críticos en el desarrollo de aplicaciones de inteligencia artificial. La correcta gestión de usuarios y permisos es esencial para garantizar que solo las personas autorizadas tengan acceso a datos sensibles y funcionalidades específicas del sistema. Para ello, se pueden implementar herramientas compatibles con el protocolo *Lightweight Directory Access Protocol* (LDAP) [**noauthor_protocolo_2024**], como lo es *Light* LDAP (LLDAP) [44]. Esta herramienta permite la autenticación y autorización de usuarios, lo que simplifica la gestión centralizada de credenciales y permisos. Además, se pueden establecer políticas de acceso basadas en roles, y así contribuir a una mayor seguridad y control sobre el sistema.

2.2.4. Gestión de etiquetado de datos

La calidad de los modelos de VPC dependen en gran medida de la precisión y consistencia del etiquetado de datos. Para optimizar este proceso, existen herramientas especializadas como CVAT [45], que facilitan la anotación de conjuntos de datos para el entrenamiento y validación de modelos. Su interfaz intuitiva y funcionalidades colaborativas permiten una asignación precisa de etiquetas en imágenes, lo que reduce errores y mejora la calidad de los datos. Entre las funcionalidades de CVAT se encuentra la integración con protocolos LDAP y la gestión de datos utilizando *object storage*.

FiftyOne [46], otra herramienta ampliamente utilizada en la industria, permite visualizar, analizar y gestionar los conjuntos de datos etiquetados. Facilita la identificación de inconsistencias y errores en las anotaciones, lo que resulta crucial para la mejora continua de los modelos de aprendizaje.

2.2.5. Gestión de repositorio de datos

El manejo eficiente de grandes volúmenes de datos es un aspecto indispensable en los proyectos de IA, especialmente las imágenes en proyectos de VPC. Entre las opciones *open source* se encuentran:

- MinIO [47]: solución de almacenamiento de objetos compatible con S3 [48] (*S3-compliant*). Permite gestionar grandes volúmenes de datos de manera escalable y segura. Su implementación posibilita la integración tanto en entornos de almacenamiento en la nube como en sistemas locales, lo que asegura la disponibilidad continua de la información. Se integra fácilmente con herramientas de anotaciones como CVAT.
- MongoDB [49]: base de datos NoSQL que se utiliza para almacenar metadatos y gestionar información asociada a los conjuntos de datos. La flexibilidad y la escalabilidad de Mongo facilitan la administración de datos estructurados y no estructurados, lo que mejora la eficiencia al realizar consultas y análisis detallados.

La integración de estas herramientas de soporte establecen un flujo de trabajo sistematizado y robusto. Este enfoque integral asegura la reproducibilidad de los resultados, optimiza la gestión de recursos y facilita tanto el mantenimiento como la evolución del sistema a lo largo del tiempo.

2.3. Modelos de visión por computadora

El desarrollo de los modelos de VPC ha permitido avances notables en la automatización y análisis de imágenes. Entre las tareas de clasificación y detección de objetos, las CNN se destacan con altos niveles de precisión y eficiencia.

2.3.1. Clasificación de objetos

La tarea de clasificación consiste en asignar una etiqueta o categoría a una imagen o a regiones específicas de la misma. Entre los modelos más utilizados en esta área se encuentra la familia EfficientNet. Esta familia fue introducida en 2019 por investigadores de Google en el artículo *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks* [50]. Este conjunto de arquitecturas se caracteriza por su innovador enfoque de escalado compuesto, que equilibra de forma simultánea la profundidad, el ancho y la resolución de la red para así lograr mejores resultados que los enfoques tradicionales.

2.3.2. Detección de objetos

La detección de objetos se enfoca no solo en clasificar, sino también en localizar de forma precisa los elementos de interés dentro de una imagen. Esta tarea se ha abordado mediante dos grandes paradigmas: los modelos de dos etapas y los modelos de una etapa.

Modelos de dos etapas

Los modelos de dos etapas, representados de manera destacada por Faster R-CNN, se caracterizan por un proceso secuencial en el que se generan propuestas de regiones de interés (ROI, del inglés *Region of Interest*) para luego clasificarlas y refinar sus límites. La primera etapa utiliza una red de propuestas de regiones (RPN, por su sigla en inglés de *Region Proposal Network*) que sugiere ubicaciones potenciales de objetos. En la segunda etapa, estas propuestas se analizan en detalle mediante un clasificador que asigna una etiqueta y ajusta la caja delimitadora. Esta aproximación, aunque computacionalmente más costosa, ofrece una

alta precisión en la detección, lo que la hace adecuada para aplicaciones donde la exactitud es prioritaria.

Modelos de una etapa

En contraste, los modelos de una etapa, como YOLO, abordan el problema de la detección en un único paso. Estos modelos dividen la imagen en una cuadrícula y, de forma simultánea, predicen la probabilidad de presencia de un objeto y la localización de su correspondiente caja delimitadora. Esta integración de procesos permite una mayor velocidad de inferencia, lo que hace de YOLO una opción idónea para aplicaciones en tiempo real, aunque hay antecedentes del uso de este tipo de modelos para la detección de objetos con una alta eficiencia en la sección 1.2.

2.4. Métricas en visión por computadora

La evaluación cuantitativa de los modelos de VPC es esencial para determinar su desempeño y compararlos de manera objetiva. En este contexto, se utilizan diversas métricas que permiten analizar tanto la precisión en la clasificación de imágenes como la efectividad en la detección y localización de objetos. A continuación, se describen las métricas más utilizadas en el campo.

2.4.1. Precisión, Recuperación y *F1-Score*

Para evaluar el rendimiento de los modelos de visión por computadora, es fundamental comprender ciertos términos básicos que se utilizan en el cálculo de las métricas. En este contexto, los verdaderos positivos (TP, del inglés *True Positives*) corresponden a los casos en los que el modelo identifica correctamente la presencia de un objeto de interés. Los falsos positivos (FP, del inglés *False Positives*) se refiere a las instancias en las que el modelo indica la presencia de un objeto de interés cuando en realidad este no está presente. Los falsos negativos (FN, del inglés *False Negatives*) y verdaderos negativos (TN, del inglés *True Negatives*) son términos que describen situaciones opuestas: los FN representan los casos en los que el modelo no detecta un objeto de interés que sí existe en la imagen, mientras que los TN indican las ocasiones en las que el modelo reconoce correctamente la ausencia de un objeto de interés.

- Precisión (*Precision*): mide la proporción de verdaderos positivos sobre el total de predicciones positivas realizadas por el modelo. Una alta precisión indica que, de todos los objetos identificados, la mayoría corresponde efectivamente a objetos de interés. Se define como:

$$\text{Precisión} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2.1}$$

- Recuperación (*Recall*): representa la proporción de verdaderos positivos detectados sobre el total de objetos de interés presentes en el conjunto de datos. Un valor elevado de *recall* sugiere que el modelo es capaz de capturar la mayoría de los objetos existentes. Se define como:

$$\text{Recuperación} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.2}$$

- Exactitud (*Accuracy*): mide la proporción de predicciones correctas (tanto verdaderos positivos como verdaderos negativos) sobre el total de predicciones realizadas. Esta métrica es útil en contextos donde las clases están equilibradas, pero puede ser engañosa en conjuntos de datos desbalanceados. Se define como:

$$\text{Exactitud} = \frac{\text{TP} + \text{TN}}{\text{Total de predicciones}} \tag{2.3}$$

- *F1-Score*: es la media armónica entre *precision* y *recall*, y proporciona una medida equilibrada cuando se requiere considerar ambas métricas simultáneamente. Se define como:

$$F1 = 2 \cdot \frac{\text{Precisión} \cdot \text{Recuperación}}{\text{Precisión} + \text{Recuperación}} \tag{2.4}$$

Un resumen intuitivo de estas métricas se puede visualizar en la imagen 2.1.

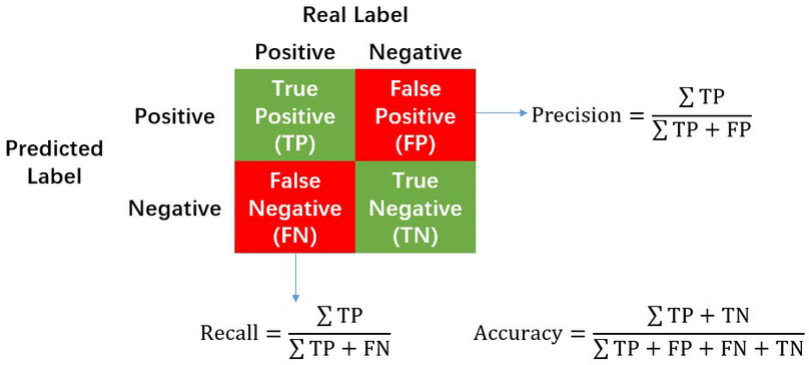

FIGURA 2.1. Matriz de confusion [1].

2.4.2. Intersección sobre unión

La intersección sobre unión (IOU, por su sigla en inglés de *Intersection Over Union*) es una métrica que evalúa la superposición entre la caja delimitadora predicha y la caja delimitadora real de un objeto. Se calcula como el cociente entre el área de intersección y el área de unión de ambas cajas. Esta métrica es fundamental en tareas de detección, ya que un IOU alto indica una buena coincidencia espacial entre la predicción y la referencia, como se puede intuir en la siguiente fórmula:

$$\text{IoU} = \frac{\text{Área de intersección}}{\text{Área de unión}} \tag{2.5}$$

2.4.3. *Average Precision* y *Mean Average Precision*

La evaluación del desempeño en tareas de detección de objetos requiere considerar cómo varía la precisión del modelo ante distintos umbrales de decisión. Se suele utilizar dos métricas claves: la precisión promedio (AP, del inglés *Average*

[1] Imagen tomada de https://www.researchgate.net/figure/Calculation-of-Precision-Recall-and-Accuracy-in-the-confusion-matrix_fig3_336402347

Precision) y media de la precisión promedio (mAP, del inglés *Mean Average Precision*).

- AP: para cada clase de objeto, se calcula el promedio de la precisión, integrando la curva de precisión-recuperación (ROC, por su sigla en inglés de *Receiver Operating Characteristic*) [51]. La AP resume el rendimiento del detector en diferentes umbrales de decisión.
- mAP: es la media aritmética de la AP calculada para cada clase. El mAP es ampliamente utilizado en competencias y evaluaciones de modelos de detección de objetos, ya que proporciona una única medida que resume la efectividad global del modelo.

2.4.4. Otras métricas relevantes

Además de las métricas anteriormente descritas, en la práctica se pueden utilizar otros indicadores, dependiendo de la aplicación y la naturaleza de los datos:

- Curva ROC y área bajo la curva (AUC, del inglés *Area Under the Curve*): la curva ROC y el AUC se emplean comúnmente en tareas de clasificación para evaluar la capacidad del modelo en discriminar entre clases.
- *Mean Square Error* (MSE) y *Mean Absolute Error* (MAE): en aplicaciones que requieren regresión, como la estimación de posiciones o dimensiones, se utilizan estas métricas para cuantificar la diferencia entre los valores predichos y los reales.

La correcta selección y combinación de estas métricas permite una evaluación integral de los modelos de VPC, lo que facilita la identificación de áreas de mejora así como la comparación objetiva entre diferentes enfoques o arquitecturas implementadas.

Capítulo 3

Diseño e implementación

En este capítulo se describen los aspectos más relevantes del diseño e implementación de la plataforma. Inicialmente se presenta su arquitectura en donde se describen sus componentes e interacciones. Luego se describe el proceso de despliegue de la infraestructura de soporte, tanto en ambiente local como en ambiente de producción. Posteriormente se enfatiza la tarea de etiquetado de datos, incluyendo los problemas y soluciones enfrentados. Finalmente se presenta el entrenamiento de los modelos, seguido del despliegue e integración con el resto de la plataforma.

3.1. Arquitectura de la plataforma

El objetivo principal de este trabajo fue brindar una introducción a un sistema completo de monitoreo y detección de la plaga del picudo rojo mediante imágenes aéreas generadas por drones y/o aviones. El enfoque se basó en el módulo de VPC, donde se destacan los hitos que fueron necesarios ejecutar dentro de la IM para llevar a cabo el trabajo. Entre estos hitos se encuentra el proceso de gestión de datos, la correcta administración de los modelos de VPC y la infraestructura que soporta estos procesos. Para ello, se planteó la plataforma de la figura 3.1. Esta plataforma se basa en un enfoque modular, donde cada componente desempeña un papel específico en el proceso de detección y monitoreo de la infección.

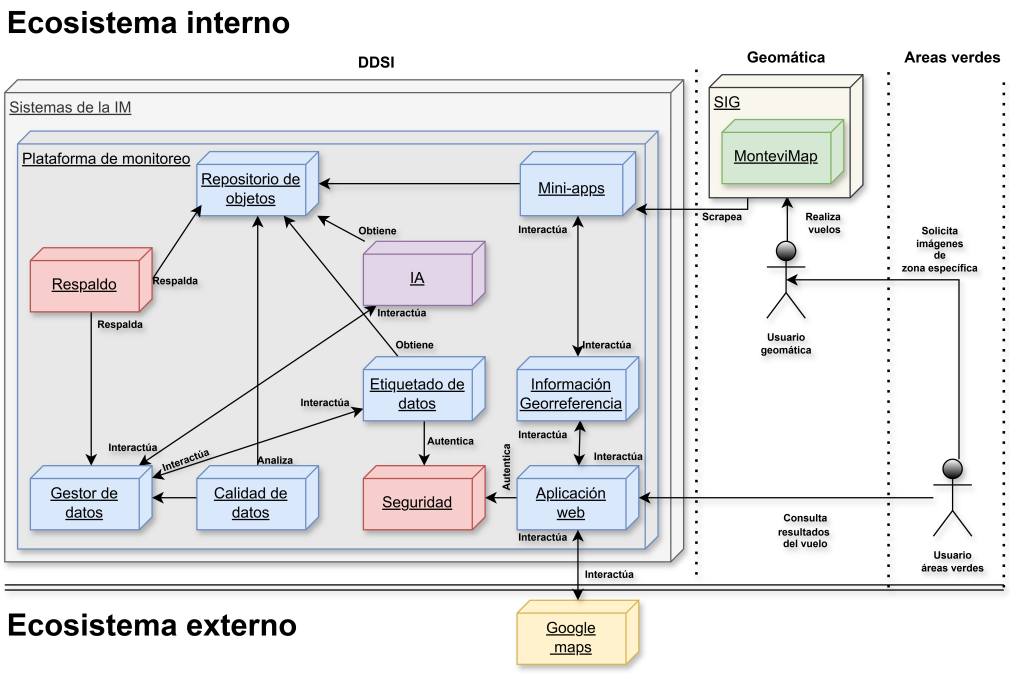

FIGURA 3.1. Diagrama de arquitectura del sistema de monitoreo.

El sistema se organiza en dos ecosistemas diferenciados: uno constituido por herramientas externas a la infraestructura de la IM, y otro integrado por herramientas internas.

El ecosistema externo se encarga de las interaccione con plataformas de servicios externos, como lo es Google Maps, en donde al momento de realizar el trabajo se encontraba la información relevada por el servicio de arbolado. En un futuro, esta información estaría integrada a sistemas dentro de la IM, como lo es QGIS [52].

Por otra parte, en el ecosistema interno se incluyen todas las herramientas necesarias para llevar a cabo un proyecto de VPC, encapsuladas dentro de la infraestructura de la IM. Este ecosistema se basa en software de código abierto y se divide en varios módulos, lo que aumenta la sustituibilidad de piezas de software (atributo comúnmente conocido intercambiabilidad en patrones de arquitecturas de software). La comunicación se basa en protocolos ligeros como REST [53], que generan un bajo acoplamiento y una alta cohesión entre módulos. Esta plataforma cuenta también con varios módulos definidos para procesos auxiliares como la seguridad, los respaldos y la gestión de la configuración.

El módulo de seguridad permite la administración de usuarios y permisos. Esto garantiza que solo los usuarios autorizados tengan acceso a la información sensible. Este módulo se basa en el protocolo LDAP, que favorece una gestión centralizada de autenticación y autorización. Adicionalmente, el módulo puede ser sustituido por el software utilizado en la IM (WSO2 [54]) o directamente interactuar con este (WSO2 puede comunicarse mediante el protocolo LDAP).

El módulo de respaldo de datos cerciora que los datos estén siempre disponibles y que se puedan recuperar en caso de pérdida. Gestiona procesos de respaldo y

recuperación de datos, lo que mitiga riesgos asociados a la pérdida de información. Al ser un módulo independiente, puede ser sustituido por otro software de respaldo, como por ejemplo: Restic [55] o Bacula [56].

El módulo gestor de datos es el encargado de almacenar los metadatos de las imágenes y los resultados de los modelos. Está conformado por una base de datos orientada a documentos, que permite almacenar varios tipos de datos (explicado en la sección 2.2.5). Esta forma de almacenamiento, que mejora la eficiencia, facilita la gestión de grandes volúmenes de información. Asimismo, esta estructura también soluciona el desafío guardar las etiquetas generadas por los usuarios en el proceso de etiquetado de datos.

El módulo de etiquetado de datos permite ejecutar el proceso de etiquetado de imágenes y se comunica tanto con el módulo de seguridad para el manejo de diferentes niveles de información, así como con el módulo gestor de datos para almacenar las etiquetas generadas por los usuarios. Este módulo intenta estar bajamente acoplado en todo momento, por lo que la integración con repositorios de datos y proveedores de identidad es esencial para su funcionamiento.

El módulo de calidad de datos se ocupa de realizar un análisis cualitativo de los datos, lo que mejora la eficiencia de los modelos al utilizar datos de alta calidad. Este módulo permite que un equipo de control de calidad pueda analizar y generar informes específicos sobre los datos, lo que habilita una traza y evolución de lás imágenes, esencial en proyectos de VPC donde la variable temporal afecta el resultado de las predicciones, como lo es este trabajo.

El módulo de información georreferenciada permite resolver la ubicación de los elementos de interés. Esto concede que los datos estén correctamente posicionados para así emparejar las predicciones de los modelos y las coordenadas reales. Además, este módulo brinda información con mayor granularidad y un historial de los elementos, como lo son las palmeras y sus antecedentes de infección.

El módulo de repositorio de objetos es el responsable del almacenamiento y gestión de las imágenes, en este caso las generadas por drones y aviones. Este módulo administra grandes volúmenes de diferentes tipos de datos y garantiza la disponibilidad continua de esta información mediante APIs. Para esta tarea, usualmente se utilizan sistemas de almacenamiento de objetos como Amazon S3 [57] o algún otro proveedor de nube.

El módulo de aplicaciones web se encarga de gestionar la interfaz de usuarios, lo que les permite interactuar con el sistema y acceder a la información de manera sencilla. Interactúa con el sistema de información georreferenciada y muestra información sobre los elementos de interés, como las palmeras y su estado de actual de infección. También permite ingresar nueva información al sistema como nuevos focos de detección de la plaga. Esto asegura que el sistema esté actualizado en todo momento y facilita a los usuarios la visualización e identificación de estos focos.

El módulo mini-apps centraliza aplicaciones utilizadas en varias tareas, como la actualización, migración y generación de datos que usualmente son insumos para los procesos de entrenamiento. Este módulo realiza tareas específicas que no están directamente relacionadas con el proceso de detección, pero que son necesarias para el correcto funcionamiento del sistema. Un ejemplo de esto es la aplicación que efectúa *web scrapping* del sistema de información geográfica de la IM, para

descargar las imágenes de los vuelos allí publicadas y guardarlas en el repositorio de imágenes.

Finalmente, el módulo de IA se encarga de cumplir con los procesos asociados al entrenamiento y despliegue de los modelos. Tiene el objetivo de manejar sus el ciclos de vida, desde su desarrollo hasta su despliegue en producción. Incluye la administración de experimentos y expone una API para la interacción con otros elementos del sistema, como las aplicaciones web o las mini-apps.

La interacción entre estos componentes permite que el sistema funcione de manera eficiente y escalable. Cada módulo se encarga de una tarea específica, lo que se alinea con SRP (siglas del inglés, *Single Responsibility Principle* [58]). Esto posibilita una fácil sustitución y mejora continua del sistema. Este tipo de arquitecturas también favorece la integración de nuevas funcionalidades y la adaptación a cambios en los requisitos de la solución. Si bien el enfoque principal del trabajo fue sobre el módulo de IA, se utilizaron herramientas de soporte que satisfacen este enfoque modular, como las que se describen en la siguiente sección.

3.2. Despliegue de la infraestructura de soporte

El despliegue de la infraestructura de soporte se llevó a cabo en dos ambientes, uno local y otro de producción.

El entorno local se utilizó para el desarrollo y la ejecución de pruebas preliminares, mientras que el de producción se destinó al despliegue final del sistema, el cual replica fielmente la infraestructura operativa de la IM.

En este contexto, el dominio local se configuró para emular el entorno de producción mediante la adopción de herramientas idénticas o análogas, tal como se ilustra en la figura 3.2.

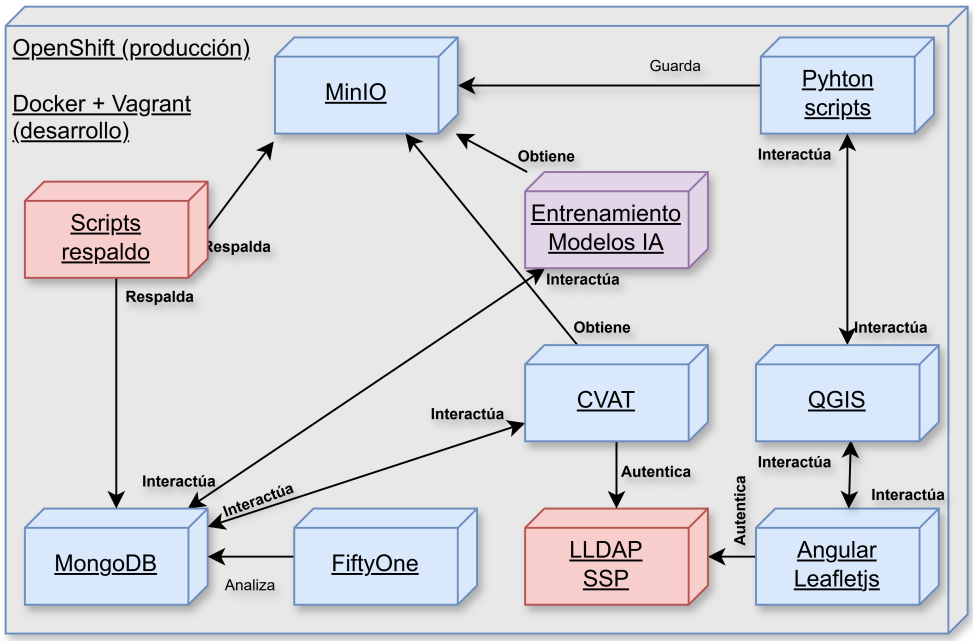

FIGURA 3.2. Diagrama de arquitectura del sistema de monitoreo.

Otro usuario desarrollador puede tener el ambiente directamente desde su máquina, simplemente clonando el repositorio [59] mediante Git y luego importar el proyecto utilizando un IDE [60]. En este caso, se utilizó Visual Studio Code. Una vez importado el proyecto, la estructura de directorios emula cada módulo del proyecto de la siguiente manera:

```
desarrollo/
├── modulo-aplicaciones-web/
│   ├── entrypoint/
│   └── landing-page/
├── modulo-calidad-datos/
│   └── fiftyone/
├── modulo-etiquetado-datos/
│   └── cvat/
├── modulo-gestor-datos/
│   └── mongo/
├── modulo-ia/
│   ├── YoloV11/
│   ├── Faster R-CNN/
│   ├── ConvNeXt/
│   ├── EfficientDet/
│   └── DETR/
├── modulo-informacion-georreferenciada/
│   └── QGIS/
├── modulo-mini-apps/
│   └── webscrapping-SIG/
├── modulo-repositorio-objetos/
│   └── MinIO/
├── modulo-respaldo/
│   └── scripts/
├── modulo-seguridad/
│   ├── SSP/
│   └── LLDAP/
├── vagrant-scripts/
├── readme.md
└── Vagrantfile
```

CÓDIGO 3.1. Estructura de directorios utilizada.

Esta estructura permite que luego se puedan sustituir estas herramientas por su análogo en la infraestructura de la IM. Por ejemplo, LLDAP puede ser sustituido por WSO2.

En ingeniería de software es importante asegurar la reproducibilidad de los ambientes de trabajo. Una de las herramientas utilizadas para asegurar esto es Docker, que habilita crear contenedores que encapsulan una aplicación y sus dependencias. Esto asegura un entorno de ejecución uniforme y reproducible en diferentes sistemas. Para la orquestación de múltiples contenedores se utilizó Docker Compose, que brinda una infraestructura en entornos complejos en donde interactúan varias aplicaciones.

Cada directorio tiene su propio archivo de Docker Compose (docker-compose.yml) que habilita el despliegue de las herramientas del módulo. Esto, además de las ventajas mencionadas anteriormente en la sección 3.1, también permite que cada componente pueda ser desarrollado y probado de manera independiente (con sus debidos *stubs* [61] o *mocks* [62]), lo que reduce el Lead Time [63]. Estos archivos de configuración pueden ser adaptados a distintas situaciones. Por ejemplo, el archivo docker-compose.yml de MinIO fue adaptado para que, si no existe el *bucket* llamado "picudo-rojo-bucket", se cree automáticamente.

Asimismo cada módulo tiene sus variables de entorno definidos como archivos .env su directorio raíz. Esto permite separar la configuración de los distintos ambientes para cada módulo. En este sentido, concede la fácil transición entre el ambiente de desarrollo y el ambiente de producción. Cuando se ejecute en el ambiente de producción, estos archivos de configuración se pueden migrar a ConfigMaps [64] y Secrets [65] de OpenShift, que permiten gestionar la configuración de los contenedores de manera segura y eficiente. Esto asegura que la información sensible, como contraseñas o claves de acceso, no se exponga en el código fuente.

Aunque Docker permite crear contenedores que emulan el ambiente de producción, esto no es suficiente para asegurar la reproducibilidad de este dominio en el entorno de desarrollo. Esto se debe a que Docker se ejecuta sobre una plataforma, que puede ser diferente en cada máquina. Por lo tanto, es necesario utilizar una herramienta que permita crear y gestionar la infraestructura de soporte de manera uniforme y reproducible. Para esto se utilizó Vagrant.

Vagrant concede la habilidad de crear una o varias máquinas virtuales (VM, del inglés *virtual machine*) que emulan una plataforma objetivo, en este caso, se podría crear una VM con la misma versión del sistema operativo que se utiliza en el ambiente real (RHEL [66]). Se maneja con un archivo de configuración llamado Vagrantfile que brinda la posibilidad de realizar la definición de la configuración de la máquina virtual como *infrastructure as a code*. Esto implica definir parámetros como la cantidad de memoria, el sistema operativo o las aplicaciones a instalar. Se puede utilizar una imagen de un sistema operativo ya configurado o crear una desde cero. En este caso se utilizó una imagen de Ubuntu 24.04 LTS [67], del repositorio oficial de Vagrant [68], que incluye la mayoría de las herramientas necesarias para el desarrollo y despliegue del sistema. Esta imagen se puede utilizar en cualquier máquina que tenga instalado Vagrant y VirtualBox [69] (también se puede utilizar otro software de virtualización, como lo es Hyper-V [70] o VMWare [71]), por lo que si otro desarrollador se incorpora al proyecto, al utilizar la misma imagen puede tener el mismo ambiente de desarrollo. Esto asegura que el sistema funcione de la misma manera en diferentes máquinas y evita problemas de compatibilidad entre diferentes versiones de software.

Asimismo, Vagrant también permite aprovisionar *scripts* de configuración a las máquinas virtuales. Estos pueden ejecutarse al iniciar estas máquinas, lo que permite instalar y configurar las aplicaciones necesarias para poner el funcionamiento el sistema. En este caso, se utilizaron cuatro scripts de aprovisionamiento:

```
vagrant-scripts/
├── 01-setup-network.sh
├── 02-install-dependencies.sh
├── 03-configure-firewall.sh
└── 04-start-dev-services.sh
```

CÓDIGO 3.2. Scripts de vagrant utilizados.

En donde el último se encarga de iniciar los servicios dentro de cada módulo (usualmente se ejecuta mediante un script de shell el comando docker compose up).

Una vez que se ejecuta el comando vagrant up, se crea una máquina virtual (si ya no fue creada), se ejecutan los scripts de aprovisionamiento, lo que resulta en el ambiente de desarrollo listo para utilizarse.

Entre los servicios que se inician se encuentran:

- MongoDB: herramienta que implementa la base de datos que permite almacenar metadatos de las imágenes ortorectificadas y los resultados de los modelos.
- MinIO: herramienta almacena las imágenes generadas por los drones y aviones.
- CVAT: herramienta utiliza para etiquetar las imágenes almacenadas en el repositorio de objetos.
- FiftyOne: herramienta de análisis, organización, visualización y generación de informes sobre los datos.
- LLDAP: herramienta de gestión de usuarios y permisos, utilizada para administrar la seguridad del sistema.
- LDAP Self Service Password: herramienta de gestión de contraseñas, utilizada para que los usuarios manejen sus credenciales.
- Landing Page: conjunto de herramientas basadas en tecnologías de desarrollo web (HTML, CSS, Javascript) que brindan información sobre el proyecto. Es la cara visible del sistema desde internet.
- Mini-apps: scripts que ejecutan varias tareas, como por ejemplo, obtener las imágenes desde el SIG de la IM.
- Entrypoint: proxy reverso que expone el ambiente de desarrollo a internet o intranet para su fácil acceso.

3.3. Proceso de etiquetado de datos

3.4. Entrenamiento de los modelos

3.5. Despliegue de los modelos

Capítulo 4

Ensayos y resultados

Todos los capítulos deben comenzar con un breve párrafo introductorio que indique cuál es el contenido que se encontrará al leerlo. La redacción sobre el contenido de la memoria debe hacerse en presente y todo lo referido al proyecto en pasado, siempre de modo impersonal.

4.1. Pruebas funcionales del hardware

La idea de esta sección es explicar cómo se hicieron los ensayos, qué resultados se obtuvieron y analizarlos.

Capítulo 5

Conclusiones

Todos los capítulos deben comenzar con un breve párrafo introductorio que indique cuál es el contenido que se encontrará al leerlo. La redacción sobre el contenido de la memoria debe hacerse en presente y todo lo referido al proyecto en pasado, siempre de modo impersonal.

5.1. Conclusiones generales

La idea de esta sección es resaltar cuáles son los principales aportes del trabajo realizado y cómo se podría continuar. Debe ser especialmente breve y concisa. Es buena idea usar un listado para enumerar los logros obtenidos.

En esta sección no se deben incluir ni tablas ni gráficos.

Algunas preguntas que pueden servir para completar este capítulo:

- ¿Cuál es el grado de cumplimiento de los requerimientos?
- ¿Cuán fielmente se puedo seguir la planificación original (cronograma incluido)?
- ¿Se manifestó algunos de los riesgos identificados en la planificación? ¿Fue efectivo el plan de mitigación? ¿Se debió aplicar alguna otra acción no contemplada previamente?
- Si se debieron hacer modificaciones a lo planificado ¿Cuáles fueron las causas y los efectos?
- ¿Qué técnicas resultaron útiles para el desarrollo del proyecto y cuáles no tanto?

5.2. Próximos pasos

Acá se indica cómo se podría continuar el trabajo más adelante.

Bibliografía

[1] Anticimex. *Picudo Rojo | Daños y tratamientos - Anticimex*. URL: https://www.anticimex.es/picudo-rojo/ (visitado 21-03-2025).

[2] A. Poplin et al. *Palm Weevils*. Mayo de 2014.

[3] MGAP. «Información actualizada sobre el picudo rojo de las palmeras a setiembre 2024». En: (). URL: https://www.gub.uy/ministerio-ganaderia-agricultura-pesca/comunicacion/noticias/informacion-actualizada-sobre-picudo-rojo-palmeras-setiembre-2024 (visitado 21-03-2025).

[4] Alfonso Arcos. *PICUDO ROJO (Rynchophorus ferrugineus) EN MONTEVIDEO MONITOREO Y CONTROL (Avances)*. Montevideo, ago. de 2024. (Visitado 21-03-2025).

[5] Intendencia de Montevideo. *Acciones de la Intendencia de Montevideo contra el Picudo Rojo | Portal institucional*. URL: https://montevideo.gub.uy/noticias/urbanismo-y-obras/acciones-de-la-intendencia-de-montevideo-contra-el-picudo-rojo (visitado 26-03-2025).

[6] Pedro Hernández Sánchez. «Cirugía especializada en palmeras». es. En: ().

[7] BMVA. *What is computer vision?* 2017. URL: https://web.archive.org/web/20170216180225/http://www.bmva.org/visionoverview.

[8] A. Torralba, P. Isola y W.T. Freeman. *Foundations of Computer Vision*. Adaptive Computation and Machine Learning series. MIT Press, 2024. ISBN: 978-0-262-37866-6. URL: https://mitpress.mit.edu/9780262048972/foundations-of-computer-vision/.

[9] Mingliang Gao et al. «Recent Advances in Computer Vision: Technologies and Applications». en. En: *Electronics* 13.14 (ene. de 2024). Number: 14 Publisher: Multidisciplinary Digital Publishing Institute, pág. 2734. ISSN: 2079-9292. DOI: 10.3390/electronics13142734. URL: https://www.mdpi.com/2079-9292/13/14/2734 (visitado 21-03-2025).

[10] Manav Sutar. «Convolutional Neural Networks (CNNs): Advancements and Future Trends». En: *Convolutional Neural Networks (CNNs): Advancements and Future Trends* (ene. de 2025). URL: https://www.academia.edu/128256115/Convolutional_Neural_Networks_CNNs_Advancements_and_Future_Trends (visitado 21-03-2025).

[11] Joseph Redmon et al. *You Only Look Once: Unified, Real-Time Object Detection*. arXiv:1506.02640 [cs]. Mayo de 2016. DOI: 10.48550/arXiv.1506.02640. URL: http://arxiv.org/abs/1506.02640 (visitado 21-03-2025).

[12] Dima Kagan, Galit Fuhrmann Alpert y Michael Fire. «Automatic large scale detection of red palm weevil infestation using street view images». en. En: *ISPRS Journal of Photogrammetry and Remote Sensing* 182 (dic. de 2021), págs. 122-133. ISSN: 09242716. DOI: 10.1016/j.isprsjprs.2021.10.004. URL: https://linkinghub.elsevier.com/retrieve/pii/S0924271621002665 (visitado 21-11-2024).

[13] Shaoqing Ren et al. *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. arXiv:1506.01497 [cs]. Ene. de 2016. DOI: 10.485

50/arXiv.1506.01497. URL: http://arxiv.org/abs/1506.01497 (visitado 21-03-2025).

[14] Naufal Najiv Zhorif et al. «Implementation of Slicing Aided Hyper Inference (SAHI) in YOLOv8 to Counting Oil Palm Trees Using High-Resolution Aerial Imagery Data». en. En: *International Journal of Advanced Computer Science and Applications* 15.7 (2024). ISSN: 21565570, 2158107X. DOI: 10.14569/IJACSA.2024.0150786. URL: http://thesai.org/Publications/ViewPaper?Volume=15&Issue=7&Code=ijacsa&SerialNo=86 (visitado 22-03-2025).

[15] M. S. Muna et al. «Development of Automatic Counting System for Palm Oil Tree Based on Remote Sensing Imagery». En: *Proceedings of the International Conference on Sustainable Environment, Agriculture and Tourism (ICOSEAT 2022)*. Atlantis Press, ene. de 2023. DOI: 10.2991/978-94-6463-086-2_68.

[16] H. Wibowo et al. «Large-Scale Oil Palm Trees Detection from High-Resolution Remote Sensing Images Using Deep Learning». En: *Big Data and Cognitive Computing* 6.3 (sep. de 2022). DOI: 10.3390/bdcc6030089.

[17] Adel Ammar, Anis Koubaa y Bilel Benjdira. «Deep-Learning-Based Automated Palm Tree Counting and Geolocation in Large Farms from Aerial Geotagged Images». en. En: *Agronomy* 11.8 (jul. de 2021), pág. 1458. ISSN: 2073-4395. DOI: 10.3390/agronomy11081458. URL: https://www.mdpi.com/2073-4395/11/8/1458 (visitado 22-03-2025).

[18] D. P. T. Wardana, R. S. Sianturi y R. Fatwa. «Detection of Oil Palm Trees Using Deep Learning Method with High-Resolution Aerial Image Data». En: *ACM International Conference Proceeding Series*. Association for Computing Machinery, 2023, págs. 90-98. DOI: 10.1145/3626641.3626667.

[19] D. Sandya Prasvita, D. Chahyati y A. M. Arymurthy. *Automatic Detection of Oil Palm Growth Rate Status with YOLOv5*.

[20] Y. Nuwara, W. K. Wong y F. H. Juwono. «Modern Computer Vision for Oil Palm Tree Health Surveillance using YOLOv5». En: *2022 International Conference on Green Energy, Computing and Sustainable Technology, GECOST 2022*. Institute of Electrical y Electronics Engineers Inc., 2022, págs. 404-409. DOI: 10.1109/GECOST55694.2022.10010668.

[21] M. H. Junos et al.

[22] Stephanie Delalieux et al. «Red Palm Weevil Detection in Date Palm Using Temporal UAV Imagery». en. En: *Remote Sensing* 15.5 (ene. de 2023). Number: 5 Publisher: Multidisciplinary Digital Publishing Institute, pág. 1380. ISSN: 2072-4292. DOI: 10.3390/rs15051380. URL: https://www.mdpi.com/2072-4292/15/5/1380 (visitado 21-03-2025).

[23] Intendencia de Montevideo. *Sistema de Información Geográfica*. es. URL: https://sig.montevideo.gub.uy/ (visitado 26-03-2025).

[24] Izham Mokhtar et al. *Image Processing Systems for Unmanned Aerial Vehicle: State-of-the-art*. Ago. de 2023. DOI: 10.20944/preprints202308.1855.v1.

[25] Unmanned System technology. *UAV/Drone Cameras for Commercial, Industrial, and Military Applications*. en-US. URL: https://www.unmannedsystemstechnology.com/expo/cameras/ (visitado 27-03-2025).

[26] 2000 Aviation. *Ortoimágenes – 2000 Aviation*. es. URL: https://www.2000aviation.com/ortoimagenes/ (visitado 27-03-2025).

[27] Agisoft. *Agisoft Metashape: Agisoft Metashape*. URL: https://www.agisoft.com/ (visitado 27-03-2025).

[28] DJI. *Support for Phantom 4 Pro*. en. URL: https://www.dji.com/global/support/product/photo (visitado 27-03-2025).

[29] DJI. *Specs - DJI Mavic 3 Enterprise - DJI Enterprise*. en. URL: https://enterprise.dji.com/mavic-3-enterprise/photo (visitado 27-03-2025).
[30] Wikipedia. *RTK (navegación)*. es. Page Version ID: 161740605. Ago. de 2024. URL: https://es.wikipedia.org/w/index.php?title=RTK_(navegaci%C3%B3n)&oldid=161740605 (visitado 27-03-2025).
[31] ComNav Technology Ltd. *Receptor GNSS T300 Plus-Receptor GNSS-ComNav Technology Ltd.* zh-CN. URL: https://www.comnavtech.com/sp/product/receiver/T300Plus.html (visitado 28-03-2025).
[32] Wikipedia. *Networked Transport of RTCM via Internet Protocol*. en. Page Version ID: 1277712218. Feb. de 2025. URL: https://en.wikipedia.org/w/index.php?title=Networked_Transport_of_RTCM_via_Internet_Protocol&oldid=1277712218 (visitado 27-03-2025).
[33] Ian Sommerville. *Software Engineering*. Addison-Wesley, mar. de 2015.
[34] Github. *Build software better, together*. en. URL: https://github.com (visitado 27-03-2025).
[35] Dirk Merkel. «Docker: lightweight Linux containers for consistent development and deployment». En: *Linux journal* 2014.239 (mar. de 2014), págs. 2-. URL: https://dl.acm.org/citation.cfm?id=2600239.2600241.
[36] Docker. *Docker documentation*. en. 0. URL: https://docs.docker.com/ (visitado 27-03-2025).
[37] HashiCorp. *Vagrant*. 2023.
[38] HashiCorp. *Documentation | Vagrant | HashiCorp Developer*. en. URL: https://developer.hashicorp.com/vagrant/docs (visitado 27-03-2025).
[39] Red Hat. *OpenShift*. 2023.
[40] Red Hat. *Red Hat Product Documentation*. URL: https://docs.redhat.com/en (visitado 27-03-2025).
[41] The Helm Authors. *Helm*. 2023.
[42] The Helm Authors. *Docs Home*. en. URL: https://helm.sh/docs/ (visitado 27-03-2025).
[43] Wikipedia. *Kubernetes*. es. Page Version ID: 166046716. Mar. de 2025. URL: https://es.wikipedia.org/w/index.php?title=Kubernetes&oldid=166046716 (visitado 27-03-2025).
[44] LLDAP Authors. *LLDAP: Light LDAP implementation*. 2023.
[45] Boris Sekachev et al. *opencv/cvat: v1.1.0*. Ago. de 2020. DOI: 10.5281/zenodo.4009388. URL: https://doi.org/10.5281/zenodo.4009388.
[46] B. E. Moore y J. J. Corso. «FiftyOne». En: *GitHub. Note: https://github.com/voxel51/fiftyone* (2020).
[47] MinIO, Inc. *MinIO: High Performance Object Storage*. 2023.
[48] Amazon. *Amazon S3*. es. Page Version ID: 164024822. Dic. de 2024. URL: https://es.wikipedia.org/w/index.php?title=Amazon_S3&oldid=164024822 (visitado 28-03-2025).
[49] MongoDB, Inc. *MongoDB: The Developer Data Platform*. 2023.
[50] Mingxing Tan y Quoc V. Le. *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*. arXiv:1905.11946 [cs]. Sep. de 2020. DOI: 10.48550/arXiv.1905.11946. URL: http://arxiv.org/abs/1905.11946 (visitado 27-03-2025).
[51] Wikipedia. *Curva ROC*. es. Page Version ID: 166160029. Mar. de 2025. URL: https://es.wikipedia.org/w/index.php?title=Curva_ROC&oldid=166160029 (visitado 01-04-2025).

[52] Wikipedia. *QGIS*. es. Page Version ID: 166170815. Mar. de 2025. URL: https://es.wikipedia.org/w/index.php?title=QGIS&oldid=166170815 (visitado 09-04-2025).
[53] Wikipedia. *Protocolo ligero de acceso a directorios*. es. Page Version ID: 162588716. Sep. de 2024. URL: https://es.wikipedia.org/w/index.php?title=Protocolo_ligero_de_acceso_a_directorios&oldid=162588716 (visitado 27-03-2025).
[54] WSO2. *Deliver Seamless Login Experiences with WSO2 Identity Server*. URL: https://wso2.com/identity-server/,https://wso2.com/identity-server/ (visitado 04-04-2025).
[55] Restic. *restic · Backups done right!* URL: https://restic.net/ (visitado 09-04-2025).
[56] Bacula. *Documentation*. en-US. URL: https://www.bacula.org/documentation/ (visitado 04-04-2025).
[57] Amazon Web Services. *AWS | Almacenamiento de datos seguro en la nube (S3)*. es-ES. URL: https://aws.amazon.com/es/s3/ (visitado 04-04-2025).
[58] M. Soni y S. Soni. *Software Engineering Text Book*. 2024. URL: https://books.google.com.uy/books?id=L_YxEQAAQBAJ.
[59] Bruno Masoller. *brunomaso1/uba-ceia at ceia-proy-final*. URL: https://github.com/brunomaso1/uba-ceia/tree/ceia-proy-final (visitado 04-04-2025).
[60] Wikipedia. *Entorno de desarrollo integrado*. es. Page Version ID: 165982855. Mar. de 2025. URL: https://es.wikipedia.org/w/index.php?title=Entorno_de_desarrollo_integrado&oldid=165982855 (visitado 09-04-2025).
[61] Wikipedia. *Talón de prueba*. es. Page Version ID: 164505596. Ene. de 2025. URL: https://es.wikipedia.org/w/index.php?title=Tal%C3%B3n_de_prueba&oldid=164505596 (visitado 04-04-2025).
[62] Wikipedia. *Objeto simulado*. es. Page Version ID: 157309935. Ene. de 2024. URL: https://es.wikipedia.org/w/index.php?title=Objeto_simulado&oldid=157309935 (visitado 04-04-2025).
[63] Wikipedia. *Lead time*. en. Page Version ID: 1279081155. Mar. de 2025. URL: https://en.wikipedia.org/w/index.php?title=Lead_time&oldid=1279081155 (visitado 09-04-2025).
[64] Red Hat. *Chapter 21. ConfigMaps | Red Hat Product Documentation*. URL: https://docs.redhat.com/en/documentation/openshift_container_platform/3.11/html/developer_guide/dev-guide-configmaps (visitado 10-04-2025).
[65] Red Hat. *Chapter 20. Secrets | Red Hat Product Documentation*. URL: https://docs.redhat.com/en/documentation/openshift_container_platform/3.11/html/developer_guide/dev-guide-secrets (visitado 10-04-2025).
[66] Red Hat. *Sistema operativo Red Hat Enterprise Linux*. URL: https://www.redhat.com/es/technologies/linux-platforms/enterprise-linux (visitado 10-04-2025).
[67] Progress Chef's Bento. *bento/ubuntu-24.04*. URL: https://portal.cloud.hashicorp.com/vagrant/discover/bento/ubuntu-24.04 (visitado 04-04-2025).
[68] HashiCorp. *HashiCorp Cloud Platform | Bento*. URL: https://portal.cloud.hashicorp.com/vagrant/discover/bento (visitado 04-04-2025).
[69] Wikipedia. *VirtualBox*. es. Page Version ID: 166376510. Mar. de 2025. URL: https://es.wikipedia.org/w/index.php?title=VirtualBox&oldid=166376510 (visitado 04-04-2025).
[70] meaghanlewis. *Información general sobre la tecnología Hyper-V*. es-es. Ene. de 2025. URL: https://learn.microsoft.com/es-es/windows-server/virtualization/hyper-v/hyper-v-overview (visitado 04-04-2025).

[71] Wikipedia. *VMware*. es. Page Version ID: 165895175. Mar. de 2025. URL: https://es.wikipedia.org/w/index.php?title=VMware&oldid=165895175 (visitado 04-04-2025).