UNIVERSIDAD NACIONAL JORGE BASADRE GROHMANN

Facultad de Ingeniería

Escuela Profesional de Ingeniería en Informática y Sistemas

MACHINE LEARNING PARA LA CLASIFICACIÓN DE IMÁGENES DE PLAGAS DEL OLIVO EN LA YARADA LOS PALOS, TACNA

TESIS

Presentada por:

Bach. Franz Chanini Mena

Para optar el Título Profesional de:

INGENIERO EN INFORMÁTICA Y SISTEMAS

TACNA - PERÚ

2025

UNIVERSIDAD NACIONAL JORGE BASADRE GROHMANN

FACULTAD DE INGENIERÍA

ESCUELA PROFESIONAL DE INGENIERÍA EN INFORMÁTICA Y SISTEMAS

"MACHINE LEARNING PARA LA CLASIFICACIÓN
DE IMÁGENES DE PLAGAS DEL OLIVO EN
LA YARADA LOS PALOS, TACNA"

TESIS PRESENTADA Y APROBADA EL 27 DE DICIEMBRE DEL 2024
ESTANDO EL JURADO CALIFICADOR INTEGRADO POR:

Presidente : Dr. Edwin Antonio Hinojosa Ramos

Secretario : Mtro. Hugo Manuel Barraza Vizcarra

Vocal : Dr. Erbert Francisco Osco Mamani

CERTIFICADO DE SIMILITUD

Yo, Dr. Erbert Francisco Osco Mamani, en mi condición de asesor acreditado por la Resolución de Facultad N° 8260-2023-FAIN/UNJBG, de la Tesis:

"MACHINE LEARNING PARA LA CLASIFICACIÓN DE IMÁGENES DE PLAGAS DEL OLIVO EN LA YARADA LOS PALOS, TACNA".

Presentado por el bachiller Franz Chanini Mena (2018-119024), para optar el Título Profesional de Ingeniero en Informática y Sistemas.

Habiendo cumplido con lo establecido en el reglamento de originalidad y de similitud de trabajos de investigación y producción intelectual, considerando que según la revisión, evaluación y análisis realizado a través del software de similitud textual Turnitin, cuenta con el NIVEL DE SIMILITUD de 8%. Por lo que certifico la similaridad de la tesis, que está de acuerdo al NIVEL PERMITIDO, para continuar con los trámites correspondientes y para su publicación en el repositorio Institucional.

Se emite el presente certificado con fines de continuar con los trámites respectivos para su obtención del título profesional.

Dr. Erbert Francisco Osco Mamani
DNI. 00409196

Bach. Franz Chanini Mena
DNI. 74808802

DEDICATORIA

A mis padres, Benita Mena Limachi y Juan Chanini Arana, por su apoyo incondicional y por ser mi fuente constante de inspiración. Su esfuerzo y sacrificio han sido invaluables para alcanzar mis metas. Les dedico este logro con todo mi amor y gratitud.

A mis hermanos, cuyo apoyo, consejos y hasta reprimendas, me han permitido seguir el camino correcto. Les dedico este logro de todo corazón, agradeciéndoles por su aprecio constante y por estar a mi lado en cada paso del camino.

A mi cuñado, Omar Santiago Mamani Rivera, quien ha sido fundamental para alcanzar este objetivo. Su constante apoyo y los consejos que me ha brindado en momentos difíciles han sido de gran ayuda y me han dado ánimo para seguir adelante.

AGRADECIMIENTO

A Dios, por ser el soporte fundamental en mi vida, brindándome fortaleza emocional y manteniéndome siempre con salud. Sé que siempre está a mi lado, permitiéndome contar con él en cualquier instante. Sin su presencia y bendiciones, este logro no habría sido posible.

Al proyecto "Estimación de productividad de olivares usando técnicas de visión artificial y aprendizaje profundo en la Región de Tacna" de la Universidad Nacional Jorge Basadre Grohmann, Tacna – Perú, por brindarme la ayuda necesaria para resolver mis dudas relacionadas con esta investigación. También agradezco por facilitarme la cámara y proporcionarme los fondos necesarios para los gastos de pasaje, lo cual me permitió desplazarme a La Yarada Los Palos, Tacna, para la toma de fotografías esenciales para esta investigación. Además, valoro el seguimiento constante de mi estado de tesis, lo cual me motivó y me ayudó a mantenerme enfocado en los objetivos establecidos.

A mis tutores, Dr. Erbert Francisco Osco Mamani y Mgr. Israel Nazareth Chaparro Cruz, cuya orientación experta y asesoramiento dedicado fueron pilares fundamentales en cada etapa de la presente investigación.

A mi colega, Alejandro Daniel Yanapa Chicalla por su valiosa contribución durante el transcurso de esta investigación. Su disposición para resolver mis dudas, sus explicaciones claras y su apoyo constante fueron importantes en el desarrollo de este trabajo.

Finalmente, a todos los docentes de la Escuela Profesional de Ingeniería en Informática y Sistemas, por brindarme sus valiosos conocimientos y orientación durante mi formación. Su compromiso, paciencia y dedicación en la enseñanza han sido fundamentales para mi desarrollo académico y personal.

CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

El presente trabajo de investigación denominado "Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna", tiene como objetivo principal comparar el rendimiento de diversas configuraciones de un modelo de Machine Learning para la clasificación de imágenes de plagas del olivo en esta región. La investigación, de naturaleza aplicada y de nivel comparativo, se basa en un diseño no experimental.

El estudio se centró en evaluar el rendimiento de un modelo de Machine Learning basado en redes neuronales convolucionales (CNN) para la clasificación de plagas del olivo. A través de un análisis comparativo, se investigaron dos configuraciones clave: las dimensiones de las imágenes de entrada y el uso de técnicas de aumento de datos.

Los resultados mostraron diferencias significativas en el rendimiento del modelo, evidenciando que configuraciones como el tamaño de las imágenes de entrada y la aplicación de técnicas de aumento de datos tienen un impacto considerable en la precisión de la clasificación.

Palabras clave: Machine Learning, Redes Neuronales Convolucionales, VGG-16, Olivo, Plagas.

ABSTRACT

The present research work entitled “Machine Learning for the classification of olive pest images in La Yarada Los Palos, Tacna”, has as its main objective to compare the performance of different configurations of a Machine Learning model for the classification of olive pest images in this region. The research, of applied nature and comparative level, is based on a non-experimental design.

The study focused on evaluating the performance of a Machine Learning model based on convolutional neural networks (CNN) for olive pest classification. Through a comparative analysis, two key configurations were investigated: the dimensions of the input images and the use of data augmentation techniques.

The results showed significant differences in model performance, evidencing that configurations such as the size of the input images and the application of data augmentation techniques have a considerable impact on classification accuracy.

Keywords: Machine Learning, Convolutional Neural Networks, VGG-16, Olive, Pests.

INTRODUCCIÓN

El sector de la agricultura juega un papel muy importante en el crecimiento de la economía del país. En particular, Tacna se destaca como el principal productor y exportador de aceitunas a nivel nacional (MINCETUR, 2023). Esta área dispone de aproximadamente 35,000 hectáreas dedicadas exclusivamente al cultivo de olivos. Desafortunadamente, Tacna enfrenta no solo un grave déficit hídrico, sino también diversas plagas que afectan la producción de olivos y sus frutos (Trinidad, 2023).

La situación se ve agravada por el impacto del Fenómeno El Niño, que ha elevado las temperaturas más allá de los niveles óptimos para el cultivo del olivo. Estas condiciones climáticas extremas favorecen la aparición y propagación de plagas (SENAMHI, 2023). Según Víctor Manuel Morales Ordoñez, presidente de la Asociación Pro Olivo, el Fenómeno El Niño ha afectado ya más del 80% de la producción total de aceitunas en Tacna (Canal Exitosa Noticias, 2024).

Ante estos desafíos, resulta esencial adoptar tecnologías modernas, siendo clave la identificación temprana y la clasificación precisa de las plagas para implementar estrategias de control más efectivas.

El presente trabajo comparará el rendimiento de diversas configuraciones de modelos de Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna. Se evaluarán configuraciones como las dimensiones de las imágenes de entrada y el uso de técnica de aumento de datos, con el objetivo de identificar las configuraciones óptimas que mejoren la precisión y eficiencia para clasificar imágenes de las tres principales plagas que afectan los cultivos de olivo en la región: la Queresa Móvil (Orthezia olivicola), el Chanchito o Piojo Harinoso (Pseudococcus adonidum) y la Mosquita Blanca del Fresno (Siphoninus phillyreae). Se empleará un modelo de red neuronal convolucional (CNN) preentrenado en ImageNet, específicamente el VGG-16.

CAPÍTULO I
PLANTEAMIENTO DEL PROBLEMA

1.1. Antecedentes del problema a investigar

Perú se distingue no solo por su deliciosa gastronomía, sino también por la gran variedad de productos agrícolas que se cultivan en nuestra tierra y se exportan globalmente. En este contexto, Tacna, conocida como la “Capital del Olivo”, representa un eje central en la producción de aceitunas, con aproximadamente 35,000 hectáreas dedicadas exclusivamente a este cultivo (Trinidad, 2023). Según el Ministerio de Comercio Exterior y Turismo (MINCETUR, 2023), Tacna lidera la producción y exportación de aceitunas, alcanzando un 77% y 64% del total nacional en 2023, respectivamente.

En los últimos años, el desarrollo de tecnologías de inteligencia artificial, especialmente el Machine Learning, ha mostrado un gran potencial para mejorar la detección y clasificación de plagas en cultivos. Investigaciones previas han demostrado que el uso de algoritmos de Machine Learning, en particular redes neuronales convolucionales (CNN), permite realizar clasificaciones precisas a partir de imágenes, facilitando así la identificación temprana de plagas. Por ejemplo, en el estudio de Sánchez (2022), se aborda la necesidad de detectar el aumento de las poblaciones de la mosca del olivo (Bactrocera oleae) sin la intervención de un operario en la zona de estudio. Este enfoque permite una mayor eficacia en la prevención de plagas y reduce el número de viajes al área, disminuyendo así la huella de carbono asociada al transporte y facilitando el trabajo de biólogos y agricultores.

Asimismo, en la investigación de Burgos (2023), se busca abordar los resultados insuficientes que se obtienen con el uso de drones o la inspección visual para el control de plagas en cultivos de aguaymanto. Esta investigación implementa una red neuronal convolucional y una red neuronal profunda con el objetivo de tomar acciones correctivas a tiempo y mejorar el rendimiento de las cosechas.

Por otro lado, la investigación de Cayllante (2024) se centra en la necesidad de identificar y clasificar plagas en los cultivos de quinua en la Asociación de Productores

Agropecuarios de Jopopamba (ASPRAJO) mediante el desarrollo de un modelo de aprendizaje profundo. Este modelo, basado en imágenes de las plantas y de los insectos, tiene el potencial de mejorar la precisión y eficiencia del proceso de detección, reduciendo el daño a los cultivos y las pérdidas económicas para los productores.

Un aspecto fundamental en el rendimiento de los modelos de Machine Learning son los ajustes en las técnicas de preprocesamiento y optimización del modelo, tales como la dimensión de la imagen y las técnicas de aumento de datos o Data Augmentation (DA). La dimensión de la imagen de entrada puede influir drásticamente en la capacidad del modelo para aprender patrones relevantes, y su adecuado ajuste es esencial para maximizar la precisión de la clasificación. Asimismo, la técnica de aumento de datos se ha vuelto indispensable para enriquecer los conjuntos de datos, especialmente en escenarios donde la cantidad de imágenes disponibles es limitada. Las investigaciones han demostrado que la implementación de diversas técnicas de aumento, como rotaciones, escalados, desplazamientos, volteos y giros, contribuye a mejorar tanto la robustez como la precisión de los modelos. Por ejemplo, en la investigación de Calderon Ortiz et al. (2021), se desarrolló una herramienta para facilitar un primer diagnóstico mediante la identificación de lunares en la piel que se ajusten a los criterios de un posible melanoma. Para ello, se recopilaron un total de 1,087 imágenes de melanomas y 1,029 de lunares benignos. Dado que muchas de estas imágenes contenían vellos, se utilizó un removedor digital llamado DigitalHairRemoval, junto con técnicas de aumento de datos, para mejorar la precisión de los modelos desarrollados con redes neuronales convolucionales.

Por otro lado, en la investigación de De Luca et al. (2021), se lleva a cabo una comparación de distintos modelos preentrenados para optimizar el ingreso de productos al sistema del Banco Alimentario de La Plata, utilizando el reconocimiento de imágenes. Se consideran varios factores que afectan el rendimiento de los modelos. Uno de estos factores es el procesamiento de cada imagen, que se representa como una matriz de tres dimensiones que debe ser recorrida durante la convolución a través del kernel. Este aspecto depende en gran medida del tamaño de la imagen y de la configuración de la convolución, lo que influye directamente en el tiempo de procesamiento. Además, se destaca el uso de técnicas de aumento de datos (Image Augmentation), que permiten girar

e invertir las imágenes del conjunto de datos, logrando un aumento considerable de este y, como resultado, mejorando la calidad del aprendizaje.

A pesar de los avances en la aplicación de modelos Machine Learning en la agricultura, la clasificación de plagas en el entorno local aún depende en gran medida de la inspección humana, lo que genera errores debido a la subjetividad de los observadores. Esto ha llevado a una notable falta de estudios específicos que aborden la clasificación de plagas del olivo en la región de La Yarada Los Palos, Tacna.

1.2. Descripción del problema

La agricultura es un pilar fundamental de la economía en la región de Tacna, donde el cultivo del olivo tiene un rol significativo tanto en términos de producción como de comercio. Sin embargo, la presencia de plagas representa un desafío crítico que afecta la salud de los cultivos, afectando directamente la rentabilidad de los agricultores y, por ende, la economía local. Especies como Queresa Móvil (*Orthezia olivicola*), el Chanchito o piojo harinoso (*Pseudococcus adonidum*), y la Mosquita blanca del fresno (*Siphoninus phillyreae*) pueden causar daños graves si no se identifican y gestionan a tiempo. Por esta razón, la detección temprana y la clasificación precisa de estas plagas son esenciales para implementar estrategias de control eficaces que minimicen los daños en la producción agrícola.

En este contexto, la aplicación de modelos Machine Learning (ML) ofrece una solución prometedora para la clasificación automática de imágenes de plagas del olivo. Sin embargo, el rendimiento de estos modelos depende en gran medida de configuraciones técnicas como las dimensiones de las imágenes de entrada y el uso de la técnica de aumento de datos. Estas configuraciones son fundamentales para mejorar la capacidad del modelo de generalizar correctamente las plagas y clasificarlas con precisión.

A pesar de su importancia, en la región de Tacna aún no se ha investigado cómo estas configuraciones afectan el desempeño de los modelos de ML en la clasificación de plagas del olivo, lo que deja una brecha de conocimiento que necesita ser abordada. Si estas configuraciones no se optimizan adecuadamente, el modelo podría presentar bajas tasas de precisión o altos márgenes de error, lo que limitaría su efectividad en la práctica agrícola.

En resumen, en esta investigación se comparó el rendimiento de diversas configuraciones del modelo de Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna. Se evaluaron configuraciones como las dimensiones de las imágenes de entrada y el uso de la técnica de aumento de datos, con el fin de identificar las configuraciones óptimas que permitan mejorar la precisión y eficiencia en la clasificación de plagas en los olivares de la región. Una solución eficaz contribuirá a una gestión más eficiente de las plagas y fortalecerá la sostenibilidad agrícola en la región de Tacna.

1.3. Formulación del problema

Por lo expuesto anteriormente, para dar formalidad a la investigación, se plantean los problemas:

1.3.1. Problema general

¿Cómo es el rendimiento de distintas configuraciones del modelo Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna?

1.3.2. Problemas específicos

a) ¿Cómo es el rendimiento modificando las dimensiones de la imagen de entrada del modelo Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna?
b) ¿Cómo es el rendimiento de la técnica aumento de datos del modelo Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna?

1.4. Objetivos de la investigación

1.4.1. Objetivo general

Comparar el rendimiento de distintas configuraciones del modelo Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna.

1.4.2. Objetivos específicos

a) Comparar el rendimiento de diferentes modificaciones de las dimensiones de la imagen de entrada del modelo Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna.
b) Comparar el rendimiento de la técnica aumento de datos del modelo Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna.

1.5. Justificación e importancia de la investigación

La región Tacna se caracteriza por ser primera en la producción de aceitunas y sus beneficios económicos están determinados por la productividad y eficiencia de la rentabilidad alcanzada en los últimos años, contribuyendo significativamente en la economía local y regional. Sin embargo, la incidencia de plagas en los cultivos de olivos presenta un desafío constante, afectando la salud y productividad de los cultivos de olivo.

En ese contexto, el uso de tecnologías avanzadas como Machine Learning puede transformar la manera en que los agricultores gestionan las plagas. La capacidad de clasificar automáticamente imágenes de plagas permite una identificación más rápida y precisa, lo que es esencial para implementar estrategias de control eficaces.

Esta investigación es de suma importancia porque:

Mejora de la productividad agrícola: La optimización de los modelos de ML para la clasificación de plagas permitirá a los agricultores de La Yarada Los Palos, Tacna realizar diagnósticos más precisos y oportunos, mejorando el manejo de plagas y, en última instancia, la productividad, rentabilidad, valor comercial y la calidad de los cultivos.

Relevancia tecnológica: Aportará al campo del Machine Learning en agricultura, especialmente en la región de Tacna, proporcionando información valiosa sobre configuraciones óptimas de modelo Machine Learning para clasificar imágenes de plagas en términos de dimensiones de imagen de entrada y aumento de datos.

Sostenibilidad: Una detección y clasificación más precisa de las plagas permitirá una intervención más focalizada y menos invasiva, lo cual reducirá el uso de productos químicos y fomentará prácticas agrícolas más sostenibles.

1.6. Limitaciones

El conjunto de datos utilizado en este estudio presenta las siguientes limitaciones:

Tamaño del conjunto de datos: Para complementar la cantidad de imágenes de plagas capturadas en La Yarada Los Palos, se buscaron repositorios de imágenes, alcanzando un total de 552. Sin embargo, este número puede ser insuficiente para que el modelo logre una buena generalización a nuevos datos.

Diversidad del conjunto de datos: La ausencia de imágenes de otros tipos de plagas podría reducir la capacidad del modelo para generalizar y clasificar adecuadamente plagas no representadas en el conjunto de datos.

Variabilidad estacional: La ausencia de imágenes recolectadas a lo largo de múltiples temporadas limita la capacidad del modelo para adaptarse a cambios estacionales y variaciones en la aparición de plagas.

Condiciones de captura: Las imágenes pueden haberse tomado en condiciones de iluminación y ángulos de visión limitados, lo que podría impactar negativamente la capacidad del modelo para generalizar situaciones reales. Esta falta de diversidad en las condiciones de captura puede influir en la efectividad del sistema de clasificación.

1.7. Viabilidad del estudio

Este estudio se sostiene sobre un respaldo sólido que garantiza su viabilidad. Hemos recibido todas las facilidades de los tutores, incluyendo acceso completo al área donde se recolectó el conjunto de datos de imágenes de plagas del olivo. Además, se han proporcionado los recursos necesarios y el acceso a la tecnología adecuada para llevar a cabo la investigación de manera efectiva.

La viabilidad del estudio se fortalece con el consentimiento explícito para utilizar estos datos en el entrenamiento de los modelos, lo que asegura el cumplimiento de los estándares éticos pertinentes. También contamos con las herramientas y tecnologías

necesarias para implementar el modelo de Machine Learning, así como con la infraestructura computacional adecuada para su desarrollo eficiente.

Por último, este proyecto cumple con todas las normativas legales vigentes, asegurando un manejo responsable y ético de la información y los resultados obtenidos.

1.8. Formulación de hipótesis

1.8.1. Hipótesis general

H0: No existen diferencias significativas en el rendimiento de distintas configuraciones del modelo Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna.

H1: Existen diferencias significativas en el rendimiento de distintas configuraciones del modelo Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna.

1.8.2. Hipótesis específicas

a) No existen diferencias significativas en el rendimiento del modelo Machine Learning al modificar las dimensiones de la imagen de entrada para la clasificación de plagas del olivo en La Yarada Los Palos, Tacna.

b) No existen diferencias significativas en el rendimiento de la técnica de aumento de datos del modelo Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna.

1.9. Variables

En esta investigación, se han identificado dos variables principales:

Variable 1: Configuración del modelo Machine Learning.

Variable 2: Clasificación de imágenes de plagas del Olivo.

1.10. Operacionalización de variables

Según Arias (2006), la variable es una característica o atributo; magnitud o cantidad, que puede variar y que se encuentra bajo análisis, evaluación, manipulación o dominio en un estudio.

Para ayudar a ilustrar la variable, la siguiente tabla muestra cómo se operacionaliza las variables de la investigación.

Tabla 1

Tabla de operacionalización de variables

Variable	Definición conceptual	Definición operacional	Dimensiones	Indicadores	Escala
V1: Configuración del modelo Machine Learning	Conjunto de métodos computacionales que utilizan la información disponible para hacer predicciones o mejorar su propio rendimiento (Fidalgo, 2021)	Ajustes en técnicas de preprocesamiento y optimización del modelo para mejorar el rendimiento en la clasificación.	- Dimensión de la imagen de entrada	-Número de pixeles	Nominal
			-Técnica de aumento de datos	-Imágenes con uso de DA - Imágenes sin uso de DA	
V2: Clasificación de imágenes de plagas del olivo	La clasificación de imágenes es una técnica de Computer Vision en la que se entrena un modelo para predecir una etiqueta de clase para una imagen en función de su contenido (Microsoft, 2024).	Proceso de identificación correcta de las imágenes de plagas que afectan a los olivos mediante modelos de Machine Learning.	-Rendimiento	-Accuracy	Razón

Nota. La tabla muestra la descripción de las variables.

CAPÍTULO II
MARCO TEÓRICO

2.1. Antecedentes del trabajo de investigación

2.1.1. A nivel internacional

Gavilanez & Saragosin (2024), en su estudio titulado “Desarrollo de un prototipo para la identificación automática de plagas y enfermedades en el cultivo de papa, utilizando técnicas de inteligencia artificial en la ciudad de Latacunga”, abordaron la identificación de plagas y enfermedades en el cultivo de papa mediante técnicas de Inteligencia Artificial. Para el desarrollo y entrenamiento del modelo, emplearon la metodología KDD y herramientas de aprendizaje automático. Durante el procedimiento de entrenamiento y prueba del modelo, utilizaron 2501 imágenes de plagas y enfermedades basadas en fotografías, que se clasificaron en cuatro categorías: hojas sanas, hojas con la plaga pulguilla de la papa, hojas con tizón temprana y tardío. Para determinar el mejor modelo, analizaron cinco arquitecturas basadas en aprendizaje por transferencia: ResNet50, VGG19, InceptionV3, EfficientNetB0 y MobileNet. Además, crearon su propio modelo, que alcanzó una precisión del 95,34%, el mismo que finalmente se incorporó en un prototipo de software destinado a detectar automáticamente plagas y enfermedades en los campos de cultivo de papa.

Salehin et al. (2024), en el artículo: BAU-Insectv2: An agricultural plant insect dataset for deep learning and biomedical image analysis, tuvo como objetivo aprovechar las metodologías de aprendizaje profundo, empleando redes neuronales convolucionales (CNN) y técnicas avanzadas de análisis de imágenes para la detección precisa de insectos, la clasificación y la comprensión de los patrones relacionados con los insectos dentro de los ecosistemas agrícolas, para ello el estudio nos ofrece un conjunto de datos agrícolas “BAU-Insectv2” que varía entre 200 y 295 muestras por género y que abarcan una colección diversa de imágenes de alta resolución que captan detalles complicados de las interacciones planta-insecto en diversos entornos y la utilidad del conjunto de datos se extendía al análisis de imágenes biomédicas, fomentando vías de investigación interdisciplinares entre la agricultura y las ciencias biomédicas. Se centran

principalmente en abordar los problemas relacionados con los insectos en los cultivos del Sur de Asia.

Tannous et al. (2023), en el artículo: A Deep-Learning-Based Detection Approach for the Identification of Insect Species of Economic Importance, tuvo como objetivo desarrollar un enfoque de clasificación basado en aprendizaje automático para reconocer especies de insectos de importancia económica. Se utilizaron como organismos modelo dos especies de plagas tefrítidos con forma y patrones locomotoras similares (p. ej., la mosca del Mediterráneo Ceratitis Capitata y la mosca del olivo Bactrocera Oleae). Desarrollaron un método de detección basado en una Red Neuronal Convolucional (CNN) que puede clasificar con precisión en tiempo real dos especies de tefrítidos libres para moverse y cambiar de postura. Los resultados mostraron una detección automática exitosa (es decir, una tasa de precisión de alrededor del 93%) en tiempo real de aductos de C. Capitata y B. Oleae utilizando un sensor de cámara a una altura fija. Estos resultados contribuyeron de manera importante al desarrollo de métodos autónomos de monitoreo de plagas, para intervenir con medidas personalizadas de manera instantánea y remota, promoviendo enfoques de protección de cultivos sostenibles y eficientes basados en el manejo integrado de plagas y técnicas de precisión.

Aguilar & Campoverde (2020) en su estudio llamado “Clasificación de frutas basadas en redes neuronales convolucionales”, se centra en la clasificación de frutas utilizando redes neuronales convolucionales, abordando aspectos como la captura y procesamiento de imágenes de frutas, el entrenamiento de la red neuronal, la extensión del conjunto de datos para incluir más clases de variantes de frutas, y la importancia de la resolución de las imágenes para la eficiencia del modelo neuronal. El proceso de la investigación consistió en capturar imágenes de frutas, procesar las imágenes, entrenar la red neuronal y validar el modelo entrenado. Como resultado, se logró obtener un 87% de eficiencia en la clasificación de frutas utilizando este enfoque. Además, se mencionan otros estudios relacionados con la clasificación de frutas y plantas utilizando técnicas de inteligencia artificial.

Hernández (2021) en su tesis titulada “Detección de plagas y enfermedades del limón persa mediante una aplicación móvil con aprendizaje profundo a partir de redes neuronales”, se centra en la detección de plagas y enfermedades en los limones persas utilizando una aplicación móvil basada en redes neuronales convolucionales y

aprendizaje profundo. El marco teórico describe la metodología de desarrollo móvil, así como las metodologías de desarrollo ágil y tradicional. En cuanto a los modelos utilizados, se aplicó el aprendizaje por transferencia utilizando un modelo previamente entrenado (MobileNet V2) del repositorio de Tensorflow. Durante el entrenamiento, el modelo alcanzó una precisión alta, llegando al 98%, pero en la validación solo obtuvo un valor del 87%, que luego disminuyó al 84%. Después de su desarrollo, la aplicación se sometió a pruebas en un entorno real. Aunque cumplió con su función, el margen de error en las predicciones se considera alto, aproximadamente del 15%, principalmente debido a la falta de datos para el entrenamiento del modelo de aprendizaje profundo.

2.1.2 A nivel nacional

Burgos (2023), en su estudio titulado "Detección y clasificación de plaga pulguilla en el cultivo de aguaymanto mediante redes neuronales", tuvo como objetivo crear y evaluar un sistema basado en redes neuronales para identificar la plaga pulguilla en los campos de aguaymanto. Para ello, utilizaron técnicas de visión artificial para el procesamiento y análisis de imágenes que fueron tomadas de los cultivos, generando un conjunto de datos compuesto por 1,666 imágenes de hojas de aguaymanto, tanto afectadas como no afectadas. Se probaron varios algoritmos de clasificación, destacando la eficacia de las redes neuronales convolucionales (CNN) y profundas. Los resultados mostraron una precisión del 88% para las redes neuronales profundas y del 81-83% para las redes neuronales convolucionales en la clasificación binaria.

Córdova (2021), en su investigación denominada: "Aplicación de aprendizaje profundo para la detección y clasificación automática de insectos agrícolas en trampas pegantes", realizada en la Pontificia Universidad Católica del Perú, en Lima-Perú, se propuso emplear los modelos de detección preentrenados Faster R-CNN y YOLOv4 y aplicar aprendizaje por transferencia para ajustarlos al reto de identificar tres tipos de plagas de insectos: mosca blanca, mosca minadora y pulgón verde del melocotonero en sus estados adultos alados. Debido a la limitada disponibilidad de un conjunto de imágenes de trampas pegantes con plagas, utilizaron un generador de imágenes realistas de trampas pegajosas de insectos, que tiene en cuenta factores realistas como la iluminación y el nivel de ruido de las imágenes, además, utilizaron técnicas de segmentación y ampliación de imágenes, para crear un conjunto de imágenes adecuadas para la fase de entrenamiento. Por último, determinaron la métrica mAP para cada uno de

los tres tipos de plagas, utilizando ambos modelos. El modelo Faster R-CNN obtuvo un 94,06%, mientras que el modelo YOLOv4 obtuvo un 95,82%, concluyendo que el rendimiento de ambos detectores es aceptable.

Gomez & Sandoval (2020), en su investigación denominada: "Una revisión: Uso de imágenes satelitales para la detección de plagas y enfermedades en cultivos", realizada en la Universidad César Vallejo, en Lima-Perú, se enfocaron en explicar los procesos metodológicos para procesar imágenes satelitales con el objetivo de detectar plagas y enfermedades en cultivos. Los resultados mostraron que el coeficiente de determinación R2 del SR es el más alto, alcanzando un 89.00%. Del mismo modo, los índices PRI, NDVI y GNDVI se identificaron como los más empleados en los estudios para la detección de plagas y enfermedades, obteniendo porcentajes de uso del 70%, 50% y 50%, respectivamente. En cuanto a los algoritmos de clasificación, el SVM obtuvo un coeficiente de determinación R2 más alto, llegando al 92.73%, mientras que el ANN alcanzó un 90.64%. Por otro lado, en cuanto a los algoritmos de segmentación, el HSI mostró un rendimiento sobresaliente con un 95.23%, seguido por el umbral de Otsu con un 89.10% y el K-means Clustering con un 83.55%. En consecuencia, los estudios demuestran que los índices de vegetación como REDSI, PRI, PRSSa, VARI y NDVI pueden clasificar eficazmente las plagas y enfermedades gracias a su elevado coeficiente de determinación R2. Además, los enfoques de segmentación más utilizados son los algoritmos de clasificación ANN y SVM, mientras que el HSI destaca por su precisión en la detección de plagas y enfermedades en los cultivos.

Gallardo et al. (2019), en su investigación denominada "Identificación de plagas en el cultivo de Olivo utilizando percepción remota con vehículo aéreo no tripulado en el Valle la Yarada los Palos", utiliza la percepción remota con un vehículo aéreo no tripulado para la identificación de plagas. La primera etapa consiste en acondicionar el vehículo aéreo no tripulado, que consta de una estructura de vuelo y un sistema de cámaras que facilita la recogida de datos, en este caso fotografías aéreas. Posterior al acondicionamiento y la ejecución de planes de vuelo, las imágenes se procesaron mediante dos algoritmos: SIFT, BRISK, SURF y FREAK, resultando el algoritmo SIFT, que tiene más puntos de interés, pero tarda más en encontrar una zona común, frente al algoritmo FREAK, que graba más rápido, pero encuentra menos puntos de interés.

Además, se determinó que la identificación de plagas como la Orthezia Olivícola se realiza con mejor precisión mediante la clasificación supervisada.

Delgado & Obeso (2019), en su investigación denominada: "Solución de Machine Learning en el reconocimiento de plagas para plantones de arándano", realizada en la Universidad Privada del Norte, en Trujillo-Perú, tuvieron como propósito determinar el impacto de una solución de Machine Learning en la detección de plagas para plantones de arándano en Trujillo. El estudio fue cuasi-experimental, utilizando una muestra de diez evaluadores. Las dimensiones del reconocimiento de plagas en plantones de arándano fueron sensibilidad y especificidad, mientras que las dimensiones en la solución de aprendizaje automático fueron adecuación funcional y usabilidad. La certeza media en el reconocimiento de plagas para plantones de arándano fue del 90.14% cuando se utilizó la solución de aprendizaje automático, concluyendo que una solución de aprendizaje automático tiene un efecto beneficioso en el reconocimiento de plagas para plantones de arándanos.

2.2. Bases teóricas

2.2.1 Machine Learning

Según Fidalgo (2021), el Machine Learning, también conocido como aprendizaje de máquina o aprendizaje automático, es una rama de la inteligencia artificial. Se define como un conjunto de métodos computacionales que utilizan la información disponible para hacer predicciones o mejorar su propio rendimiento. Esta información proviene de grandes volúmenes de datos recolectados, conocidos como Big Data. A partir de estos datos, los algoritmos pueden aprender patrones y proporcionar resultados precisos.

Según Samuel (1959), el Machine Learning se define como la habilidad de las máquinas para adquirir conocimiento sin necesidad de programación explícita. Esto se logra mediante el uso de algoritmos que posibilitan la mejora continua del rendimiento a medida que se les suministra más información.

Tipos de Machine Learning

Según Pereyra (2020), hay distintos tipos de aprendizaje en Machine Learning, entre las que sobresalen el aprendizaje supervisado, el no supervisado, el semisupervisado

y el aprendizaje por refuerzo. Estas se diferencian en la manera en que el algoritmo adquiere conocimiento.

a) Aprendizaje supervisado

En el aprendizaje supervisado, los algoritmos trabajan con datos “etiquetados”, con el objetivo de identificar una función que, dada una entrada (datos de entrada), pueda asignar correctamente un resultado. Estos algoritmos se entrenan con un historial de datos para "aprender" patrones y reglas que les permitan hacer inferencias sobre nuevos datos de entrada.

El aprendizaje supervisado se emplea comúnmente en:

- Problemas de clasificación, en los cuales la predicción resulta en una o varias etiquetas o clases discretas.
- Problemas de regresión, donde la predicción resulta en un valor o cantidad continua.

b) Aprendizaje no supervisado

El aprendizaje no supervisado ocurre cuando no se cuenta con datos "etiquetados" para el proceso de entrenamiento, ya que obtener estas etiquetas puede resultar difícil, no se posee información previa sobre los datos, o el etiquetado implica costos significativos.

c) Aprendizaje semisupervisado

Es una técnica que combina elementos del aprendizaje supervisado y no supervisado, implicando la necesidad de tener un conjunto de datos con etiquetas y otro sin etiquetar. Esta aproximación se vuelve especialmente valiosa cuando etiquetar los datos resulta una tarea que consume mucho tiempo o es intensiva en esfuerzo. El procedimiento del aprendizaje semi supervisado se puede esquematizar de la siguiente manera:

1. Recolectar datos con resultados conocidos.
2. Recolectar datos sin resultados asociados.
3. Construir un modelo que aprenda de manera supervisada los datos con resultados.
4. Utilizar dicho modelo para etiquetar automáticamente los datos restantes (sin etiquetas).

5. Desarrollar un nuevo modelo que se entrene con aprendizaje supervisado utilizando tanto los datos etiquetados inicialmente como los datos etiquetados automáticamente.
6. Aplicar el nuevo modelo a datos adicionales.

d) Aprendizaje por refuerzo

Es una técnica que persigue mejorar el resultado de un problema mediante el método de prueba y error. Cada problema de aprendizaje por refuerzo consta de un agente (algoritmo) y su entorno. El agente, caracterizado como una entidad con habilidades de memoria y deducción, tiene como objetivo entrenarse en el entorno hasta lograr un rendimiento óptimo. Por otro lado, el entorno representa el problema que se debe resolver, el contexto con el cual interactúa el agente y sirve como su fuente de información; este entorno está organizado como una secuencia de alternativas. El procedimiento de aprendizaje por refuerzo es el siguiente: en cada alternativa, el agente ejecuta una acción y recibe una recompensa como consecuencia. Con la acumulación de recompensas, el conocimiento del agente sobre el entorno aumenta. El proceso de aprendizaje concluye cuando el agente logra identificar una secuencia de acciones, entre las muchas posibles, que le proporciona la mayor recompensa acumulada.

2.2.2 Redes Neuronales Convolucionales (CNN)

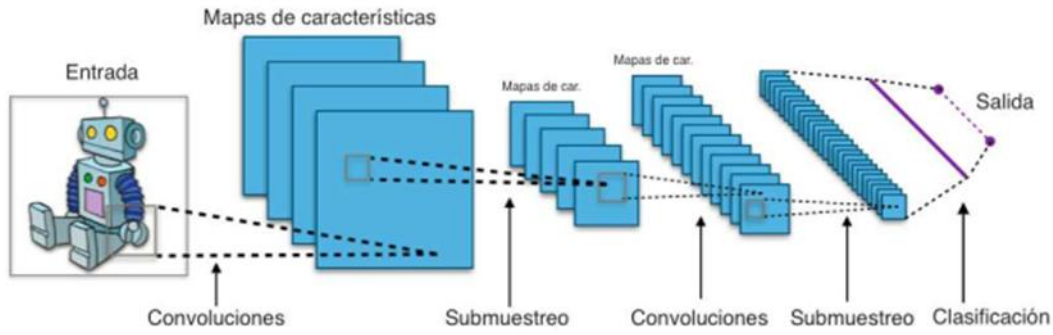
De acuerdo a Durán (2017), se define una red neuronal convolucional como una red multicapa formada por numerosas capas convolucionales y de pooling (sub muestreo) alternadas, que culminan en una secuencia de capas full-connected similar a una red perceptrón multicapa. A continuación, en los siguientes puntos, profundizaremos en las estructuras de las redes neuronales, explicando específicamente el esquema general de una red neuronal convolucional y realizando análisis en profundidad de cada una de sus capas.

a). Esquema general de una Red Neuronal Convolucional

En la Figura 1 se muestra un diseño básico de las capas que conforman una Red Neuronal Convolucional, ofreciendo una visión general de su estructura.

Figura 1

Esquema general de una Red Neuronal Convolucional (CNN)



Nota: Adaptado de Durán Suárez, J. (2017). *Redes Neuronales Convolucionales en R - Reconocimiento de caracteres escritos a mano*.

b). Capa de entrada

La primera capa de una red convolucional, conocida como capa de entrada, tiene como objetivo recibir la imagen inicial y prepararla para su procesamiento en las capas posteriores.

b). Capa convolucional

Es una de las capas fundamentales para el procesamiento de imágenes. Esta capa aplica filtros (también conocidos como kernels) a la imagen de entrada mediante la operación de convolución. Esta operación consiste en deslizar el filtro sobre la imagen de entrada y calcular la suma ponderada de los píxeles correspondientes, lo que facilita la detección de características locales en la imagen.

b). Capa de Pooling (Submuestreo)

La capa de pooling, también conocida como capa de submuestreo, la función principal de la capa de pooling es reducir la dimensionalidad de las características extraídas por las capas convolucionales, lo que ayuda a controlar el sobreajuste y a mejorar la eficiencia computacional del modelo.

b). Capa full-connected

La capa fully connected, también conocida como capa densa, es la capa final que se utiliza para clasificar la imagen de entrada en una de las clases predefinidas. Esta capa

se sitúa inmediatamente después de la última capa de agrupación y está formada por el mismo número de neuronas que de clases.

En la capa fully connected, cada neurona está conectada a todas las neuronas de la capa anterior, lo que significa que cada neurona de esta capa recibe información de todas las características extraídas por las capas convolucionales y de pooling anteriores. Finalmente, esta capa utiliza la información de las características para realizar la clasificación final de la imagen de entrada.

2.2.3 Dimensión de imagen de entrada

Según García y Romero (2020), las dimensiones de la imagen, o el tamaño de la capa de entrada, afectan directamente al rendimiento de una Red Neuronal Convolucional (CNN). De acuerdo con De Luca et al. (2021), cada imagen se representa como una matriz tridimensional que debe ser procesada mediante convolución a través de un kernel. Este proceso depende en gran medida de las dimensiones de la imagen de entrada y de la configuración de la operación de convolución.

García y Romero (2020) señalan que utilizar dimensiones más grandes conlleva varias consecuencias:

a) Mayor tiempo de entrenamiento,

b) Mayor consumo de memoria debido al tamaño del modelo,

c) Incremento en el tiempo de procesamiento al aplicar el modelo en un sistema real.

Por otro lado, las dimensiones más pequeñas limitan la capacidad de aumentar el tamaño de las capas convolucionales en la arquitectura, debido a la reducción progresiva de las dimensiones resultante de las operaciones propias de las CNNs.

Para calcular el número total de píxeles en una imagen, se debe multiplicar sus dimensiones: el alto (o altura) por el ancho. Es decir, se multiplican los píxeles verticales por los horizontales (Pechuan & Rosso, 2015).

2.2.4 Técnica de aumento de datos de Keras

Según Brownlee (2019), el aumento de datos de imagen es una técnica que se puede utilizar para expandir artificialmente el tamaño de un conjunto de datos de entrenamiento mediante la creación de versiones modificadas de imágenes en el conjunto de datos.

El aumento de datos es beneficioso en el aprendizaje automático, ya que nos permite aumentar artificialmente el tamaño y la diversidad de nuestro conjunto de datos, mejorando la capacidad de los modelos para generalizar (Rosebrock, 2019).

De acuerdo con Rosebrock (2019), existen tres tipos principales de aumento de datos, los cuales son:

a) Generación de conjuntos de datos y expansión de datos mediante aumento de datos (menos común)

Los modelos de aprendizaje automático, especialmente las redes neuronales, suelen requerir grandes cantidades de datos de entrenamiento. Pero, ¿qué sucede si no se dispone de suficientes datos en primer lugar?

Consideremos el caso más simple: si solo tienes una imagen y deseas aplicar aumento de datos para crear un conjunto completo de imágenes, todas basadas en esa única imagen, el proceso sería el siguiente:

1. Carga la imagen original desde el disco.
2. Aplica transformaciones aleatorias a la imagen original, como traslaciones, rotaciones, etc.
3. Guarda la imagen transformada nuevamente en el disco.
4. Repite los pasos 2 y 3 un total de N veces.

Al completar este proceso, habrás generado un directorio lleno de imágenes "nuevas" que son transformaciones aleatorias de la imagen original, las cuales podrás usar para el entrenamiento.

Es probable que tengas más de una imagen. En ese caso, si cuentas con decenas o cientos de imágenes, el objetivo sería convertir ese conjunto pequeño en miles de imágenes para entrenamiento.

En estas situaciones, puede ser útil explorar técnicas de expansión y generación de conjuntos de datos.

Sin embargo, hay un inconveniente con este enfoque: no hemos incrementado realmente la capacidad de generalización de nuestro modelo. Si bien hemos aumentado el número de ejemplos de entrenamiento, todos estos ejemplos siguen basándose en un conjunto de datos muy limitado.

b) Aumento de datos in situ y sobre la marcha (más común)

El segundo tipo de aumento de datos se denomina aumento de datos in situ o aumento de datos sobre la marcha. Este tipo de aumento es implementado por la clase ImageDataGenerator en Keras.

Al utilizar este enfoque, nuestro objetivo es garantizar que, durante el entrenamiento, la red vea nuevas variaciones de los datos en cada una de las épocas.

Hay dos puntos importantes que quisiera destacar:

1. El ImageDataGenerator no devuelve tanto los datos originales como los transformados: la clase solo devuelve los datos transformados aleatoriamente.
2. Llamamos a esto aumento de datos "en el lugar" y "sobre la marcha" porque las transformaciones se realizan en el momento del entrenamiento (es decir, no generamos los ejemplos de antemano).

Cuando nuestro modelo está siendo entrenado, podemos pensar en la clase ImageDataGenerator como un mecanismo que "intercepta" los datos originales, los transforma aleatoriamente y luego los devuelve a la red neuronal para su entrenamiento, mientras que la red neuronal no tiene idea de que los datos han sido modificados.

Es importante señalar que el objetivo de la técnica de aumento de datos descrita en esta sección es asegurar que la red vea imágenes "nuevas" que nunca antes había "visto" durante cada época de entrenamiento.

Si incluyéramos los datos de entrenamiento originales junto con los datos aumentados en cada lote, la red "vería" los mismos datos de entrenamiento varias veces, lo cual frustraría el objetivo. Además, es crucial recordar que el propósito del aumento de datos es mejorar la capacidad de generalización del modelo. Para lograrlo, "reemplazamos" los datos de entrenamiento con versiones aumentadas y transformadas aleatoriamente.

En la práctica, esto lleva a un modelo que suele funcionar mejor con los datos de validación/prueba, aunque quizás un poco peor con los datos de entrenamiento (debido a las variaciones generadas por las transformaciones aleatorias).

c) Combinación de generación de conjuntos de datos y ampliación in situ

El último tipo de aumento de datos combina la generación de conjuntos de datos y el aumento in situ. Este enfoque se puede observar en la clonación de comportamiento.

Un ejemplo destacado de clonación de comportamiento se encuentra en las aplicaciones de vehículos autónomos.

La creación de conjuntos de datos para entrenar automóviles autónomos puede ser un proceso costoso y que consume mucho tiempo. Una posible solución a este desafío es el uso de videojuegos y simuladores de conducción.

Los gráficos de los videojuegos han alcanzado un nivel de realismo tal que ahora pueden utilizarse como datos de entrenamiento. Así, en lugar de conducir un vehículo real, puedes:

1. Jugar un videojuego.
2. Desarrollar un programa para jugar al videojuego.
3. Utilizar el motor de renderizado del videojuego.

Este proceso permite generar datos reales que pueden usarse para el entrenamiento de modelos de vehículos autónomos.

Una vez que se han recopilado los datos de entrenamiento, puedes aplicar el aumento de datos de tipo 2 (es decir, aumento in situ o sobre la marcha) a los datos obtenidos a través de la simulación.

2.2.5 Transfer Learning

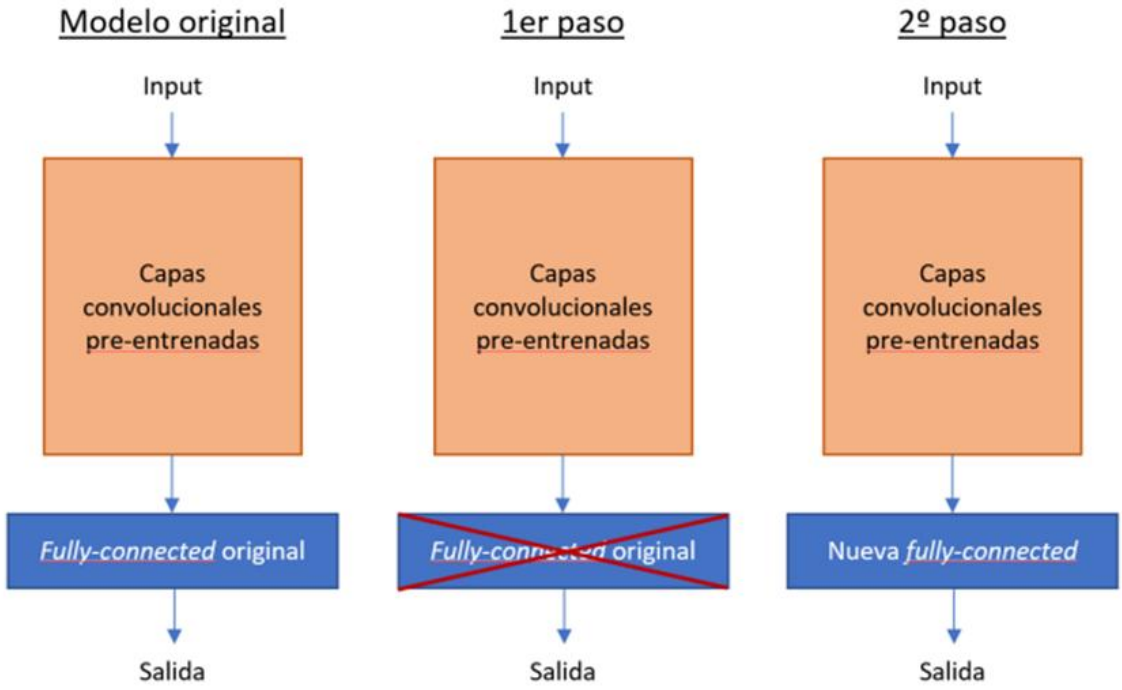
De acuerdo a Santos (2021), la técnica de Aprendizaje por Transferencia consiste en reutilizar un modelo preentrenado con una gran cantidad de imágenes para abordar problemas de clasificación o reconocimiento de imágenes cuando se dispone de pocos datos de entrenamiento, con el fin de obtener resultados satisfactorios en una Red Neuronal sin comenzar desde cero. Al emplear un modelo preentrenado a uno nuevo por desarrollar, debemos tener en cuenta que las capas del primer modelo ya han sido entrenadas; al tratarse de una red neuronal convolucional, la estructura y comportamiento de cada capa será similar a cualquier otra red neuronal que desarrollemos.

Por eso, cuando se despliega una nueva red sobre una ya existente, el primer paso es eliminar la red totalmente conectada (full connected) de la red preentrenada y sustituirla por una nueva que aprenda de las nuevas imágenes de entrenamiento. También es necesario congelar todas las capas de la red preentrenada para evitar el reentrenamiento. Esto garantiza que se conserven todos los mapas de características predeterminados, ya que son bastante genéricos y valiosos para cualquier nuevo procedimiento de clasificación de imágenes.

A continuación, en la Figura 2, se presenta una representación simplificada y sencilla de los procedimientos descritos anteriormente, lo que facilita su comprensión y visualización.

Figura 2

Proceso de Transfer Learning



Nota: Adaptado de Santos, A. (2021). *Clasificación de logos de marcas mediante Deep Learning*. http://hdl.handle.net/20.500.12880/779

2.2.6 Puntuación de precisión

La función accuracy_score calcula la precisión, que es la fracción (por defecto) o el recuento (cuando normalizar=False) de las predicciones correctas (scikit-learn, s.f.).

En la clasificación de etiquetas múltiples, la función devuelve la precisión del subconjunto. Si todo el conjunto de etiquetas predicho para una muestra coincide estrictamente con el conjunto de etiquetas verdadero, entonces la precisión del subconjunto es 1.0; de lo contrario, es 0.0 (scikit-learn, s.f.).

Si $\hat{y}_i$ es el valor predicho de el i– muestra y y_i es el valor verdadero correspondiente, entonces la fracción de predicciones correctas terminó $n_{samples}$ es definido como:

$$accuracy(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} 1(\hat{y}_i = y_i) \quad (1)$$

donde 1(x) es la función indicadora.

2.2.7 Clasificación de imágenes de plagas del olivo

2.2.7.1 Clasificación de imágenes

La clasificación de imágenes es una técnica de Computer Vision en la que se entrena un modelo para predecir una etiqueta de clase para una imagen en función de su contenido (Microsoft, 2024).

2.2.7.2 Plagas del Olivo

A continuación, se presentan las principales plagas que afectan a los olivos en La Yarada Los Palos, Tacna.

a). Queresa móvil

En este insecto "queresa móvil" de nombre científico: Orthezia olivicola (Hem.: Ortheziidae), el cuerpo de las hembras adultas tiene forma oval, provista de una cola denominada ovisaco, de forma cilíndrica, encorvada hacia arriba y de 11 mm de longitud. Las hembras tienen la particularidad de ser partenogenética; es decir, pueden multiplicarse sin la intervención del macho, originando poblaciones entre 82 a 116 individuos (Casanova, 2022).

Figura 3

Plaga Orthezia olivicola atacando hojas de olivo con presencia de Fumagina en Tacna.

Nota: Elaboración propia

Ciclo biológico

De acuerdo con Casanova (2022), el ciclo de vida de la "Queresa móvil" u "Orthezia" es de 103 a 185 días.

- Periodo de ninfa: 43 a 75 días
- Longevidad de adultos: 60 a 110 días
- De generación a generación: 64 a 84 días

Los ciclos cortos de desarrollo se manifiestan en épocas calurosas de noviembre a febrero, cuyas temperaturas máximas promedio, fluctúan entre 23 °C a 29 °C, mientras que los más largos en épocas de frío de mayo a agosto, con temperaturas mínimas de 14 °C a 12 °C. (SENAMHI, 2019). Este insecto succiona la savia de las hojas y excreta miel que sirve de medio de cultivo para el hongo conocido como "fumagina" Capnodium sp. que ennegrece las hojas del árbol, afectando la capacidad fotosintética de la planta. La "fumagina" produce daños en los frutos con un manchado, ocasionando la calidad y pérdida del valor comercial

Daño

De acuerdo a Casanova (2022), los daños de esta plaga son:

- El oscurecimiento de las plantas por la presencia de la fumagina y de mielecilla en la estructura vegetativa aérea de la planta.
- Este insecto por las características que presentan sus daños se pueden distinguir rápidamente, manifestándose por el lado en donde la planta recibe los rayos del sol por las mañanas.

b). Chanchito o piojo harinoso

Esta plaga denominada "piojo harinoso" o "chanchito" Pseudococcus adonidum (Hem.: Pseudococcidae), es un insecto que produce daño en diversas plantas cultivadas, como el olivo. La hembra adulta presenta 3.5 mm de tamaño y son ovovivíparas; es decir, no depositan sus huevos fuera de su cuerpo, sino que los almacena en su interior, saliendo directamente de la madre como ninfas, las mismas que son ubicadas en la unión del tronco con las ramas. Los machos son de menor tamaño y presentan alas. Se detecta la presencia de esta plaga por presentar abundante mielecilla en las hojas y en los frutos durante en épocas de calor (diciembre a marzo). En las épocas de frío el insecto se aloja en el suelo (Casanova, 2022).

Figura 4

Plaga Piojo Harinoso atacando hojas de olivo en Tacna.

Nota: Elaboración propia

Daño

De acuerdo a Casanova (2022), los daños de esta plaga son:

- Directos, debilitamiento de las plantas por succión de la savia en los brotes, hojas y frutos.
- Indirectos, mediante las excreciones que favorecen la presencia y desarrollo del hongo denominado “fumagina” (Capnodium sp.), que ocasiona el manchado de los frutos disminuyendo, su calidad y valor comercial, así mismo en daños severos ocasiona su caída.

c). Mosquita blanca del fresno

Esta es una especie del grupo de las moscas blancas, Siphoninus phillyreae (Hall.) (Hem.: Aleyrodidae) es muy pequeño, mide aproximadamente 2 mm de longitud en estado adulto, vive agrupado en colonias, principalmente en la zona adaxial (envés) de las hojas en todos sus estados (huevos, ninfas, pupas y adultos) (Casanova, 2022).

Los huevos son pequeños, pedicelados y oblongos. Al poco tiempo eclosionan las ninfas. El primer estadío es el único móvil, después de unas horas de desplazamiento, se establece en un punto de la hoja, pierde las patas, permaneciendo inmóvil por el resto de este período. Las ninfas pasan por cuatro estadios. Presentan una fecundidad de: 66-140 huevos/hembra. Las ninfas y pupas tienen 40 a 50 espinas tubiformes que producen una

gran cantidad de cera que puede llegar a cubrir el insecto. Las pupas miden 0.7 a 0.8 mm de longitud por 0.5 mm de ancho. Las ninfas y adultos presentan una banda de cera gruesa y finos tubos con una gota de cera en su extremo (Casanova, 2022).

Figura 5

Plaga Mosquita blanca del fresno atacando hojas de olivo en Tacna.

Nota: Elaboración propia

Ciclo biológico

De acuerdo a Casanova (2022), el desarrollo de este insecto ocurre entre los 10 y 30 °C, con una temperatura óptima de 25 °C. El adulto alado coloca sus huevos en el envés de las hojas; luego aparecen las ninfas, las que se alimentan de la savia de los árboles hasta que empupan sobre las hojas.

En relación al ciclo biológico de la mosquita blanca del fresno en condiciones de laboratorio a una temperatura de 15.7 °C y con una humedad relativa de 79.2 %, el ciclo promedio es de:

- Huevo a adulto, es de 72 días.
- La incubación de huevos comprende 20 días y con un desarrollo ninfal de 52 días.

Daño

De acuerdo a Casanova (2022), los daños de esta plaga son:

- Altas poblaciones de ninfas causan la caída prematura de las hojas y reducen severamente los rendimientos en árboles frutales.
- En algunos casos, causan la muerte de los árboles jóvenes.

2.3. Definiciones conceptuales

2.3.1 Arquitectura VGG-16

La arquitectura VGG-16, también conocida como Visual Geometry Group-16, es un modelo de aprendizaje profundo creado por investigadores de la Universidad de Oxford. Esta red neuronal convolucional (CNN) está diseñada específicamente para tareas de clasificación y reconocimiento de imágenes. VGG-16 es conocido por su simplicidad y complejidad computacional relativamente menor en comparación con otras arquitecturas de aprendizaje profundo como ResNet e Inception. La arquitectura consta de un total de 16 capas de peso, incluidas 13 capas convolucionales, 3 capas completamente conectadas y 5 capas de agrupación máxima (Gayathri et al., 2023).

2.3.2 Aprendizaje

Se refiere al proceso mediante el cual un algoritmo o modelo estadístico ajusta sus parámetros o estructura para hacer predicciones o tomar decisiones basadas en datos de entrada.

2.3.3 Training Dataset

La muestra de datos utilizada para ajustarse al modelo (Shah, 2017).

2.3.4 Validation Dataset

La muestra de datos utilizada para proporcionar una evaluación imparcial del ajuste de un modelo en el conjunto de datos de entrenamiento mientras se ajusta los hiperparámetros del modelo (Shah, 2017).

2.3.5 Test Dataset

La muestra de datos utilizada para proporcionar una evaluación imparcial de un ajuste final del modelo en el conjunto de datos de entrenamiento (Shah, 2017).

2.3.6 Adam (Adaptive Moment Estimation)

Es un algoritmo de optimización ampliamente utilizado en el campo del Machine Learning y Deep Learning. Es una variante del descenso de gradiente estocástico (SGD)

y combina conceptos de otros algoritmos de optimización para mejorar la eficiencia y la velocidad de convergencia.

2.3.7 Época

Una época consiste en un ciclo completo de entrenamiento, sobre todo el conjunto de datos, lo que implica que cada ejemplo ha sido procesado una vez. En otras palabras, una época abarca N/iteraciones de entrenamiento, donde N representa el número total de instancias.

2.3.8 Overfitting

Es un fenómeno común en Machine Learning que ocurre cuando un modelo se ajusta demasiado bien a los datos de entrenamiento y, como resultado, tiene un rendimiento deficiente en datos no vistos o nuevos. En otras palabras, el modelo memoriza los detalles y el ruido presentes en el conjunto de entrenamiento en lugar de aprender patrones generales que se pueden aplicar a datos más amplios.

2.3.9 JupyterLab

JupyterLab es una herramienta ampliamente empleada en programación y ciencia de datos a través de cuadernos computacionales o "notebooks" (Olney & Fleming, 2021).

2.3.10 CUDA

Ofrece un entorno de desarrollo diseñado para crear aplicaciones de alto rendimiento aprovechando la aceleración mediante la unidad de procesamiento gráfico. Incluye bibliotecas para la aceleración con GPU, así como herramientas para depuración y optimización (NVIDIA, n.d.).

CAPÍTULO III
MARCO METODOLÓGICO

3.1. Planteamiento metodológico

De acuerdo a Sánchez & Reyes (2015), el método es el camino a seguir mediante una serie de operaciones y reglas prefijadas que nos permiten alcanzar un resultado o un objetivo. En resumen, es la ruta para alcanzar un objetivo o un fin.

3.1.1. Tipo de investigación

Según Sánchez & Reyes (2015), la investigación aplicada se caracteriza por su interés en la aplicación de los conocimientos teóricos a determinada situación concreta.

La presente investigación se clasifica como aplicada, ya que emplea estos conocimientos teóricos para comparar el rendimiento de distintas configuraciones del modelo Machine Learning en la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna.

3.1.2. Nivel de investigación

Según Sánchez et al. (2018), los estudios de nivel comparativo son investigaciones que se centran en comparar dos o más muestras, seleccionadas de manera específica, con el objetivo de identificar y analizar las similitudes y diferencias en relación con una o varias variables clave.

La presente investigación es de nivel comparativo, ya que se centra en comparar el rendimiento de distintas configuraciones del modelo de Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna.

3.1.3. Diseño de investigación

Según Hernández Sampieri et al. (2014), una investigación de diseño no experimental se refiere a estudios en los que el investigador no manipula las variables,

si no que observa y analiza la realidad tal como se presenta. En este tipo de investigación, se busca establecer relaciones entre variables sin intervenir en el entorno, lo que permite una descripción y análisis más natural de los fenómenos.

En ese contexto, la investigación que nos ocupa se enmarca en el diseño de investigación no experimental, ya que no se manipuló las variables y se observarán los hechos tal como se presentan en su contexto natural para su análisis.

3.2. Población y muestra

3.2.1. Población

La población se define como el conjunto de todos los casos que cumplen con ciertas características específicas (Hernández Sampieri et al., 2014, p. 174).

En el presente estudio la población de estudio es de 552 imágenes capturadas de tres tipos de plagas en los cultivos de La Yarada Los Palos, ubicados en la región de Tacna, Perú: la Queresa móvil (Orthezia olivicola) en su fase de hembra con saco ovígero completamente desarrollado, el chanchito o piojo harinoso (Pseudococcus adonidum) en su fase de Ninfa III y hembra adulta, y la mosquita blanca del fresno (Siphoninus phillyreae) en su fase adulta, como se muestra en la Figura 6. Además, se incluyeron datos obtenidos de otros recursos en línea.

Figura 6

Población de estudio compuesta por tres tipos de plagas

Nota: Imágenes de la plaga Queresa Móvil, Chanchito o Piojo Harinoso, Mosquita Blanca del Fresno.

Los resultados indican que 198 imágenes corresponden a la plaga Queresa Móvil, 186 imágenes representan la plaga Chanchito o Piojo Harinoso, y 168 imágenes pertenecen a la plaga Mosquita Blanca del Fresno. Todas estas imágenes se guardaron en formato JPG para que puedan ser utilizadas en el entrenamiento.

Tabla 2
Población de estudio

Descripción	Cantidad	Cantidad
Queresa Móvil	198	35.87%
Chanchito o Piojo Harinoso	186	33.70%
Mosquita Blanca del Fresno	168	30.43%
Total	552	100%

Fuente. Elaboración propia.

3.2.2. Muestra

La muestra es un subgrupo de la población de interés sobre el cual se recolectarán datos, y que tiene que definirse y delimitarse de antemano con precisión, además de que debe ser representativo de la población (Hernández Sampieri et al., 2014, p. 173).

De acuerdo a Hernández Sampieri et al. (2014), no siempre se trabaja con una muestra en una investigación, pero en la mayoría de las situaciones si realizamos el estudio en una muestra debido a limitaciones de tiempo y recursos. Solo cuando queremos efectuar un censo debemos incluir todos los casos de una población. Esto implica realizar un censo en lugar de una muestra para tener una visión completa y detallada del fenómeno estudiado, eliminando cualquier error de muestreo y aumentando así la exactitud y precisión de los resultados. Esta metodología es apropiada para poblaciones relativamente pequeñas o cuando es crucial obtener datos muy precisos y exhaustivos.

En el contexto de esta investigación, se emplearán las 552 imágenes que conforman la población completa, las cuales fueron capturadas en la Parcela F5 Ampliación Cooperativa 60 del Distrito La Yarada Los Palos. Este enfoque permitirá tanto entrenar como validar el modelo en su totalidad, mejorando así su precisión y minimizando los posibles sesgos derivados de una muestra no representativa.

3.2.3. Muestreo

El muestreo es el procedimiento que se sigue para seleccionar una muestra de una población. Este procedimiento incluye el diseño de estrategias y métodos para elegir los elementos que formarán parte de la muestra (Hernández Sampieri et al., 2014).

Según Hernández Sampieri et al. (2014), un censo es un estudio que consiste en la recopilación de datos de todos los elementos de una población específica. A diferencia de una muestra, donde solo se observa un subconjunto, el censo busca obtener información exhaustiva y detallada sobre cada miembro de la población

En el presente estudio se consideran todas las 552 imágenes de la población, ya que se lleva a cabo un censo que abarca todos los casos existentes.

3.3. Equipos y materiales

Los equipos y materiales usados en esta investigación se detallan en la siguiente tabla.

Tabla 3

Equipos y materiales

Nombre	Cantidad	Características
Laptop	1 unidad	Laptop Asus X507UB-BR267U con procesador Intel Core i5-8250U 1.6GHz, 8GB RAM, HDD 1TB, tarjeta gráfica Nvidia GeForce MX-110 de 2GB, LED 15.6" HD
Workstation	1 unidad	Procesador Intel® Xeon® Silver 4214 2.20 GHz, memoria RAM: 64 GB DDR4 2933 366 MHz, almacenamiento de 1 TB SSD, unidad de procesamiento gráfico NVIDIA RTX A5000 de 24 GB, para el entrenamiento de los modelos.
Memoria SD	1 unidad	Marca Kingston, color negro, compatibilidad Android, capacidad 128GB, rendimiento 100MB/seg Lectura, formato exFAT, dimensiones 11x15x1 mm (micro SD) y 24x32x2.1 mm (con adaptador SD)
Cámara	1 unidad	Canon EOS Rebel T6i, Sensor CMOS (APS-C) de 24.2 megapíxeles, procesador de imágenes DIGIC 6 de Canon, tecnología AF del CMOS Híbrido III, pantalla táctil LCD Clear View II de ángulo variable, tomas continuas de alta velocidad, sistema AF de 19 puntos tipo cruz.
Impresora	1 unidad	Impresora Multifuncional Epson Ecotank L3250
Papel bond	1 paquete	Resma Papel Bond Xerox A4 de 75g - 500h

Lapicero	2 unidades	Lapicero Trilux 032 Azul Faber Castell
Google Colab		Se utilizó en su versión Free que nos proporciona acceso gratuito a infraestructura informática. La versión Free permite tiempos de ejecución de GPU y TPU por hasta 12 horas. El tiempo de ejecución de GPU incluye una CPU Intel Xeon a @2.20GHz, 13 GB de RAM, un acelerador Tesla K80 y 12 GB de VRAM GDDR5 y el tiempo de ejecución de TPU está equipado con una CPU Intel Xeon a @2.30GHz, 13 GB de RAM y una TPU en la nube con una potencia computacional de 180 teraflops. Se utilizó Keras, con las versiones 2.15.0 y TensorFlow 2.15.0, como backend, dentro de un entorno Python versión 3.10.12.
JupyterLab v4.0.6		Se usó arquitectura CUDA v12.2 para proporcionar un incremento del rendimiento del sistema y los modelos entrenados en el entorno JupyterLab v4.0.6.

Fuente. Elaboración propia.

3.4. Procedimiento de las pruebas experimentales

Dado que se trata de un estudio totalmente descriptivo, no es necesario ni apropiado realizar pruebas experimentales.

3.5. Técnicas de recolección de datos

a). Técnica

Las técnicas de recolección de datos son los diversos métodos utilizados para obtener información (Arias, 2006).

Según Arias (2006), la observación libre o no estructurada se lleva a cabo con un propósito específico, pero no sigue una guía predefinida que detalle todos los aspectos a observar.

En este contexto, en este estudio se empleó la técnica de observación no estructurada, que se lleva a cabo con un objetivo definido, pero sin una guía preestablecida que delimite todos los aspectos a observar. Esta metodología ofrece una mayor flexibilidad y libertad en la recolección de información.

b). Instrumento

De acuerdo con Arias (2006), en la observación libre o no estructurada se emplean varios instrumentos, como el diario de campo, una libreta o cuaderno de notas, además de cámaras fotográficas y de vídeo.

En el contexto de esta investigación, el instrumento utilizado para capturar y almacenar las fotografías es una cámara fotográfica de alta resolución marca Canon EOS Rebel T6i.

c). Validación y confiabilidad del instrumento

De acuerdo con Hernández Sampieri et al. (2014), la confiabilidad de un instrumento de medición se refiere al grado en que su aplicación repetida al mismo individuo u objeto produce resultados iguales.

De acuerdo con Hernández Sampieri et al. (2014), la validez se refiere al grado en que un instrumento de medición realmente mide lo que se propone medir. Esto implica que los resultados obtenidos realmente representan lo que se pretende medir y son útiles para alcanzar los objetivos del estudio.

En esta investigación, se utilizó la cámara fotográfica como instrumento de recolección de datos. Para fundamentar esta elección, se hace referencia al estudio de Alves et al. (2021) titulado “Use of photography in qualitative research in the health area: scoping review”. En dicho estudio, se recopilaron datos de 12 bases de datos académicas, que incluyeron artículos, tesis y disertaciones, abarcando investigaciones realizadas entre 2001 y 2018 en 21 países. En total, se incluyeron 138 estudios en la muestra, la mayoría de ellos realizados por investigadores del área de enfermería, quienes emplearon principalmente la cámara fotográfica, photovoice y foto-elicitación para la recolección de datos.

La conclusión del estudio destaca los principales beneficios de la fotografía como instrumento de investigación: promueve la reflexión y creatividad de los participantes, captura experiencias subjetivas de manera visual y enriquece la calidad de la información recolectada. Estos elementos permiten obtener datos más profundos y significativos.

Dado el uso exitoso de la cámara fotográfica en diversas investigaciones, tal como se detalla en el artículo, es posible afirmar que este instrumento es tanto válido como confiable para la recolección de datos, respaldando su eficacia en el ámbito investigativo.

3.6. Técnicas para el procesamiento de datos

El procesamiento de datos se llevará a cabo mediante el uso de un computador, aplicando el modelo estudiado. A continuación, se detallarán los pasos y las instrucciones seguidas en esta investigación.

1. Generación de datos

La generación de los conjuntos de imágenes para los tres tipos de plagas se obtuvo mediante una cámara digital Canon EOS Rebel T6i, ssuensor CMOS (APS-C) de 24.2 megapíxeles. Las imágenes fueron almacenadas en una tarjeta SD Kingston de 128 GB, y además, se complementaron con algunas imágenes provenientes de diversos recursos en línea.

Las imágenes fueron capturadas en los cultivos de La Yarada Los Palos, en la región de Tacna en Perú, con diferentes tipos de iluminación y escenarios: hojas, troncos, ramas, frutos, entre otras superficies, como se muestra en la Figura 7.

Figura 7
Fotografiando plagas del Olivo en La Yarada Los Palos, Tacna-Perú

Nota: Elaboración propia

2. Etiquetado de datos

El conjunto total de datos recopilado consistió en 552 imágenes. A continuación, se realizó una etiquetación manual de las imágenes, basada en las características de cada tipo de plaga. Los resultados de la etiquetación indicaron que 198 imágenes correspondían a la plaga Queresa Móvil, 186 imágenes representaban la plaga Chanchito o Piojo Harinoso, y 168 imágenes pertenecían a la plaga Mosquita Blanca del Fresno. Todas las imágenes fueron guardadas en formato JPG, lo que facilitó su uso en el proceso de entrenamiento del modelo.

Posteriormente, el conjunto de datos se dividió en tres subconjuntos: 70% (385 imágenes) para entrenamiento, 20% (109 imágenes) para validación y 10% (58 imágenes) para pruebas. Para realizar esta división, se utilizó la librería “splitfolders”, que permite dividir las carpetas que contienen archivos (en este caso, imágenes) de manera aleatoria en los conjuntos de entrenamiento, validación y prueba. En la Figura 8 se muestra el código utilizado para llevar a cabo esta división.

Figura 8

División del conjunto de datos

Nota: Código para dividir aleatoriamente imágenes usando la librería “splitfolders”.

La Tabla 4 presenta el número final de imágenes por tipo de plaga, organizadas según el conjunto de datos correspondiente.

Tabla 4

Etiquetado por tipo de plaga y conjunto de datos

Tipo	Conjunto de datos para entrenamiento	Conjunto de datos para validación	Conjunto de datos para prueba	Total
Queresa móvil (Orthezia olivicola)	138	39	21	**198**
Chanchito o piojo harinoso (Pseudococcus adonidum)	130	37	19	**186**

Mosquita blanca del fresno (Siphoninus phillyreae)	117	33	18	**168**
Total	**385**	**109**	**58**	**552**

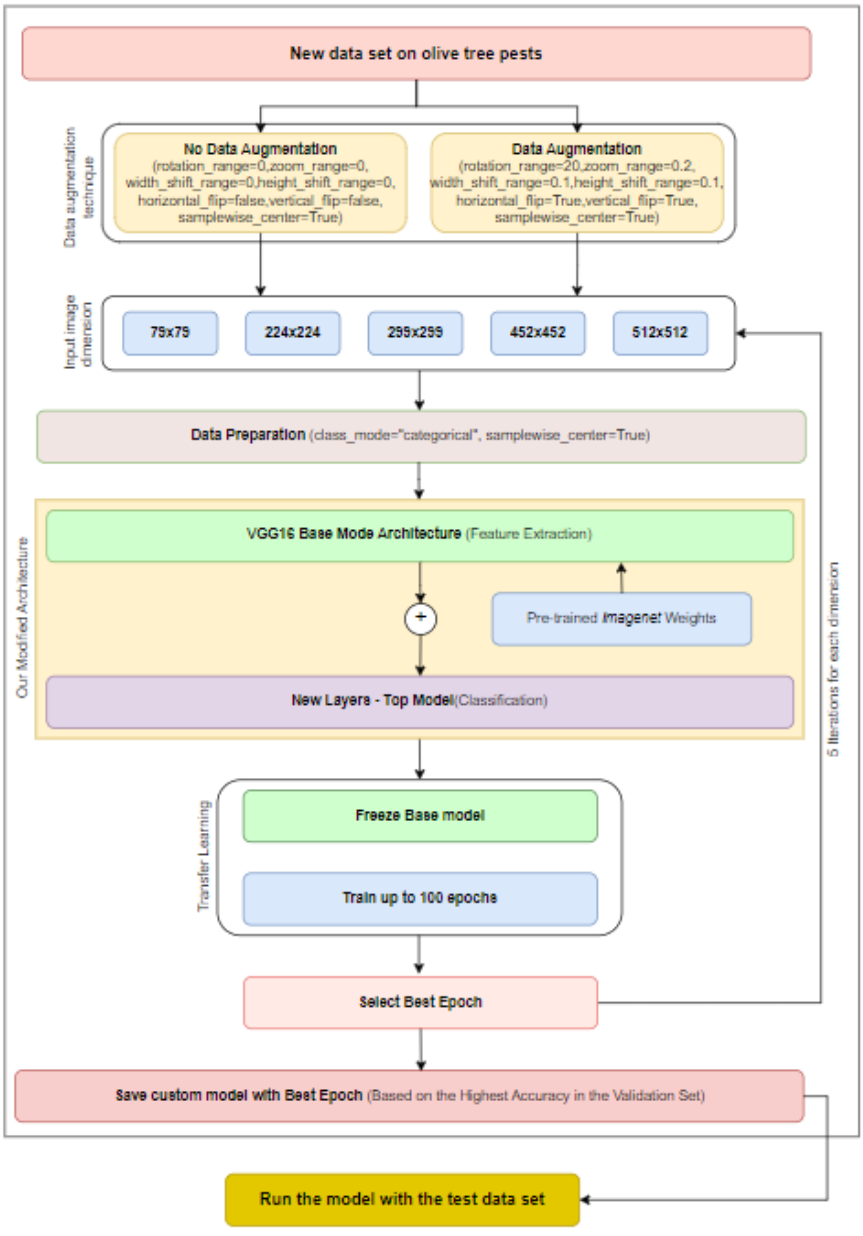
Nota. La tabla presenta la distribución de la población de plagas según cada conjunto de datos.

3. Procesamiento

La Figura 9 presenta el proceso completo del experimento realizado con el modelo Machine Learning, que se detalla a continuación.

Figura 9

Proceso completo del experimento



Nota. Elaboración propia

Para el experimento, se configuraron dos escenarios: uno con la aplicación de la técnica de aumento de datos y otro sin ella. En cada uno de estos escenarios, se probaron cinco dimensiones diferentes de las imágenes de entrada al modelo: 79x79, 224x224, 299x299, 452x452 y 512x512. Cada configuración se ejecutó cinco veces para evaluar su rendimiento de manera más precisa.

En el escenario donde se aplique la técnica de aumento de datos, se realizaron los siguientes ajustes: rotación aleatoria de entre 0 y 20 grados, zoom en el rango de 0.8 a 1.2, desplazamiento horizontal y vertical de hasta un 20% del tamaño total de la imagen, volteo horizontal y rotación vertical.

Además, se normalizó la media de cada muestra a cero y se utilizó una clasificación categórica porque deseamos trabajar con tres tipos diferentes de plagas.

Para construir la arquitectura del modelo Machine Learning, se usó el modelo VGG-16 que está constituida por 13 capas convolucionales, cada grupo de capas convolucionales le sigue una capa de agrupación máxima que compone la extracción de características o base model, a la que le siguen tres capas completamente conectadas como clasificación o top model, de ahí que en su nombre incluye las 16 Capas. Finalmente, se agrega una capa softmax al clasificador.

La clasificación se llevó a cabo mediante una red neuronal densa. La arquitectura original VGG-16 utiliza tres capas densas secuenciales como clasificador (top model), que permiten mapear la entrada del modelo a 1000 clases (porque fue concebido por Imagenet). La arquitectura se modificó reemplazando la última capa densa por otra capa densa que asigna la entrada del modelo a 3 clases. La tabla 5 presenta nuestra arquitectura VGG-16 modificada.

Tabla 5

Arquitectura VGG-16 modificada

Capa(componente)
Input_1(Input Layer)
Vgg16_base_model
Base Model (Feature Extracción)
flatten (Flatten)
fc1 (Dense)
fc2 (Dense)
output (Dense)
Top Model(Classification)

Nota. La tabla muestra la arquitectura modificada de VGG-16.

Para aplicar Aprendizaje por Transferencia (Transfer Learning), se cargaron pesos de Imagenet previamente entrenados en el modelo base de nuestra arquitectura modificada. Luego, el modelo base se congeló y solo se entrenó el Top model hasta 100 épocas, utilizando el optimizador Adam y la entropía cruzada categórica como función de pérdida para medir pérdida de entropía cruzada entre etiquetas y predicciones.

Durante el proceso de entrenamiento, se elige la mejor época según el valor máximo de exactitud (accuracy) obtenido en el conjunto de datos de validación. Luego, se guarda el modelo personalizado con la mejor época para su posterior evaluación con el conjunto de datos de prueba.

Al finalizar, se generaron los reportes correspondientes a los indicadores de desempeño en cada época, los cuales fueron almacenados en un archivo .csv para su posterior análisis y evaluación.

CAPÍTULO IV
RESULTADOS

4.1. Descripción de las pruebas experimentales

En el entorno de JupyterLab, se configuró el entrenamiento del modelo utilizando los siguientes parámetros: 100 épocas, un tamaño de lote de 8 y el optimizador ADAM. Para aprovechar el aprendizaje por transferencia, se cargaron los modelos preentrenados con los pesos provenientes del conjunto de datos ImageNet. Esto permitió que el modelo aprovechara los conocimientos adquiridos previamente en una tarea similar y pudiera generalizar mejor en el nuevo conjunto de datos.

A continuación, se llevaron a cabo cinco ejecuciones del entrenamiento para cada dimensión de la imagen de entrada, tanto con el uso de técnicas de aumento de datos como sin ellas. Al finalizar, se generaron los reportes correspondientes a los indicadores de desempeño en cada época, los cuales fueron almacenados en un archivo .csv para su posterior análisis y evaluación.

Además, los modelos entrenados se guardaron para facilitar su reutilización y análisis en etapas posteriores del proyecto.

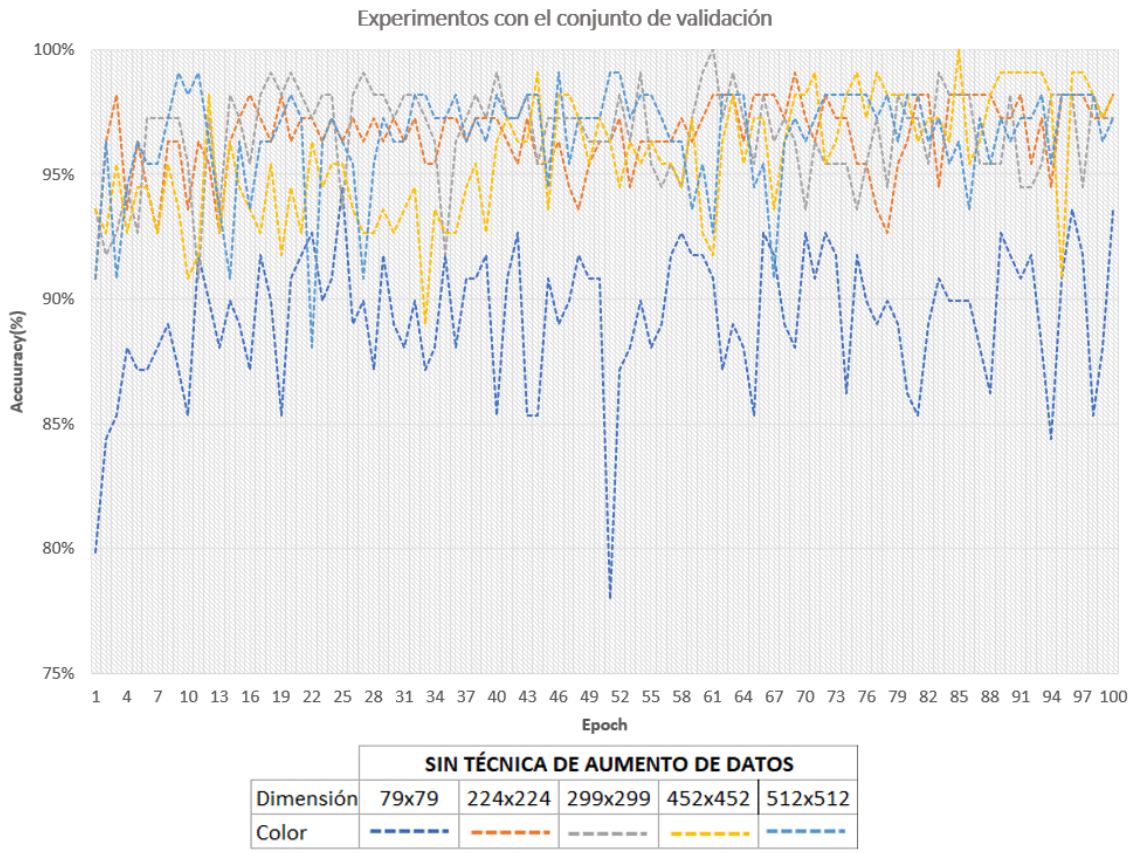
Toda esta codificación se muestra en el Anexo 03.

4.2. Presentación y análisis de los resultados

La Figura 10 presenta los experimentos realizados sin la aplicación de la técnica de aumento de datos, utilizando diferentes colores para representar cada dimensión de la imagen.

Figura 10

Diagrama de líneas de los experimentos realizados sin uso de técnica aumento de datos.

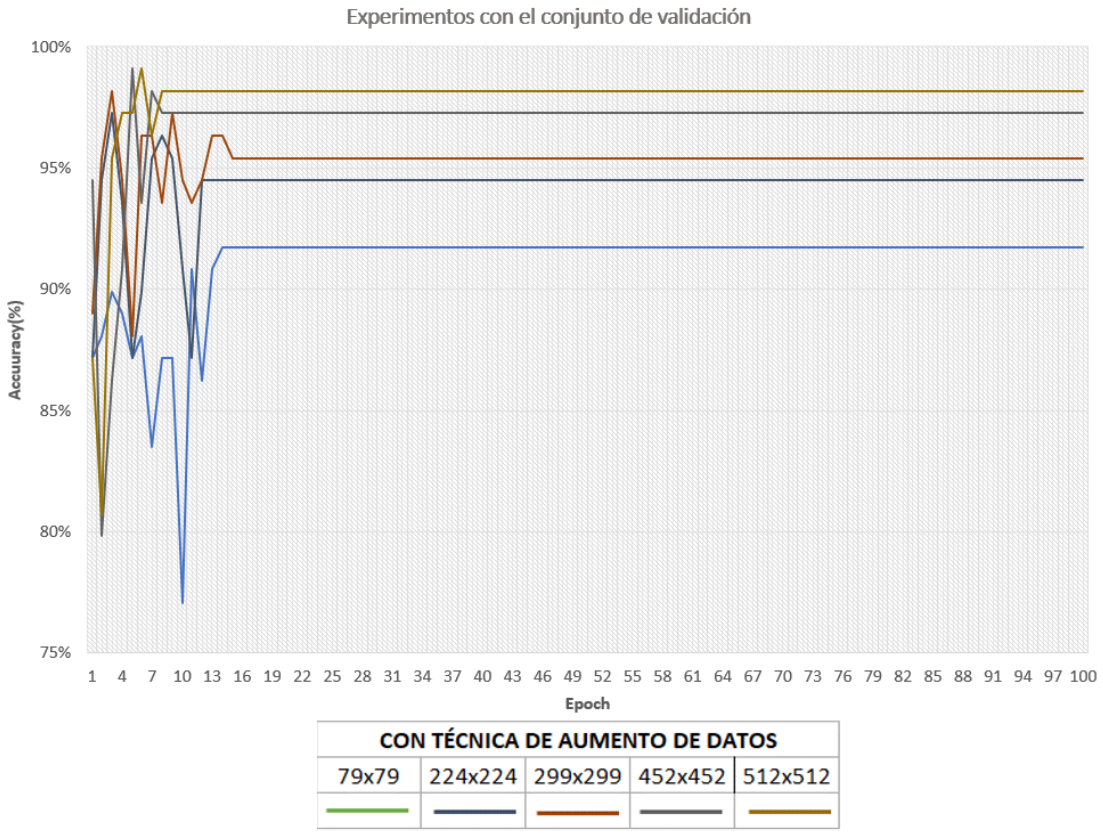


Nota: La gráfica muestra la variación de la precisión a lo largo de las épocas para cada una de las dimensiones de las imágenes de entrada sin uso de la técnica aumento de datos.

La Figura 11 presenta los experimentos realizados con la aplicación de la técnica de aumento de datos, utilizando diferentes colores para representar cada dimensión de la imagen.

Figura 11

Diagrama de líneas de los experimentos realizados con uso de técnica aumento de datos.



Nota: La gráfica muestra la variación de la precisión a lo largo de las épocas para cada una de las dimensiones de las imágenes de entrada con uso de la técnica aumento de datos.

La Figura 10 y 11 ilustra los resultados de experimentos con el conjunto de validación, comparando la precisión (Accuracy) de un modelo de aprendizaje automático con y sin el uso de técnicas de aumento de datos. El eje X del gráfico representa las épocas de entrenamiento, que van de 1 a 100, y el eje Y representa la precisión en porcentaje, que varía de 75% a 100%.

4.2.1 Resultados para la dimensión “dimensión de la imagen de entrada”

En la Figura 10 y 11 se destacan los mejores resultados obtenidos para cada dimensión de la imagen de entrada, tanto con la técnica de aumento de datos como sin ella. A partir de estos resultados, se extraen los valores correspondientes a los puntos más altos de precisión alcanzados en los experimentos para cada dimensión, los cuales se detallan en la Tabla 6.

Tabla 6

Máxima precisión en experimentos para “dimensión de la imagen de entrada”

	Sin técnica de aumento de datos		**Con técnica de aumento de datos**	
	Accuracy	Epoch	Accuracy	Epoch
Dimensión 79x79	94,50%	25	91,74%	14
Dimensión 224x224	99,08%	69	97,25%	3
Dimensión 299x229	100%	61	98,17%	3
Dimensión 452x452	100%	85	99,08%	5
Dimensión 512x512	99,08%	9	99,08%	6

Fuente. Elaboración propia.

En la Tabla 6 muestra que para la dimensión de la imagen de entrada 79x79, sin técnica de aumento de datos, la mayor precisión se alcanza en la época 25, con un valor de 76,50%. Con técnica de aumento de datos, la precisión máxima es de 91,74%, lograda en la época 14.

Para la dimensión de la imagen de entrada 224x224, sin técnica de aumento de datos, se obtiene una precisión de 99,08% en la época 69, mientras que, con técnica de aumento de datos, la mayor precisión es de 97,25% en la época 3.

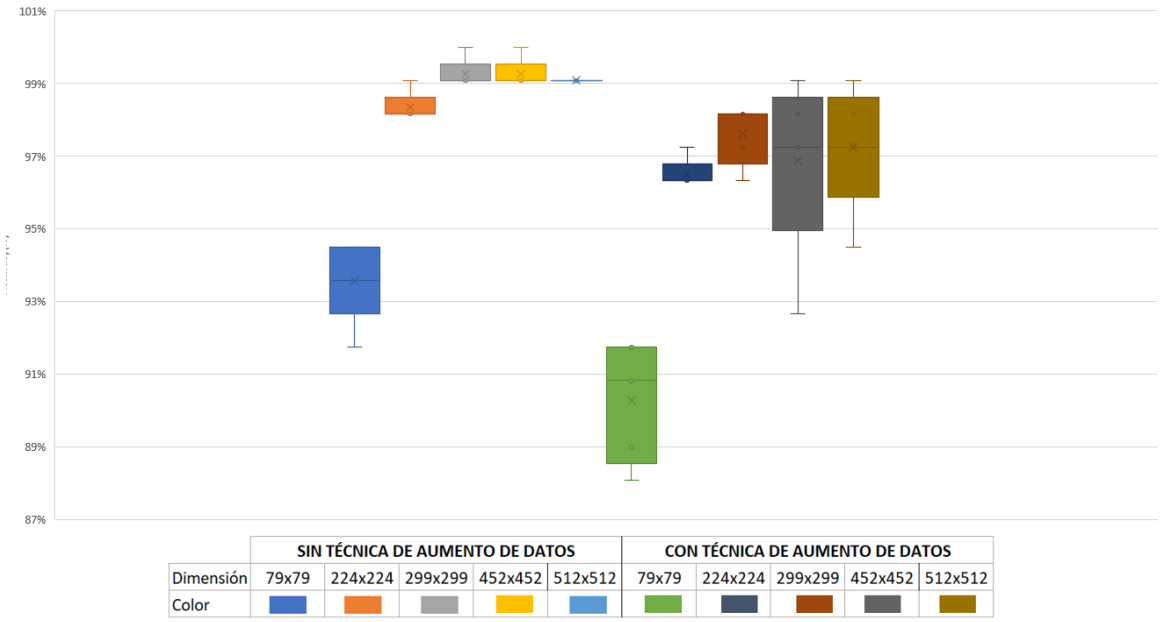
En el caso de la dimensión de la imagen de entrada 299x299, sin técnica de aumento de datos, la precisión alcanza el 100% en la época 61, mientras que, con técnica de aumento de datos, la mayor precisión es de 98,17% en la época 3.

Para la dimensión de la imagen de entrada 452x452, sin técnica de aumento de datos, se logra una precisión del 100% en la época 85, y con técnica de aumento de datos, la mayor precisión es de 99,08% en la época 5.

Finalmente, para la dimensión de la imagen de entrada 512x512, sin técnica de aumento de datos, se obtiene una precisión de 99,08% en la época 9, mientras que, con técnica de aumento de datos, la mayor precisión es de 99,08% en la época 6.

Figura 12

Diagrama de cajas para las dimensiones



Nota. Elaboración propia

En la Figura 12 presentan los valores de la dimensión "dimensión de la imagen de entrada" para 10 ejecuciones por cada dimensión de la imagen de entrada, representados en un gráfico de cajas, tanto con el uso de la técnica de aumento de datos como sin ella. Podemos observar que el rango de valores de precisión para la dimensión de la imagen de entrada de 299x299 y 452x452 sin la técnica de aumento de datos es superior al de las otras dimensiones de la imagen de entrada. Además, la dimensión de la imagen de entrada de 512x512 presenta una dispersión mínima de datos en comparación con las otras

dimensiones de la imagen de entrada. No se observa ningún valor atípico en ninguna de las dimensiones de la imagen de entrada.

4.2.2 Resultados para la dimensión “técnica de aumento de datos”

En la Figura 12, también presenta los valores para la dimensión "técnica de aumento de datos" para cinco ejecuciones por cada dimensión de la imagen de entrada, sumando un total de 25 ejecuciones, tanto con el uso de la técnica de aumento de datos como sin ella. A partir de este gráfico, se puede observar que el rango de valores de precisión es superior cuando no se utiliza la técnica de aumento de datos en comparación con cuando se utiliza. Además, el uso de la técnica de aumento de datos presenta una mayor dispersión en los resultados, lo que sugiere que la precisión varía más en este caso, mientras que el no uso de la técnica muestra una dispersión más reducida.

En resumen, el gráfico sugiere que el modelo, cuando no emplea aumento de datos, logra una mayor estabilidad y precisión en comparación con cuando se aplica dicha técnica.

4.3. Contrastación de hipótesis

4.3.1 Prueba de normalidad de datos

Según Romero (2016), cuando el tamaño muestral es igual o inferior a 50, la prueba más adecuada para evaluar la normalidad de los datos es la prueba de Shapiro-Wilk.

En la investigación, se optó por aplicar la prueba de normalidad de Shapiro-Wilk debido a que el tamaño de la muestra por cada modelo no supera los cincuenta datos. La prueba se realizó utilizando el software SPSS sobre los resultados de ambas dimensiones, estableciendo un nivel de confianza del 95% y un valor de significancia (α) del 5%.

- Si la significancia $\geq \alpha$, entonces se acepta Ho
- Si la significancia $< \alpha$, entonces se acepta Hi

4.3.1.1 Prueba de normalidad para la dimensión “dimensión de la imagen de entrada”

H0: Los datos de la dimensión “dimensión de la imagen de entrada” provienen de una distribución normal

H1: Los datos de la dimensión “dimensión de la imagen de entrada” no provienen de una distribución normal

Figura 13

Prueba de normalidad para la dimensión “dimensión de la imagen de entrada”

Pruebas de normalidad

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Estadístico	gl	Sig.	Estadístico	gl	Sig.
DimensionImagenEntrada	.220	50	<.001	.816	50	<.001

a. Corrección de significación de Lilliefors

Nota: En esta tabla se muestran los resultados de las pruebas estadísticas con Kolmogorov-Smirnov y Shapiro-Wilk

En la Figura 13 muestra que el p-valor obtenido para la dimensión “dimensión de la imagen de entrada” fue menor a 0,001, lo que es inferior al **nivel** de significancia del 5%. Por lo tanto, se rechaza la hipótesis nula (H0) y se acepta la hipótesis alternativa (H1), lo que indica que los datos no provienen de una distribución normal. Dado que los datos no cumplen con el supuesto de normalidad, se concluye que la dimensión 'dimensión de la imagen de entrada' no sigue una distribución normal, por lo que corresponde aplicar una prueba no paramétrica.

4.3.1.2 Prueba de normalidad para la dimensión “técnica de aumento de datos”

H0: Los datos de la dimensión “técnica de aumento de datos” provienen de una distribución normal

H1: Los datos de la dimensión “técnica de aumento de datos” no provienen de una distribución normal

Figura 14

Prueba de normalidad para la dimensión "técnica de aumento de datos"

Pruebas de normalidad

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Estadístico	gl	Sig.	Estadístico	gl	Sig.
TecnicaAumentoDatos	.220	50	<.001	.816	50	<.001

a. Corrección de significación de Lilliefors

Nota: En esta tabla se muestran los resultados de las pruebas estadísticas con Kolmogorov-Smirnov y Shapiro-Wilk

En la Figura 14 muestra que el p-valor obtenido para la dimensión "técnica de aumento de datos"" fue menor a 0,001, lo que es inferior al **nivel** de significancia del 5%. Por lo tanto, se rechaza la hipótesis nula (H0) y se acepta la hipótesis alternativa (H1), lo que indica que los datos no provienen de una distribución normal. Dado que los datos no cumplen con el supuesto de normalidad, se concluye que la dimensión "técnica de aumento de datos" no sigue una distribución normal, por lo que corresponde aplicar una prueba no paramétrica.

4.3.2 Prueba de hipótesis

4.3.2.1 Prueba de hipótesis para la dimensión "dimensión de la imagen de entrada"

H0: No existen diferencias significativas en el rendimiento del modelo Machine Learning al utilizar diferentes dimensiones de la imagen de entrada para la clasificación de plagas del olivo en La Yarada Los Palos, Tacna.

H1: Existen diferencias significativas en el rendimiento del modelo Machine Learning al utilizar diferentes dimensiones de la imagen de entrada para la clasificación de plagas del olivo en La Yarada Los Palos, Tacna.

En la Figura 13 muestra que los datos no cumplen con el supuesto de normalidad. Por lo tanto, se empleó una prueba no paramétrica, específicamente la prueba H de Kruskal-Wallis, para evaluar si existe una diferencia significativa entre los cinco grupos independientes.

Para la presente prueba se ha establecido un nivel de confianza de 95% (significancia α=0,05).

Se utilizó el software estadístico SPSS para realizar la prueba de hipótesis con la prueba estadística H de Kruskal-Wallis dando los siguientes resultados:

Figura 15

Prueba de H de Kruskal-Wallis para la dimensión "dimensión de la imagen de entrada"

Estadísticos de prueba[a,b]

	Accuracy
H de Kruskal-Wallis	26.630
gl	4
Sig. asin.	<.001

a. Prueba de Kruskal Wallis

b. Variable de agrupación: DimensiónImagenEntrada

Nota: Se muestra que el valor p (Sig. Asintótica tiene un valor menor a 0,001).

Interpretación del resultado:

La Figura 15 muestra que el p-valor (Sig. Asintótica) obtenido para la dimensión "dimensión de la imagen de entrada" fue menor a 0,001, lo que está por debajo del nivel de significancia del 5%. Por lo tanto, se rechaza la hipótesis nula (H_0) y se acepta la hipótesis alternativa (H_1). Esto afirma que existen diferencias significativas en el rendimiento del modelo Machine Learning al utilizar diferentes dimensiones de la imagen de entrada para la clasificación de plagas del olivo en La Yarada Los Palos, Tacna.

4.3.2.2 Prueba de hipótesis para la dimensión "técnica de aumento de datos"

H0: No existen diferencias significativas en el rendimiento de la técnica de aumento de datos del modelo Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna.

H1: Existen diferencias significativas en el rendimiento de la técnica de aumento de datos del modelo Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna.

En la Figura 14 muestra que los datos no cumplen con el supuesto de normalidad. Por lo tanto, se empleó una prueba no paramétrica, específicamente la prueba T de

Wilcoxon, para evaluar si existe una diferencia significativa entre los dos grupos relacionados.

Para la presente prueba se ha establecido un nivel de confianza de 95% (significancia α=0,05).

Se utilizó el software estadístico SPSS para realizar la prueba de hipótesis con la prueba estadística T de Wilcoxon dando los siguientes resultados:

Figura 16

Prueba de T de Wilcoxon para la dimensión “técnica de aumento de datos”

Estadísticos de prueba[a]

	ConTecnicaAumentoDatos - SinTecnicaAumentoDatos
Z	-4.225[b]
Sig. asin. (bilateral)	<.001

a. Prueba de rangos con signo de Wilcoxon

b. Se basa en rangos positivos.

Nota: Se muestra que el valor p (Sig. Asintótica bilateral tiene un valor menor a 0,001).

Interpretación del resultado:

La Figura 16 muestra que el p-valor (Sig. Asintótica) obtenido para la dimensión 'técnica de aumento de datos' fue inferior a 0,001, lo que está por debajo del nivel de significancia del 5%. Por lo tanto, se rechaza la hipótesis nula (H_0) y se acepta la hipótesis alternativa (H_1). Esto afirma que existen diferencias significativas en el rendimiento de la técnica de aumento de datos en el modelo de Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna.

CAPÍTULO V
DISCUSIÓN

5.1. Pruebas de validación del modelo experimental

Para validar el indicador de precisión (Accuracy), se tomó como referencia un estudio que utiliza esta métrica como indicador de rendimiento, la cual es ampliamente reconocida en la comunidad científica. En el trabajo de De Luca et al. (2021), se evaluó la precisión (Accuracy) de varios modelos pre-entrenados, como Faster RCNN Inception COCO, SSD MobileNet COCO y SSD Inception COCO. El objetivo de la investigación fue comparar estos modelos con el fin de optimizar el proceso de ingreso de productos al sistema del Banco Alimentario de La Plata, utilizando el reconocimiento de imágenes como herramienta principal para automatizar y agilizar la identificación de los productos.

5.2. Aplicación de la tecnología encontrada

Esta investigación tiene un enfoque descriptivo y comparativo, ya que no se orienta a la aplicación directa de soluciones, sino a la evaluación y comparación de los rendimientos de diversas configuraciones de un modelo de Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna. En este estudio, se analizaron configuraciones clave, como las dimensiones de las imágenes de entrada y la implementación de la técnica de aumento de datos.

5.3. Contraste con trabajos de investigación similares

En el presente trabajo, se evaluaron diversas configuraciones al modelo Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna. Se consideraron configuraciones como las dimensiones de las imágenes de entrada y el uso de la técnica de aumento de datos. En cuanto a las dimensiones de las imágenes, la mayor precisión alcanzada fue del 100% sin aplicar aumento de datos, utilizando dimensiones de 299x299 y 452x452. Sin embargo, cuando se implementó la técnica de aumento de datos, la precisión máxima alcanzada fue del 99,08%, con las dimensiones de imagen de entrada de 452x452 y 512x512.

Al comparar estos resultados con otros estudios que han utilizado configuraciones similares, se observa que las tendencias en los rendimientos obtenidos son consistentes, aunque con algunas variaciones según el contexto y las configuraciones empleadas. Por ejemplo, en el trabajo de Burgos (2023), en su tesis "Detección de plaga de pulguilla en el cultivo de aguaymanto mediante redes neuronales", se aplicó la técnica de aumento de datos y se utilizó una dimensión de entrada de 224x224 en los modelos VGG-16 y VGG-19. En su investigación, la precisión alcanzada con ambos modelos fue del 85%, utilizando un total de 1666 imágenes de pulguilla de aguaymanto. En nuestra investigación, al utilizar la misma configuración de dimensión de entrada (224x224) en el modelo VGG-16, se logró una precisión significativamente mayor, del 97,25%. Esto sugiere que, en nuestro caso, las configuraciones utilizadas (en particular, la técnica de aumento de datos y la elección de dimensiones de entrada) tuvieron un impacto positivo en el rendimiento del modelo.

De manera similar, en otro estudio de Castro (2019), denominado “Aplicación de algoritmos inteligentes para reconocimiento automático de enfermedades foliares de cultivo de palta”, se utilizó una dimensión de imagen de entrada de 100x100 para la clasificación automática de enfermedades foliares en el cultivo de palta, obteniendo una precisión de 95,24% con redes neuronales, 93,65% con Random Forest y 89,68% con Naive Bayes. Este estudio empleó un conjunto de datos compuesto por 630 imágenes y no aplicó la técnica de aumento de datos. Si bien los resultados son buenos, la precisión alcanzada en este caso fue más baja en comparación con los obtenidos en nuestra investigación, especialmente cuando se usaron dimensiones de entrada mayores (como 452x452 y 512x512) y la técnica de aumento de datos.

Estos contrastes evidencian la importancia de la selección de las configuraciones adecuadas para cada contexto y modelo. En nuestra investigación, el uso de las dimensiones 299x299 y 452x452 junto con el aumento de datos mostró mejoras sustanciales en la precisión, en comparación con trabajos similares que no emplearon técnicas de aumento de datos o utilizaron diferentes configuraciones de tamaño de imagen.

CONCLUSIONES

Primero:

Se observó que, entre las diferentes dimensiones de la imagen de entrada utilizadas en el modelo de Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna, las dimensiones con mayor precisión fueron 299x299 y 452x452, sin el uso de la técnica de aumento de datos. Ambas alcanzaron un valor máximo de 1,0000 (100%) y un valor promedio en las cinco ejecuciones de cada uno 0,9927 (99,27%). A continuación, se encuentran las dimensiones de imagen de entrada 512x512, que obtuvieron un valor máximo de 0,9908 (99,08%) y un valor promedio en las cinco ejecuciones de 0,9908 (99,08%), tanto con y sin el uso de la técnica de aumento de datos. La dimensión de imagen de entrada 79x79, sin el uso de la técnica de aumento de datos, alcanzó un valor máximo de 0,9450 (94,50%) y un valor promedio de 0,9358 (93,58%) en las cinco ejecuciones. Finalmente, la dimensión de imagen de entrada 224x224, presentó un valor máximo de 0,9908 (99,08%) y un valor promedio en las cinco ejecuciones de 0,9853 (98,53%) sin el uso de la técnica de aumento de datos.

Segundo:

Se pudo comprobar que, en cuanto al rendimiento de la técnica de aumento de datos en el modelo de Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna, la precisión es mayor cuando no se utiliza dicha técnica. Específicamente, las dimensiones de imagen de entrada 299x299 y 452x452 lograron un valor máximo de 1.0000 (100%) y un valor promedio de 0,9791 (97,91%) sin aumento de datos. En cambio, cuando se aplicó la técnica de aumento de datos, el valor máximo obtenido fue de 0,9908 (99,08%) para las dimensiones 452x452 y 512x512, con un valor promedio de 0,9571 (95,71%).

Tercero:

Se han comprobado que existen diferencias significativas en el rendimiento de las configuraciones del modelo de Machine Learning para la

clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna, en función de las dimensiones evaluados: la dimensión de la imagen de entrada y la aplicación de la técnica de aumento de datos. Para la dimensión "dimensión de la imagen de entrada", el valor p obtenido en la prueba de Kruskal-Wallis fue menor a 0,001, lo que es significativamente menor que el umbral de 0,05, lo que nos lleva a aceptar la hipótesis alternativa. De manera similar, para la dimensión "técnica de aumento de datos", el valor p en la prueba de Wilcoxon también fue menor a 0,001, lo que nos permite concluir que existe una diferencia significativa en el rendimiento en función de esta técnica. En ambos casos, los resultados demuestran que las configuraciones del modelo afectan de manera significativa el desempeño en la clasificación de imágenes de plagas del olivo en el área de estudio.

RECOMENDACIONES

Primera

Para futuras investigaciones sobre el mismo tema que el proyecto actual, se recomienda realizar un análisis inicial para descubrir qué plagas son más frecuentes en el campo de cultivo, su fenología y la época del año en que son más abundantes. Esto permitirá capturar un mayor número de fotografías durante el período adecuado, lo que enriquecerá el conjunto de datos para el entrenamiento del modelo.

Segunda

Aunque esta investigación se centró en tres tipos de plagas, se sugiere ampliar el catálogo en futuros estudios para desarrollar un modelo mucho más robusto.

Tercera

Al recolectar imágenes de plagas en el campo de cultivo, se recomienda capturar fotografías de alta calidad utilizando una cámara que pueda capturar detalles finos, incluso en imágenes de tamaño reducido.

Cuarta

En futuras investigaciones, se recomienda analizar las posibles causas del sobreajuste que el modelo puede experimentar al utilizar la técnica de aumento de datos, empleando el conjunto de datos generado en este estudio.

Quinta

Para futuras investigaciones, se recomienda explorar otras configuraciones que puedan influir en el rendimiento del modelo para la clasificación de imágenes de plagas del olivo en la región de La Yarada Los Palos, Tacna, utilizando el conjunto de datos generado en este estudio.

Sexta

Se sugiere llevar a cabo investigaciones que empleen métodos computacionales de disciplinas relacionadas con la inteligencia artificial, como el aprendizaje automático (machine learning), la minería de datos (data mining), entre otros, con el fin de aplicarlos en sectores como la agricultura.

REFERENCIAS BIBLIOGRÁFICAS

Aguilar, V., & Campoverde, M. (2020). *Clasificación de frutas basadas en redes neuronales convolucionales*. *5*(01), 3–22. https://doi.org/10.23857/pc.v5i01.1210

Andrade Alves, K. Y., Martins Rodrigues, C. C. F., De Oliveira Salvador, P. T. C., & De Mendonça Fernandes, S. D. (2021). Use of photography in qualitative research in the health area: scoping review. *Ciencia & Saude Coletiva*, *26*(2), 521–529. https://doi.org/10.1590/1413-81232021262.41052020

Arias, F. G. (2006). *El Proyecto de Investigación -Introducción a la metodología científica* (Episteme, Ed.; 6th ed.).

Brownlee, J. (2019, July). *How to Configure Image Data Augmentation in Keras - MachineLearningMastery.com*. https://machinelearningmastery.com/how-to-configure-image-data-augmentation-when-training-deep-learning-neural-networks/

Burgos, D. A. (2023). Detección y clasificación de plaga pulguilla en el cultivo de aguaymanto mediante redes neuronales. *Repositorio Institucional - USS*. http://repositorio.uss.edu.pe//handle/20.500.12802/12030

Calderon Ortiz, J. D., Morales Ticliahuanca, L. F., Roncal Moscol, M. E., & Solórzano Requejo, W. G. (2021). *Uso de algoritmos de Machine Learning para el diagnóstico de melanomas*.

Canal Exitosa Noticias. (2024). *FEN afectó más del 80% de la producción total del olivo en Tacna: "No queda nada por hacer" - YouTube* [Broadcast]. https://www.youtube.com/watch?v=NEkmE5pA-5o

Casanova, D. P. (2022). *Guía técnica del cultivo de Olivo en la región Tacna*. https://repositorio.midagri.gob.pe/handle/20.500.13036/1203

Cayllante Capia, J. A. (2024). Modelo de Aprendizaje Profundo para identificar plagas en la producción de quinua. *Revista Ingeniería*, *8*(20), 31–48. https://doi.org/10.33996/revistaingenieria.v8i20.116

Córdova, C. S. (2021). *Aplicación de aprendizaje profundo para la detección y clasificación automática de insectos agrícolas en trampas pegantes*. https://tesis.pucp.edu.pe/repositorio//handle/20.500.12404/20520

De Luca, A., Irigoitia, M., Pérez, G., & Pons, C. (2021). *Uso de la Técnica de Transfer Learning en Machine Learning para la Clasificación de Productos en el Banco Alimentario de La Plata*. http://bancoalimentario.org.ar

Delgado, R. M., & Obeso, G. I. (2019). Solución de Machine Learning en el reconocimiento de plagas para plantones de arándano. *Universidad Privada Del Norte*. https://repositorio.upn.edu.pe/handle/11537/22493

Durán, J. (2017). *Redes Neuronales Convolucionales en R Reconocimiento de caracteres escritos a mano*. https://hdl.handle.net/11441/69564

Fidalgo, I. Á. (2021). *Introducción al Machine Learning con TensorFlow*. http://hdl.handle.net/10347/30077

Gallardo Torres, J. E., Avalos Condori, G. E., & Córdova Miranda, T. L. (2019). *Identificación de plagas en el cultivo de Olivo utilizando precepción remota con un vehículo Aéreo no tripulado en el Valle La Yarada Los Palos*. http://hdl.handle.net/20.500.12969/1310

Gavilanez, A. R., & Saragosin, B. A. (2024). *Desarrollo de un prototipo para la identificación automática de plagas y enfermedades en el cultivo de papa, utilizando técnicas de inteligencia artificial en la ciudad de Latacunga*. http://repositorio.utc.edu.ec/handle/27000/12029

Gayathri, P., Dhavileswarapu, A., Ibrahim, S., Paul, R., & Gupta, R. (2023). Exploring the Potential of VGG-16 Architecture for Accurate Brain Tumor Detection Using Deep Learning. *Journal of Computers, Mechanical and Management*, *2*(2), 13–22. https://doi.org/10.57159/GADL.JCMM.2.2.23056

Gomez, E. M., & Sandoval, E. (2020). Una revisión: Uso de imágenes satelitales para la detección deplagas y enfermedades en cultivos. *Repositorio Institucional - UCV*. https://repositorio.ucv.edu.pe/handle/20.500.12692/53274

Hernández, A. (2021). *Detección de plagas y enfermedades del limón persa mediante una aplicación móvil con aprendizaje profundo a partir de redes neuronales*. http://51.143.95.221/bitstream/TecNM/5144/1/TS46DE~1.PDF

Hernández Sampieri, R., Fernández Collado, C., & Baptista Lucio, M. del P. (2014). *Metodología de la Investigación* (S. A. D. C. V. McGRAW-HILL / INTERAMERICANA EDITORES, Ed.; Sexta edición).

Keras ImageDataGenerator and Data Augmentation - PyImageSearch. (n.d.). Retrieved December 8, 2024, from https://pyimagesearch.com/2019/07/08/keras-imagedatagenerator-and-data-augmentation/

Microsoft. (2024). *Comprender la clasificación de imágenes*. https://learn.microsoft.com/es-es/training/modules/classify-images/3-understand-image-classification

MINCETUR. (2023). *Reporte de Comercio Regional I Semestre 2023 - Tacna*. https://cdn.www.gob.pe/uploads/document/file/5688904/5051942-rcr-tacna-i-semestre-2023.pdf

NVIDIA. (n.d.). *CUDA-X | NVIDIA*. Retrieved November 11, 2024, from https://www.nvidia.com/es-la/technologies/cuda-x/

Olney, A. M., & Fleming, S. D. (2021). JupyterLab Extensions for Blocks Programming, Self-Explanations, and HTML Injection. *CEUR Workshop Proceedings*, *3051*. https://digitalcommons.memphis.edu/facpubs/2917

Pereyra, M. E. (2020). *Detección de enfermedades y plagas en cultivos mediante Machine Learning*.

Romero, M. (2016). *Metodología de la investigación. Pruebas de bondad de ajuste a una distribución normal*.

Rosebrock, A. (2019). *Reconocimiento de dígitos escritos a mano mediante métodos de tratamiento de imagen y modelos de clasificación*. https://www.researchgate.net/publication/278157473

Salehin, I., Khan, M. R., Habiba, U., Badhon, N. H., & Moon, N. N. (2024). BAU-Insectv2: An agricultural plant insect dataset for deep learning and biomedical image analysis. *Data in Brief*, *53*, 110083. https://doi.org/10.1016/J.DIB.2024.110083

Samuel, A. L. (1959). *Some Studies in Machine Learning Using the Game of Checkers*. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5392560

Sánchez, D. F. (2022). *Sistema Autónomo de detección de plagas en el Olivar*. https://hdl.handle.net/10953.1/19831

Sánchez, H., & Reyes, C. (2015). *Metodología y diseños en la Investigación Científica* (Business Support Aneth S.R.L, Ed.; Quinta).

Sánchez, H., Reyes, C., & Mejía, K. (2018). *Manual de términos en investigación científica, tecnológica y humanística*.

Santos, A. (2021). *Clasificación de logos de marcas mediante Deep Learning*. http://titula.universidadeuropea.com/handle/20.500.12880/779

SENAMHI. (2023). *Boletín de pronóstico del riesgo Agroclimático para el cultivo de Olivo en la Cuenca Caplina de la región Tacna, Trimestre Diciembre 2023 - Febrero 2024*. https://www.senamhi.gob.pe/load/file/04106SENA-66.pdf

Shah, T. (2017). *About Train, Validation and Test Sets in Machine Learning*. Towards Data Science. https://towardsdatascience.com/train-validation-and-test-sets-72cb40cba9e7

Tannous, M., Stefanini, C., & Romano, D. (2023). A Deep-Learning-Based Detection Approach for the Identification of Insect Species of Economic Importance. *Insects 2023, Vol. 14, Page 148*, *14*(2), 148. https://doi.org/10.3390/INSECTS14020148

Trinidad, S. (2023). *Tacna | Plantaciones de olivo se encuentran en peligro por plagas: ¿Cuáles son?* https://larepublica.pe/sociedad/2023/10/12/tacna-plantaciones-de-olivo-se-encuentran-en-peligro-por-plagas-cuales-son-barrenillo-quesera-blanca-movil-senasa-lrsd-629628

ANEXOS

ANEXO 01: MATRIZ DE CONSISTENCIA

Título: Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna.

<table>
<tr><th>Problemas</th><th>Objetivos</th><th>Hipótesis</th><th>Variables</th><th>Dimensiones</th><th>Indicadores</th></tr>
<tr><td><u>Problema general</u>

¿Cómo es el rendimiento de distintas configuraciones del modelo Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna?</td><td><u>Objetivo general</u>

Comparar el rendimiento de distintas configuraciones del modelo Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna.</td><td><u>Hipótesis general</u>

H1: Existen diferencias significativas en el rendimiento de distintas configuraciones del modelo Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna.</td><td rowspan="2">V1:

Configuración del modelo Machine Learning</td><td rowspan="2">-Dimensión de la imagen de entrada

-Técnica de aumento de datos</td><td>-Número de pixeles

-Imágenes con uso de DA
- Imágenes sin uso de DA</td></tr>
<tr><td><u>Problema específico 1</u>

¿Cómo es el rendimiento modificando las dimensiones de la imagen de entrada del modelo Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna?</td><td><u>Objetivo específico 1</u>

Comparar el rendimiento de diferentes modificaciones de las dimensiones de la imagen de entrada del modelo Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna.</td><td><u>Hipótesis específica 1</u>

H1: Existen diferencias significativas en el rendimiento del modelo Machine Learning al modificar las dimensiones de la imagen de entrada para la clasificación de plagas del olivo en La Yarada Los Palos, Tacna.</td><td rowspan="2">V2:

Clasificación de imágenes de plagas del olivo</td><td rowspan="2">-Rendimiento</td><td rowspan="2">- Accuracy</td></tr>
<tr><td><u>Problema específico 2</u>

¿Cómo es el rendimiento de la técnica aumento de datos del modelo Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna?</td><td><u>Objetivo específico 2</u>

Comparar el rendimiento de la técnica aumento de datos del modelo Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna.</td><td><u>Hipótesis específica 2</u>

H1: Existen diferencias significativas en el rendimiento de la técnica de aumento de datos del modelo Machine Learning para la clasificación de imágenes de plagas del olivo en La Yarada Los Palos, Tacna.</td></tr>
<tr><td colspan="3">Tipo de Investigación: Aplicada
Nivel: Comparativa
Diseño: No experimental</td><td colspan="2">Población: 552 imágenes de plagas del olivo
Muestra: 552 imágenes de plagas del olivo</td><td></td></tr>
</table>

ANEXO 02: INTRUMENTO DE RECOLECCIÓN DE DATOS

Nombre del instrumento	Cámara fotográfica
Objetivo	Obtener Data Set de plagas del olivo de La Yarada Los Palos, Tacna.
Población	La población está compuesta por 512 imágenes de tres tipos de plagas del olivo: Queresa Móvil, Chanchito o Piojo Harinoso, y Mosquita Blanca del Fresno.
Muestra	La muestra es igual a 512 imágenes de plagas del olivo.
Dimensión que mide	Dimensión de la imagen de entrada
Detalles del instrumento	
Atributos	Detalle
Marca	Canon
Modelo	EOS Rebel T6i
Alto	100.9
Ancho	131.9
Profundidad	77.8
Peso	510
Megapixeles	24.2
Tipo de cámara	SLR Camera Kit
Tipo de sensor	CMOS
Máxima resolución de imagen	6000 x 4000
Relaciones de aspecto compatibles	1:1, 3:2, 4:3, 16:9
Total de megapixeles	24.7
Formato de sensor	Advanced Photo System type-C (APS-C)
Formatos de imagen soportados	JPG, RAW
Intervalo de longitud focal	18 - 55
Tipo de lente	Standard zoom
Enfoque	TTL
Ajustes de enfoque	Auto/Manual
Sensibilidad ISO	100,400,6400,12800,25600

Fuente. Elaboración propia.

ANEXO 03: CÓDIGO DE PROGRAMACIÓN

Código para dividir el conjunto de datos

```
pip install split-folders
pip install split-folders[full]
from google.colab import drive
drive.mount('/content/drive')
import splitfolders
# Split with a ratio.
# To only split into training and validation set, set a tuple to `ratio`, i.e, `(.8, .2)`.
splitfolders.ratio("/content/drive/MyDrive/PROYECTO TESIS - FRANZ CHANINI
MENA/DataSetCompleto", output="/content/drive/MyDrive/PROYECTO TESIS - FRANZ CHANINI
MENA/DataSetPlagas",
    seed=1337, ratio=(.7, .2, .1), group_prefix=None, move=False) # default values
```

Código para entrenar el modelo Machine Learning

```
from keras.models import Sequential, Model
from keras.layers import Conv2D, MaxPool2D, Dense, Flatten, Dropout, BatchNormalization, Input
from keras.optimizers import Adam
from keras.callbacks import TensorBoard, ModelCheckpoint
import os
import numpy as np
from keras.preprocessing import image
from keras.applications.imagenet_utils import preprocess_input, decode_predictions
from keras.applications.vgg16 import VGG16
from keras.preprocessing.image import ImageDataGenerator
from sklearn.utils import shuffle
from sklearn.model_selection import train_test_split
import cv2
from tensorflow import keras
from tensorflow import keras
import pandas as pd
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D
from tensorflow.keras.models import Model
import matplotlib.pyplot as plt
%matplotlib inline

width_shape = 224
height_shape = 224
num_classes = 3
epochs = 100
batch_size = 8
train_data_dir='/content/drive/MyDrive/PROYECTO TESIS - FRANZ CHANINI MENA/DataSetPlagas/train'


validation_data_dir='/content/drive/MyDrive/PROYECTO TESIS - FRANZ CHANINI
MENA/DataSetPlagas/val'
train_datagen = keras.preprocessing.image.ImageDataGenerator(
    rotation_range=20,
    zoom_range=0.2,
    width_shift_range=0.1,
    height_shift_range=0.1,
    horizontal_flip=True,
    vertical_flip=True,
    samplewise_center=True
   # preprocessing_function=preprocess_input
    )
valid_datagen = keras.preprocessing.image.ImageDataGenerator(samplewise_center=True)
#Ejecutamos el dataAugmentation
train_generator = train_datagen.flow_from_directory(
    train_data_dir,
    target_size=(width_shape, height_shape),
    batch_size=batch_size,
    #save_to_dir='',
    class_mode='categorical')
```

```
validation_generator = valid_datagen.flow_from_directory(
    validation_data_dir,
    target_size=(width_shape, height_shape),
    batch_size=batch_size,
    shuffle=False,
    #save_to_dir='',
    class_mode='categorical')
nb_train_samples = 385
nb_validation_samples = 109
image_input = Input(shape=(width_shape, height_shape, 3))
model = VGG16(input_shape=(224, 224, 3), include_top=True,weights='imagenet')
#input_tensor=image_input,
last_layer = model.get_layer('fc2').output
out = Dense(num_classes, activation='softmax', name='output')(last_layer)
custom_vgg_model = Model(model.input, out)
for layer in custom_vgg_model.layers[:-3]:
        layer.trainable = False
custom_vgg_model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
custom_vgg_model.summary()
model_checkpoint_callback = keras.callbacks.ModelCheckpoint(
            filepath=str('/content/drive/MyDrive/PROYECTO TESIS - FRANZ CHANINI
MENA/ModeloGuardado')+'/VGG16'+'.h5',
            #save_weights_only=True,
            monitor='val_accuracy',
            mode='max',
            save_best_only=True)
model_history = custom_vgg_model.fit(
    train_generator,
    epochs=epochs,
    validation_data=validation_generator,
    steps_per_epoch=nb_train_samples/batch_size,
    validation_steps=nb_validation_samples/batch_size,
                callbacks=[model_checkpoint_callback])
hist_df = pd.DataFrame(model history.history)
hist_csv_file = str('/content/drive/MyDrive/PROYECTO TESIS - FRANZ CHANINI
MENA/GuarArchivoCSV')+'/history VGG16'+'.csv'
with open(hist_csv_file, mode='w') as f:
         hist_df.to_csv(f)
```

Código para graficar los resultados del entrenamiento

```
def plotTraining(hist, epochs, typeData):

    if typeData=="loss":
        plt.figure(1,figsize=(10,5))
        yc=hist.history['loss']
        xc=range(epochs)
        plt.ylabel('Loss', fontsize=24)
        plt.plot(xc,yc,'-r',label='Loss Training')
    if typeData=="accuracy":
        plt.figure(2,figsize=(10,5))
        yc=hist.history['accuracy']
        for i in range(0, len(yc)):
            yc[i]=100*yc[i]
        xc=range(epochs)
        plt.ylabel('Accuracy (%)', fontsize=24)
        plt.plot(xc,yc,'-r',label='Accuracy Training')
    if typeData=="val_loss":
        plt.figure(1,figsize=(10,5))
        yc=hist.history['val_loss']
        xc=range(epochs)
        plt.ylabel('Loss', fontsize=24)
        plt.plot(xc,yc,'--b',label='Loss Validate')
    if typeData=="val_accuracy":
        plt.figure(2,figsize=(10,5))
        yc=hist.history['val_accuracy']
        for i in range(0, len(yc)):
            yc[i]=100*yc[i]
        xc=range(epochs)
        plt.ylabel('Accuracy (%)', fontsize=24)
        plt.plot(xc,yc,'--b',label='Training Validate')

    plt.rc('xtick',labelsize=24)
    plt.rc('ytick',labelsize=24)
    plt.rc('legend', fontsize=18)
    plt.legend()
    plt.xlabel('Number of Epochs',fontsize=24)
    plt.grid(True)

plotTraining(model_history,epochs,"loss")
plotTraining(model_history,epochs,"accuracy")
plotTraining(model_history,epochs,"val_loss")
plotTraining(model_history,epochs,"val_accuracy")
```

Código para graficar la matriz de confusión de entrenamiento

```
from google.colab import drive
drive.mount('/content/drive')
width_shape = 224
height_shape = 224
num_classes = 3
epochs = 100
batch_size = 8

names = ['MosquitaBlanca','PiojoHarinoso','QueresaMovil']
train_data_dir = '/content/drive/MyDrive/PROYECTO TESIS - FRANZ CHANINI
MENA/DataSetPlagas/train'
train_datagen = keras.preprocessing.image.ImageDataGenerator(samplewise_center=True)
train_generator = train_datagen.flow_from_directory(
    train_data_dir,
    target_size=(width_shape, height_shape),
    batch_size=batch_size,
    shuffle=False,
    #save_to_dir='',
    class_mode='categorical')
custom_Model= load_model("/content/drive/MyDrive/PROYECTO TESIS - FRANZ CHANINI
MENA/ModeloGuardado/VGG16.h5")
predictions = custom_Model.predict(train_generator)
y_pred = np.argmax(predictions, axis=1)
y_real = train_generator.classes


matc=confusion_matrix(y_real, y_pred)
plot_confusion_matrix(conf_mat=matc, figsize=(9,9), class_names = names,
show_normed=False)
plt.tight_layout()
print(metrics.classification_report(y_real,y_pred, digits = 4))
import pandas as pd
def class_matric(confusion_matrix, class_id):
    confusion_matrix = np.float64(confusion_matrix)
    TP = confusion_matrix[class_id,class_id]
    FN = np.sum(confusion_matrix[class_id]) - TP
    FP = np.sum(confusion_matrix[:,class_id]) - TP
    TN = np.sum(confusion_matrix) - TP - FN - FP
    # sensitivity = 0 if TP == 0
    if TP != 0:
        recall = TP/(TP+FN)
    else:
        recall = 0.
    precision = TP/(TP+FP)  #Precision
    accuracy = (TP+TN)/(TP+FP+FN+TN)
    error=(FP+FN)/(TP+FP+FN+TN)
    f1_score=(2*precision*recall)/(precision+recall)
    return accuracy,error,precision,recall,f1_score
confusion_matrix=matc
result=[[],[],[],[],[]]
for i in range(len(names)):
  class_id=i
  accuracy,error,precision,recall,f1_score=class_matric(confusion_matrix, class_id)
  result[0].append(accuracy)
  result[1].append(error)
  result[2].append(precision)
  result[3].append(recall)
  result[4].append(f1_score)
df_train =
pd.DataFrame(result,index=['accuracy','error','precision','recall','f1_score'],
columns=['MosquitaBlanca','PiojoHarinoso','QueresaMovil'])
df_train
```

Código para graficar la matriz de confusión de validación

```
from google.colab import drive
drive.mount('/content/drive')
width_shape = 224
height_shape = 224
num_classes = 3
epochs = 100
batch_size = 8
from sklearn.metrics import confusion_matrix, f1_score, roc_curve, precision_score,
recall_score, accuracy_score, roc_auc_score
from sklearn import metrics
from mlxtend.plotting import plot_confusion_matrix
from keras.models import load_model
import numpy as np
import matplotlib.pyplot as plt
from tensorflow import keras
%matplotlib inline

names = ['MosquitaBlanca','PiojoHarinoso','QueresaMovil']

validation_data_dir = '/content/drive/MyDrive/PROYECTO TESIS - FRANZ CHANINI
MENA/DataSetPlagas/val'

valid_datagen = keras.preprocessing.image.ImageDataGenerator(samplewise_center=True)

validation_generator = valid_datagen.flow_from_directory(
    validation_data_dir,
    target_size=(width_shape, height_shape),
    batch_size=batch_size,
    shuffle=False,
    #save_to_dir='',
    class_mode='categorical')

custom_Model= load_model("/content/drive/MyDrive/PROYECTO TESIS - FRANZ CHANINI
MENA/ModeloGuardado/VGG16.h5")

predictions = custom_Model.predict(validation_generator)

y_pred = np.argmax(predictions, axis=1)
y_real = validation_generator.classes

matc=confusion_matrix(y_real, y_pred)

plot_confusion_matrix(conf_mat=matc, figsize=(9,9), class_names = names,
show_normed=False)
plt.tight_layout()
print(metrics.classification_report(y_real,y_pred, digits = 4))

import pandas as pd
def class_matric(confusion_matrix, class_id):
    confusion_matrix = np.float64(confusion_matrix)
    TP = confusion_matrix[class_id,class_id]
    FN = np.sum(confusion_matrix[class_id]) - TP
    FP = np.sum(confusion_matrix[:,class_id]) - TP
    TN = np.sum(confusion_matrix) - TP - FN - FP

    # sensitivity = 0 if TP == 0
    if TP != 0:
        recall = TP/(TP+FN)
    else:
        recall = 0.
```

```
    precision = TP/(TP+FP)  #Precision
    accuracy = (TP+TN)/(TP+FP+FN+TN)
    error=(FP+FN)/(TP+FP+FN+TN)
    f1_score=(2*precision*recall)/(precision+recall)

    return accuracy,error,precision,recall,f1_score

confusion_matrix=matc
result=[[],[],[],[],[]]
for i in range(len(names)):
  class_id=i
  accuracy,error,precision,recall,f1_score=class_matric(confusion_matrix, class_id)
  result[0].append(accuracy)
  result[1].append(error)
  result[2].append(precision)
  result[3].append(recall)
  result[4].append(f1_score)
df_val =
pd.DataFrame(result,index=['accuracy','error','precision','recall','f1_score'],
columns=['MosquitaBlanca','PiojoHarinoso','QueresaMovil'])
df_val
```

Código para graficar la matriz de confusión de prueba(test)

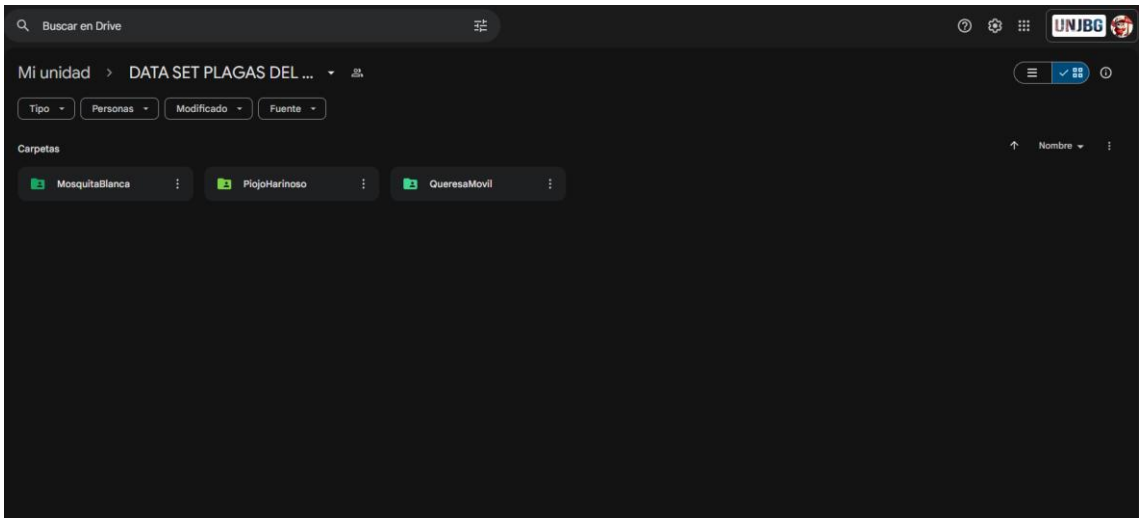
```
from google.colab import drive
drive.mount('/content/drive')

width_shape = 224
height_shape = 224
num_classes = 3
epochs = 100
batch_size = 8

from sklearn.metrics import confusion_matrix, f1_score, roc_curve, precision_score,
recall_score, accuracy_score, roc_auc_score
from sklearn import metrics
from mlxtend.plotting import plot_confusion_matrix
from keras.models import load_model
import numpy as np
import matplotlib.pyplot as plt
from tensorflow import keras
%matplotlib inline


names = ['MosquitaBlanca','PiojoHarinoso','QueresaMovil']

test_data_dir = '/content/drive/MyDrive/PROYECTO TESIS - FRANZ CHANINI
MENA/DataSetPlagas/test'
```

```
test_datagen = keras.preprocessing.image.ImageDataGenerator(samplewise_center=True)

test_generator = test_datagen.flow_from_directory(
    test_data_dir,
    target_size=(width_shape, height_shape),
    batch_size=batch_size,
    shuffle=False,
    #save_to_dir='',
    class_mode='categorical')

custom_Model= load_model("/content/drive/MyDrive/PROYECTO TESIS - FRANZ CHANINI
MENA/ModeloGuardado/VGG16.h5")

predictions = custom_Model.predict(test_generator)

y_pred = np.argmax(predictions, axis=1)
y_real = test_generator.classes

matc=confusion_matrix(y_real, y_pred)

plot_confusion_matrix(conf_mat=matc, figsize=(9,9), class_names = names,
show_normed=False)
plt.tight_layout()

print(metrics.classification_report(y_real,y_pred, digits = 4))

import pandas as pd
def class_matric(confusion_matrix, class_id):
    confusion_matrix = np.float64(confusion_matrix)
    TP = confusion_matrix[class_id,class_id]
    FN = np.sum(confusion_matrix[class_id]) - TP
    FP = np.sum(confusion_matrix[:,class_id]) - TP
    TN = np.sum(confusion_matrix) - TP - FN - FP

    # sensitivity = 0 if TP == 0
    if TP != 0:
        recall = TP/(TP+FN)
    else:
        recall = 0.
    precision = TP/(TP+FP)  #Precision
    accuracy = (TP+TN)/(TP+FP+FN+TN)
    error=(FP+FN)/(TP+FP+FN+TN)
    f1_score=(2*precision*recall)/(precision+recall)

    return accuracy,error,precision,recall,f1_score

confusion_matrix=matc
result=[[],[],[],[],[]]
for i in range(len(names)):
  class_id=i
  accuracy,error,precision,recall,f1_score=class_matric(confusion_matrix, class_id)
  result[0].append(accuracy)
  result[1].append(error)
  result[2].append(precision)
  result[3].append(recall)
  result[4].append(f1_score)
df_test =
pd.DataFrame(result,index=['accuracy','error','precision','recall','f1_score'],
columns=['MosquitaBlanca','PiojoHarinoso','QueresaMovil'])
df_test
```

ANEXO 04: CONJUNTO DE DATOS

El conjunto de datos, que consiste en imágenes de plagas del olivo, puede ser accedido a través del siguiente enlace: https://drive.google.com/drive/folders/1NAKIDCPSXH6iXTiXNUinazZsFcttqGga?usp=sharing

Conjunto de datos disponible públicamente en Google Drive.



ANEXO 05: FOTOS TOMADAS EN LA YARADA LOS PALOS, TACNA

Fotografiando las plagas del Olivo en La Yarada Los Palos, Tacna

Plaga Queresa móvil en las hojas del Olivo

Plaga Queresa móvil en el tallo del Olivo

Plaga Piojo Harinoso en el tallo del Olivo

Plaga Piojo Harinoso en el fruto del Olivo

Plaga Piojo Harinoso en la hoja del Olivo

Plaga Mosquita Blanca del Fresno en la hoja del Olivo