Desarrollo de un modelo para la detección de drones de ala rotatoria

Iván Vargas Carmona

Iván Darío Higuera González

José Alejandro Cárdenas Ríos

Asesor

William Alexander Cuevas Carrero

Universidad Nacional Abierta y a Distancia UNAD

Escuela de Ciencia Básicas Tecnología e Información ECBTI

Ingeniería Electrónica

2024

Dedicatoria

A nuestros seres queridos, cuyo apoyo incondicional han sido nuestra mayor fuente de inspiración a lo largo de este arduo pero gratificante viaje académico. A nuestras familias, por su paciencia y aliento constante, amigos, compañeros por su comprensión y alegría compartida.

Este trabajo está dedicado a todos aquellos que creen en la fusión de la tecnología y la responsabilidad, en la capacidad de la inteligencia artificial para contribuir a la seguridad y bienestar de nuestra sociedad. Agradecemos a quienes han sido parte de este proyecto, guiándonos con su experiencia y brindando su apoyo en cada paso.

A la memoria de aquellos que siempre creyeron en la importancia de avanzar en el conocimiento y la innovación. Que este esfuerzo contribuya, al progreso tecnológico y a la construcción de un futuro más seguro.

Gracias a todos los que formaron parte de este trayecto, su presencia es la esencia de este logro.

Resumen

Este proyecto presenta una solución basada en inteligencia artificial a los retos de seguridad debidos al aumento y el uso inadecuado de drones de ala rotatoria. La proliferación de estos dispositivos ha planteado problemas de seguridad, privacidad y control del espacio aéreo. La propuesta se enfoca en el desarrollo de un modelo de detección eficiente utilizando el framework YOLOv8 y técnicas de fine tuning, Maximizando la precisión y eficiencia en la identificación de drones, lo que contribuirá a la seguridad de infraestructura y espacio aéreo. Con este proyecto se sustenta en la relevancia actual de abordar estas problemáticas y aprovechar tecnologías avanzadas. La solución propuesta permite detectar patrones distintivos de drones de ala rotatoria, superando las limitaciones de enfoques convencionales. Cuenta con una interfaz gráfica intuitiva para acceder a información en tiempo real, facilitando su uso para diferentes usuarios. El modelo presentado, fue entrenado para identificar drones de ala rotatoria en diversos entornos, evaluando su desempeño mediante métricas precisión, recall, loss, Map50, Map50-95, en un entorno controlado y con distancias máximo de 50 metros entre la cámara y el objeto a analizar. Siendo un proyecto que impacta positivamente como un desarrollo que permite optimizar la seguridad aérea con una solución que permite la detección de drones y que puede ser aplicado en áreas sensibles como aeropuertos y eventos masivos. La combinación de inteligencia artificial avanzada y la interfaz basada en un modelo de entrenamiento proporciona una herramienta sólida y accesible, contribuyendo al progreso tecnológico y a la seguridad en la era de los drones.

Palabras clave: Drones, Inteligencia artificial, modelo, Seguridad, Interfaz

Abstract

This project presents an AI-based solution to the security challenges posed by the increasing number and misuse of rotary-wing drones. The proliferation of these devices has raised concerns about security, privacy, and airspace control. The proposal focuses on developing an efficient and accurate detection model using the YOLOv8 neural network and fine-tuning techniques. Maximizing the accuracy and efficiency of drone identification will contribute to the security of critical areas and the prevention of incidents. This project highlights the importance of addressing these issues and leveraging advanced technologies. The proposed solution allows for the detection of distinctive patterns of rotary-wing drones, overcoming the limitations of conventional approaches. It features an intuitive graphical interface for accessing real-time information, facilitating its use by different users. The presented model was trained to identify rotary-wing drones in various environments. Its performance was evaluated using precision, recall, loss, Map50, and Map50-95 metrics in a controlled environment with maximum distances of 50 meters between the camera and the object to be analyzed. This project has a positive impact as a development that optimizes air security with a solution that enables drone detection and can be applied in sensitive areas such as airports and mass events. The combination of advanced artificial intelligence and a model-based training interface provides a robust and accessible tool, contributing to technological progress and security in the age of drones.

Keywords**:** Drones, Artificial Intelligence, Detection Algorithm, Security, Graphical Interface

Tabla de Contenido

Lista de Tablas

Lista de Figuras

Introducción

En la era contemporánea, la proliferación de drones de ala rotatoria ha revolucionado diversos sectores, desde la industria hasta la seguridad, generando una serie de desafíos significativos. La creciente popularidad y versatilidad de estos dispositivos plantean inquietudes cruciales en términos de seguridad, privacidad y control del espacio aéreo. En este contexto, el presente proyecto de grado se posiciona como una respuesta estratégica y aplicada para abordar los desafíos derivados del aumento en el uso de drones, enfocándose específicamente en el desarrollo de un modelo de entrenamiento basado en inteligencia artificial para la detección eficiente de drones de ala rotatoria.

La relevancia de este proyecto está respaldada por la importancia estratégica del sector aeronáutico y la necesidad de enfrentar amenazas emergentes, como el uso ilícito de drones, como se señaló en la (Estrategia de Seguridad Nacional, 2021). La capacidad de estos dispositivos para paralizar aeropuertos o infraestructuras críticas, así como su potencial uso en acciones terroristas, subraya la urgencia de desarrollar soluciones efectivas.

La propuesta se enfoca en la aplicación del framework YOLOv8 y técnicas de fine tuning para maximizar la precisión y eficiencia en la detección de drones, contribuyendo directamente a la seguridad de áreas críticas y la prevención de incidentes. Este enfoque aplicado implica la creación de un modelo entrenado capaz de identificar patrones distintivos de drones de ala rotatoria, superando las limitaciones de enfoques convencionales.

La investigación aplicada se fundamenta en la necesidad de ofrecer soluciones prácticas y tangibles para los problemas emergentes en la detección de drones, destacando la implementación de una interfaz gráfica para acceder a información en tiempo real. La evaluación del rendimiento del modelo de entrenamiento mediante métricas específicas garantizará la

eficacia y confiabilidad del sistema, reforzando su aplicación en entornos aeroespaciales diversos.

Este proyecto de grado se alinea con la urgente demanda de soluciones innovadoras en el ámbito de la seguridad aeroespacial, integrando tecnologías avanzadas con un enfoque práctico para enfrentar los desafíos contemporáneos asociados con la proliferación de drones de ala rotatoria de altos mandos, se discutieron medidas para mitigar estos riesgos, incluyendo el establecimiento de protocolos, sistemas de detección y alerta temprana, y tácticas defensivas. Estas acciones se llevan a cabo en respuesta a los recientes ataques contra la minga indígena en Toribio, Cauca, y en medio del cese al fuego pactado con las disidencias por el presidente actual de Colombia. El general del ejército destacó la importancia de implementar sistemas de detección y alerta temprana, así como el establecimiento de zonas de exclusión aérea en áreas sensibles y estratégicas. Además, enfatizó la necesidad de desarrollar tácticas defensivas para neutralizar el potencial uso de drones por parte de grupos criminales. Estas medidas buscan garantizar la seguridad y protección de las comunidades afectadas y prevenir posibles ataques con drones en el futuro.

Es por esto por lo que la proliferación de drones de ala rotatoria ha generado desafíos significativos en términos de seguridad y control del espacio aéreo. La creciente popularidad y versatilidad de estos dispositivos han llevado a su uso en diversas aplicaciones, desde la industria y la agricultura hasta la seguridad y la logística. Sin embargo, este aumento en el uso de drones también ha dado lugar a riesgos relacionados con la seguridad y la privacidad, especialmente en entornos críticos como aeropuertos, eventos masivos, infraestructuras críticas y zonas restringidas (Vargas, 2024).

Planteamiento del Problema

Ante la creciente incidencia de incidentes relacionados con drones, como la suspensión de un partido de la Premier League debido a la presencia de un dron en el estadio y la preocupación de las Fuerzas Militares sobre su posible uso por parte de grupos criminales, surge la necesidad de mejorar los procesos de detección de drones de ala rotatoria. Esta optimización es crucial para desarrollar una solución tecnológica que fortalezca la seguridad y el control del espacio aéreo.

El objetivo es implementar un enfoque eficiente que permita detectar rápidamente drones no autorizados, utilizando tecnologías avanzadas como sistemas de radar y cámaras de alta resolución, junto con algoritmos de análisis de datos en tiempo real. Esta solución integral busca garantizar la seguridad en eventos y operaciones críticas, protegiendo al mismo tiempo la privacidad y los derechos individuales. El proyecto aspira a contribuir al desarrollo de una infraestructura de seguridad aérea adaptable a los desafíos actuales asociados con el uso de drones en la sociedad moderna.”

¿Cómo optimizar los procesos para la detección de drones de ala rotatoria de forma eficiente, con el fin de aportar una solución tecnológica que permita mejorar la seguridad y el control del espacio aéreo?

Justificación

Para abordar estos desafíos y minimizar los riesgos asociados con la proliferación de drones de ala rotatoria, este proyecto de grado propone el desarrollo de un modelo de entrenamiento de detección basado en YOLOv8 y técnicas de fine tuning. La implementación de esta solución se enfocará en maximizar la precisión de la detección de UAV (Unmanned Aerial Vehicle, Vehículo aéreo no tripulado), permitiendo una identificación oportuna y confiable en diversos entornos. Al utilizar tecnologías de vanguardia como YOLOv8 y fine tuning se espera superar las limitaciones de los enfoques tradicionales y proporcionar una herramienta efectiva para mejorar la seguridad en el espacio aéreo y la protección de áreas críticas. Asimismo, el desarrollo de una interfaz gráfica el cual le permitirá a los usuarios acceder a la información de detección en tiempo real, facilitando su utilización por personal con diferentes niveles de experiencia técnica. Con esta solución, se pretende impulsar el progreso tecnológico y contribuir al bienestar y seguridad de la sociedad en la era de los drones.

La propuesta pretende desarrollar un modelo entrenado innovador que permitirá la detección eficiente y precisa de drones de ala rotatoria mediante inteligencia artificial. La creciente popularidad y versatilidad de los drones ha generado una amplia variedad de aplicaciones en diversos campos, desde la industria y la agricultura hasta la seguridad y la logística. El aumento en el uso de drones ha provocado desafíos relacionados con su detección y control, lo que ha impulsado la necesidad de implementar soluciones avanzadas y eficaces para su identificación. La relevancia del proyecto radica en abordar un problema social relevante, relacionado con la seguridad, privacidad, y el control del espacio aéreo. La detección de drones de ala rotatoria se ha vuelto una tarea cada vez más crítica, especialmente en entornos sensibles como aeropuertos, eventos masivos, infraestructuras críticas y zonas restringidas. En este

contexto, la aplicación de técnicas de inteligencia artificial se presenta como una solución prometedora para mejorar la capacidad de identificación y monitoreo de estos dispositivos. La creación de este modelo de entrenamiento permitirá el análisis y reconocimiento de patrones característicos de los drones de ala rotatoria, lo que a su vez permitirá diferenciarlos de otros objetos en el espacio aéreo. La adopción de enfoques de inteligencia artificial, con el modelo de entrenamiento, garantizará una mayor precisión y eficiencia en la detección, lo que contribuirá significativamente a la seguridad y protección de áreas sensibles.

Esta interfaz será la base para futuras investigaciones que permitan al personal de seguridad física o controladores de tráfico aéreo, obtener información en tiempo real sobre la presencia y ubicación de drones de ala rotatoria en un área determinada. La integración de una interfaz gráfica basado en un modelo de entrenamiento garantizará que la tecnología desarrollada sea accesible garantizando que el uso pueda ser de uso personal con diferentes niveles de experiencia técnica, maximizando así su impacto en la sociedad.

El resultado de este proyecto de grado ofrecerá una herramienta eficiente para la detección de drones de ala rotatoria, contribuyendo a la seguridad y el control del espacio aéreo. La combinación de la inteligencia artificial con el desarrollo de una interfaz gráfica accesible la cual representa una solución innovadora y sólida, con un impacto significativo en el ámbito tecnológico y social.

Objetivos

Objetivo General

Entrenar un modelo de inteligencia artificial para la detección de drones de ala rotatoria usando visión computacional.

Objetivos Específicos

Realizar el entrenamiento de modelo de inteligencia artificial para la detección de drones.

Desarrollar la interfaz gráfica para la detección de drones de ala rotatoria.

Evaluar las métricas de rendimiento del modelo de entrenamiento de inteligencia artificial, como lo son: precisión, exhaustividad, exactitud y F1 Score.

Marco Referencial

Estado del Arte

Investigaciones Nacionales

Título: CyberDrone: una plataforma de ciberseguridad para detección de ataques a drones.

El aumento en el uso de drones en la infraestructura eléctrica global, incluyendo Colombia, ha llevado a mejoras significativas en la eficiencia y seguridad de las operaciones. Codensa, como ejemplo, ha implementado drones para inspecciones termográficas, mapeo topográfico y otras tareas, reduciendo costos y mejorando la productividad (Zapata y García, 2021).

El mercado de UAV en la infraestructura eléctrica está en crecimiento, con aplicaciones que van desde inspecciones solares hasta respuesta a desastres. Sin embargo, se destaca que la ciberseguridad en los drones es un área crítica y subdesarrollada. La falta de mecanismos de defensa cibernética eficientes y la ausencia de protocolos seguros pueden poner en riesgo la integridad y confidencialidad de los datos operativos (Zapata y García, 2021).

Se evidencia una brecha en la investigación sobre ataques al flujo de comunicaciones de datos en los drones, lo que representa un riesgo potencialmente grave. Se menciona la necesidad de desarrollar soluciones innovadoras que cumplan con los requisitos de seguridad y desempeño en la operación de UAV (Zapata y García, 2021).

Se presentan casos de ciberataques a drones, destacando la importancia de abordar estos problemas de seguridad. Además, se relaciona la importancia de la ciberseguridad en la infraestructura crítica, como se evidenció en casos como Stuxnet y otros ataques a instalaciones estratégicas.

La introducción de conceptos como el “cyber deception” se propone como una estrategia para confundir y desorientar a los atacantes. Se discute la necesidad de aumentar la complejidad de los sistemas de protección para prevenir el reconocimiento de red y mejorar la detección de intrusiones (Zapata y García, 2021).

Se resalta que las herramientas basadas en el engaño pueden proporcionar ventajas significativas sobre los controles de seguridad tradicionales. Estas herramientas se centran en las percepciones de los atacantes, buscando inducir acciones beneficiosas para la defensa.

Finalmente, se menciona un enfoque sistemático para incorporar tácticas de engaño en el diseño de software, destacando la importancia de continuar el desarrollo de técnicas basadas en el “cyber deception”.

Este estado del arte proporciona un contexto sólido para la introducción de su plataforma CyberDrone y destaca la necesidad de abordar las vulnerabilidades en la ciberseguridad de los drones utilizados en la infraestructura eléctrica (Zapata y García, 2021).

Investigaciones Internacionales

Título: Drones y seguridad nacional un estudio multidimensional. El objetivo del GT-Drones es estudiar la situación actual de los drones en el contexto de la Seguridad Aeroespacial, identificar carencias y limitaciones, y proponer soluciones para garantizar su uso eficaz y seguro (López, 2022).

La introducción subraya la relevancia estratégica del sector aeronáutico y la creciente inquietud ante el uso ilícito de drones, que representa una amenaza significativa para la seguridad nacional. El impacto potencial de estos vehículos no tripulados en aeropuertos, infraestructuras críticas y operaciones aéreas ha llevado al Consejo Nacional de Seguridad Aeroespacial (CNSA) a establecer el Grupo de Trabajo GT-Drones.

El objetivo primordial del GT-Drones es abordar la disrupción que los drones pueden generar en el ámbito de la seguridad, tanto en operaciones aéreas como en la protección de personas e instalaciones. La Estrategia de Seguridad Nacional 2021 destaca la importancia de prevenir posibles ataques terroristas y asegurar la integridad del sector aeronáutico (López, 2022).

En respuesta a esta preocupación, el GT-Drones se embarcó en un estudio interministerial y estratégico. Su misión es analizar la situación actual de los vehículos aéreos no tripulados, identificar problemas, carencias y limitaciones, y proponer soluciones globales y definitivas. Este estudio se divide en varios ejes temáticos, como registros, difusión normativa, zonas de vuelo, control del tráfico aéreo, capacidades contra drones (C-UAS) se refiere a “Counter-Unmanned Aircraft Systems” o “Counter-Unmanned Aerial Systems”. Estos son sistemas diseñados para detectar, identificar, rastrear y, en algunos casos, neutralizar drones o aeronaves no tripuladas que representan una amenaza o se encuentran en un espacio aéreo no autorizado. (), normativa y legislación, necesidades de otros organismos y desarrollo tecnológico (López, 2022).

Los resultados del estudio reflejan avances efectivos en la resolución de problemas vinculados a los drones. Se destacan iniciativas y regulaciones implementadas por organismos relacionados con el sector aeronáutico y de seguridad y defensa. Aunque muchos problemas han sido abordados, algunos aún carecen de soluciones viables o no han sido completamente identificados (López, 2022).

Los resultados recogidos demuestran que se ha avanzado eficazmente en la resolución de los problemas que la aparición de los drones ha supuesto. Numerosas iniciativas procedentes de los principales organismos relacionados con el sector, principalmente el aeronáutico y de seguridad y defensa, se han puesto en marcha, consiguiendo que un alto número de problemas ya

estén solucionados o en vías de solución, aunque, por el contrario, otros o no se han encontrado soluciones viables o no han sido identificados (López, 2022).

Título: Optimización conjunta de abortos de tareas y rutas de sistemas de camiones y drones bajo ataques aleatorios (Joint optimization of task abortions and routes of truck-and-drone systems under random attacks). El presente artículo se centra en la optimización de sistemas de camiones y drones en entornos afectados por ataques aleatorios, considerando estrategias de interrupción de tareas y planificación de rutas. A través de la revisión de la literatura existente, se destacan diversas contribuciones y enfoques relevantes para contextualizar y fundamentar el trabajo de tesis (BBC, 2023).

Sistemas Colaborativos de Camiones y Drones: Se ha observado un creciente interés en el desarrollo de sistemas colaborativos que integran camiones y drones para llevar a cabo diversas tareas, como vigilancia militar, reconocimiento logístico y búsqueda y rescate en situaciones de desastre (Giordan y Adams, 2020; Wang et al., 2022). Estos sistemas ofrecen soluciones eficientes y rápidas para la realización de tareas críticas.

Importancia de la Confiabilidad en Sistemas Críticos: La literatura resalta la importancia de la confiabilidad en sistemas críticos y de seguridad, donde la capacidad de realizar tareas con éxito es fundamental. Sin embargo, se observa una brecha en la investigación, ya que la mayoría de los modelos existentes se centran en la maximización de la probabilidad de éxito de tareas (Myers, 2022; Cui et al., 2023).

Estrategias de Interrupción de Tareas: La toma de decisiones respecto a la interrupción de tareas se ha abordado en diferentes contextos. Myers (2022) propuso estrategias para sistemas de espera con un procedimiento de rescate, mientras que Cui et al. (2023) desarrollaron políticas de interrupción basadas en procesos de falla en dos etapas. Estos trabajos enfatizan la importancia

de equilibrar la confiabilidad del sistema y la minimización de costos asociados a fallas (Wu and Castro, 2023).

Planificación de rutas en entornos hostiles: El diseño de rutas para sistemas de camiones y drones bajo condiciones adversas ha sido abordado en diversos contextos. Se destacan modelos que consideran la seguridad del sistema en entornos hostiles (Tamke y Buscher, 2022). Sin embargo, pocos estudios han integrado estrategias de interrupción de tareas en la planificación de rutas, lo cual constituye una brecha que este trabajo de tesis busca abordar.

Ataques Aleatorios y Estrategias de Aborto en Sistemas: La literatura revisada reconoce la posibilidad de ataques aleatorios en sistemas complejos y destaca la necesidad de estrategias de aborto de tareas para minimizar pérdidas en casos de fallos catastróficos (Peng y Murray, 2022; Zhu et al., 2023). Este enfoque es relevante en sistemas de camiones y drones bajo amenazas imprevisibles.

Contribuciones del Trabajo Propuesto: El trabajo de tesis propuesto presenta contribuciones significativas al abordar la interrupción de tareas y la planificación de rutas en sistemas de camiones y drones bajo ataques aleatorios. Se destaca la caracterización de ataques mediante evaluación probabilística, un modelo híbrido que considera la fiabilidad del sistema y la optimización de estrategias de interrupción y rutas (López, 2019).

Título: el dispositivo del dron: entre la vigilancia securitaria y la necro política. El artículo El dispositivo del dron: entre la vigilancia securitaria y la necro política, de Mendiola (2018), publicado en la revista Convergencia en 2019, analiza el papel que juega el dron en el contexto de la actual vigilancia securitaria. El artículo se basa en los planteamientos conceptuales de los estudios críticos de la seguridad, que ponen en cuestión las narrativas hegemónicas sobre la seguridad y la vigilancia (Mendiola, 2018).

El artículo comienza con una reflexión sobre la imbricación entre visibilidad y poder. Mendiola (2018) señala que la modernidad se ha caracterizado por un afán por articular un régimen de observación que torne la sociedad en una trama transparente. Este afán se manifiesta en la proliferación de dispositivos de vigilancia, como el dron (Mendiola, 2018).

El dron es un dispositivo de vigilancia que posee una serie de características que lo hacen especialmente eficaz para la vigilancia securitaria. En primer lugar, el dron tiene una gran capacidad de movilidad y autonomía, lo que le permite acceder a zonas de difícil acceso. En segundo lugar, el dron puede llevar cámaras de alta resolución que permiten capturar imágenes de gran calidad. En tercer lugar, el dron puede ser operado a distancia, lo que reduce el riesgo para los operadores (Iglesias, 2022). Mendiola (2018) analiza el papel del dron en la vigilancia securitaria desde tres perspectivas:

La redefinición de la vigilancia: El dron redefine la vigilancia al hacerla más omnipresente y omnipresente. El dron puede estar presente en cualquier lugar, en cualquier momento, lo que dificulta la posibilidad de escapar de la mirada del poder.

Las lógicas bélico-policiales: El dron está inmerso en lógicas bélico-policiales. El dron se utiliza para la vigilancia y el control de las poblaciones, así como para la guerra.

La reconfiguración geográfica del poder soberano: El dron reconfigura la geografía del poder soberano. El dron permite al poder soberano actuar en cualquier lugar del mundo, sin tener que desplegar tropas en el terreno.

Mendiola (2018) concluye que el dron es un dispositivo de vigilancia que tiene un impacto significativo en la sociedad. El dron contribuye a la seguridad de la sociedad, al reforzar el poder del Estado y al limitar los derechos y libertades de los ciudadanos.

Principales contribuciones del artículo, el artículo es una contribución importante a los estudios sobre el dron. El artículo ofrece una visión crítica del dron, que pone en cuestión las narrativas hegemónicas sobre este dispositivo (Mendiola, 2018).

El artículo destaca las siguientes contribuciones:

Analiza el papel del dron en la vigilancia securitaria desde una perspectiva crítica.

Señala que el dron tiene una serie de características que lo hacen especialmente eficaz para la vigilancia securitaria.

Expone las implicaciones del uso del dron para la sociedad.

Discusión

El autor inicia su reflexión destacando la relación entre visibilidad y poder en la modernidad. Señala cómo la sociedad contemporánea busca establecer un régimen de observación que convierta a la sociedad en un entramado transparente. En este contexto, el dron emerge como un dispositivo de vigilancia con características únicas que lo hacen altamente efectivo en la vigilancia securitaria. El cual es un trabajo riguroso y bien argumentado. El artículo ofrece una perspectiva crítica del dron que es necesaria para comprender el impacto de este dispositivo en la sociedad. A pesar de estas limitaciones, el artículo de Mendiola-Gonzalo es una contribución importante a los estudios sobre el dron.

Línea de Tiempo y Estudios a través de la Historia del DRON

El desarrollo de los drones ha sido un proceso continuo y significativo en la historia de la tecnología. En 1907, los hermanos Jacques y Louis Bréguet crearon el primer dron, aunque su diseño era limitado y requería la participación de cuatro personas para mantenerlo estable. A medida que la tecnología avanzaba, los drones que podían ser controlados por radio se volvieron

disponibles para el público en general a un precio razonable en la década de los 60 (Bestechnology Group, 2019).

Esto abrió las puertas a una amplia variedad de aplicaciones, incluyendo operaciones policiales.

En 2002, se llevó a cabo la primera operación policial en Estados Unidos utilizando drones. Esto marcó un hito importante en el uso de estos dispositivos para fines de seguridad. En 2006, la FAA (Federal Aviation Administration) emitió los primeros permisos comerciales para el uso de drones (Bestechnology Group, 2019).

En 2010, una empresa francesa desarrolló el primer dron que podía ser controlado a través de un teléfono inteligente conectado a través de Wi-Fi. Esto representó un avance significativo en la tecnología de los drones y abrió nuevas posibilidades para su uso en diversas áreas (Bestechnology Group, 2019).

En 2016, se creó el primer dron con visión inteligente y tecnología de aprendizaje automático, lo que permitió volar evitando obstáculos y capturar imágenes del entorno. Esto ha llevado a una mayor precisión y eficiencia en la utilización de drones para una variedad de aplicaciones.

Recientemente, en la conferencia MARS (Machine Learning, Automation, Robotics and Space) de 2019, Amazon presentó una nueva versión de su dron mensajero, Prime Air. Este dron es completamente eléctrico y está equipado con cámaras térmicas, sensores y configurado utilizando Machine Learning. Esto le permite volar evitando obstáculos como tendidos eléctricos, aves y otros objetos. Según Amazon, tiene planeado realizar entregas en menos de 30 minutos en un radio de 24 kilómetros, exclusivamente para paquetes de un peso determinado (Bestechnology Group, 2019).

Marco Teórico

Antecedentes Históricos

Los orígenes de los drones se remontan al final del siglo XIX, cuando Nikola Tesla, inventor serbio-americano, demostró la viabilidad de controlar a distancia pequeños dispositivos mediante frecuencias de radio. A lo largo del siglo XX, los militares expandieron y adaptaron el concepto de control remoto de Tesla, dando lugar al primer avión no tripulado al término de la Primera Guerra Mundial. Durante este periodo, se emplearon bombas radiocontroladas, equipos con cámaras y designadores láser como estrategias militares (Mendiola, 2018).

En 1907, los hermanos Louis y Jacques Bréguet desarrollaron el primer cuadricóptero, aunque requería ser tripulado por cuatro personas, una por cada motor, lo que lo excluía de la categoría de dron. A pesar de esta limitación, el vehículo logró elevarse unos pocos pies del suelo, sentando las bases para futuros avances en la aeronáutica (Mendiola, 2018).

Posteriormente, a lo largo del año 1922, el ingeniero aeronáutico T. Claude Ryan fundó la “Compañía Aeronáutica Ryan”, donde se destacó por la construcción de aviones de relevancia histórica y técnica. Entre sus logros se encuentran cuatro diseños innovadores de V/STOL (acrónimo en inglés de Vertical/Short Take-Off and Landing, despegue y aterrizaje verticales/cortos), aunque su producción más exitosa fue la línea de drones no tripulados Ryan Firebee.

En este orden de ideas, estos drones, utilizados con el fin de ser blancos y vehículos aéreos no tripulados, representaron uno de los primeros drones propulsados a chorro y se convirtieron en uno de los modelos más empleados en la historia de la aviación no tripulada (Mendiola, 2018).

Revisión de Algoritmos de Detección y Seguimiento de Objetos con Redes Profundas para Videovigilancia Inteligente

Algorithms for detection and tracking objects with deep networks for intelligent video surveillance: A review:

El artículo aborda el uso creciente de redes neuronales profundas en problemas de visión por computadora, especialmente en el reconocimiento y seguimiento de personas mediante una red de cámaras. Se revisan los algoritmos principales para detectar y rastrear objetos basados en redes profundas, para establecer la arquitectura de un sistema de videovigilancia inteligente. Se concluye que los algoritmos de un solo paso son más rápidos, destacando SSD (Single Shot Multibox Detector), mientras que los algoritmos de seguimiento sin conexión ofrecen mayor precisión, siendo DeepSort el más eficiente (Sánchez et al., 2020).

Palabras clave: Videovigilancia inteligente, Redes neuronales profundas, Clasificación de objetos, Seguimiento de objetos.

Introducción: Se menciona la evolución de la videovigilancia desde los sistemas analógicos hasta los digitales con conexiones IP. La videovigilancia encuentra aplicaciones en seguridad, control del tráfico, prevención del delito, entre otros. Se destaca la necesidad de sistemas inteligentes de videovigilancia y se presenta el proceso de reconocimiento, incluyendo preprocesamiento, extracción de rasgos, clasificación y seguimiento de objetos (Sánchez et al., 2020).

Métodos o Metodología Computacional: Se discuten la detección y rastreo de objetos como elementos fundamentales en sistemas de reconocimiento. Se hace hincapié en la integración de algoritmos para resolver ambas tareas y se menciona la importancia del

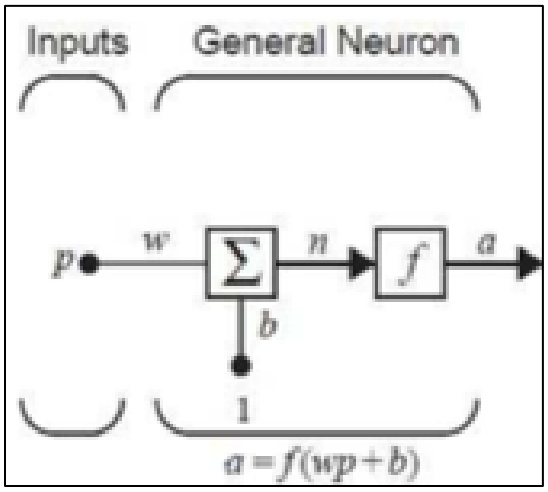
aprendizaje automático en este contexto. Se aborda la minería de datos y el aprendizaje automático supervisado, con énfasis en redes neuronales artificiales (Sánchez et al., 2020).

Aprendizaje Automático. Se explora la minería de datos y el aprendizaje automático supervisado, describiendo las variables predictoras y la variable objetivo. Se establece una distinción entre problemas supervisados y no supervisados. La investigación se ubica en el marco de los métodos de aprendizaje supervisado (Sánchez et al., 2020).

Redes Neuronales Artificiales. Se presenta un modelo simplificado de redes neuronales artificiales, destacando su capacidad para modelar problemas complejos y grandes de manera eficiente. Se señala su enfoque inductivo basado en datos y su capacidad para descubrir relaciones sin requerir especificaciones funcionales detalladas (Sánchez et al., 2020).

Figura 1

Red Neuronal de una Única Entrada y Salida



Fuente. Sánchez et al. (2020)

El siguiente proceso es simple. A la entra llega un escalar p que es multiplicado por su peso w, quedando $w * p$, que es una de las entradas de la sumatoria. La siguiente entrada siempre

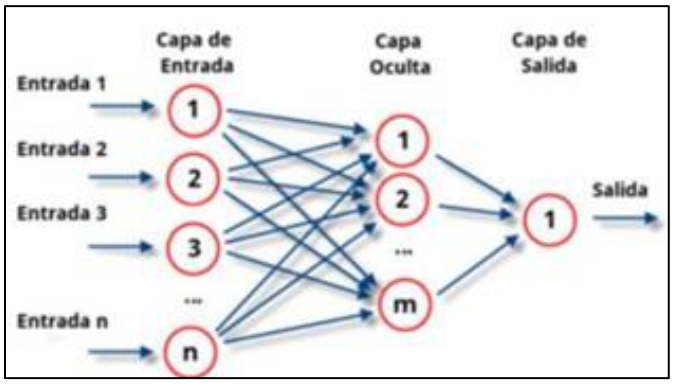
tiene valor 1 y es multiplicado por un bias. Una vez ambos se han sumado se obtiene la entrada n a la función de transferencia $w * p + b$. Finalmente se tiene la salida de la función a de la neurona:

En vista de esto, se observa que la salida del sistema depende de la función de transferencia que se tenga. Es importante conocer que la función de transferencia la decide el diseñador y que los parámetros w y b son ajustables y calculados mediante una regla de aprendizaje (Suárez, 2017).

Normalmente con una única neurona no bastará para resolver los problemas prácticos. Para resolver problemas más complejos se tendrá que hacer un uso conjunto de muchas de estas neuronas simples, dando lugar a una verdadera red neuronal, en la que se tendrán cientos o incluso miles de neuronas. Es ahí donde aparece el concepto de capa, que es la agrupación de todas estas neuronas en varios conjuntos dentro de la red neuronal completa (Suárez, 2017) como se muestra en la figura 2.

Figura 2

Esquema de una red Neuronal de Tres Capas



Fuente. Sánchez et al. (2020)

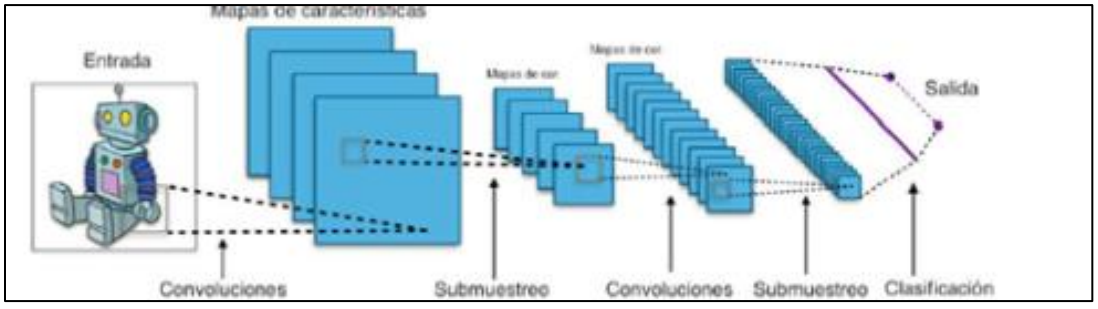
El aprendizaje es la clave de la adaptabilidad de la red neuronal y esencialmente es el proceso en el que se adaptan las sinapsis, para que la red responda de un modo distinto a los estímulos del medio. Entrenar una red neuronal modifica los pesos y vías asociados a cada neurona, para que la red pueda generar una salida con datos de entrada.

Las ANN (Artificial Neural Networks o Redes Neuronales Artificiales) son un método para resolver problemas, de forma individual o combinadas con otros métodos, en tareas de clasificación, identificación, diagnóstico, optimización o predicción . Estas redes son especialmente útiles cuando el balance entre datos y conocimiento se inclina hacia los datos y donde, adicionalmente, puede haber la necesidad de aprendizaje en tiempo de ejecución y de cierta tolerancia a fallos. En estos casos, las RNAs (Redes Neuronales Artificiales) se adaptan dinámicamente reajustando constantemente los pesos de sus interconexiones (Salas, 2020).

Redes Neuronales Convolucionales. Una red neuronal convolucional según (Suárez, 2017) es un tipo de red multicapa que consta de diversas capas convolucionales y de pooling (submuestreo) alternadas, y al final tiene una serie de capas full-connected como una red perceptrón multicapa, como se muestra en la figura 3.

Figura 3

Esquema de una Red Neuronal Convolucional



Fuente. Sánchez et al. (2020)

Redes Neuronales Convolucionales (CNN) en Videovigilancia. La entrada típica a una capa convolucional es una imagen de dimensiones $\mathbf{m} \times \mathbf{m} \times \mathbf{x} \times \mathbf{r}$, donde m es la altura y el ancho de la imagen, x es el tamaño y r es el número de canales. Las capas convolucionales utilizan l filtros (kernels) de dimensiones n × n × q, generando mapas de características de tamaño $(\mathbf{m} - \mathbf{n} + \mathbf{1}) \times (\mathbf{m} - \mathbf{n} + \mathbf{1}) \times \mathbf{q}$. Se realiza un submuestreo en capa de pooling (mean o max pooling) y se aplica una función de activación sigmoidal (Sánchez et al., 2020).

Redes Neuronales Convolucionales para Reconocimiento de Objetos. Las CNN están diseñadas asumiendo la entrada como imágenes, siendo poderosas para el análisis visual, detectando rasgos simples como líneas y bordes. Se abordan algoritmos de reconocimiento de objetos genéricos, destacando la detección de categorías naturales. Se enfoca en objetos estructurados y articulados (Sánchez et al., 2020).

Algoritmos de Reconocimiento de Objetos Basados en Redes Neuronales. Se examinan algoritmos como YOLO (You Only Look Once) que predice cuadros delimitadores y probabilidades de clase simultáneamente. YOLO es rápido, alcanzando hasta 150 fps, logrando eficiencia en tiempo real y doble precisión en comparación con otros sistemas. La arquitectura de YOLO se basa en GoogLeNet, con 24 capas convolucionales y 2 capas completamente conectadas. Existe una versión más pequeña para mayor velocidad (Sánchez et al., 2020).

Regression/Classification Based Framework (YOLO, SSD). Se mencionan marcos basados en propuestas regionales, incluyendo la generación de propuestas, extracción de características con CNN, clasificación y regresión de caja delimitadora. Se señala que estos marcos pueden ser un cuello de botella en aplicaciones de tiempo real (Sánchez et al., 2020).

Conclusiones del Articulo Científico. Los algoritmos one-stage, son considerablemente más rápidos que los basados en propuestas de regiones, aunque tienen dificultades para detectar

objetos pequeños, sobre todo si están agrupados. Entre los analizados SSD supera a YOLO en velocidad y precisión en todos los escenarios (incluyendo objetos pequeños). Serían más factibles en sistemas de vigilancia en entornos controlados como pasillos, cuartos, almacenes, bancos y parqueos (Sánchez et al., 2020).

Algoritmos como YOLO y SSD tienen más adaptabilidad; pueden ser usados con menos recursos computacionales (YOLO en su versión reducida y SSD utilizando Mobilenet como extractor de características), en estos casos, sacrificando precisión, pudiéndose ejecutar en dispositivos libres de GPU, como celulares de gama media baja y Rasperry Pi (Sánchez et al., 2020).

Algoritmos como Faster-RCNN (Region-based Convolutional Neural Networks) y R-FCN (Region-based Fully Convolutional Networks) son muy precisos y, dependiendo del extractor de características, pueden utilizarse para funciones de videovigilancia, especialmente en el preprocesamiento de video, debido a su capacidad de detectar objetos pequeños. Aunque no alcanzan velocidades que les permitan ser utilizados en sistemas en tiempo real, son muy eficientes detectando posibles amenazas en imágenes, como la identificación de objetos peligrosos como armas durante los escaneos de equipajes en los aeropuertos (Sánchez et al., 2020).

El aprendizaje en el paso de detección puede mejorar considerablemente el rendimiento de un algoritmo de rastreo. Las CNN son esenciales en la extracción de características: el uso de características de apariencia es fundamental para un buen rastreador y las CNN son muy eficaces en su extracción. Además, los rastreadores fuertes tienden a usarlos junto con características de movimiento, que pueden ser computadas usando LSTM, filtro Kalman (Sánchez et al., 2020).

Los algoritmos offline funcionan mejor que los algoritmos en línea; usan toda la secuencia de video, aunque en aplicaciones de videovigilancia es más factible usar los segundos, especialmente si el sistema funciona en tiempo real, dejando los algoritmos offline para tareas de preprocesamiento, como rastrear el comportamiento de un individuo en una secuencia de video previamente obtenida. Los algoritmos Sort y Deep Sort son los más eficientes en el rastreo de objetos, teniendo en cuenta el balance entre la precisión y la velocidad, por ejemplo, en el seguimiento de peatones y autos (Sánchez et al., 2020).

Se recomienda para el uso de un sistema de videovigilancia en tiempo real utilizar los algoritmos SSD con Mobilenet de base para la tarea de detección, y Deep Sort como rastreador; de esta forma se consigue un sistema balanceado entre la precisión y la velocidad con una adaptabilidad a distintos escenarios sin disponer de muchos recursos computacionales (Sánchez et al., 2020).

Marco Conceptual

DRON. Este es un dispositivo aéreo no tripulado que ha evolucionado desde sus inicios en el siglo XIX, siendo utilizado tanto en aplicaciones militares como civiles, gracias a su capacidad de vuelo autónomo o control remoto (Bestechnology Group, 2019).

Seguridad Aeroespacial. La seguridad aeroespacial se refiere a las medidas y prácticas destinadas a garantizar la protección integral de aeronaves, infraestructuras aeroportuarias y operaciones en el espacio aéreo, considerando amenazas y desafíos emergentes, como los asociados al uso de drones.

Vehículos Aéreos no Tripulados (UAV). Los Vehículos Aéreos no Tripulados, o drones, son aeronaves que operan sin la presencia de un piloto a bordo, controladas remotamente o de manera autónoma. Su diversidad de tamaños y usos los hace una herramienta versátil con

aplicaciones en diversos sectores. Tipos, incluye diversas categorías según su tamaño, alcance y capacidad, estos drones pueden ser monócopteros, bicópteros, cuadricópteros, octocópteros, etc.

Seguridad Aeroespacial. Área de estudio y práctica que se enfoca en garantizar la seguridad de las operaciones aéreas y la protección contra amenazas aeroespaciales.

Incluye medidas para prevenir acciones ilegítimas, accidentes y minimizar riesgos en el espacio aéreo.

Inteligencia Artificial (IA). Campo de la informática que se centra en el desarrollo de sistemas capaces de realizar tareas que requieren inteligencia humana, como el aprendizaje, la percepción y la toma de decisiones.

Aplicación: Utilización de modelos de entrenamiento de IA, como YOLOv8, para mejorar la detección y clasificación de drones en tiempo real.

YOLOv8 (You Only Look Once, version 8). Un Framework de detección de objetos en imágenes que se caracteriza por su eficiencia y velocidad.

Aplicación: Implementación en el proyecto para lograr una detección rápida y precisa de drones en diferentes entornos.

Fine Tuning. El “fine-tuning” (ajuste fino, en español) es un proceso en el aprendizaje automático (machine learning) que implica ajustar y optimizar un modelo pre entrenado en una tarea específica. En vez de entrenar uno desde cero, se parte de uno ya entrenado en un conjunto de datos más grande y general. Luego, este modelo pre entrenado se ajusta o “sintoniza finamente” para adaptarse a una tarea más específica o a un conjunto de datos más especializado.

El proceso de fine-tuning permite aprovechar el conocimiento general capturado durante el entrenamiento previo del modelo en datos diversos. Esto es útil cuando se dispone de conjuntos de datos más pequeños o específicos para una tarea en particular. Al ajustar ciertos

parámetros o capas del modelo pre entrenado, se busca optimizar su rendimiento para la tarea específica de interés.

Este enfoque se usa en el aprendizaje automático, incluyendo el procesamiento del lenguaje natural (NLP), la visión por computadora y otros dominios. Fine-tuning permite acelerar el proceso de entrenamiento y mejorar el rendimiento de un modelo en tareas específicas, capitalizando el conocimiento aprendido durante el entrenamiento inicial en conjuntos de datos más amplios.

Interfaz gráfica. Sistema visual que facilita la interacción entre usuarios y tecnologías complejas. Aplicación, desarrollo de una interfaz gráfica basada en un modelo de entrenamiento para acceder y comprender la información de detección de drones en tiempo real.

Métricas de Rendimiento. Parámetros utilizados para evaluar la eficacia y precisión del modelo y sistemas. Incluye métricas como precisión, recall, loss, Map50 y Map50-95 en el contexto del proyecto.

Espacio Aéreo Restringido. Áreas designadas donde el acceso y la operación de aeronaves, incluidos los drones, están limitados y regulados. Importancia, consideración crucial para la detección y control de drones, especialmente en entornos críticos como aeropuertos y eventos masivos.

Privacidad y Seguridad. Protección de información sensible y prevención de acciones que puedan poner en riesgo la seguridad de las personas y las instalaciones. Relevancia, inegración de tecnologías para evitar usos malintencionados de drones y garantizar la seguridad nacional.

Metodología

Método

Tipo de Estudio

Esta investigación se enmarca en la categoría de investigación aplicada, destacando su orientación práctica y su intención directa de abordar desafíos concretos en la detección de drones de ala rotatoria. A diferencia de investigaciones puramente teóricas, que buscan expandir el conocimiento en un área sin una aplicación inmediata, la presente indagación se centra en desarrollar soluciones prácticas y aplicables en entornos reales.

La necesidad de abordar la proliferación de drones y sus implicaciones en la seguridad aeroespacial ha llevado a la adopción de un enfoque aplicado. Se busca crear un modelo de entrenamiento basado en inteligencia artificial que no solo sea conceptualmente sólido, sino que también pueda implementarse de manera efectiva en situaciones del mundo real. Este enfoque práctico es esencial para garantizar que los resultados de la investigación no sean simplemente teóricos, sino que tengan un impacto tangible y directo en la mejora de la seguridad en áreas críticas.

La investigación aplicada también se destaca por su orientación hacia la resolución de problemas específicos y la transferencia de conocimiento a situaciones prácticas. Al utilizar tecnologías avanzadas como YOLOv8 y técnicas de fine tuning se busca no solo expandir el conocimiento en el campo de la detección de drones, sino también desarrollar una herramienta efectiva que pueda aplicarse en entornos aeroespaciales diversos.

La metodología de esta investigación involucra la experimentación práctica y pruebas con drones en vuelo, lo que refuerza su enfoque aplicado. La efectividad del modelo entrenado

se evaluará mediante métricas, asegurando que los resultados sean medibles y proporcionen una base sólida para futuras implementaciones.

Características,

Orientación Práctica: El proyecto se enfoca en abordar problemas específicos y aplicar soluciones prácticas en entornos reales.

Transferencia de Conocimiento: Se proveen los resultados técnicos de la investigación en s para que futuros investigadores los apliquen en situaciones reales.

Objetivos

Desarrollo de Soluciones Prácticas: El objetivo principal es desarrollar modelo de entrenamiento basado en inteligencia artificial para la detección de drones, con aplicaciones directas en la seguridad aeroespacial.

Impacto Tangible: Busca lograr resultados que medibles que beneficien la seguridad en áreas críticas y contribuyan al progreso tecnológico.

Aplicabilidad

Resolución de Problemas Actuales: Se centra en abordar desafíos específicos asociados con la proliferación de drones de ala rotatoria y los riesgos asociados.

Relevancia Práctica: Los resultados de la investigación tienen el potencial de tener un impacto directo en la seguridad aeroespacial y el control del espacio aéreo.

Implementación Práctica: Busca facilitar la aplicación de la tecnología desarrollada en entornos del mundo real, como aeropuertos y áreas críticas.

Uso Prolongado: La investigación tiene como objetivo la creación de una herramienta sostenible que pueda utilizarse de manera continua para abordar la problemática.

Evaluación del Impacto

Medición de Resultados: La efectividad del modelo se evalúa mediante métricas para determinar la precisión,

Contribución a la Seguridad Aérea: Se busca aportar en búsqueda de soluciones para la seguridad y protección de áreas sensibles.

Diseño Metodológico

Este estudio adopta un diseño de investigación aplicada, centrado en el desarrollo de una solución práctica para la detección de drones de ala rotatoria. La metodología se estructura en varias fases, que abarcan desde la revisión bibliográfica hasta la implementación y evaluación del modelo propuesto.

Para la metodología del procedimiento en programación, entrenamiento y otros aspectos, se emplearon los siguientes métodos:

Experimentación Práctica: Incluye la implementación y evaluación de un modelo de entrenamiento utilizando tecnologías como YOLOv8 y técnicas de fine tuning.

Pruebas en Entornos Reales: La investigación implica la aplicación y evaluación del modelo de entrenamiento en situaciones prácticas para validar su eficacia.

Metodología Cuantitativa para Desarrollo de un Sistema de Reconocimiento de Drones de Ala Rotatoria:

Definición de Requisitos: Para desarrollar y entrenar el modelo de detección de drones con YOLO v8, se usaron herramientas y bibliotecas especializadas en visión por ordenador y aprendizaje profundo. A continuación, se detallan las principales:

LabelImg: LabelImg es una herramienta de código abierto utilizada para el etiquetado de objetos en imágenes, lo que facilita la creación de conjuntos de datos de entrenamiento anotados

para tareas de detección de objetos. Se empleó para marcar y delimitar las ubicaciones de los drones en las imágenes de entrenamiento, proporcionando datos de entradas los cuales deben ser etiquetados para el modelo YOLO v8.

Python: Python se empleó como el lenguaje principal de programación debido a su facilidad de uso, su amplia adopción en el campo de la inteligencia artificial y su extensa colección de bibliotecas especializadas. Se utilizaron diversas bibliotecas de Python para tareas específicas, como el entrenamiento del modelo, la evaluación de su rendimiento y el diseño de la interfaz gráfica.

CUDA: CUDA (Compute Unified Device Architecture) de NVIDIA se aprovechó para la aceleración del entrenamiento del modelo en la GPU de nuestra propiedad. La capacidad de CUDA para ejecutar cálculos de manera paralela en la GPU permitió reducir significativamente el tiempo necesario para entrenar modelos complejos, como YOLO v8.

Ultralytics: Ultralytics es una biblioteca de código abierto desarrollada por la comunidad de inteligencia artificial, centrada en ofrecer soluciones avanzadas de detección de objetos y visión por computadora. Su principal contribución es la implementación del popular algoritmo YOLO (You Only Look Once) para la detección de objetos en imágenes y videos.

PyTorch: Se utilizó PyTorch como biblioteca de aprendizaje profundo para implementar YOLO v8. Esta es una biblioteca que ofrece herramientas poderosas y flexibles para implementaciones eficiente de algoritmos de aprendizaje profundo y el desarrollo de modelos de detección de objetos, así mismo Pytorch cuenta con la gran ventaja de estar completamente integrada dentro del ecosistema de Python lo que nos permitió desarrollar las actividades requeridas para este proyecto con una mayor facilidad de uso y rendimiento optimo.

OpenCV**:** OpenCV es una biblioteca de visión por computadora de código abierto, se utilizó para el preprocesamiento de imágenes, manipulación de datos y operaciones de bajo nivel, como la lectura de archivos de imagen, la conversión de formatos y la visualización de resultados.

Tkinter: Tkinter permite crear ventanas y widgets (elementos de la interfaz de usuario) de manera sencilla. Se utilizó para diseñar y organizar la disposición de la interfaz gráfica, incluyendo botones, etiquetas, cuadros de texto y otros elementos necesarios para interactuar con el modelo de detección de drones.

Preparación del conjunto de datos de entrenamiento: El proceso de preparación del conjunto de datos de entrenamiento para el modelo de detección de drones comprendió varias etapas clave, desde la recolección de imágenes hasta la distribución adecuada de los datos. A continuación, se describe detalladamente cada una de estas etapas:

Recolección de Imágenes de Drones: Se hizo una búsqueda en internet para recolectar imágenes de drones en diferentes entornos y situaciones. Se priorizó la variedad en condiciones de iluminación, fondos, ángulos de visión y poses de los drones, para mejorar la robustez y generalización del modelo.

Etiquetado de Imágenes con LabelImg: Cada imagen recolectada fue etiquetada manualmente utilizando el programa LabelImg. Se delinearon y etiquetaron con precisión los drones presentes en cada imagen, definiendo sus ubicaciones y contornos mediante cuadros delimitadores. Este proceso garantizó que el modelo de detección de drones recibiera datos de entrenamiento óptimos y correctamente etiquetados.

Distribución en Carpetas de Entrenamiento: Una vez etiquetadas, las imágenes se distribuyeron en tres carpetas principales: train, val y test, siguiendo una proporción adecuada

para el entrenamiento, la validación y la evaluación del modelo. Se asignó aproximadamente el 70% de las imágenes al conjunto de entrenamiento, el 15% al conjunto de validación y el 15% restante al conjunto de pruebas.

Creación del Archivo Dataset.yaml: Para facilitar la gestión y el uso del conjunto de datos de entrenamiento, se creó un archivo dataset.yaml. Este archivo incluyó información detallada sobre las rutas de archivo de las imágenes de los conjuntos de imágenes de entrenamiento, validación y pruebas. Esta estructura organizada permitió cargar fácilmente el conjunto de datos durante el entrenamiento del modelo.

Verificación y Control de Calidad: Se realizó una verificación de la calidad y la consistencia del conjunto de datos de entrenamiento para garantizar la precisión y la integridad de las etiquetas. Se corrigieron algunos errores de etiquetado, se eliminaron imágenes de baja calidad o irrelevantes, así como imágenes que pudieran afectar negativamente el rendimiento del modelo. El proceso de preparación del conjunto de datos de entrenamiento aseguró la disponibilidad de datos de alta calidad y variados para el entrenamiento del modelo de detección de drones, sentando así una base sólida para el desarrollo y la evaluación del modelo.

Entrenamiento del Modelo con YOLOv8: El proceso de entrenamiento del modelo de detección de drones se llevó a cabo utilizando el algoritmo YOLOv8, una versión mejorada y optimizada del popular modelo YOLO (You Only Look Once) para la detección de objetos en imágenes y videos. A continuación, se detalla el proceso de entrenamiento:

Configuración del Modelo: Se utilizó la implementación de YOLOv8 disponible en la biblioteca Ultralytics, configurando el modelo con parámetros apropiados para la detección de drones. Se ajustaron aspectos como el tamaño de la entrada de la imagen, el número de clases (en

este caso, la clase “drone”), y los hiper parámetros de entrenamiento para optimizar el rendimiento del modelo.

Inicialización de Pesos: Los pesos del modelo se inicializaron utilizando un enfoque de preentrenamiento con el conjunto de datos masivo COCO. Esta inicialización permitió al modelo comenzar con representaciones aprendidas previamente, acelerando el proceso de convergencia durante el entrenamiento en el conjunto de datos específicos de detección de drones.

Entrenamiento del Modelo: El modelo se entrenó utilizando el conjunto de datos de entrenamiento preparado previamente, que consistía en una variedad de imágenes etiquetadas con la presencia de drones. Se utilizó un algoritmo de optimización, como el descenso de gradiente estocástico (SGD), para ajustar los pesos del modelo iterativamente con el objetivo de minimizar una función de pérdida definida.

Evaluación del Rendimiento: Una vez finalizado el entrenamiento, se evaluó el rendimiento del modelo en un conjunto de datos de prueba independiente para medir su precisión y su capacidad para detectar drones en nuevas imágenes no vistas durante el entrenamiento. Se calcularon métricas de evaluación como la precisión, el recall y la puntuación F1 para cuantificar el rendimiento del modelo de detección de drones.

El proceso de entrenamiento del modelo con YOLOv8 permitió desarrollar un sistema de detección de drones eficiente, capaz de identificar drones con alta precisión en una variedad de entornos y condiciones.

Diseño de la Interfaz de Video: Como parte del desarrollo del proyecto, se diseñó una interfaz de video para visualizar y analizar la detección de drones en tiempo real. Esta interfaz se implementó utilizando Python y las bibliotecas OpenCV y Tkinter, aprovechando las

capacidades de procesamiento de imágenes y la facilidad de creación de interfaces gráficas de usuario de ambas bibliotecas. A continuación, se describe el proceso de diseño de la interfaz:

Captura de Video: Se utilizó la biblioteca OpenCV para capturar el video en tiempo real desde una cámara web o un archivo de video pregrabado. Esta funcionalidad permitió la entrada de video en tiempo real en la interfaz, sobre la cual se realizaría la detección de drones.

Integración del Modelo Detección de Drones: El modelo de detección de drones previamente entrenado se integró en la interfaz de video utilizando OpenCV. Los cuadros delimitadores de los drones detectados se superpusieron en el video en tiempo real, lo que permitió a los usuarios visualizar de manera interactiva las detecciones realizadas por el modelo.

Controles de Interfaz de Usuario: Se diseñaron controles de interfaz de usuario utilizando Tkinter para proporcionar funcionalidades adicionales a los usuarios.

Estos controles incluyeron opciones para seleccionar la fuente de origen del video (cámara o archivo de video pregrabado) botones para iniciar y detener la detección de drones, ajustar parámetros de detección.

Retroalimentación y Mejoras: Durante el desarrollo de la interfaz, se recopiló retroalimentación de los usuarios para identificar áreas de mejora y funcionalidades adicionales deseadas. Se realizaron ajustes iterativos en el diseño de la interfaz y las funcionalidades implementadas para satisfacer las necesidades y preferencias de los usuarios finales.

La interfaz de video diseñada proporcionó una herramienta fácil de usar para visualizar y analizar la detección de drones en tiempo real, mejorando así la capacidad de los usuarios para monitorear y responder a la presencia de drones en diferentes escenarios y aplicaciones.

Configuración de Ordenador y Resultados

Actividad 1

Configuración del entorno de trabajo para el entrenamiento del modelo de detección de objetos con un ordenador propio:

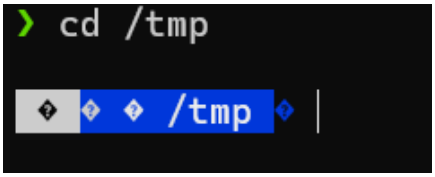
Instalar Anaconda

La mejor opción para instalar Anaconda es descargar la última secuencia de comandos bash del instalador de Anaconda, verificarlo y ejecutarlo.

Se busca la última versión de Anaconda para Python 3 en la página de descargas de Anaconda, a continuación, posiciónese en el directorio /tmp en su servidor. Este es un buen directorio para descargar elementos temporales, como la secuencia de comandos bash de Anaconda, que no necesitaremos después de la ejecución.

Figura 4

Comando para Ubicarse en el Directorio /TMP



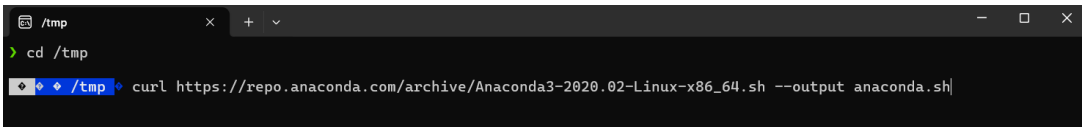
Nota. Pantallazos de como configurar el ordenador.

Instalación de Programas

Utilice **CURL** para descargar el enlace que copió desde el sitio web de Anaconda. Lo enviaremos a un archivo llamado anaconda.sh para un uso más rápido.

Figura 5

Comando Curl para Descargar Anaconda

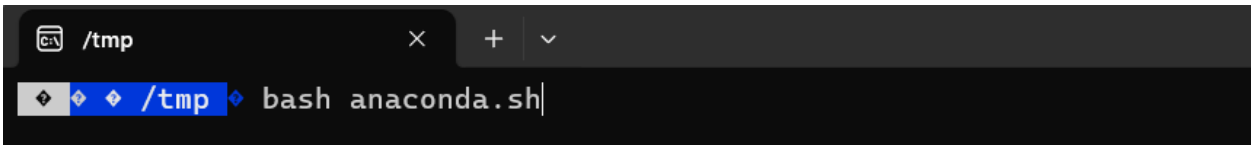


Nota. Pantallazos de como configurar el ordenador

Una vez descargado, procedemos a ejecutar la siguiente secuencia de comandos:

Figura 6

Comando para Iniciar la Instalación de Anaconda

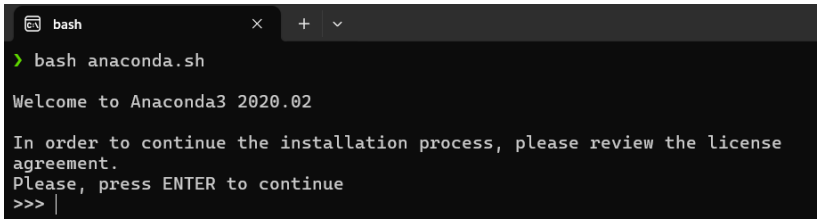


Nota. Pantallazos de como configurar el ordenador

Obteniendo el siguiente resultado:

Figura 7

Pantalla del Programa de Instalación de Anaconda

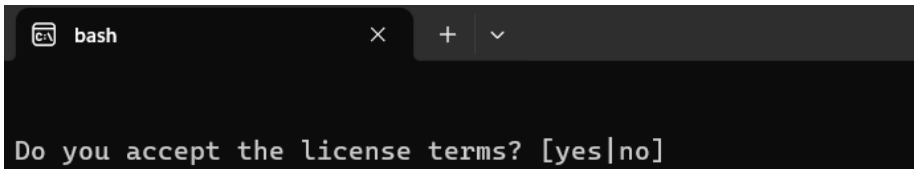


Nota. Pantallazos de como configurar el ordenador

Se presiona enter para continuar y luego se presiona nuevamente enter para leer la licencia hasta que obtengamos el siguiente mensaje:

Figura 8

Mensaje Aceptación Términos y Condiciones Anaconda

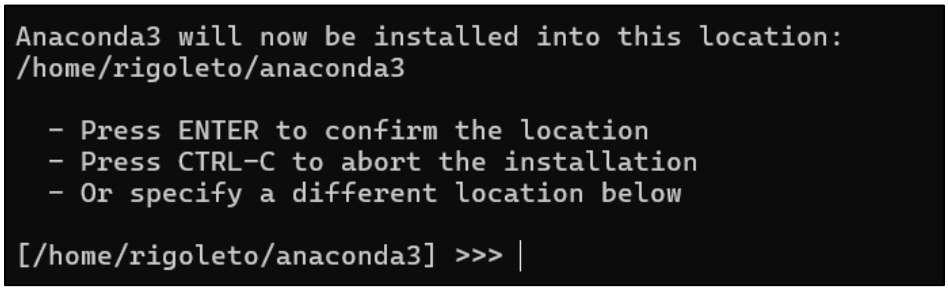


Nota. Escribir yes y presionar la tecla **enter**

Luego se nos solicitara una ubicación para instalar anaconda, podemos dejar la ubicación predeterminada dando **enter**

Figura 9

Confirmación del Directorio para Instalación de Anaconda

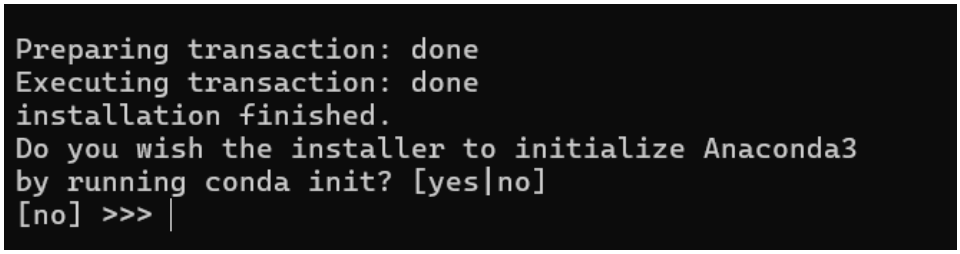


Nota. Pantallazos de como configurar el ordenador

La instalación tardará unos momentos en completarse y luego se recibirá un mensaje como este:

Figura 10

Mensaje de Confirmación para Inicializar Anaconda

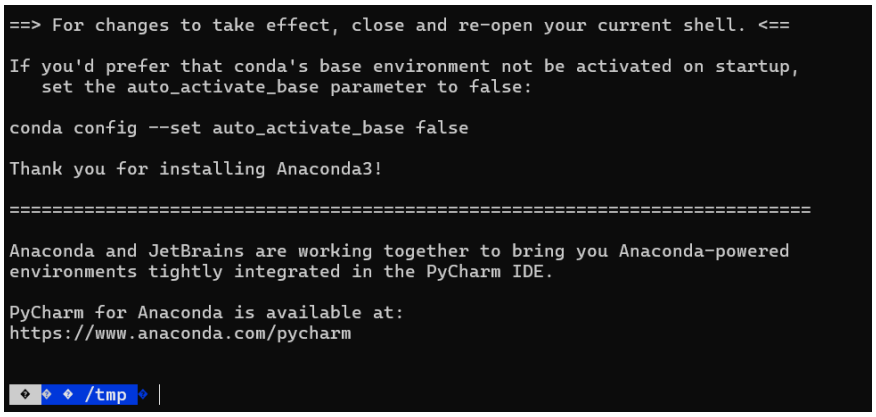


Nota. Pantallazos de como configurar el ordenador

Se procede a escribir yes para iniciar anaconda y posteriormente recibimos el siguiente mensaje que nos indica que la instalación fue terminada correctamente:

Figura 11

Mensaje Final de Instalación Correcta de Anaconda

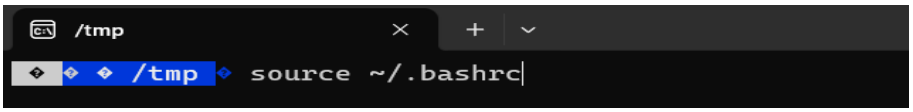


Nota. Pantallazos de como configurar el ordenador

Ahora se debe activar la instalación por medio del archivo ~/.bashrc

Figura 12

Comando para Activar la Instalación de Anaconda



Nota. Pantallazos de como configurar el ordenador

Una vez hecho esto, estaremos ubicados en el entorno base de anaconda.

Figura 13

Correcta Activación del Entorno base en Anaconda

Nota. Pantallazos de como configurar el ordenador

Aunque anaconda provee un entorno base de programación, es una buena práctica crear un entorno personalizado para cada uno de nuestros proyectos, a continuación, vamos a crear un entorno con Python 3.8 para el desarrollo de nuestro proyecto.

Creación de un Entorno con Anaconda y Python

Lo primero que se realiza es verificar las versiones de Python disponibles para el entorno virtual, esto lo haremos ejecutando el siguiente comando:

Figura 14

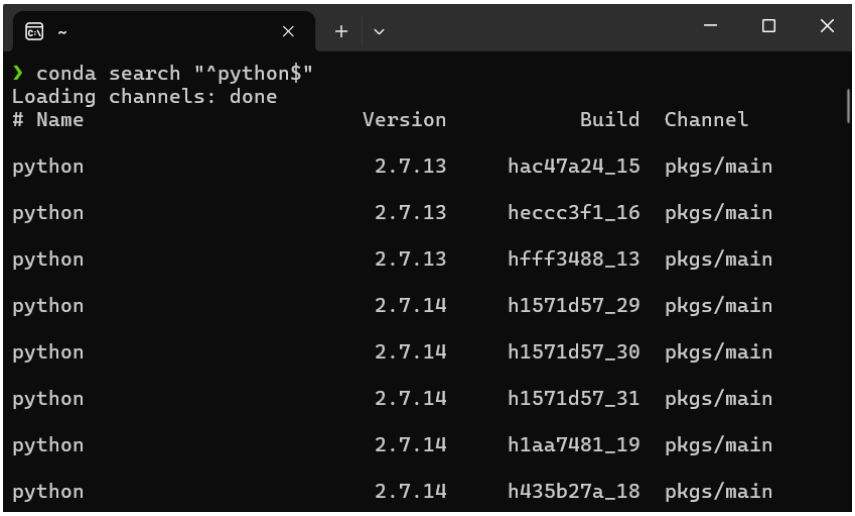
Comando de Búsqueda de las Versiones Disponibles de Python

Nota. Pantallazos de como configurar el ordenador

Esto nos mostrara la lista de versiones de Python 2 y 3 disponibles para el entorno virtual, para este proyecto utilizaremos Python 3.8 para evitar errores de compatibilidad con las demás herramientas que usar posteriormente.

Figura 15

Lista de Versiones Disponibles de Python

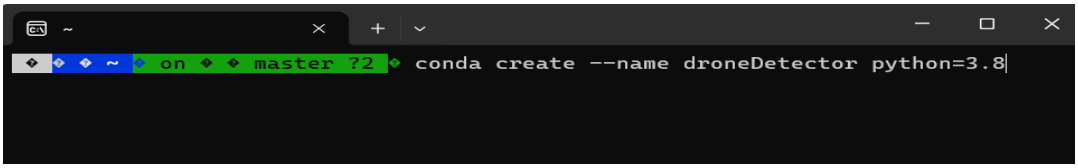


Nota. Se selecciona la mejor versión

Se crea un entorno utilizando la versión de Python 3.8, para esto se asigna la versión 3.8 al argumento python. Llamaremos al entorno droneDetector, para esto ejecutaremos el siguiente comando.

Figura 16

Comando para Crear el Entorno Virtual droneDetector con Python 3.8

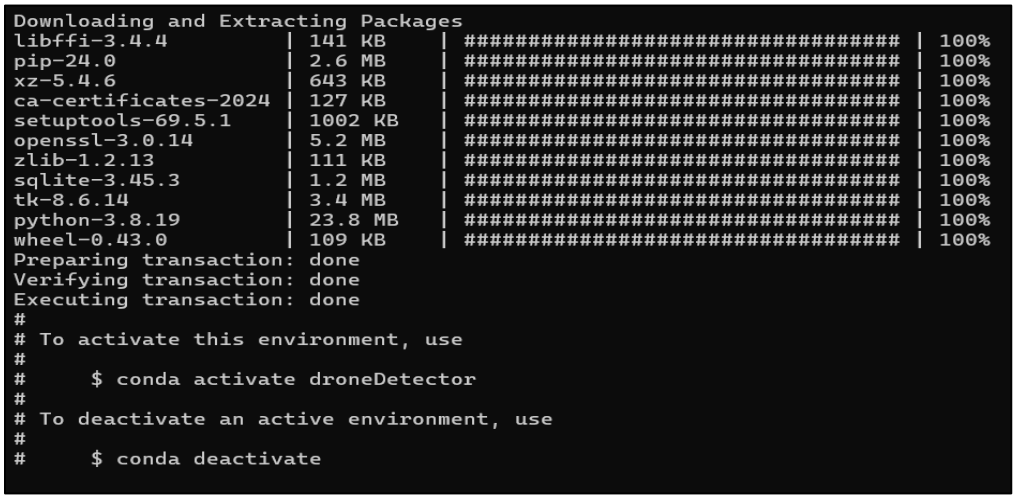


Nota. Pantallazos de como configurar el ordenador

Durante el proceso de instalación se nos pedirá digitar la letra y para continuar con el proceso, una vez terminado el proceso de descarga e instalación veremos el siguiente mensaje de confirmación de la creación del entorno y las instrucciones para activarlo o desactivarlo

Figura 17

Mensaje de Confirmación de Creación del Entorno Virtual droneDetector

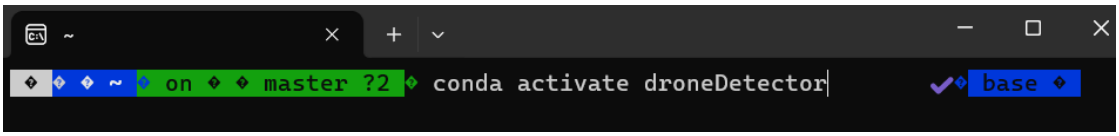


Nota. Pantallazos de como configurar el ordenador

Para activar el nuevo entorno procedemos a escribir el siguiente comando.

Figura 18

Comando para Activar el Entorno Virtual droneDetector



Nota. Pantallazos de como configurar el ordenador

Si la activación del entorno virtual fue exitosa, se debería ver el nombre de nuestro entorno virtual en la terminal de comandos en lugar del entorno base.

Figura 19

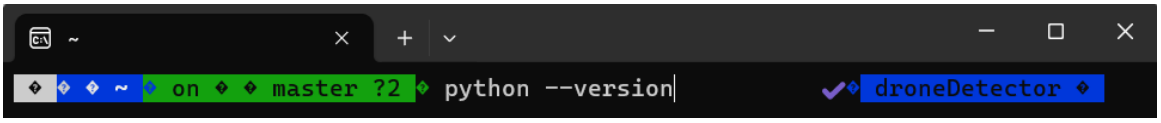
Activación Exitosa del Entorno Virtual droneDetector

Nota. Pantallazos de como configurar el ordenador

Al final, se debe asegurar que se trabaje en la versión correcta de Python con el comando:

Figura 20

Comando para Verificar la Versión de Python



Nota. Pantallazos de como configurar el ordenador

El resultado será la versión de Python instalada en el entorno virtual “droneDetector”

Figura 21

Versión de Python en el Entorno Virtual droneDetector

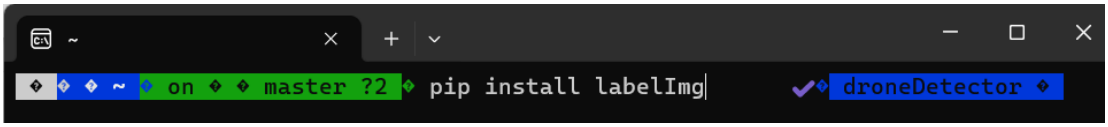
Nota. Pantallazos de como configurar el ordenador

Instalar labelImg

Ahora se debe instalar labelImg que nos ayudara en el proceso de etiquetado de las imágenes de nuestro dataset, para esto ejecutaremos el siguiente comando:

Figura 22

Comando para Instalar LabelImg

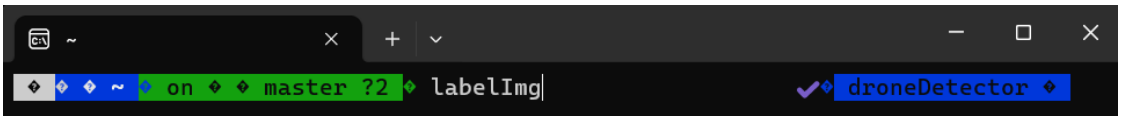


Nota. Pantallazos de como configurar el ordenador

Una vez instalado labelImg, podemos ejecutarlo mediante el siguiente comando

Figura 23

Comando para Ejecutar Labelimg



Nota. Pantallazos de como configurar el ordenador

Esto ejecutara la interfaz gráfica de labelImg, en la cual podremos cargar la carpeta que contiene las imágenes de nuestro dataset y empezar con el proceso de etiquetado de las imágenes

Figura 24

Interfaz Gráfica del Programa LabelImg

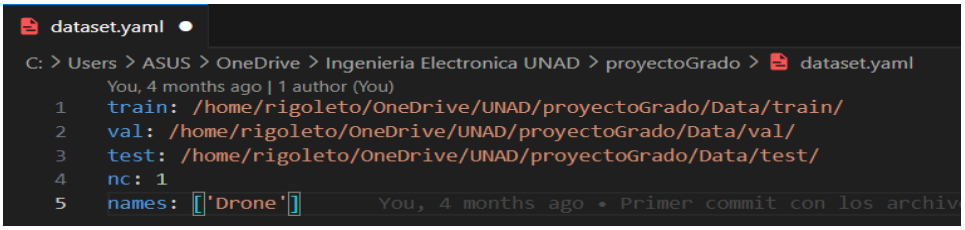
Nota. Entorno del programa LabelImg

Configuración del Archivo Dataset.yaml

Una vez que se ha realizado la tarea de etiquetado de las imágenes y se han separado las imágenes en las carpetas **Train**, val y test, es necesario crear el archivo dataset.yaml que es donde vamos a indicarle a **Yolo** las rutas de las imágenes que se usaran para entrenamiento, evaluación y test, así como los nombres de las etiquetas creadas en el proceso de etiquetado de las imágenes, para nuestro caso únicamente existe la etiqueta “Drone”.

Figura 25

Configuración del Archivo Dataset.yaml



Nota. Pantallazos de como configurar el ordenador

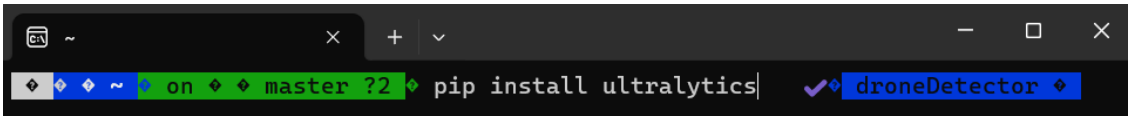
La anterior es la configuración del archivo dataset.yaml para el proyecto.

Instalación de la Librería Ultralytics

Para la instalación de la librería ultralytics que es necesaria para el entrenamiento del modelo con Yolo, vamos a ejecutar el siguiente comando.

Figura 26

Comando para Instalar la Librería Ultralytics



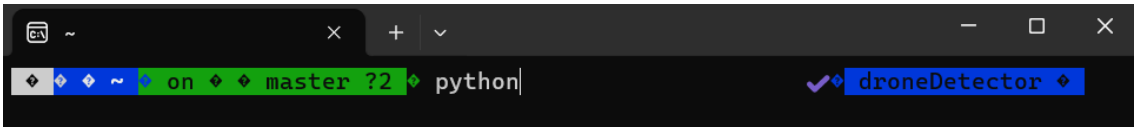
Nota. Verificar si Torch se está utilizando en la tarjeta gráfica

Para verificar si Torch está utilizando la tarjeta gráfica de la computadora debemos seguir los siguientes pasos:

Ingresar a Python con el siguiente comando

Figura 27

Comando para Iniciar Python en la Terminal

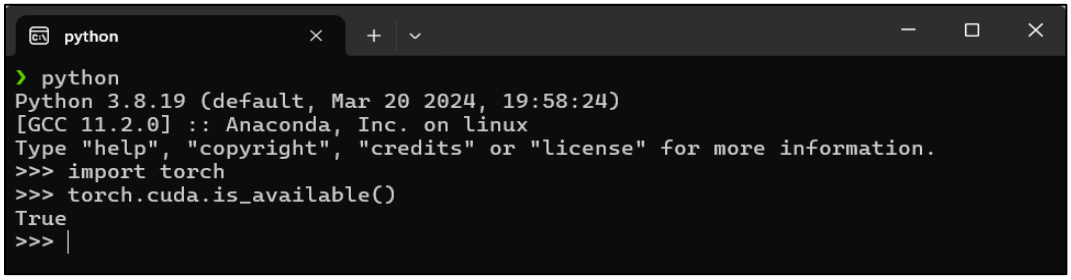


Nota. Pantallazos de como configurar el ordenador

Una vez dentro de Python se ejecutan las siguientes líneas para importar torch y luego preguntar si CUDA está disponible.

Figura 28

Comandos para Verificar si la Tarjeta Gráfica está Disponible con CUDA



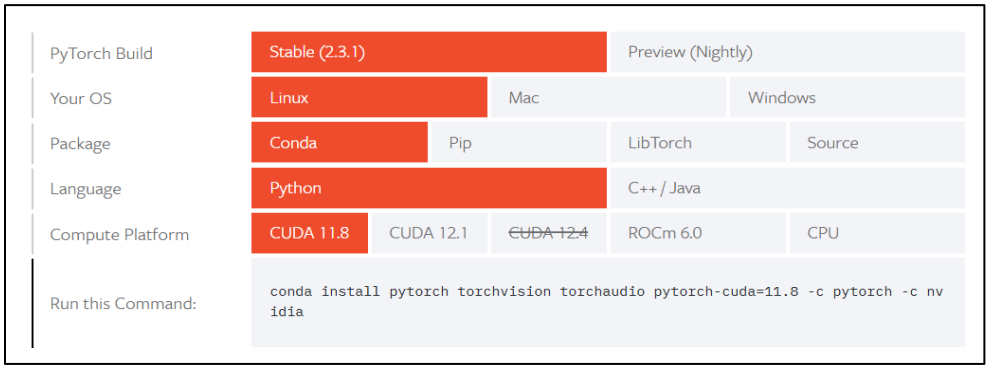
Nota. Pantallazos de como configurar el ordenador

En este caso, la respuesta es “True”, por lo cual ahora estamos seguros de que la tarjeta gráfica está disponible para ser usada en nuestro entrenamiento.

En caso de que la respuesta sea “False”, es necesario que nos dirijamos a la página https://pytorch.org/get-started/locally/ y allí seleccionemos nuestra configuración para que nos indiquen cual es el comando adecuado para instalar pytorch y poder disponer de la tarjeta gráfica.

Figura 29

Opciones para la Descarga de PyTorch



Nota. Pantallazos de como configurar el ordenador teniendo en cuenta la versión

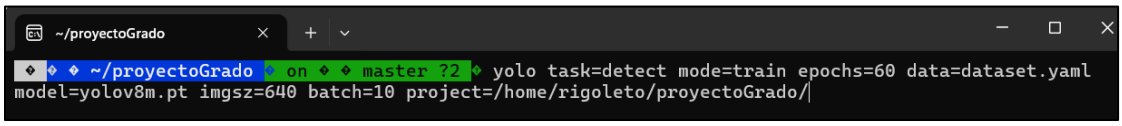
La figura 29 corresponde a la configuración que se realizará en este proyecto, por lo cual, luego de correr el comando indicado en la terminal de Linux e instalar pytorch, se debe volver a ingresar a Python, importar torch y preguntar si cuda está disponible, como se realizó en el paso anterior.

Entrenar el Modelo con Yolo

Finalmente podemos entrenar el modelo de detección con **Yolo**, para esto debemos estar ubicados en la carpeta de nuestro proyecto y luego ejecutar el siguiente comando:

Figura 30

Comando para Ejecutar el Entrenamiento con Yolo



Nota. Pantallazos de como configurar el ordenador

Explicación del comando anterior:

Yolo: Este es el comando principal que llama a la herramienta YOLO (You Only Look Once) para la detección de objetos.

task=detect: Especifica que la tarea que se va a realizar es la detección de objetos. YOLO puede ser utilizado para varias tareas, como detección (detect), segmentación (segment), entre otras.

mode=train: Indica que el modo de operación es el entrenamiento del modelo. Otros modos pueden incluir predict para predicción y val para validación.

epochs=60: Define el número de épocas para el entrenamiento. Una época es un ciclo completo a través del conjunto de datos de entrenamiento. Aquí se especifica que el modelo entrenará durante 60 ciclos.

data=dataset.yaml: Proporciona la ruta al archivo de configuración de datos (dataset.yaml). Este archivo contiene información sobre el conjunto de datos, como la ubicación de los archivos de entrenamiento, validación y las clases de los objetos.

model=yolov8m.pt: Especifica el modelo pre entrenado que se va a utilizar como punto de partida para el entrenamiento. En este caso, se está utilizando el modelo yolov8m.pt, que es una versión específica de YOLOv8.

imgsz=640: Define el tamaño de las imágenes de entrada al modelo. Aquí, las imágenes se redimensionan a 640x640 píxeles antes de ser procesadas por el modelo.

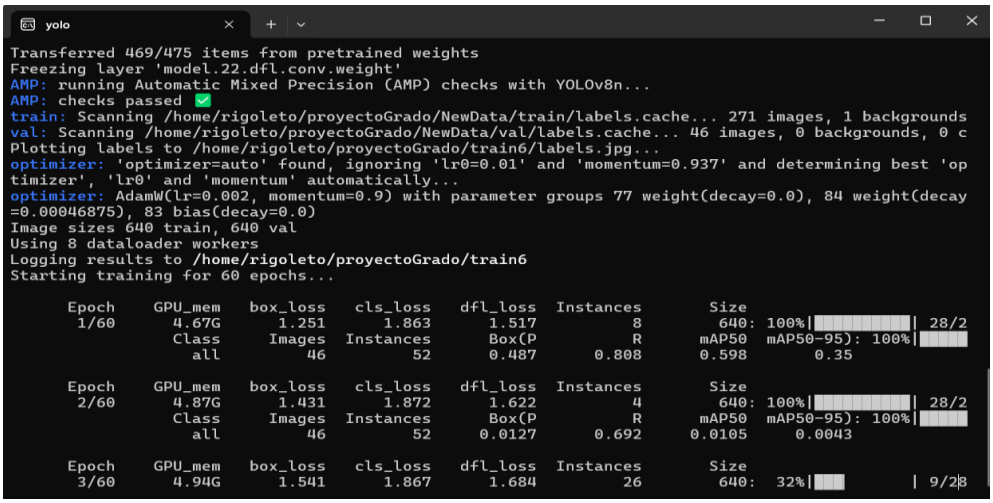
batch=10: Establece el tamaño del lote de datos que se procesará en cada iteración del entrenamiento. Un lote es un subconjunto de datos que se utiliza para actualizar los pesos del modelo. Aquí, cada lote contiene 10 imágenes.

project=/home/rigoleto/proyectoGrado/: Indica la ruta del directorio donde se guardarán los resultados del entrenamiento, como los pesos del modelo entrenado, gráficos de entrenamiento, y registros. En este caso, se guardarán en /home/rigoleto/proyectoGrado/

Si el proceso de entrenamiento se ejecuta sin problemas, deberíamos ver en la pantalla lo siguiente:

Figura 31

Ejecución de Yolo de Forma Correcta

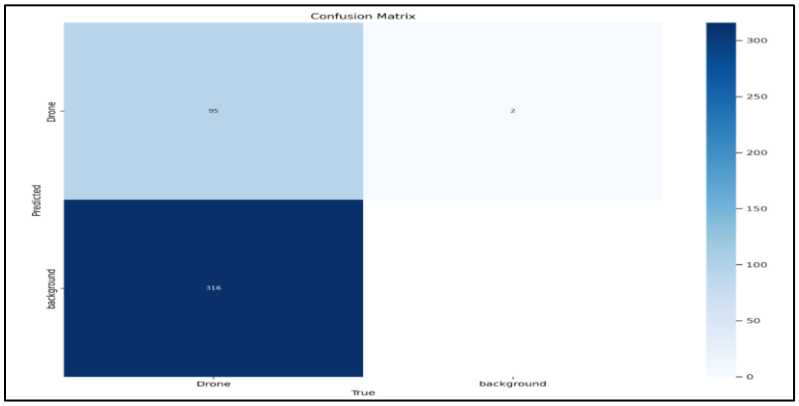


Nota. Pantallazo ejecución modelo entrenamiento Yolo correcto

Cuando finaliza el proceso de entrenamiento el programa automáticamente genera un archivo de tendencias graficas para cada una de las variables que usa para lograr el objetivo, se evidencia como es el comportamiento a través de las épocas y el resultado. Archivo ejecutable para visualización: **Confusion Matrix**

Figura 32

Confusion Matrix



Nota. Matriz de confusión representada en una gama de colores en el etiquetado del modelo.

Para analizar la matriz de confusión del modelo, se analizan los elementos que se encuentran en la gráfica:

True Positives (TP): Drones correctamente clasificados como “Drone”.

False Positives (FP): Backgrounds incorrectamente clasificados como “Drone”.

True Negatives (TN): Backgrounds correctamente clasificados como “background”.

False Negatives (FN): Drones incorrectamente clasificados como “background”.

En la matriz de confusión:

True Positives (TP): 95 (Drones predichos correctamente como “Drone”).

False Positives (FP): 2 (Backgrounds predichos incorrectamente como “Drone”).

True Negatives (TN): No hay datos en este caso (cero Backgrounds predichos correctamente como “background”).

False Negatives (FN): 316 (Drones predichos incorrectamente como “background”).

Métricas derivadas de la matriz de confusión

Precisión. La precisión se define como la proporción de verdaderos positivos entre el total de positivos predichos (la suma de verdaderos positivos y falsos positivos).

$$\text{Precisión} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{95}{95 + 2} = \frac{95}{97} \approx 0.979 \approx 97.9\%$$

Exhaustividad (Recall o Sensibilidad). La exhaustividad se define como la proporción de verdaderos positivos entre el total de positivos reales (la suma de verdaderos positivos y falsos negativos).

$$\text{Exhaustividad } \frac{\text{TP}}{\text{FN+TP}} = \frac{95}{316+95} = \frac{95}{411} \approx 0.231 \approx 23.1\%$$

Exactitud (Accuracy). La exactitud se define como la proporción de todas las predicciones correctas (tanto verdaderos positivos como verdaderos negativos) entre el total de casos evaluados.

$$\text{Exactitud}=\frac{\text{TP+TN}}{(\text{TP+TN+FP+FN})}=\frac{95+0}{95+0+2+316}=\frac{95}{413}\approx 0.23\approx 23\%$$

F1 Score. El F1 Score es la media armónica de la precisión y la exhaustividad, proporcionando un balance entre las dos métricas.

$$\text{F1 Score}=2\cdot\frac{\text{Precisión}*\text{Exhaustividad}}{\text{Precisión}+\text{Exhaustividad}}\approx 2\cdot\frac{0.979*0.231}{0.979+0.231}\approx 0.373\approx 37.3\%$$

La precisión del modelo es alta, lo que significa que cuando predice que algo es un dron, generalmente es correcto.

La exhaustividad es baja, indicando que el modelo no está capturando una gran cantidad de drones que realmente están presentes (muchos falsos negativos).

La exactitud también es baja debido a la gran cantidad de falsos negativos.

El F1 Score es relativamente bajo, reflejando el desequilibrio entre la precisión y la exhaustividad.

La “confusion_matrix.png” proporciona una visualización clara de cómo el modelo está clasificando las diferentes clases y si hay algún patrón de errores comunes, como la confusión entre clases similares. Esto permite una evaluación más detallada del rendimiento del modelo y puede ayudar a identificar áreas de mejora para futuras iteraciones de entrenamiento o ajuste del modelo. En la matriz de confusión, las filas representan las clases reales, mientras que las columnas representan las clases predichas por el modelo. Cada celda de la matriz muestra el recuento de instancias que pertenecen a una clase específica según las predicciones del modelo.

Se entrega, además un archivo “labels.jpg” que describe cuadros azules claros y oscuros, junto con ejes “width” (ancho) y “height” (alto), probablemente es una representación visual de las etiquetas o anotaciones utilizadas durante el entrenamiento del modelo.

Durante el proceso del entrenamiento se utilizó un conjunto de datos de 996 imágenes, 189 imágenes para validación y 144 de prueba, el modelo ahora está listo para su

implementación en el mundo real. Durante el entrenamiento, el modelo ha aprendido a reconocer y clasificar una amplia variedad de objetos, desde múltiples perspectivas y en diversas condiciones ambientales.

Para ejecutar el modelo con éxito, se lleva a cabo un proceso cuidadoso que implica la preparación y el procesamiento de los datos de entrada, la carga del modelo entrenado en la memoria del sistema y la ejecución de la inferencia en las imágenes de prueba o en tiempo real capturadas por el dron.

Durante la ejecución del modelo, se implementan técnicas avanzadas de procesamiento de imágenes y algoritmos de detección para identificar objetos de interés en el campo de visión del dron. Estos objetos pueden ser personas, vehículos, edificios u otros elementos relevantes para la tarea de reconocimiento.

La correcta ejecución del modelo implica no solo la detección precisa de objetos, sino también la toma de decisiones en tiempo real basadas en la información obtenida. Esto puede incluir el seguimiento de objetos en movimiento, la estimación de trayectorias futuras y la generación de alertas o notificaciones según sea necesario. Tras completar el entrenamiento en YOLOv8, el sistema genera archivos importantes fundamentales para implementar y usar el modelo entrenado. Algunos de los archivos típicos que YOLOv8 puede generar son:

1. Pesos del modelo: (weights) Este archivo contiene los pesos aprendidos por la red neuronal durante el entrenamiento. Es esencial para realizar inferencias con el modelo entrenado.

En el contexto de YOLOv8 y otros modelos entrenados con técnicas de aprendizaje profundo, como redes neuronales convolucionales (CNN), los archivos “best.pt” y “last.pt” generalmente se refieren a los pesos del modelo en diferentes etapas del proceso de entrenamiento.

best.pt (Mejor peso del modelo): Este archivo contiene los pesos del modelo en el punto en el que el modelo obtuvo el mejor rendimiento en un conjunto de datos de validación o evaluación durante el entrenamiento. Durante el entrenamiento, el modelo se evalúa periódicamente en un conjunto de datos de validación para controlar su rendimiento y evitar el sobreajuste. El “best.pt” representa los pesos del modelo en el punto donde se alcanzó la mejor métrica de evaluación, como la precisión o la pérdida mínima en el conjunto de validación.

last.pt (Último peso del modelo): Este archivo contiene los pesos del modelo en la última iteración del entrenamiento, es decir, después de que se completó el número predefinido de iteraciones o épocas. A menudo, se usa como punto de partida para continuar el entrenamiento si es necesario, o como punto final si el modelo ha alcanzado la convergencia y no se planea realizar más iteraciones de entrenamiento. Un archivo PT es un proyecto de aplicación creado con el marco Panther para entornos de desarrollo basados en componentes. Contiene información sobre guiones, fuentes de medios y metadatos.

En la carpeta weights encontramos los archivos del modelo entrenado

Figura 33

Archivos del Modelo Entrenado

Nota. Archivos ejecutables generados una vez finaliza entrenamiento.

En la anterior figura se evidencian los dos archivos ejecutables del modelo creado una finalizó el entrenamiento a través de las diferentes épocas de YOLOV8.

2. Configuración del modelo: Este archivo describe la arquitectura y la configuración del modelo de YOLOv8 utilizado durante el entrenamiento. Incluye información sobre la estructura de la red neuronal, como el número de capas, filtros, tamaños de entrada y otras configuraciones específicas.

3. Archivos de clases: Estos archivos enumeran las clases o categorías de objetos que el modelo ha sido entrenado para reconocer. Cada clase tiene un identificador único que se utiliza durante la inferencia para etiquetar los objetos detectados.

4. Configuración de hiper parámetros: Este archivo puede contener información sobre los hiper parámetros utilizados durante el entrenamiento, como la tasa de aprendizaje, la función de pérdida, el tamaño del lote, entre otros. Estos hiper parámetros son importantes para comprender cómo se entrenó el modelo y pueden ajustarse durante la fase de ajuste fino o afinación del modelo.

5. Archivos de registro y métricas: Estos archivos contienen información sobre el progreso del entrenamiento, incluidas las métricas de rendimiento como la precisión, la pérdida y otras estadísticas relevantes. Son útiles para evaluar la calidad del modelo entrenado y realizar ajustes adicionales si es necesario.

El entrenamiento de modelos de detección de objetos, como YOLOv8, es un proceso clave en el desarrollo de sistemas de visión son capaces de identificar y localizar objetos en imágenes. Los resultados de este entrenamiento se registran en una serie de métricas y pérdidas que proporcionan información crucial sobre el rendimiento y la capacidad de generalización del modelo. En este análisis, exploraremos en detalle los componentes de los resultados del entrenamiento de YOLOv8, desde pérdidas específicas hasta métricas que evalúan la precisión y eficacia del modelo.

El entrenamiento genera un informe ejecutable con extensión .CSV el cual nos muestran los datos en forma de tabla lo cual facilita el cual evaluar los siguientes resultados:

EPOCH: Número de época o iteración durante el entrenamiento. Cada época representa un ciclo completo a través de todo el conjunto de datos de entrenamiento.

TRAIN/BOX_LOSS: Pérdida (loss) asociada a la regresión de coordenadas de las cajas delimitadoras (bounding boxes) durante el entrenamiento.

TRAIN/CLS_LOSS: Pérdida asociada a la clasificación de objetos durante el entrenamiento.

TRAIN/DFL_LOSS: Pérdida asociada a la regresión de características de puntos de referencia (landmarks) de las cajas delimitadoras durante el entrenamiento.

METRICS/PRECISION(B): Precisión del modelo en el conjunto de entrenamiento, específicamente para objetos grandes (B).

METRICS/RECALL(B): Recall del modelo en el conjunto de entrenamiento, específicamente para objetos grandes (B).

METRICS/MAP50(B): Media del Área Bajo la Curva de Precisión (mAP) a un umbral del 50% para objetos grandes (B). Es una métrica comúnmente utilizada en problemas de detección de objetos.

METRICS/MAP50-95(B): mAP en un rango de umbrales del 50% al 95% para objetos grandes (B).

VAL/BOX_LOSS: Pérdida en el conjunto de validación asociada a la regresión de coordenadas de las cajas delimitadoras.

VAL/CLS_LOSS: Pérdida en el conjunto de validación asociada a la clasificación de objetos.

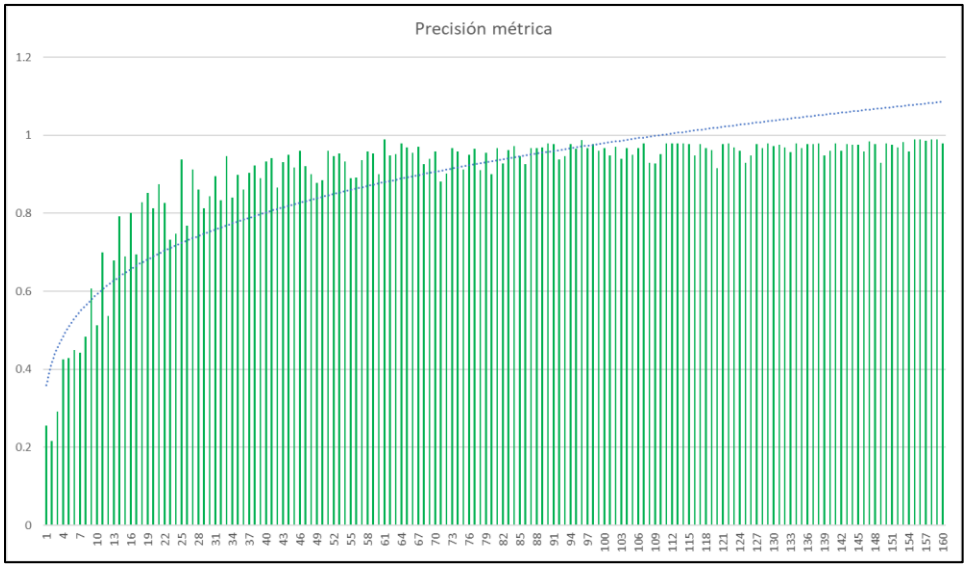
VAL/DFL_LOSS: Pérdida en el conjunto de validación asociada a la regresión de características de puntos de referencia de las cajas delimitadoras.

LR/PG0, LR/PG1, LR/PG2: Tasas de aprendizaje para los diferentes grupos de parámetros (pg0, pg1, pg2).

Es común utilizar tasas de aprendizaje diferenciadas para distintos conjuntos de parámetros dentro del modelo. El entrenamiento del modelo se realiza en 160 épocas o ciclos donde se realiza el aprendizaje del modelo, una vez termina el entrenamiento, el programa suministra un archivo ejecutable en formato .CSV Para representar datos en forma de tabla, en las que las columnas se separan por comas y las filas por saltos de línea. Donde se analizan los datos por medio de tablas dinámicas, gráficos dinámicos dependiendo del análisis que se requiera a continuación, se muestra los procesos para cada una de las épocas y el comportamiento del entrenamiento durante el aprendizaje.

Figura 34

Precisión Métrica



Nota. Autoría Propia.

En la Figura 34, se contrastan los resultados de la métrica de precisión durante el entrenamiento del modelo con respecto a la precisión del proceso. En las etapas iniciales, se observa un valor que no supera el 40%. No obstante, al continuar refinando el modelo en las subsiguientes épocas, la precisión experimenta un notable aumento, alcanzando aproximadamente un 98%. Estos valores resultan significativos para las evaluaciones de campo necesarias.

Teniendo en cuenta los datos

Tendencia General: Se observa una variabilidad en la precisión a lo largo de las épocas. Es crucial considerar si hay una tendencia general de mejora o estabilización.

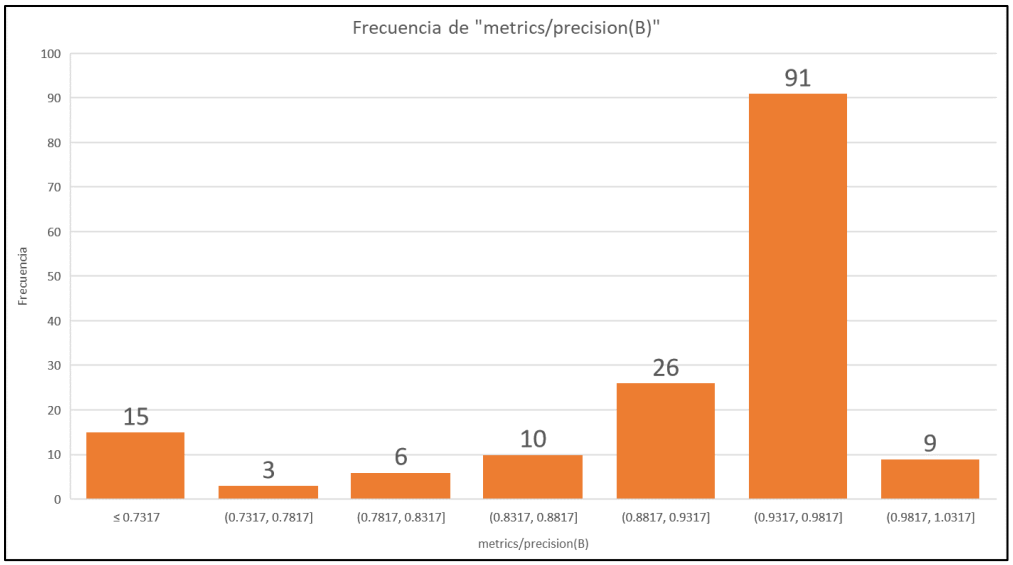
Descenso Inicial y Estabilización: En las primeras épocas, la precisión puede variar significativamente. Luego, a medida que avanzan las épocas, parece haber una estabilización y una mejora general en la precisión.

Picos y Valles: Se logran observar si hay picos o valles notables en la precisión. Estos puntos pueden indican momentos críticos durante el entrenamiento, como la introducción de nuevas características o ajustes en los hiper parámetros.

Estabilidad a Largo Plazo: A medida que avanzan las épocas, verifica si la precisión se mantiene relativamente estable o sigue mejorando. La estabilidad a largo plazo es esencial para garantizar que el modelo generalice bien a datos nuevos.

Figura 35

Frecuencia de Precisión Métrica



Nota. Autoría Propia.

En la Figura 35. Se realiza un análisis estadístico por grupos de la cantidad de datos de la precisión métrica, donde en el grupo con mayor incidencia en la precisión de datos está en el rango de 0.9317 a 0.9817 los cuales representan alta confiabilidad en los datos suministrados.

Tabla 1

Resumen Estadístico de Datos Precisión

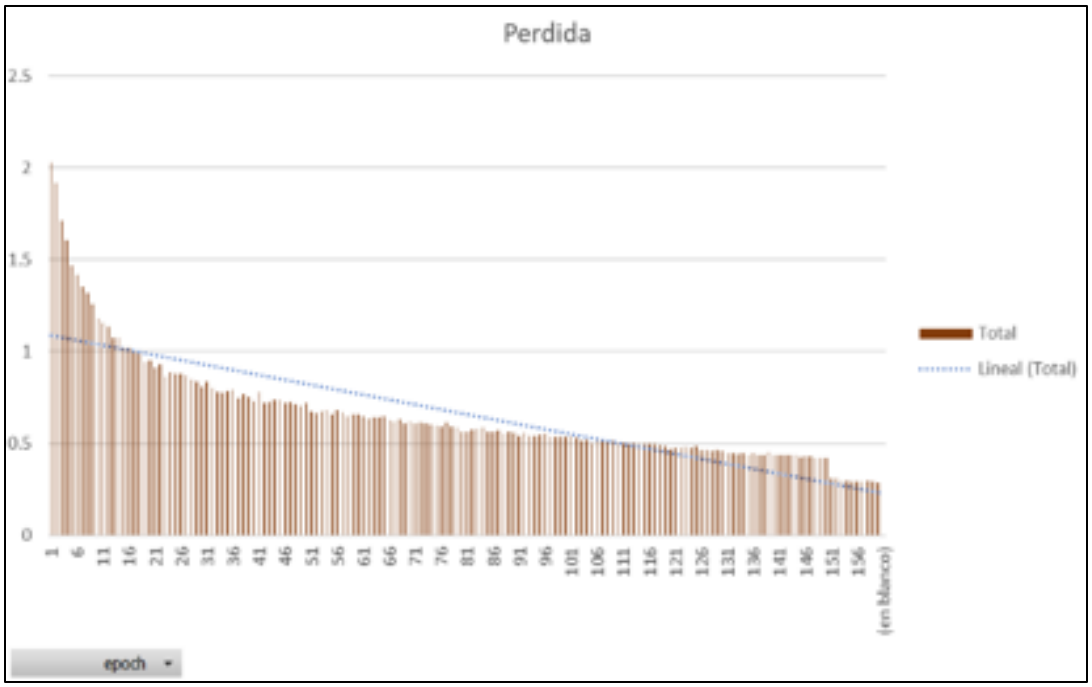
Precisión	
Media	0,89591013
Error típico	0,01159639
Mediana	0,9502
Moda	0,97811
Desviación estándar	0,14668401
Varianza de la muestra	0,0215162
Curtosis	795,117342
Coeficiente de asimetría	-281,399894
Rango	0,77427
Mínimo	0,21541
Máximo	0,98968
Suma	14,334562
Cuenta	160
Nivel de confianza (95.0%)	0,02290282

Nota. Autoría Propia.

La métrica de precisión en el contexto de un modelo de detección de objetos refleja la habilidad del modelo para realizar detecciones precisas. Según los resultados estadísticos, el modelo exhibe una precisión promedio del 89.6%, lo que indica que, en general, alrededor del 89.6% de las detecciones propuestas por el modelo son correctas.

Figura 36

Perdidas



Nota. Autoría Propia.

Descenso de la Pérdida de Clasificación: En las primeras épocas, la pérdida de clasificación es relativamente alta, indicando que el modelo inicialmente tiene dificultades para clasificar los objetos con precisión.

A medida que avanzan las épocas, se observa una tendencia general de descenso en la pérdida de clasificación. Este descenso sugiere que el modelo está mejorando su capacidad para realizar clasificaciones precisas a lo largo del tiempo.

Estabilización en las Últimas Épocas: Después de cierto número de épocas, la pérdida de clasificación parece estabilizarse. Esto indica que el modelo ha alcanzado un punto donde su rendimiento en la tarea de clasificación ha convergido o ha mejorado en menor medida.

Importancia de la Estabilidad:

Aunque la pérdida continúa disminuyendo en las primeras épocas, es importante observar la estabilidad del modelo en las últimas épocas. Una pérdida más baja no siempre garantiza un mejor rendimiento en la detección de objetos, ya que puede haber sobreajuste (overfitting) a los datos de entrenamiento.

Tabla 2

Datos Estadísticos de las Pérdidas Train/DLF Loss 1

Train/dfl loss	
Media	1,19814775
Error típico	0,012772893
Mediana	1,16585
Moda	1,2632
Desviación estándar	0,161565732
Varianza de la muestra	0,026103486
Curtosis	2,960297115
Coeficiente de asimetría	1,362655665
Rango	0,9057
Mínimo	9362
Máximo|	1,8419
Suma	191,70364
Cuenta	160
Nivel de confianza (95%)	0,025226415

Nota. Autoría Propia.

Tabla 3

Datos Estadísticos de las Pérdidas Train/CLS Loss 2

Train/cls loss	
Media	0,662244937
Error típico	0,023323571
Mediana	0,583125
Moda	
Desviación estándar	0,295022429
Varianza de la muestra	0,087038234
Curtosis	5,505329664
Coeficiente de asimetría	2,055203968
Rango	1,74485
Mínimo	0,28625
Máximo	2,0311
Suma	105,95919
Cuenta	160
Nivel de confianza (95%)	0,025226415

Nota. Autoría Propia.

Tabla 4

Datos Estadísticos de las Pérdidas Train/CLS Loss 3

Train/cls loss	
Media	1,006525875
Error típico	0,018450358
Mediana	0,96188
Moda	
Desviación estándar	0,233380616
Varianza de la muestra	0,054466512
Curtosis	2,0633866658
Coeficiente de asimetría	1,229152212
Rango	1,22092
Mínimo	0,65148
Máximo|	1,8724
Suma	161,04414
Cuenta	160
Nivel de confianza (95%)	0,036439387

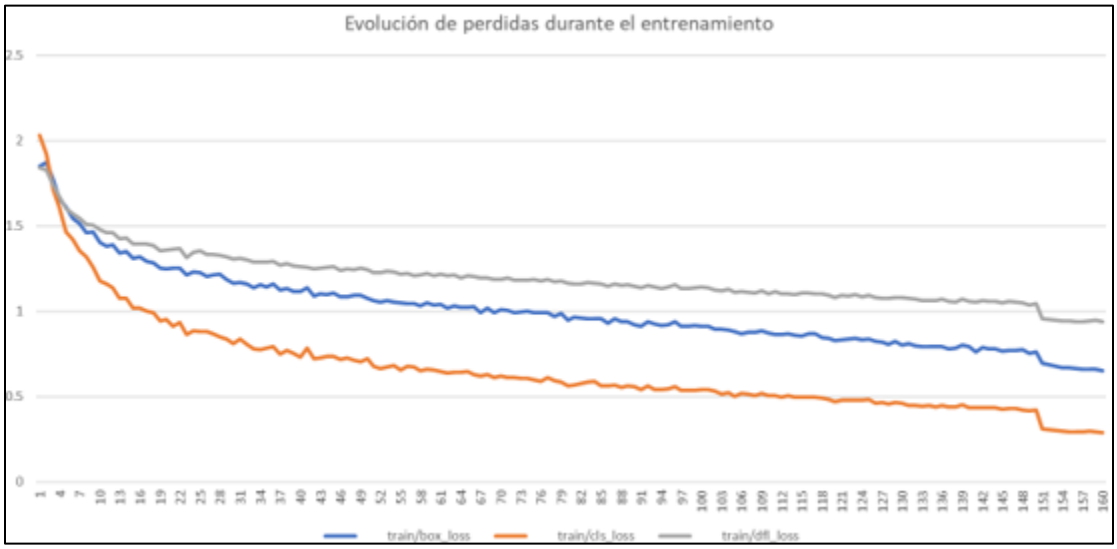
Nota. Autoría Propia.

Dentro de las tablas 2, 3 y 4 se obtienen los datos estadísticos para las diferentes épocas del modelo de entrenamiento en sus pérdidas del parámetro medido.

En el caso de la tabla 3 y 4 se evidencia que en la moda existe un valor “#N/D” no hay un valor disponible, puesto que no existe un valor que aparezca con mayor frecuencia en el conjunto de datos obtenidos durante el modelo de entrenamiento.

Figura 37

Evolución de Pérdidas Durante el Entrenamiento



Nota. Autoría Propia.

Este gráfico de líneas presenta la evolución de las pérdidas durante el entrenamiento del modelo a lo largo de múltiples épocas. Las pérdidas son indicadores cruciales que reflejan cómo el modelo está ajustando sus pesos y mejorando su capacidad para realizar detecciones precisas.

Pérdida de Caja (train/box_loss): Representa la penalización asociada con la imprecisión en la predicción de las cajas delimitadoras de los objetos. Una disminución en esta pérdida indica una mejora en la capacidad del modelo para ubicar objetos correctamente.

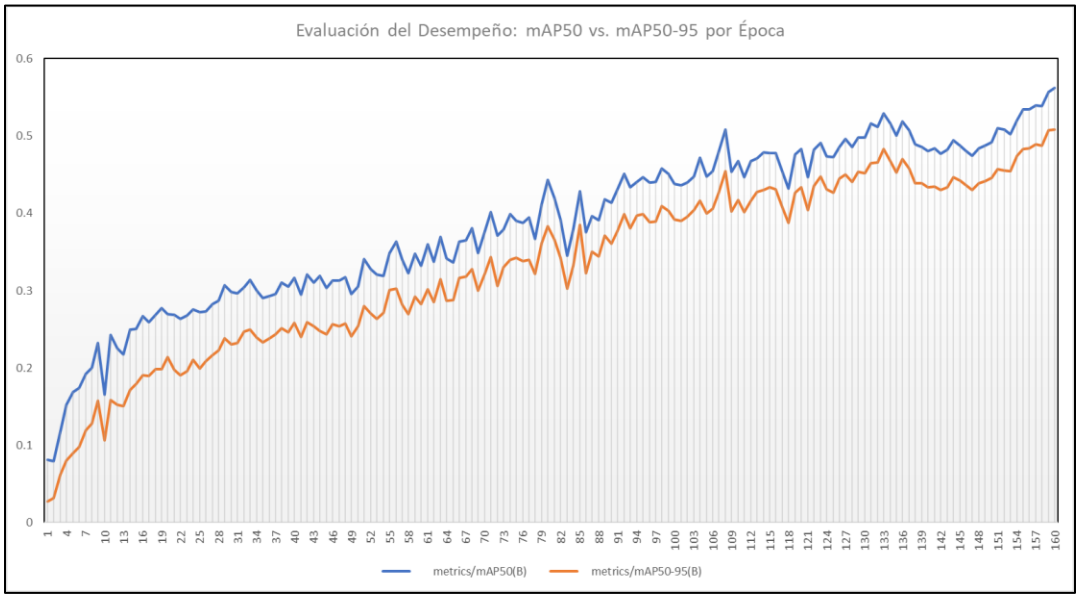
Pérdida de Clasificación (train/cls_loss): Refleja la precisión en la clasificación de los objetos detectados. Una reducción en esta pérdida indica que el modelo está mejorando en la identificación de las clases de los objetos.

Pérdida de Dificultad (train/dfl_loss): Mide la habilidad del modelo para manejar objetos con distintos niveles de dificultad. Una disminución en esta pérdida sugiere una mejora en la capacidad del modelo para detectar objetos desafiantes.

Este análisis visual proporciona información detallada sobre cómo el modelo responde a diferentes aspectos del entrenamiento. Las tendencias descendentes en las curvas de pérdida indican áreas donde el modelo está convergiendo. Es clave observar cómo estas pérdidas evolucionan para garantizar que el modelo esté aprendiendo de manera efectiva y generalizando correctamente a nuevos datos.

Figura 38

Evaluación de Desempeño mAP50 vs. mAP50-95



Nota. Autoría Propia.

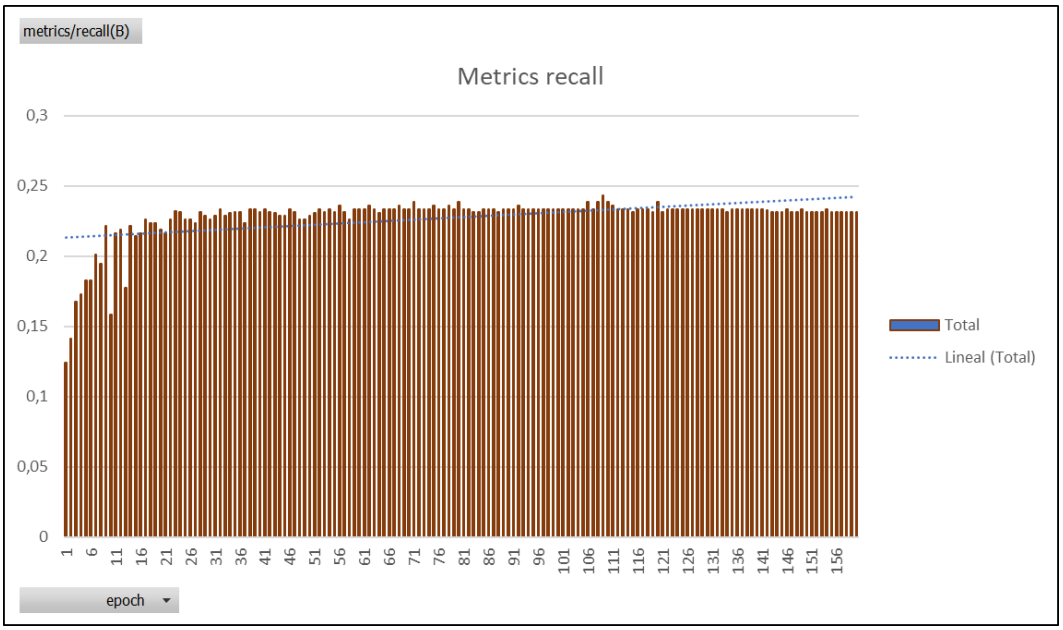
El gráfico de la figura 38 d presenta la evolución de las métricas mAP50 (Mean Average Precisión a 50%) y mAP50-95 (Mean Average Precisión de 50% a 95%) a lo largo de las épocas durante el entrenamiento del modelo. Estas métricas son indicadores clave de la precisión del modelo en la detección de objetos en imágenes.

Se tiene en cuenta que mAP50: Representa la precisión promedio de detección de objetos cuando se consideran solo las detecciones cuya confianza es igual o superior al 50%. Muestra cómo mejora la precisión general del modelo a lo largo del entrenamiento.

Y el parámetro mAP50-95: Mide la precisión promedio considerando las detecciones en un rango de confianza del 50% al 95%. Esta métrica proporciona una visión más detallada de la precisión del modelo al incluir detecciones con confianzas más altas.

Figura 39

Metric Recall

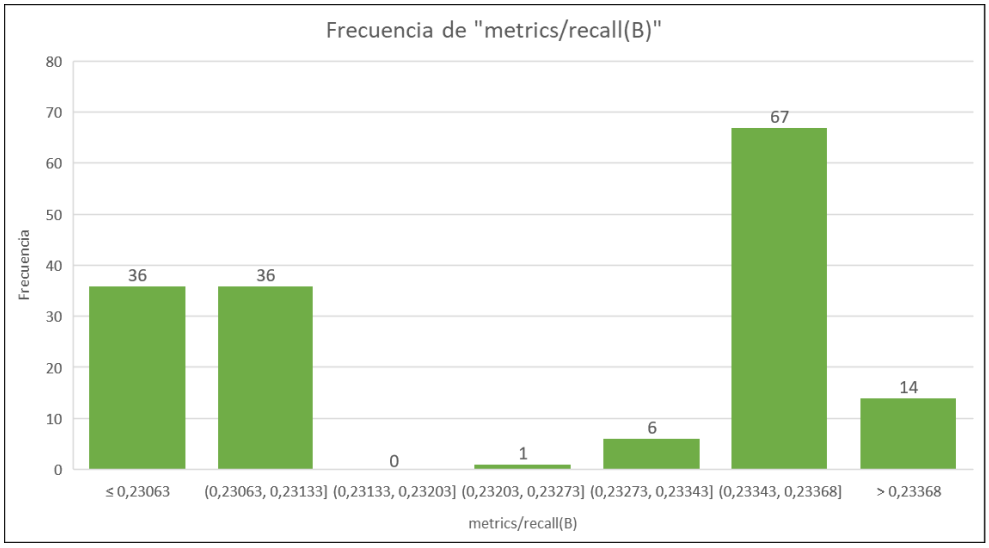


Nota. Autoría Propia.

Teniendo en cuenta que el recall mide la proporción de los casos positivos reales que fueron correctamente identificados por el modelo. Se evidencia el aumento significativo a través del paso de las épocas del modelo de entrenamiento.

Figura 40

Frecuencia Metrics Recall



Nota. Autoría Propia.

Se evidencian los siguientes resultados más significativos en la Figura 40.

Datos entre 0.23343 y 0.23368 (67 datos): Este es el rango principal de recall. Puede indicar que el modelo está funcionando bastante bien para la mayoría de las instancias de la clase ‘B’.

Datos mayores a 0.23368 (14 datos): Estos casos representan un rendimiento superior en términos de recall. Pueden ser instancias donde el modelo es especialmente competente en la detección de la clase ‘B’.

El modelo se puede optimizar aún más ya que se evidencia una tendencia de estabilización y los datos mayores a 0.23368 fueron solamente 14 datos, indicando que aún se puede mejorar los datos obtenidos del recall.

Tabla 5

Datos Estadísticos de Metrics Recall

Train/cls loss	
Media	0,227834625
Error típico	0,001303723
Mediana	0,23356
Moda	0,23358
Desviación estándar	0,016490932
Varianza de la muestra	0,000271951
Curtosis	17,56466538
Coeficiente de asimetría	-4,01484518
Rango	0,11889
Mínimo	0,12442
Máximo	0,24331
Suma	36,45354
Cuenta	160
Nivel de confianza (95%)	0,002574847

Fuente. Autoría Propia.

Teniendo en cuenta los resultados estadísticos del metric recall en la Tabla 5. Se identificaron 36 datos con recall inferior a 0.23063, indicando un área donde el modelo puede tener dificultades.

La moda de 0.23358 sugiere que este valor es frecuente en las predicciones del modelo.

El coeficiente de asimetría negativa y la curtosis alta podrían indicar una concentración de predicciones cercanas a la media.

Actividad 2.

Una vez entendemos que significa cada variable y cuáles fueron los métodos para lograr un modelo de entrenamiento, abrimos las imágenes del programa al azar, de todas las que se utilizaron en el entrenamiento donde se demuestra su funcionamiento, en el que se detecta el dron dentro de un cuadro rojo señalando el objetivo en campo.

Figura 41

Imagen de Prueba 1

Nota. Autoría Propia.

Cada una de las imágenes que se muestran son el resultado de la ejecución del entrenamiento, ya sea en campo abierto o en un atardecer, algo oscuro demostrando su eficiencia operativa.

Figura 42

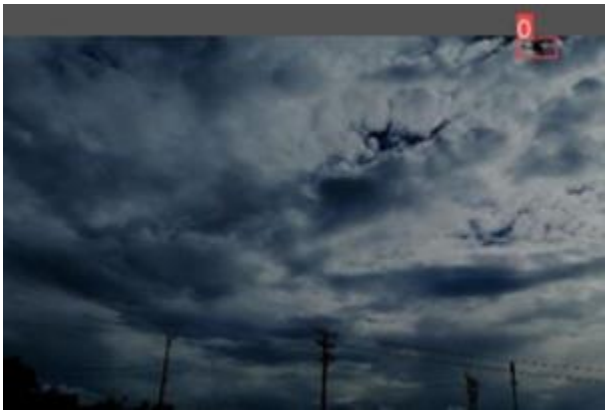
Imagen de Prueba 2

Nota. Autoría Propia.

A medida que el objetivo se acerca, su precisión aumenta; durante el día, la imagen se vuelve más clara. Esto se refleja en la ejecución del modelo entrenado, donde el recuadro rojo que rodea al objetivo se vuelve más nítido y definido a medida que el objeto se acerca al dron, permitiendo una detección más precisa y confiable.

Figura 43

Imagen de Prueba 3



Nota. Autoría Propia.

Aun reduciendo un poco la opacidad del ambiente, el modelo entrenado funciona a una distancia moderada, detectando correctamente el dron, no obstante, se aclara que no hay un porcentaje de eficiencia desplegable ya que son imágenes de prueba que el sistema entrega como resultado final del entrenamiento.

Las anteriores imágenes enmarcan el funcionamiento del proceso de detección optima con imágenes aleatorias, además dentro de la carpeta de archivos se encuentran dos tipos de imágenes una como “pred” y otra como “label”

Las imágenes con nombres como “val_batch0_pred.jpg” son imágenes de predicción generadas durante el proceso de validación del modelo durante el entrenamiento. Durante el entrenamiento de un modelo de detección de objetos como YOLOv8, se utiliza un conjunto de datos separado llamado conjunto de datos de validación para evaluar el rendimiento del modelo en datos no vistos previamente.

Estas imágenes de predicción muestran el resultado de la ejecución del modelo entrenado en el conjunto de datos de validación. Cada imagen mostrará los objetos detectados por el modelo, resaltados por recuadros, y puede proporcionar una visualización del rendimiento del modelo en términos de precisión y detección de objetos.

Es común generar estas imágenes de predicción durante el entrenamiento para realizar un seguimiento del progreso del modelo y ajustar los parámetros de entrenamiento según sea necesario para mejorar el rendimiento. Además, estas imágenes pueden ser útiles para diagnosticar problemas o errores durante el entrenamiento y realizar correcciones adecuadas.

Figura 44

Archivo PRED del Modelo Entrenado

Nota. El archivo PRED es un archivo con el cual el modelo hará la primera prueba de función.

El archivo “val_batch0_labels.jpg” contiene imágenes del conjunto de datos de validación con las etiquetas correspondientes a los objetos presentes en cada imagen. Estas imágenes suelen ser utilizadas durante el proceso de entrenamiento para verificar la precisión de las predicciones del modelo.

Cada imagen en “val_batch0_labels.jpg” probablemente muestra el contenido original del conjunto de datos de validación con las etiquetas anotadas manual o automáticamente. Estas etiquetas pueden incluir la ubicación y la clase de los objetos presentes en la imagen, lo que permite comparar las predicciones del modelo con las etiquetas reales durante la evaluación del rendimiento del modelo. Al visualizar estas imágenes junto con las imágenes de predicción (como “val_batch0_pred.jpg”), los desarrolladores pueden evaluar visualmente cómo el modelo está desempeñando en el conjunto de datos de validación y si hay discrepancias entre las predicciones del modelo y las etiquetas reales. Esto puede ayudar en el proceso de ajuste y mejora del modelo durante el entrenamiento.

Figura 45

Archivo PRED Modelo Entrenado

Nota. El archivo PRED es la muestra que el modelo fue correctamente entrenado

En la figura 45, se evidencia que el programa entrego un modelo funcional, enmarcando el objetivo logrando desarrollar una interfaz ejecutable utilizando Python para probar el sistema de detección de drones. Esta interfaz proporciona una plataforma funcional que permite a los usuarios interactuar fácilmente con el sistema, facilitando así la evaluación y validación de su funcionamiento.

Figura 46

Código en Python para Cargar el Modelo Best.pt

```
# Inicializar variables
vid = None
is_playing = False
model = YOLO("best.pt")
```

Nota. Autoría Propia.

Anteriormente en la Figura 33, se obtuvieron dos archivos del modelo entrenado, el cual en el lenguaje de programación de python llamaremos el archivo" best.pt" para ejecutar el modelo y así abrir la interfaz, tenemos:

vid = None: En esta línea, se crea una variable llamada y se le asigna el valor None. None en Python es un valor especial que representa la ausencia de un valor válido. En este caso, parece que vid se va a utilizar para almacenar alguna referencia a un objeto de video, pero en este punto no se ha asignado ningún objeto, por lo que se inicializa con None.

is_playing = False: Esta línea crea una variable llamada is_playing y se le asigna el valor booleano False. Este tipo de variable booleana se utiliza a menudo para controlar el estado de reproducción de algo, como un video o un juego. En este caso, is_playing probablemente se usará para rastrear si un video está reproduciéndose o no.

model = YOLO("best.pt"): Aquí se crea una variable llamada model y se le asigna el resultado de llamar a una función llamada YOLO con el argumento "best.pt". Parece que YOLO es una clase o función usada para cargar un modelo de detección de objetos YOLO. "best.pt" es probablemente el archivo que contiene los pesos pre entrenados del modelo. Por lo tanto, esta línea carga un modelo YOLO pre entrenado y lo asigna a la variable model para su uso posterior en la detección de drones en el sistema.

En la figura 47 se tienen funciones las cuales leen un fotograma del vídeo, lo procesa para detectar drones, marca los drones detectados en el fotograma y actualiza el lienzo de la interfaz gráfica con el fotograma procesado.

Figura 47

Código en Python para la Interfaz de Video

```python
# Función para actualizar el video en el lienzo
def update():
    global is_playing, vid
    if is_playing:
        ret, frame = vid.read()
        if ret:
            frame = cv2.resize(frame, (640, 640))  # Redimensionar el frame al tamaño
            resultados = model.predict(frame, imgsz = 640, conf = 0.50)
            annotated_frame = resultados[0].plot()
            annotated_frame = cv2.cvtColor(annotated_frame, cv2.COLOR_BGR2RGB)
            image = Image.fromarray(annotated_frame)
            image = ImageTk.PhotoImage(image=image)
            canvas.create_image(0, 0, anchor=tk.NW, image=image)
            canvas.image = image
    window.after(10, update)
```

Nota. Líneas de código para ejecutar el lienzo de video en Python

Las funciones de las líneas de código funcionan de la siguiente manera:

global is_playing, vid: Esto indica que la función utilizará las variables globales is_playing y vid, lo que significa que la función puede modificar estas variables incluso si no están definidas dentro de la función misma.

if is_playing: Comprueba si el vídeo está reproduciéndose. Si is_playing es verdadero, el siguiente bloque de código se ejecutará.

ret, frame = vid. read(): Lee un fotograma (frame) del vídeo almacenado en la variable vid. ret es un booleano que indica si la lectura fue exitosa y frame es el fotograma leído.

if ret: Verifica si la lectura del fotograma fue exitosa. Si es así, el siguiente bloque de código se ejecutará.

frame = cv2.resize(frame, (640, 640)): Redimensiona el fotograma a un tamaño específico (640x640 píxeles en este caso).

resultados = model.predict(frame, imgsz=640, conf=0.50): Utiliza un modelo de detección de objetos (presumiblemente YOLO, según el código anterior) para predecir la presencia de drones en el fotograma.

annotated_frame = resultados[0].plot(): Genera una versión del fotograma con las detecciones de drones marcadas.

annotated_frame = cv2.cvtColor(annotated_frame, cv2.COLOR_BGR2RGB): Convierte el formato del fotograma de BGR (utilizado por OpenCV) a RGB.

image = Image.fromarray(annotated_frame): Convierte el fotograma en un objeto de imagen compatible con la biblioteca Tkinter.

image = ImageTk.PhotoImage(image=image): Convierte la imagen en un objeto de imagen Tkinter para ser utilizada en el lienzo.

canvas.create_image(0, 0, anchor=tk.Nw, image=image): Crea una nueva imagen en el lienzo con la imagen actualizada.

canvas.image = image: Actualiza la referencia a la imagen en el lienzo para evitar que el recolector de basura de Python elimine la imagen accidentalmente.

Después de ejecutar todo el código en Python, se produce la apertura de una ventana emergente que muestra la interfaz gráfica resultante.

Figura 48

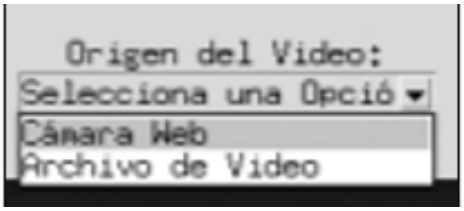
Interfaz del Sistema de Detección de Drones

Nota. Archivo ejecutable de sistema detección de drones.

Esta interfaz tiene un menú desplegable para las diferentes opciones de ejecución, si se requiere que sea por medio de cámara web o por medio de un archivo de video guardado previamente en el ordenador.

Figura 49

Selección Origen de Video



Nota. Opciones de selección de modo para correr modelo.

Figura 50

Origen de Video Local

Nota. Selección de video formato mp4.

Se selecciona el video de prueba y se inicia la reproducción con el botón iniciar.

Figura 51

Reproducción de Video 1

Nota. Reproducción de video previamente guardado en el ordenador

Una vez seleccionado el video y comenzada la reproducción, el sistema realiza la detección precisa del dron en función de la distancia observada, según la configuración establecida. Para la Figura 52, se observa una efectividad del 73%.

Figura 52

Reproducción de Video 2

Nota. Dron a una distancia moderada + 20 metros

En ocasiones en una larga distancia, se sigue detectando con una eficiencia de 62%.

Figura 53

Detección Ambiente Oscuro

Nota. Detección en video donde la visibilidad es algo reducida por el matiz del ambiente.

Debido a variaciones climáticas, el cielo puede oscurecerse, lo que resulta en una disminución en la visibilidad del dron. Sin embargo, este oscurecimiento del cielo también oscurece la figura del dron, haciéndola más distintiva como una sombra clara. Esto conduce a una eficiencia del 72% en la detección de drones en estas condiciones.

Figura 54

Alta Eficiencia Detección en Video

Nota. Detección de dron cerca al punto de origen de video.

La eficiencia de la detección fue de 80% por su cercanía e iluminación y otros factores de tonalidad durante la reproducción del video lo cual enriquecen el entorno y facilita su detección.

Figura 55

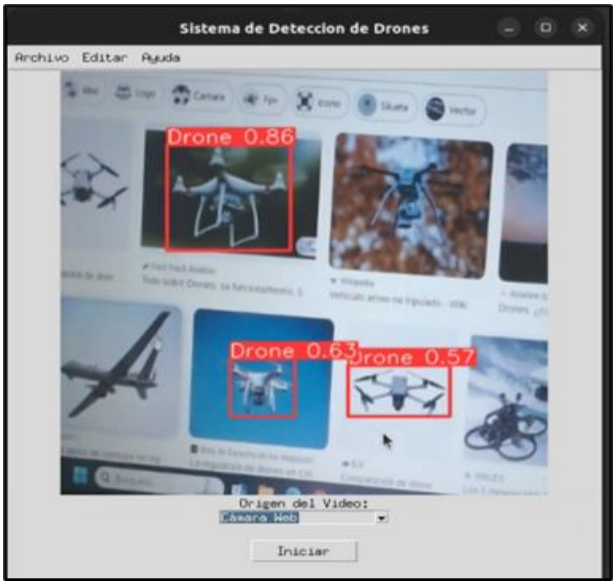
Detección no Exitosa en Video

Nota. No hay detección del dron

Por diversos factores como lo son las interferencias visuales, sombreado e iluminación cambiante, cambio en la textura y color generan errores el cual el sistema no logra detectar correctamente el dron. Para abordar estos desafíos, pueden requerirse algoritmos de detección más avanzados capaces de distinguir los drones incluso en entornos complejos y con obstrucciones visuales. Esto podría implicar el procesamiento de imágenes más sofisticadas, como redes neuronales convolucionales entrenadas para detectar drones en entornos desafiantes. Además, la integración de datos de múltiples fuentes, como cámaras de diferentes ángulos o sensores adicionales, podría mejorar la precisión de la detección en estas situaciones.

Figura 56

Origen de Video desde Cámara Web



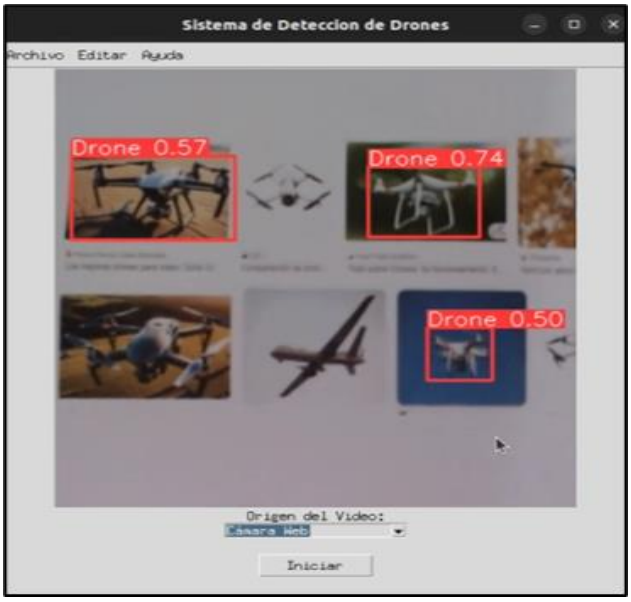
Nota. Varias imágenes de drones de Google Imágenes

En la opción “cámara web” de la interfaz, se establece la conexión con la cámara frontal del ordenador, según la configuración predeterminada en las líneas de Python. Se muestran imágenes aleatorias en la interfaz, utilizando la cámara frontal en este caso. La eficiencia de

detección en esta configuración es notablemente alta debido al fondo completamente blanco, lo que facilita una ejecución óptima y una detección precisa.

Figura 57

Detección en Imágenes desde Cámara Web

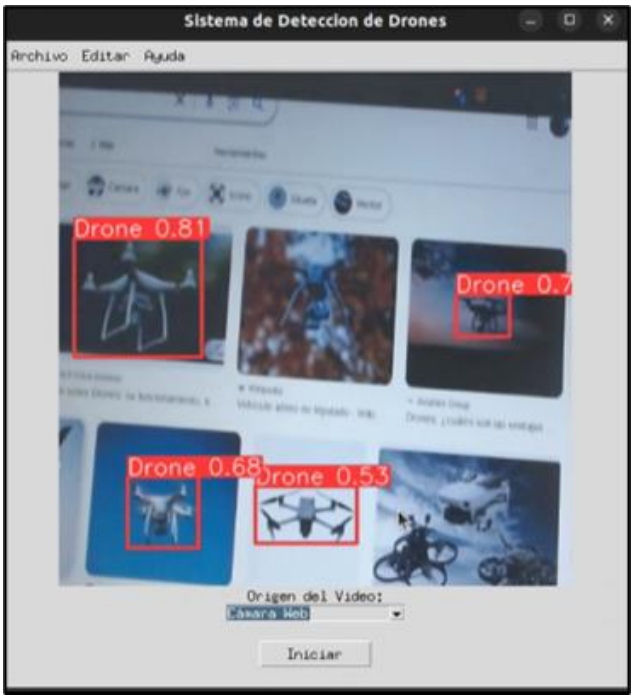


Nota. Detección con imágenes de drones

En la detección de drones la influyen varios factores. En la Figura 57, se observan drones de distintos tipos y colores, lo que puede afectar la eficiencia de la detección. Es importante destacar que, aunque se pueden experimentar algunas detecciones fallidas, en un escenario hipotético donde se presenten múltiples drones simultáneamente, se evidencia una detección eficiente de más de un dron al mismo tiempo. Este resultado resalta la robustez y capacidad del sistema para identificar y rastrear múltiples drones de manera efectiva, incluso en condiciones desafiantes.

Figura 58

Multi Detección



Nota. Prueba con más de 4 drones en imágenes de Google

En esta última prueba, se logra evidenciar un alto porcentaje de detección correcta de drones, donde se identificaron exitosamente 4 objetos. Este resultado refleja una eficiencia de detección que supera el 50%, lo que demuestra la capacidad del sistema para identificar con precisión la presencia de drones en el entorno.

Para finalizar este proyecto trasciende su naturaleza como un proyecto de grado, convirtiéndose en un pilar esencial dentro del ámbito de la seguridad. La detección de drones con YOLOv8 y Python es un logro académico y se erige como una contribución significativa a la seguridad militar aérea y a la protección de áreas restringidas. En un mundo cada vez más interconectado, la amenaza de drones no autorizados se ha convertido en un desafío creciente, y esta tecnología representa un avance crucial en la defensa contra tales amenazas.

Al dotar a las fuerzas militares y de seguridad con herramientas efectivas para detectar y neutralizar drones no deseados, este proyecto se sitúa en el centro de los esfuerzos por garantizar la integridad de nuestras fronteras y territorios críticos. Su impacto va más allá de los confines de la academia, llegando a ser una contribución tangible a la seguridad nacional y la protección de nuestros ciudadanos. Además, su aplicación en zonas prohibidas y áreas de alto riesgo fortalece la capacidad de mantener la seguridad en entornos críticos y salvaguardar la tranquilidad de nuestras comunidades.

Este proyecto de seguridad no solo representa el compromiso y la dedicación de nosotros como estudiantes, sino que también ilustra el papel crucial que desempeña la investigación académica en la promoción de la seguridad y el bienestar de la sociedad en su conjunto. Su impacto perdurará mucho más allá de los límites del campus universitario, dejando un legado de innovación y protección que beneficiará a las generaciones venideras. En última instancia, este proyecto demuestra cómo la fusión entre la academia y la seguridad puede conducir a soluciones poderosas y significativas que resguardan nuestras libertades fundamentales y aseguran un futuro más seguro para todos.

Conclusiones

Este proyecto ha desarrollado con éxito un algoritmo de inteligencia artificial basado en el framework Ultralytics YOLOv8, entrenado para la detección de drones de ala rotatoria utilizando visión computacional. Este avance representa una solución innovadora y eficiente para abordar los desafíos de seguridad y control del espacio aéreo relacionados con la proliferación de drones.

El entrenamiento del modelo utilizando framework YOLOv8 y técnicas de fine tuning ha demostrado ser efectivo para reconocer patrones distintivos de drones de ala rotatoria. Los resultados obtenidos evidencian la capacidad del modelo para detectar drones con alta precisión, superando las limitaciones de enfoques convencionales y adaptándose a condiciones variables.

La implementación de una interfaz gráfica ha ampliado la utilidad de la solución propuesta, permitiendo que usuarios con diferentes niveles de experiencia técnica puedan acceder a la información en tiempo real de manera efectiva. Este enfoque centrado en la accesibilidad fortalece la aplicabilidad práctica del modelo desarrollado en diversos entornos, desde aeropuertos hasta eventos masivos.

La evaluación del rendimiento del modelo mediante métricas rigurosas ha proporcionado una medida cuantitativa de su eficacia. Los resultados obtenidos en las pruebas destacan la capacidad del sistema para mejorar la seguridad en áreas críticas, detectando de manera oportuna la presencia de drones y demostrando un rendimiento destacado en la identificación precisa de drones en diversos entornos.

El proceso de optimización del modelo ha mejorado su eficiencia y velocidad, facilitando su implementación en hardware específico como FPGAs o GPUs embarcadas. Esta optimización

es crucial para aplicaciones en tiempo real, asegurando que el sistema pueda operar eficazmente en entornos críticos y de alta demanda.

Las pruebas en campo han confirmado la viabilidad del modelo en condiciones reales, validando su capacidad para adaptarse a diversas condiciones ambientales y reconocer patrones específicos de drones de ala rotatoria. Esta validación en campo asegura que el modelo esté listo para su implementación práctica en situaciones del mundo real, mejorando la seguridad y el control del espacio aéreo.

Recomendaciones

A continuación, las siguientes recomendaciones para diseño y elaboración de un prototipo Dron y culminar de manera satisfactoria en el proceso:

Inicia con una investigación profunda: Antes de empezar a diseñar un modelo, asegúrate de entender cómo funcionan los drones de ala rotatoria y qué características tienen en común. Analiza cómo se han abordado problemas similares en el pasado y qué técnicas se han utilizado con éxito.

Definir objetivos claros: Establece objetivos específicos para tu proyecto, como la detección de drones de ala rotatoria en un entorno determinado o la clasificación de drones según su tipo. Esto te ayudará a mantener el foco en lo que deseas lograr y a evaluar el éxito de tu modelo.

Recopila datos relevantes: Recopila datos sobre los drones de ala rotatoria que deseas detectar, incluyendo características como la forma de sus alas, el tamaño, el color, el movimiento y otros detalles visibles. Asegúrate de que los datos sean precisos y representativos de los drones que deseas detectar.

Elije una arquitectura adecuada: Considera diferentes arquitecturas para tu modelo, como redes neuronales convolucionales (CNN) o redes neuronales recurrentes (RNN). Asegúrate de que la arquitectura sea adecuada para el tipo de datos que tienes y para los objetivos que deseas lograr.

Prueba y ajusta: Prueba tu modelo con diferentes conjuntos de datos y ajusta los parámetros según sea necesario. Asegúrate de que el modelo sea robusto y pueda generalizarse a diferentes entornos y condiciones.

Considera la seguridad y la privacidad: Si tu modelo va a ser utilizado para fines de seguridad o privacidad, asegúrate de que sea diseñado con seguridad y privacidad en mente. Considera cómo proteger los datos y evitar posibles vulnerabilidades.

Documenta y comunica: Documenta cada paso del proceso de desarrollo y comunica tus resultados de manera clara y concisa. Esto te ayudará a mantener un registro de tus progresos y a compartir tus hallazgos con otros.

Aprovecha recursos adicionales: No tengas miedo de buscar ayuda o recursos adicionales si lo necesitas. Puedes encontrar tutoriales, cursos en línea o comunidades de desarrolladores que te ayuden a superar obstáculos o a mejorar tus habilidades.

Asegúrate de cumplir con las normas y regulaciones: Asegúrate de que tu modelo cumpla con las normas y regulaciones aplicables, como la privacidad de los datos o la seguridad de la información.

Recomendaciones Futuras

Aprendizaje profundo: Investigar y probar arquitecturas más avanzadas como redes neuronales profundas (DNN) o redes neuronales generativas adversarias (GAN) para mejorar la detección y clasificación de drones.

Sensor Fusion: Integrar múltiples tipos de sensores (por ejemplo, cámaras RGB, cámaras térmicas, LiDAR) para mejorar la precisión y robustez del modelo.

Aumentar el conjunto de datos: Recopilar más datos de drones en diferentes condiciones y entornos para mejorar la capacidad de generalización del modelo.

Datos sintéticos: Utilizar técnicas de generación de datos sintéticos para crear datos adicionales que simulen diferentes escenarios y condiciones.

Optimización y eficiencia:

Optimización del modelo: Trabajar en la optimización del modelo para reducir su tamaño y mejorar su velocidad, lo que es crucial para aplicaciones en tiempo real.

Implementación en hardware: Investigar la implementación del modelo en hardware específico, como FPGAs o GPUs embarcadas, para mejorar el rendimiento y la eficiencia energética.

Evaluación en campo: Realizar pruebas en condiciones reales para evaluar el rendimiento del modelo en entornos no controlados.

Comparación con otros modelos: Comparar el modelo con otros métodos y enfoques existentes para identificar fortalezas y áreas de mejora.

Aspectos de seguridad y privacidad:

Ciberseguridad: Investigar y mitigar posibles vulnerabilidades de seguridad en el sistema, incluyendo ataques adversariales que podrían engañar al modelo.

Privacidad de los datos: Desarrollar e implementar medidas para garantizar la privacidad de los datos recopilados y utilizados por el modelo.

Interfaz y usabilidad: Experiencia del usuario: Realizar estudios de usabilidad para entender mejor las necesidades y preferencias de los usuarios finales.

Colaboración y publicación: Colaboraciones interdisciplinares: Fomentar la colaboración con expertos de diferentes disciplinas (por ejemplo, ingeniería, informática, ciencias sociales) para abordar el problema desde múltiples perspectivas.

Publicación de resultados: Publicar los hallazgos y avances en revistas científicas y conferencias para compartir conocimientos y recibir retroalimentación de la comunidad investigadora.

Actualización continua: Mantenerse actualizado con las normas y regulaciones vigentes relacionadas con el uso de drones y la privacidad de los datos, y ajustar el modelo y los procedimientos según sea necesario.

Contribución a la normativa: Participar en grupos de trabajo y foros de estándares para contribuir al desarrollo de regulaciones y mejores prácticas en el campo.

Referencias Bibliográficas

Baños, J. M. (28 de junio de 2023). *letslaw Tech law y finance*. letslaw Tech law y finance: https://letslaw.es/uso-de-drones-y-la-privacidad/

BBC. (2023). *Southampton v Aston Villa: Play suspended before half-time because of drone inside stadium*. BBC: https://www.bbc.com/sport/football/64361114

Bestechnology Group. (2019). *IT Services and IT Consulting*. https://ba.linkedin.com/company/bestechnology-group

Cui, K., Tang, S., y Liu, B. (2023). Enhancement of quantum sensing in a cavity-optomechanical system around the quantum critical point. *Phys. Rev, 108,* 51-53. https://journals.aps.org/pra/abstract/10.1103/PhysRevA.108.053514

Giordan, D., y Adams, M. (2020). The use of unmanned aerial vehicles (UAVs) for engineering geology applications. *Bulletin of Engineering Geology and the Environment, 79,* 8-10. https://doi.org/10.1007/s10064-020-01766-2

Iglesias, A. (2022). *Drones espías y asesinatos en el día a día de los indígenas más aislados del mundo*. El País: https://elpais.com/planeta-futuro/3500-millones/2022-11-23/drones-espias-y-asesinatos-en-el-dia-a-dia-de-los-indigenas-mas-aislados-del-mundo.html

López, M. (2019). *Por invasión de un dron, el aeropuerto El Dorado cerró su operación unos minutos*. La República: https://www.larepublica.co/economia/por-invasion-de-un-dron-el-aeropuerto-el-dorado-cerro-su-operacion-unos-minutos-2836572

López, T. (2022). *Drones y Seguridad Nacional - Un estudio multidimensional.* https://www.dsn.gob.es/es/documento/drones-seguridad-nacional-un-estudio-multidimensional

Mendiola, G. (2018). El dispositivo del dron: entre la vigilancia securitaria y la necropolítica. *Convergencia, 26*(79), 1-24. https://www.redalyc.org/journal/105/10557887001/

Myers, J. (2022). Cryo-Computing for Infrastructure Applications: A Technology-to-Microarchitecture Co-optimization Study. *IEEE, 5*(4), 1-23. https://ieeexplore.ieee.org/abstract/document/10019436/authors#authors

Peng, L., y Murray, C. (2022). Parallel Drone Scheduling Traveling Salesman Problem with Weather Impacts. *SSRN, 1*, 1-12. https://doi.org/10.2139/ssrn.4254262

Salas, R. (2020). *Redes Neuronales Artificiales*. https://www.researchgate.net/profile/Rodrigo-Salas-5/publication/266882116_Redes_Neuronales_Artificiales/links/55943afc08ae99aa62c58eb2/Redes-Neuronales-Artificiales.pdf

Sánchez, D., González, H., y Hernández, Y. (2020). Revisión de algoritmos de detección y seguimiento de objetos con redes profundas para videovigilancia inteligente. *Revista Cubana de Ciencias Informáticas, 14*(3), 165-195. https://www.redalyc.org/journal/3783/378365834009/378365834009.pdf

Tamke, F., y Buscher, U. (2022). The vehicle routing problem with drones and drone speed selection. *Computers y Operations Research, 152*, 1-15. https://doi.org/10.1016/j.cor.2022.106112

Vargas, S. (2024). *Disidencias de las Farc estarían utilizando drones: un video tiene en alerta a las Fuerzas Armadas*. Infobae: https://www.infobae.com/colombia/2024/03/20/disidentes-estarian-manejando-drones-un-video-tiene-en-alerta-a-inteligencia/

Wang, X., Swanson, K., Liu, Z., y Jones, G. (2022). A Simulation-Heuristic Approach to Optimally Design Drone Delivery Systems in Rural Areas. *WSC '22: Proceedings of the Winter Simulation Conference, 1*, 1581-1592. https://dl.acm.org/doi/10.5555/3586210.3586340

Zapata, G., y García, R. (2021). CyberDrone: una plataforma de ciberseguridad para detección de ataques a drones. *Ingeniería y desarrollo, 39*(1), 44-65. http://www.scielo.org.co/pdf/inde/v39n1/2145-9371-inde-39-01-44.pdf

Zhu, W., Royal, P., y Waters, T. (2023). Investigating the influence of drone flight on the stability of cancer medicines. *Plos One, 18*(1), 1-10. https://doi.org/10.1371/journal.pone.0278873