UPN
UNIVERSIDAD PRIVADA DEL NORTE

FACULTAD DE INGENIERÍA

Carrera de Ingeniería Electrónica

DESARROLLO DE UN MODELO CNN PARA LA DETECCIÓN DEL ESTADO DE MADUREZ DEL TOMATE Y SU COMPARACIÓN CON MODELOS EXISTENTES MEDIANTE MÉTRICAS DE PRECISIÓN Y F1-SCORE, PARA LA AGROINDUSTRIA PERUANA, 2025

Tesis para optar el título profesional de:

Ingeniero Electrónico

Autores:

Jesus David Anaya Anticona

Brayan Andre Sanchez Chavez

Asesor:

Dr. Ricardo Manuel Rossi Valverde

Código ORCID: https://orcid.org/0000-0003-1424-8261

Trujillo – Perú

2025

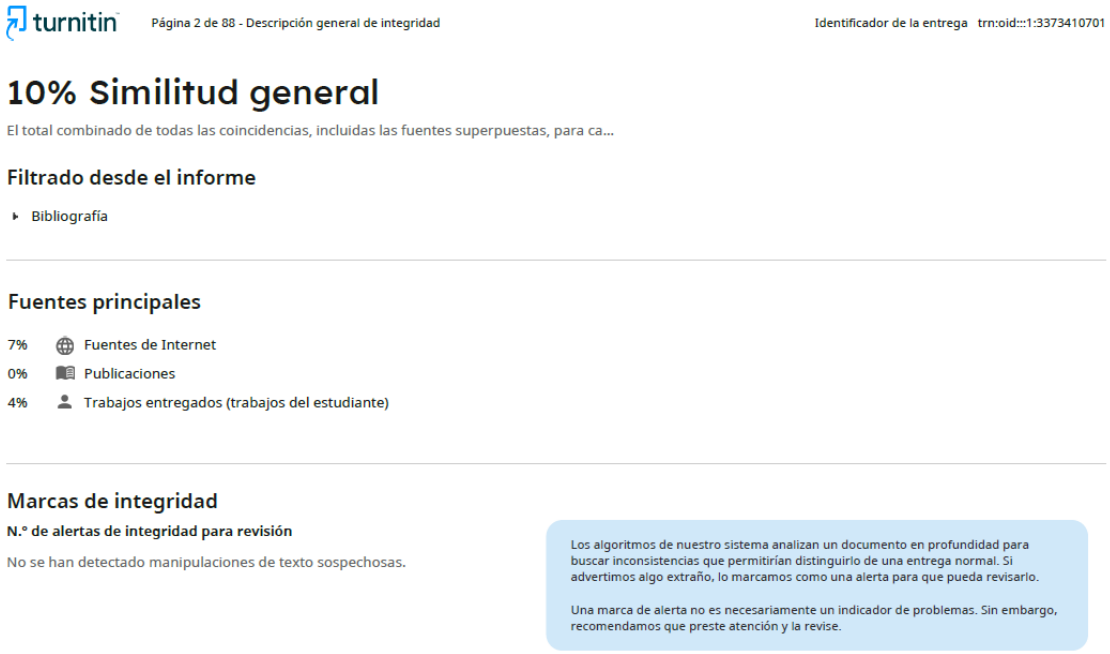
Jurado Evaluador

Jurado 1 Presidente(a)	**HENRRY JOSUE VILLANUEVA BAZAN**	
	Nombre y Apellidos	Nº DNI

Jurado 2	**CESAR AUGUSTO CIRIACO MARTINEZ**	
	Nombre y Apellidos	Nº DNI

Jurado 3	**RICARDO MANUEL ROSSI VALVERDE**	
	Nombre y Apellidos	Nº DNI

Informe de Similitud



turnitin Página 2 de 88 - Descripción general de integridad Identificador de la entrega trn:oid:::1:3373410701

10% Similitud general

El total combinado de todas las coincidencias, incluidas las fuentes superpuestas, para ca...

Filtrado desde el informe

▸ Bibliografía

Fuentes principales

7% Fuentes de Internet
0% Publicaciones
4% Trabajos entregados (trabajos del estudiante)

Marcas de integridad

N.° de alertas de integridad para revisión

No se han detectado manipulaciones de texto sospechosas.

Los algoritmos de nuestro sistema analizan un documento en profundidad para buscar inconsistencias que permitirían distinguirlo de una entrega normal. Si advertimos algo extraño, lo marcamos como una alerta para que pueda revisarlo.

Una marca de alerta no es necesariamente un indicador de problemas. Sin embargo, recomendamos que preste atención y la revise.

Dedicatoria

Dedicamos este trabajo a nuestras familias, por su apoyo a nuestros compañeros por siempre exigirnos a dar más de nosotros, y a todo el que nos brindó una palabra de aliento cuando la necesitábamos.

Agradecimiento

Agradezco de todo corazón a mi madre, por siempre brindarme un ambiente en el cual pudo florecer todo lo que ahora soy, y en el cual se encubo todo lo que un día seré.

Índice de Contenido

Índice de Tablas

Índice de Figuras

Resumen

La clasificación automática del estado de madurez del tomate es un desafío relevante para la agroindustria peruana, debido a la necesidad de optimizar los procesos de selección y mejorar la calidad del producto final. En este estudio se planteó como objetivo principal diseñar, implementar y evaluar modelos de redes neuronales convolucionales (CNN) para clasificar tomates en tres estados de madurez, comparando su rendimiento frente a modelos alternativos de la literatura. La investigación se enmarca en un diseño cuantitativo, experimental y comparativo. Se empleó un dataset de imágenes de tomates en diferentes estados de madurez, dividido en 6 folds de validación cruzada de entre 25 a 34 tomates por imagen. El instrumento de medición fueron los logs generados por Keras, de los cuales se extrajeron métricas de rendimiento (precisión y F1-score). Para el análisis estadístico se aplicó la prueba no paramétrica de Friedman y la prueba post-hoc de Nemenyi para determinar diferencias significativas entre modelos. Los resultados muestran que el modelo CNN con data augmentation supero al modelo sin data augmentation y al modelo one-shot. Se concluye que el modelo CNN optimizado propuesto presenta un desempeño superior frente a las alternativas evaluadas, confirmando su pertinencia para aplicaciones en la agroindustria peruana.

Palabras clave: Modelo CNN, Clasificación de Madurez, Embeddig, Data Augmentation.

Abstract

Automatic tomato ripeness classification is a significant challenge for the Peruvian agroindustry, driven by the need to optimize selection processes and improve final product quality. The main objective of this study was to design, implement, and evaluate convolutional neural network (CNN) models to classify tomatoes into three ripeness stages, comparing their performance with alternative models from the literature. The research was based on a quantitative, experimental, and comparative design. A dataset of tomato images at different ripeness stages was used, divided into six cross-validation folds of 25 to 34 tomatoes per image. The measurement tool was the logs generated by Keras, from which performance metrics (accuracy and F1 score) were extracted. For statistical analysis, the nonparametric Friedman test and the Nemenyi post-hoc test were applied to determine significant differences between models. The results show that the CNN model with data augmentation outperformed the model without data augmentation and the one-shot model. It is concluded that the proposed optimized CNN model presents a superior performance compared to the evaluated alternatives, confirming its relevance for applications in the Peruvian agroindustry.

Keywords: CNN model, Maturity Classification, Embedding, Data Augmentation.

Capítulo 1. Introducción

El tomate constituye la hortaliza de mayor consumo a nivel mundial, con una producción anual que supera los 130 millones de toneladas. En el contexto peruano, esta verdura representa un cultivo de gran relevancia económica y nutricional, siendo ampliamente disponible en los mercados nacionales y consumida frecuentemente por la población (Trigo Riveros & Fernández Chávez, 2024).

La región norte del Perú ha demostrado un notable potencial productivo en el cultivo de tomate. Un proyecto desarrollado por Chavimochic logró incrementar significativamente la producción mediante la implementación de diversas técnicas agrícolas, alcanzando rendimientos de 90-100 toneladas por hectárea con una sola cosecha anual, en contraste con las 60 toneladas iniciales (Numata & Itagaki, 2011). Esta capacidad productiva del norte peruano cobra especial importancia al considerar que Castillo et al. (2022) identificaron niveles elevados de metales pesados en tomates producidos en la capital del país, los cuales exceden los estándares internacionales para exportación. Por tanto, el norte del Perú se posiciona como una región estratégica para la producción de tomate de calidad, con potencial para satisfacer tanto la demanda nacional como los requerimientos del mercado internacional.

La importancia económica del sector agrícola peruano, incluyendo la producción de tomate, se refleja en su contribución del 24.7% a la población económicamente activa del país (Velazco, 2012), lo que subraya la necesidad de implementar tecnologías que optimicen los procesos productivos y mejoren la competitividad del sector.

En este contexto, la determinación del estado de madurez del tomate durante la cosecha representa un factor crítico para la calidad del producto final y su valor comercial. Tradicionalmente, esta clasificación se realiza mediante inspección visual manual, un proceso subjetivo que puede generar inconsistencias y limitaciones en términos de eficiencia y

precisión. La visión artificial emerge como una alternativa tecnológica prometedora, ya que permite analizar imágenes digitales para extraer información relevante mediante algoritmos de aprendizaje automático capaces de detectar patrones en las características visuales asociadas a diferentes estados de maduración. Esta tecnología ofrece una solución objetiva, eficiente y escalable para la clasificación automatizada del estado de madurez del tomate, con potencial para transformar los procesos de calidad en la agroindustria peruana.

En los últimos años, los modelos de visión por computadora basados en redes convolucionales han demostrado un alto desempeño en tareas de clasificación de imágenes agrícolas. Sin embargo, la mayoría de estas soluciones dependen de grandes volúmenes de datos, como podemos observar en el trabajo de Campos Ferreira y González Camacho (2021), Donde se usaron como base 569 imágenes que luego fueran aumentadas,
De igual manera Mota-Delfin et al. (2018) uso como base 2800 imágenes que generalmente obtenidos de fuentes abiertas o bases globales que no necesariamente reflejan las condiciones visuales ni las características morfológicas de los tomates cultivados localmente.

Esta dependencia limita la aplicabilidad práctica de dichos modelos en contextos donde la recolección de datos a gran escala no es viable.

En este sentido, surge la necesidad de desarrollar enfoques comparativos que evalúen el rendimiento de modelos con conjuntos de datos pequeños. La presente investigación propone un modelo de aprendizaje one-shot basado en embeddings que permite el entrenamiento con pocas imágenes, y compara sus resultados con los de modelos convolucionales tradicionales, tanto en su versión estándar como con técnicas de aumento de datos. Este enfoque busca contribuir al diseño de soluciones más adaptables y eficientes para la clasificación de tomates en escenarios de disponibilidad limitada de datos.

En relación a lo anterior, se han considerado las siguientes investigaciones como antecedentes, tomando en cuenta que corresponden al uso de la visión artificial para la clasificación de la madures de diversos frutos:

Los autores Hu et al. (2022) en su investigación “Automatic detection of pecan fruits based on Faster RCNN with FPN in orchard”, propusieron un método preciso y eficiente de detección de frutos de nuez pecanera basado en visión por computadora en huertos de nogales naturales. Para resolver el problema de iluminación, se utilizó primero un algoritmo de compensación de luz para procesar las muestras recolectadas, y luego se estableció una versión mejorada de Faster Region Convolutional Neural Network (Faster RCNN) con las Feature Pyramid Networks (FPN) para comparar las muestras. Finalmente, se introdujo un método de conteo de número de nueces pecaneras para contar la cantidad de nueces. Se probaron un total de 241 imágenes de nueces pecaneras y se llevaron a cabo experimentos de comparación. De este estudio se rescata el uso del algoritmo de compensación de luz y la aplicación del concepto del Region of Interest (ROI), el estudio presentó una exactitud del 95.932%.

El estudio académico "A Maturity Estimation of Bell Pepper (Capsicum annuum L.) by Artificial Vision System for Quality Control" realizado por Villaseñor-Aguilar et al. (2020), tuvo como objetivo principal proponer e implementar un sistema de visión artificial que describiera automáticamente los niveles de madurez del pimiento, comparando la Lógica Difusa (FL) y las Redes Neuronales (ANN) para la etapa de clasificación, y referenciando la madurez mediante la medición de sólidos solubles totales (TSS) o grados Brix. La metodología empleada consistió en cuatro etapas principales: la adquisición de imágenes de cinco vistas de cada fruto usando una cámara Raspberry Pi 5 Megapixel y un sistema de visión computarizado con OpenCV-Python, dentro de un gabinete negro para control de

iluminación; la segmentación de las imágenes para remover el fondo y el ruido, utilizando la binarización en el espacio de color HSV; la segmentación de las regiones de interés (verde, amarillo, naranja y rojo) usando el algoritmo de componentes conectados y máscaras basadas en los componentes RGB y rangos específicos de Hue; y finalmente, la clasificación y predicción de grados Brix, que se realizó utilizando modelos de Redes Neuronales Artificiales (ANN), específicamente RBF-ANN para clasificación y Fitnet para predicción de Brix, y modelos de Lógica Difusa (FL) con el modelo Takagi–Sugeno. Se analizó una muestra de 50 pimientos producidos en la región Laja-Bajío, Guanajuato, México. Estos pimientos se clasificaron en cuatro clases de madurez: Clase 1 (10 muestras verdes), Clase 2 (7 frutos amarillos), Clase 3 (14 frutos anaranjados) y Clase 4 (19 muestras rojas). Los resultados más relevantes mostraron que los modelos basados en ANN lograron una precisión del 100% en la clasificación de la madurez, superando a los modelos de FL que alcanzaron un máximo del 88% de precisión. Para la predicción de grados Brix, el modelo NETFIT-ANN (Modelo 10) obtuvo el mejor resultado con un Coeficiente de Correlación de Pearson (R) de 0.79543, superando al modelo de FL con R = 0.696. Además, se observó un aumento constante en el contenido de TSS (Brix) a medida que aumentaba el estado de madurez del fruto. Las conclusiones indican que el sistema de visión artificial propuesto evalúa eficazmente la madurez del pimiento, mejorando la predicción de grados Brix utilizando las regiones de interés asociadas al color del pericarpio. El estudio recomienda el uso de la arquitectura RBF-ANN por su superioridad en la clasificación y destaca la ventaja del sistema de utilizar una cámara RGB de bajo costo en lugar de una multispectral, siendo una técnica no destructiva y aplicable a escenarios industriales en tiempo real.

Los autores Nana Hermana et al. (2021), en su investigación “Transfer Learning for Classification of Fruit Ripeness Using VGG16”, utilizaron 4 objetos de investigación: manzanas, naranjas, mangos y tomates, con unos 9000 datos de entrenamiento. Los datos se

entrenaron utilizando 200 iteraciones de época utilizando el método de transferencia de aprendizaje con los modelos VGG16. En la capa superior de ambos modelos, se aplicó el mismo MLP con varios parámetros, los datos se convirtieron de RGB a L * a * b con el objetivo de ser un descriptor de color en la fruta. Se entrenó utilizando CNN VGG16 con el método de transferencia de aprendizaje. De este estudio se resalta que el Dropout 0.5 muestra el mejor rendimiento del experimento con 4 escenarios que utilizaron diferentes técnicas y muestra el mejor rendimiento con una tasa de precisión promedio del escenario 4 del 92%.

El estudio "Clasificación de calidad de manzana para monitoreo de cosechabilidad utilizando visión por computador y algoritmos de aprendizaje profundo" (Apple quality grading for harvestability monitoring using computer vision and deep learning algorithms) realizado por Garcés Cadena et al. (2023), tuvo como objetivo principal proponer una herramienta práctica para asistir al agricultor en el reconocimiento de la calidad de la fruta, mejorando el proceso de cuantificación de manzana y monitoreo de su estado cosechable mediante el uso de visión por computador y algoritmos de aprendizaje profundo, buscando la detección del tipo de manzana para el conteo y la clasificación de su calidad. La metodología empleó el diseño y evaluación de un sistema terrestre portable con una estrategia de detección automática y segmentación de instancias de forma y borde. Para la detección del tipo de manzana, se utilizó el modelo de red SSD-MobileNet, una arquitectura optimizada para dispositivos móviles; y para la segmentación de instancias de calidad a nivel de píxel, se implementó una red neuronal convolucional rápida FCN-ResNet 18, seleccionada por su eficiencia en hardware de capacidades limitadas. El sistema fue entrenado, validado y probado en ensayos de laboratorio (con fondo controlado y sin fondo) y en campo real, utilizando librerías de PyTorch. La muestra o datos utilizados consistieron en dos bases de datos de imágenes RGB: 5125 imágenes para la detección de cinco variedades de manzana (Granny Smith, Golden Delicious, Fuji, Royal Gala y Red Delicious), y 338 imágenes de

manzanas tipo Royal Gala para la segmentación de calidad, conteniendo 2211 frutos clasificados en grados de calidad alta, media y baja. Las imágenes se obtuvieron en condiciones de iluminación controlada en laboratorio y natural en campo. Los resultados más relevantes mostraron que para la detección, el sistema logró una precisión del 86,7% en laboratorio con fondo controlado, 92,6% en laboratorio sin fondo, y 90,4% de exactitud en las pruebas de campo, aunque con la inclusión de una clase "no detectada" debido a la complejidad del ambiente externo. Para la segmentación de calidad, se obtuvo una precisión del 94,7%. Las métricas utilizadas incluyeron precisión, exactitud, exhaustividad, valor-F y matrices de confusión para la detección, y la Intersección sobre la Unión (IsU) para la segmentación. El estudio concluye que es posible identificar y clasificar semánticamente la calidad de las manzanas in-situ antes de la cosecha, siendo una herramienta práctica para el agricultor. Se recomienda su uso en aplicaciones reales donde la inspección de calidad se realiza manualmente, y se destacó que una fuente de luz adicional puede mejorar la calidad de la imagen y la precisión de la red en ambientes externos. Aunque el artículo no menciona limitaciones explícitamente como una sección, identificó que las condiciones externas del ambiente agrícola, como la variabilidad en la iluminación, las condiciones climáticas y la oclusión, presentan dificultades en el proceso de detección. Como líneas de investigación futura, los autores proponen la fusión de características de ambos métodos y la extracción de nuevas características a nivel de píxel para combinarlas con otras arquitecturas similares.

Los autores Hu et al. (2022) en su investigación “Recognition and localization of strawberries from 3D binocular cameras for a strawberry picking robot using coupled YOLO/Mask R-CNN”, sugieren la utilización de un robot recolector de fresas que pueda identificar y encontrar las fresas. En primer lugar, se recopilaron 1000 imágenes que incluían fresas maduras, inmaduras, individuales, múltiples y ocultas. Se utilizó una red de segmentación de instancias Mask R-CNN de detección en dos etapas y una red de detección

de objetivos YOLOv3 de detección en una etapa para entrenar un modelo de identificación de fresas que clasificara las fresas en dos categorías: maduras e inmaduras. Las tasas de precisión para YOLOv3 y Mask R-CNN fueron del 93.4% y 94.5%, respectivamente. De este estudio se rescata el uso de una captura de video binocular.

El trabajo de investigación "Clasificación de frutos del durazno en maduros, no maduros y dañados hacia la cosecha automatizada" realizado por Arévalo et al. (2021), tuvieron como objetivo principal proponer una solución mediante tecnología de visión artificial y redes neuronales convolucionales para el reconocimiento de duraznos maduros y la identificación de frutos dañados, con la finalidad de obtener frutos con el nivel de calidad adecuado para su comercialización. La metodología empleó la obtención de imágenes digitales de duraznos en un ambiente no controlado, las cuales fueron recortadas hasta obtener el área de interés. Se configuraron tres conjuntos de datos para diferentes pruebas: uno de duraznos maduros e inmaduros, otro de duraznos maduros e inmaduros enfocado en un área textural, y un tercero de duraznos sanos y dañados. Para el procesamiento, se aplicó una red neuronal convolucional (CNN), programada en el lenguaje Python utilizando las librerías Keras y TensorFlow. Esta CNN contaba con una arquitectura de tres capas, incluyendo filtros de 2x2 y 3x3, con las dos primeras capas incorporando acción de pooling, y capas como Relu, Flaten y Dropout con la función Softmax para la clasificación final. La muestra o datos utilizados consistió en imágenes tomadas con una cámara profesional Nikon d3500 de 24.2 megapíxeles, distribuidas en tres conjuntos para entrenamiento y validación: 360 imágenes para el conjunto de duraznos maduros e inmaduros (160 maduros entrenamiento, 40 maduros validación, 128 inmaduros entrenamiento, 32 inmaduros validación), 360 imágenes para el conjunto de textura de maduros e inmaduros, y 600 imágenes para el conjunto de sanos y dañados (320 sanos entrenamiento, 160 sanos validación, 80 dañados entrenamiento, 40 dañados validación). Adicionalmente, se utilizó un

conjunto de 560 imágenes para una clasificación de tres categorías (maduros, inmaduros y dañados) entrenada durante 25 épocas, con 60 imágenes para pruebas. Los resultados más relevantes fueron una precisión del 95.31% al clasificar duraznos maduros e inmaduros, una precisión del 92.18% al clasificar duraznos sanos y dañados, y una precisión del 83.33% al clasificar las tres categorías (dañados, inmaduros y maduros). Se observó que las imágenes enfocadas solo en la textura del durazno resultaron en una menor precisión (84.37%). Las métricas utilizadas para evaluar el rendimiento incluyeron precisión, sensibilidad y especificidad. Las conclusiones del estudio indican que la precisión obtenida sugiere que la clasificación desarrollada puede ser implementada en una máquina para la cosecha automatizada de duraznos. Se resalta que es más efectivo utilizar la imagen completa del durazno en lugar de solo su textura para obtener mejores resultados. Además, se concluye que clasificar las tres categorías produce resultados aceptables, y se enfatiza la importancia de utilizar una gran cantidad de datos para mejorar la precisión y fiabilidad de las CNN. Aunque el trabajo no presenta una sección explícita de limitaciones, se infiere que la precisión disminuye considerablemente cuando se involucran más de dos categorías, incluso si las características físicas son explícitas, sugiriendo la necesidad de más investigaciones o redes especializadas para múltiples grupos. También se menciona que la solidez y el balanceo del conjunto de datos son esenciales para el entrenamiento de la CNN. En cuanto a las líneas de investigación futura, el estudio propone la clasificación de frutos por tamaño para discriminar aquellos que no cumplen con los requisitos del mercado, la clasificación por forma para identificar frutos deformes que puedan ser rechazados, y la generalización del método y modelo para que pueda trabajar con otras frutas, integrando el conjunto de algoritmos en una máquina clasificadora ajustando parámetros.

Los autores López-Barrios et al. (2023) en su investigación “Green Sweet Pepper Fruit and Peduncle Detection Using Mask R-CNN in Greenhouses” en este estudio se utiliza

una arquitectura llamada ResNet-101 + FPN para mejorar la extracción de características y representación de objetos en diferentes escalas. Se generan imágenes de máscara para frutas y pedúnculos, enfocándose en el pimiento dulce verde que es de color más complejo. Además de las cajas delimitadoras, Mask R-CNN proporciona máscaras binarias para mejorar la localización en el espacio 3D. Los resultados de predicción mostraron una precisión del 84.53% para las frutas y del 71.78% para los pedúnculos. La tasa media de precisión promedio con una intersección sobre unión del 50% (mAP@IoU=50) para la segmentación de instancias en todo el modelo fue del 72.64%. Estos resultados demuestran la efectividad del método propuesto en la detección y segmentación de pedúnculos y frutas de pimiento dulce verde. De este estudio se rescata el excepcional trabajo de reconocer pimientos verdes sobre una superficie de casi el mismo color, esto gracias a aplicar las fases y técnicas necesarias como son FPN para enfocarse en ROIs atraves de una RPN, siendo estas regiones de interés identificas las que finalmente serán ingresadas en la FCN

El estudio académico "Benchmark of Deep Learning and a Proposed HSV Colour Space Models for the Detection and Classification of Greenhouse Tomato" fue realizado por Moreira et al. (2022), tuvo como objetivo principal proponer e implementar un sistema de visión artificial que describiera automáticamente los niveles de madurez del tomate, comparando la efectividad de los modelos de Aprendizaje Profundo (DL), específicamente SSD MobileNet v2 y YOLOv4, con un modelo basado en el espacio de color HSV para la detección y clasificación de tomates de invernadero, con miras a la automatización de la cosecha robótica. La metodología empleó un enfoque de cuatro etapas principales: adquisición de imágenes con cámaras ZED y Raspberry Pi, segmentación para remover el fondo y ruido, identificación de regiones de interés (RoI) usando componentes RGB y rangos de Hue, y finalmente, clasificación y predicción. Para los modelos DL, los datos fueron preprocesados, divididos en conjuntos de entrenamiento, validación y prueba (60%,

20%, 20% respectivamente) y se aplicó aumento de datos; los modelos SSD MobileNet v2 y YOLOv4 fueron entrenados con transfer learning usando TensorFlow. Por otro lado, el modelo HSV se desarrolló desde cero, utilizando la conversión al espacio de color HSV y un modelo de mezcla gaussiana para generar histogramas de Hue y un clasificador estadístico cuadrático. La muestra o datos utilizados consistió en dos conjuntos de imágenes RGB de tomates variedad "Plum" de invernadero: AgRobTomato Dataset (449 imágenes) y RpiTomato Dataset (258 imágenes). Para el entrenamiento de los modelos DL, se seleccionaron y procesaron 632 imágenes que, tras operaciones de división y reescalado, se ampliaron a 1029 imágenes iniciales para los conjuntos de datos, llegando a 7598 imágenes anotadas con aumento de datos. Para el desarrollo del modelo HSV, se emplearon 40 imágenes (10 de cada clase de madurez: Verde, Amarillo (Turning), Rojo Claro (Light Red), Rojo. Los resultados más relevantes para la detección mostraron que el modelo YOLOv4 se destacó con un F1-Score de 85.81% y un mAP de 82.97%, mientras que para la tarea de clasificación de madurez, YOLOv4 fue nuevamente el mejor con un Macro F1-Score de 74.16% y una Precisión Balanceada de 68.87%. El modelo HSV propuesto, aunque menos complejo y con menor cantidad de datos de entrenamiento (solo 40 imágenes), superó al modelo SSD MobileNet v2 y logró una Precisión Balanceada similar al YOLOv4 (68.10%), destacándose en la clasificación de las clases Verde (Precisión 98.24%, Recall 98.31%) y Roja. Las métricas utilizadas incluyeron IoU, Recall, Precision, F1-Score, Mean Average Precision (mAP), Macro F1-Score y Balanced Accuracy. Las conclusiones del estudio afirman que el YOLOv4 es el modelo más eficaz para la detección y clasificación de tomates en entornos de invernadero, con el modelo HSV siendo una alternativa prometedora para la clasificación, especialmente por su simplicidad y eficiencia en recursos. Sin embargo, todos los modelos mostraron dificultad en distinguir las clases de madurez intermedias (Amarillo y Rojo Claro). Aunque el artículo no presenta una sección

explícita de limitaciones, se infiere que las principales radican en la dificultad inherente de las clases de madurez intermedias para la clasificación (incluso para el ojo humano, lo cual afecta la anotación), y que el modelo HSV, si bien es simple, no maneja tan bien los escenarios altamente complejos como los modelos DL. Las líneas de investigación futura propuestas incluyen la expansión y balanceo del dataset con más tomates rojos, la evaluación del rendimiento de los modelos en condiciones de tiempo real en invernaderos, la comparación con nuevas arquitecturas de redes neuronales más optimizadas (como ResNet y ViT transformers), y la mejora del modelo HSV para afrontar entornos complejos, posiblemente incorporando técnicas de segmentación DL

Chucos Baquerizo y Vega Ventocilla (2022) en su investigación "Evaluación de algoritmos de machine learning en la clasificación de imágenes satelitales multiespectrales, caso: Amazonia Peruana" tuvieron como objetivo desarrollar un modelo de Machine Learning para la clasificación de imágenes satelitales en la Amazonia peruana para ello se implementaron algoritmos de Machine Learning para la clasificación de imágenes satelitales multiespectrales. Se menciona que se evaluaron diferentes algoritmos y se realizaron pruebas para determinar su precisión en la clasificación de las imágenes. Los resultados mencionan que se obtuvo una precisión de 0.864 en las pruebas realizadas con los algoritmos. Se indica que esta precisión es aceptable, pero se destaca la necesidad de contar con más datos de entrenamiento para lograr una generalización adecuada en la clasificación de las imágenes

Osco-Mamani et al. (2023) en su investigación "The Detection and Counting of Olive Tree Fruits Using Deep Learning Models in Tacna, Perú", tuvieron como objetivo desarrollar un modelo de aprendizaje profundo altamente preciso para la clasificación de enfermedades de las hojas de olivo en la región de La Yarada-Los Palos en Tacna, Perú. Para lograr este objetivo, se utilizaron métodos de aprendizaje profundo, específicamente se implementó una arquitectura modificada de VGG16. Para ello utilizaron técnicas de aumento de datos, transfer

learning y fine-tuning para mejorar el rendimiento del modelo. Se realizaron experimentos comparativos exhaustivos para evaluar el impacto de estas técnicas en la clasificación de enfermedades de las hojas de olivo. Los resultados obtenidos en la investigación demostraron la efectividad de utilizar transfer learning en comparación con el aprendizaje desde cero, así como la importancia de técnicas como el aumento de datos y el fine-tuning en la clasificación precisa de enfermedades de las hojas de olivo.

Así mismo, en este trabajo de investigación se mencionan conceptos como:

Clasificación de la maduración de los tomates: La clasificación de la madurez del tomate es un aspecto relevante del control de calidad en la industria de productos agrícolas. Varios estudios han explorado métodos para clasificar los tomates según el color, el tamaño y la forma (Choi et al., 1995).

Modelo de Visión Artificial: Un modelo de visión artificial es un sistema computacional que procesa y analiza imágenes para extraer información relevante. Estos modelos pueden aplicarse en diversos campos, como el transporte público para detectar aglomeraciones (Cárdenas et al., 2024)

Clima: El clima se refiere a las condiciones atmosféricas promedio en un lugar durante períodos prolongados, mientras que el tiempo describe el estado actual de la atmósfera (Torres Puente, 2019).

Clasificación botánica: La clasificación botánica es un proceso fundamental en la biología que refleja el conocimiento acumulado y evoluciona con nuevos descubrimientos (Barkley, 1949). Tradicionalmente, las plantas se han categorizado según criterios morfológicos y funcionales, tanto en sistemas científicos como en clasificaciones tradicionales o "folk" (Gama Campillo, 2024; Da Silva & Freixo, 2020)

Manejo postcosecha: Se refiere al tratamiento de los productos agrícolas después de la cosecha, particularmente frutas y granos. Abarca varias etapas desde la recolección hasta el

almacenamiento, con el objetivo de mantener la calidad del producto y reducir las pérdidas (Puga-Muima & Chalco-Sandoval, 2021).

Reconocimiento de Patrones: El reconocimiento de patrones es una disciplina que utiliza técnicas de procesamiento de imágenes y aprendizaje automático para identificar y clasificar patrones en datos (Santa Maria Pinedo et al., 2021).

Inferencia: La inferencia en modelos de inteligencia artificial (IA) abarca diversos enfoques y aplicaciones. La abducción basada en modelos se presenta como una herramienta para el razonamiento creativo en ciencias, reforzando el papel de la abducción en el descubrimiento (Figueroa, 2011).

Visión Artificial: Visión artificial, también conocida como visión por computador, es un campo de investigación que se centra en cómo las computadoras interpretan imágenes y videos digitales (Atencio & Cruz, 2022).

OpenCV (Open Source Computer Vision Library): Es una biblioteca de código abierto ampliamente utilizada para visión artificial y procesamiento de imágenes (Cervera, 2020). Es compatible con múltiples lenguajes de programación, incluyendo C++, Python y Java, y está disponible en varias plataformas como Windows, Linux, macOS y Android (Cervera, 2020).

CNN: Las Redes Neuronales Convolucionales (CNN, por sus siglas en inglés) son un tipo de algoritmo de aprendizaje profundo con aplicaciones en el procesamiento de imágenes, la detección de objetos y el procesamiento del lenguaje natural (Purwono et al., 2023).

Yolo: (You Only Look Once) es un algoritmo de aprendizaje profundo para la detección de objetos en tiempo real que ha ganado popularidad debido a su velocidad y precisión (Kanna et al., 2024). YOLO se ha aplicado en diversos campos, incluida la agricultura para la detección de cultivos y enfermedades (Kanna et al., 2024).

FPN: Las Redes de Pirámides de Características (FPN) son un componente importante en los sistemas de detección de objetos, diseñadas para abordar los desafíos de la detección multiescala (Lin et al., 2016). FPN explota la jerarquía piramidal inherente de las redes convolucionales profundas para construir pirámides de características de manera eficiente, ofreciendo un rendimiento mejorado como un extractor de características genérico (Lin et al., 2016).

Z-RDCE: Zero-Reference Deep Curve Estimation (Zero-DCE) es un enfoque novedoso en aprendizaje profundo para mejorar imágenes con poca luz sin necesidad de datos emparejados o no emparejados durante el entrenamiento (Li et al., 2021). Utiliza una red profunda ligera llamada DCE-Net para estimar curvas de orden superior y por píxel para el ajuste dinámico del rango (Komaravalli et al., 2024).

VGG16: VGG16 es un modelo de aprendizaje profundo que consta de 16 capas, incluyendo 13 capas convolucionales y 3 capas totalmente conectadas (Yadav et al., 2024). Es reconocido por su simplicidad y eficacia en tareas de clasificación de imágenes y reconocimiento de objetos (Bala et al., 2024).

RPN: Las Redes de Propuesta de Regiones (RPNs) son un componente crucial en los sistemas de detección de objetos basados en aprendizaje profundo. Las RPNs generan regiones candidatas a objetos (RoIs) para su posterior procesamiento por detectores de objetos (Pirinen & Sminchisescu, 2018). Normalmente, se integran con redes neuronales convolucionales (CNNs) para extraer características de imagen de alto nivel (Guo et al., 2020).

ROI: Investigaciones recientes han explorado la aplicación de técnicas de aprendizaje profundo para la detección y segmentación de Regiones de Interés (ROI) en imágenes médicas. Se han utilizado Redes Neuronales Convolucionales para la segmentación de ROI de huellas dactilares, demostrando un rendimiento mejorado, especialmente en

imágenes ruidosas (Stojanovic et al., 2016).

K-meas: Deep K-means es una extensión del algoritmo K-means tradicional que incorpora técnicas de aprendizaje profundo para mejorar el rendimiento de la agrupación. Huang et al. (2020) propusieron un modelo deep K-means que aprende representaciones jerárquicas de los datos, agrupando los puntos similares más cerca capa por capa. (Du, 2019) demostró que K-means puede implementarse como una red neuronal profunda, tendiendo un puente entre los enfoques de aprendizaje convencional y profundo.

Uneven illumination correcting (UIC): La corrección de iluminación no uniforme (UIC) en el aprendizaje profundo es una técnica emergente para abordar las inconsistencias de iluminación en las imágenes. Estudios recientes han aplicado la UIC a varios dominios, incluyendo la dermatoscopia (Mei et al., 2016), el escaneo de porta objetos completos.

BBOL: (Bounding Box Object Localization) son representaciones geométricas simplificadas que se utilizan para aproximar objetos complejos, principalmente en tareas de detección de colisiones y segmentación de imágenes (Lempitsky et al., 2009).

TensorFlow: Es un sistema de aprendizaje automático a gran escala diseñado para entornos heterogéneos que utiliza grafos de flujo de datos para representar el cálculo y el estado compartido (Abadi et al., 2016). Ofrece flexibilidad en la asignación de operaciones a través de múltiples máquinas y dispositivos, incluyendo CPUs, GPUs y ASICs diseñados a medida llamados Tensor Processing Units (TPUs) (Abadi et al., 2016).

TensorBoard: TensorBoard es un potente conjunto de herramientas de visualización para TensorFlow, que ofrece diversas funciones para mejorar la experimentación del aprendizaje automático y el diseño de redes neuronales. Proporciona API para

visualizar valores escalares, imágenes, texto, audio e histogramas (Huang & Le, 2021)

ZeroShot: El aprendizaje de disparo cero (ZSL) es un paradigma de aprendizaje automático que permite a los modelos de IA clasificar instancias de categorías no vistas sin datos de entrenamiento específicos (Wang et al., 2019).

Hiperparametros: Los hiperparámetros son restricciones cruciales que controlan el proceso de aprendizaje en los modelos de aprendizaje profundo (Bhattacharjee et al., 2021). La optimización adecuada de estos parámetros es esencial para mejorar el rendimiento y la generalización del modelo (Aghaebrahimian y Cieliebak, 2019).

I.2. Formulación del problema

¿En qué medida un modelo CNN desarrollado para la detección del estado de madurez del tomate mejora las métricas de precisión y F1-score en comparación con los modelos one-shot y sin data augmentation, para su aplicación en la agroindustria peruana, 2025?

I.3. Objetivos

I.3.1. Objetivo General

Determinar en qué medida un modelo CNN desarrollado para la detección del estado de madurez del tomate mejora las métricas de precisión y F1-score en comparación con los modelos one-shot y sin data augmentation, para su aplicación en la agroindustria peruana, 2025.

I.3.2. Objetivos Específicos

OE1: Diseñar y desarrollar un modelo CNN optimizado para la clasificación del estado de madurez del tomate, considerando las características específicas de la agroindustria peruana.

OE2: Implementar el modelo CNN desarrollado utilizando un dataset de

imágenes de tomates en diferentes estados de madurez, aplicando técnicas de preprocesamiento y entrenamiento adecuadas.

OE3: Evaluar el rendimiento del modelo CNN desarrollado mediante las métricas de precisión y F1-score, utilizando técnicas de validación cruzada y conjuntos de datos de prueba.

OE4: Comparar las métricas de precisión y F1-score del modelo CNN desarrollado con los modelos existentes en la literatura, mediante análisis estadístico de los resultados obtenidos.

I.4. Hipótesis

Hipótesis Nula (H_0):

El modelo CNN desarrollado para la detección del estado de madurez del tomate no mejora las métricas de precisión y F1-score en comparación a los modelos one-shot y sin data augmentation, siendo viable para su aplicación en la agroindustria peruana, 2025.

Hipótesis de investigación (H_i):

El modelo CNN desarrollado para la detección del estado de madurez del tomate mejora significativamente las métricas de precisión y F1-score en comparación con los modelos one-shot y sin data augmentation en la agroindustria peruana, 2025.

Hipótesis Específicas:

$HE1_0$: El modelo CNN desarrollado con arquitectura optimizada no alcanza métricas de entrenamiento superiores al 85% en precisión y F1-score durante la fase de desarrollo.

$HE1_i$: El modelo CNN desarrollado con arquitectura optimizada alcanza métricas de entrenamiento superiores al 85% en precisión y F1-score durante la fase de desarrollo.

$H2_0$: La implementación del modelo CNN con técnicas de preprocesamiento

adecuadas no mejora su capacidad de generalización en datasets de validación.

$H2_i$: La implementación del modelo CNN con técnicas de preprocesamiento adecuadas mejora significativamente su capacidad de generalización en datasets de validación.

$H3_0$: El modelo CNN desarrollado no supera en al menos 5% las métricas de precisión y F1-score de los modelos de referencia identificados en la literatura.

$H3_i$: El modelo CNN desarrollado supera en al menos 5% las métricas de precisión y F1-score de los modelos de referencia identificados en la literatura.

$H4_0$: El modelo CNN desarrollado no presenta características técnicas y económicas que lo hacen viable para implementación en la agroindustria peruana.

$H4_i$: El modelo CNN desarrollado presenta características técnicas y económicas que lo hacen viable para implementación en la agroindustria peruana.

CAPÍTULO II: METODOLOGÍA

La investigación tiene un enfoque cuantitativo, ya que se basa en la recolección y análisis de datos numéricos obtenidos de métricas de evaluación como la precisión y el F1-score. Es de nivel explicativo, pues busca determinar la influencia de diferentes técnicas de entrenamiento de redes neuronales convolucionales (modelo base, CNN con data augmentation y modelo one-shot) sobre el rendimiento del modelo. Su alcance es aplicado, al orientarse a resolver un problema real de la agroindustria peruana: la detección automatizada del estado de madurez del tomate. El diseño de investigación es cuasi-experimental, dado que se manipula la variable independiente (técnica de entrenamiento) sin asignación aleatoria de los conjuntos de datos, para evaluar su efecto en las métricas de desempeño.

2.1. Tipo de Investigación

El diseño de investigación es de tipo cuasi-experimental, dado que se manipula la variable independiente (técnica de entrenamiento del modelo: modelo base, CNN con data augmentation y modelo one-shot) para observar su efecto sobre la variable dependiente (métricas de precisión y F1-score). La comparación entre los grupos experimentales no se realiza mediante asignación aleatoria de los datos, sino a partir de conjuntos previamente establecidos, lo que caracteriza el estudio como cuasi-experimental. A partir de ello la hipótesis será evaluada.

Según el enfoque: La investigación es de enfoque cuantitativo, dado que se fundamenta en la recolección y análisis de datos numéricos obtenidos a partir de las métricas de evaluación de los modelos de redes neuronales convolucionales (precisión y F1-score). El estudio busca medir y comparar de manera objetiva el rendimiento de tres enfoques distintos modelo CNN desarrollado, CNN con data augmentation y modelo one-shot, aplicando procedimientos estadísticos para determinar diferencias significativas entre ellos.

Según el propósito: Es de tipo aplicada, ya que utiliza técnicas de redes neuronales convolucionales para resolver un problema concreto en la agroindustria peruana: la detección automatizada del estado de madurez del tomate, buscando una solución tecnológica práctica.

Según su alcance: El estudio es de tipo explicativo, dado que busca identificar y demostrar la influencia de diferentes técnicas de entrenamiento (modelo CNN desarrollado, CNN con data augmentation y modelo one-shot) sobre el rendimiento de la detección del estado de madurez del tomate, evaluado mediante métricas de precisión y F1-Score. El objetivo es establecer una relación causal entre el tipo de técnica empleada y la variación en los indicadores de desempeño, en el contexto de la agroindustria peruana.

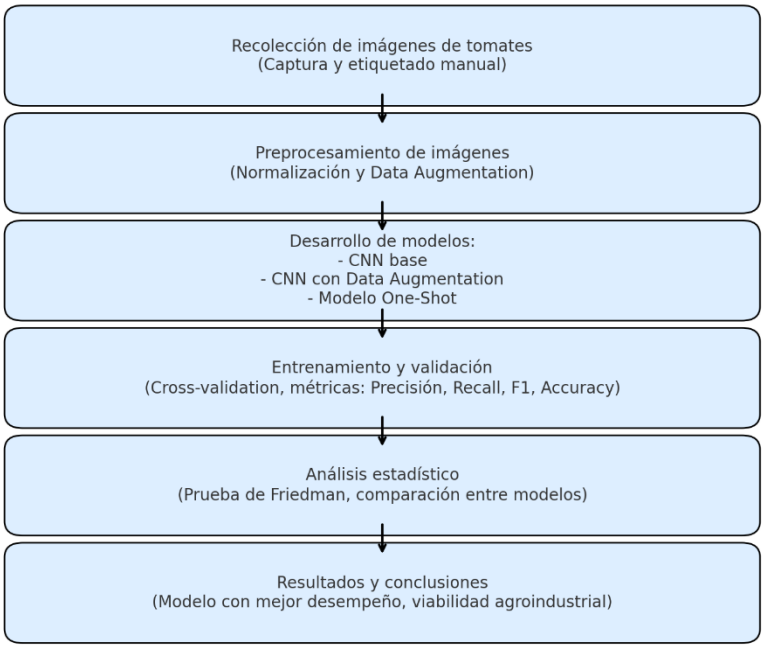
Diseño: El diseño de la investigación es cuasi-experimental, dado que se manipula la variable independiente —las técnicas de entrenamiento del modelo (CNN base, CNN con data augmentation y modelo one-shot)— para observar su efecto sobre la variable dependiente

(métricas de Precisión y F1-Score). La comparación se realiza a partir de conjuntos previamente definidos de imágenes de tomates, sin asignación aleatoria, lo que caracteriza al estudio como cuasi-experimental. Para mantener la rigurosidad metodológica, se emplea validación cruzada (cross-validation), garantizando una evaluación más estable de los modelos.

El flujo del procedimiento metodológico seguido en la investigación se resume en la siguiente

Figura 1

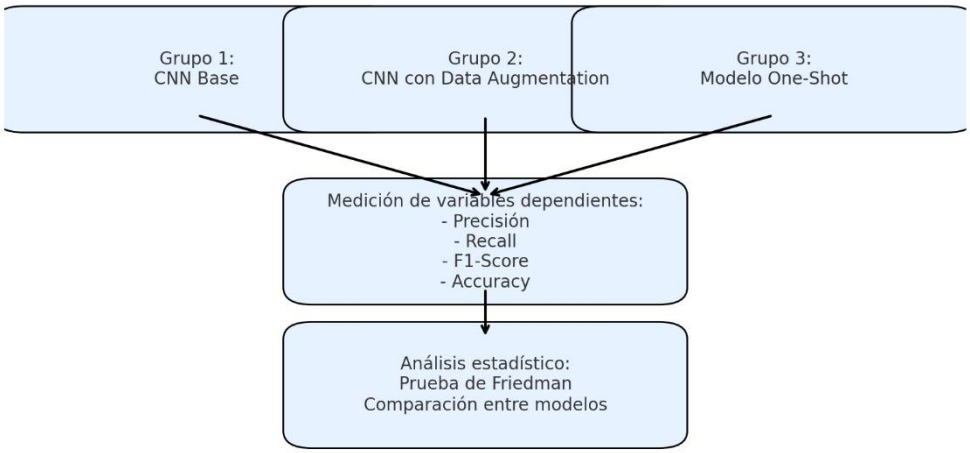
Flujo experimental de la investigación



Fuente: Elaboración propia

Figura 2

Diseño cuasiexperimental



Fuente: Elaboración propia

2.2. Población y Muestra

Población: Todos los tomates dentro de las fotos recolectadas de pertenecientes a la agroindustria peruana.

Muestra:

$$n = \frac{(Z)^2 * P * Q}{(E)^2}$$

Ecuación 1: Fórmula para determinar la muestra

Dónde:

En los antecedentes, se analizó el margen de confianza, probabilidad de éxito, probabilidad complementaria y error permitido en la muestra. Es por ello que se concluyó y seleccionó los siguientes parámetros para las distintas variables.

n = Cantidad de elementos en la muestra

Z = Margen de confianza: 90% (1.65)

P = Probabilidad de éxitos: 85% (0.85)

Q = Probabilidad complementaria: 15% (0.15)

E = Error permitido en la muestra: 6.95% (0.0695)

Sustituyendo en la fórmula:

$$n = \frac{(1.65)^2 * 0.85 * 0.15}{(0.11)^2}$$

Desarrollando:

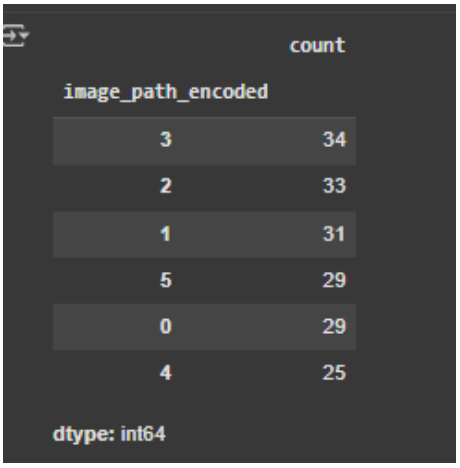
$$n = 72$$

Resultado del cálculo:

La cantidad mínima de elementos en la muestra calculada por el análisis de potencia fue de 72 imágenes; sin embargo, para garantizar métricas internas más estables por réplica y contar con un número adecuado de réplicas por clase, se utilizará un total de 138 imágenes, es decir, 6 folds de validación cruzada de entre 25 a 34 tomates por imagen. Esta decisión mantiene el requisito mínimo estadístico y añade robustez al análisis al proporcionar dos réplicas independientes por clase y seis réplicas por métrica y por modelo, facilitando así la ejecución de la prueba global entre los tres modelos.

Figura 3

Conteo de tomates detectados por imagen



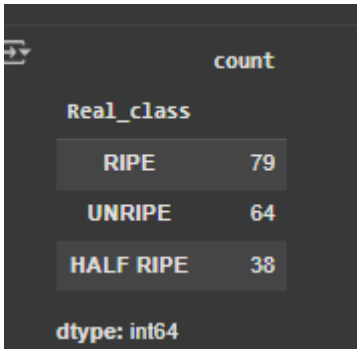
image_path_encoded	count
3	34
2	33
1	31
5	29
0	29
4	25

dtype: int64

Nota. Se aplica un value counts al DataFrame. Elaboración propia.

Figura 4

Conteo de clases del dataset de Pruebas



Real_class	count
RIPE	79
UNRIPE	64
HALF RIPE	38

dtype: int64

Nota. Se aplica un value counts al DataFrame. Elaboración propia.

Unidad de análisis:

1 tomate dentro de las fotos recolectadas de pertenecientes a la agroindustria peruana.

2.3. Técnica y análisis de datos

Técnica

Tabla 1: Técnica e instrumento de recopilación de información

Variable	Dimensión	Técnica	Instrumento	Descripción
Rendimiento de modelos de clasificación de tomates	F1 Score, Precisión, Recall	Observación	TensorBoard; Scikit-learn;keras; Hoja de Observacion(CSV)	La técnica de observación se realizará mediante TensorBoard (métricas durante evaluación/entrenamiento) y scikit-learn (evaluación sobre datos reales). La hoja de observación (CSV) contendrá las etiquetas reales y metadatos por imagen; desde Jupyter se unifican predicciones y ground-truth para el cálculo de métricas y análisis estadístico.

Fuente: Elaboración propia

Instrumento de medición: Los logs generados automáticamente por la librería Keras durante el entrenamiento y validación de los modelos, que registran métricas de rendimiento (accuracy, precision, F1-score).

Validez y confiabilidad del instrumento: La validez se sustenta en que dichas métricas son estándares internacionalmente aceptados en la evaluación de modelos de clasificación supervisada. La confiabilidad se garantiza al ejecutar múltiples repeticiones con validación cruzada (6 folds), minimizando el sesgo y asegurando consistencia en los resultados.

Análisis de datos

Con el propósito de llevar un análisis y posterior interpretación fehaciente de los datos la ejecución del notebook de Google Colab que implementa los modelos desarrollados así mismo almacena la información pertinente en un dataframe para su posterior análisis. Se efectuó el test de Friedman el cual es particularmente útil para analizar múltiples muestras relacionadas (Berlanga-Silvente & Rubio-Hurtado, 2012). La prueba es equivalente al diseño de bloques completos al azar en el análisis paramétrico y se considera una opción válida cuando no se cumplen los supuestos del ANOVA (Núñez Colín, 2018; Núñez-Colin, 2019)

2.4. Procedimiento

Para llevar a cabo la medición de las dimensiones se usará la librería de Python Scikit-learn para el cálculo de las mismas y la especializada en estadística SciPy para aplicar las pruebas respectivas a las dimensiones obtenidas de cada modelo.

Así mismo se usaron las siguientes características de entorno propia del Google Colab.

Tabla 2 : Entorno de experimentación en Google Colab

Recurso	Detalle
Plataforma	Google Colab (versión actual, octubre 2025)
Sistema operativo	Ubuntu 22.04.3 LTS (entorno virtualizado en Colab)
CPU	2 × Intel Xeon @ 2.20GHz (asignados dinámicamente)
Memoria RAM	12 GB (aproximadamente, asignación estándar de Colab)
GPU (opcional)	NVIDIA Tesla T4 (16 GB VRAM) *cuando se habilita GPU en Colab*
Librerías principales	Python 3.10, TensorFlow 2.15, Keras 2.15, scikit-learn 1.5, SciPy 1.14

Nota. El hardware asignado por Google Colab es dinámico y puede variar entre sesiones. Para este trabajo se usó la configuración estándar de Colab en octubre de 2025, con GPU NVIDIA Tesla T4 habilitada.

Para conseguir las imágenes de los tomates a clasificar se visitó el mercado más importante de la ciudad “La Hermelinda” así como mercados menos importantes, se tomaron diversas fotos de tomates en diversos contextos, de las cuales se seleccionaron solo 6 por contar con la iluminación más neutra posible, así como la cantidad de tomates necesaria, estas 6 fotos serán nuestros 6 clusters de los obtendremos una muestra de cada una de las dimensiones a calcular.

Tabla 3: Elementos de análisis

Cluster	IMÁGENES POR CLASIFICAR
1	Imagen 01
2	Imagen 02
3	Imagen 03
4	Imagen 04
5	Imagen 05
6	Imagen 06

Fuente: Elaboración propia.

La recopilación de los datos de cada clase de real para cada una de las imágenes

obtenidas, se llevó a cabo en simultaneo con su obtención haciendo consulta a los proveedores que interactúan con dicho producto a diario, y esta información fue guardada de forma tabular en un csv con la siguiente estructura para cada imagen.

Tabla 4: Imágenes por clasificar

IMÁGENES POR CLASIFICAR	Index de tomate	Estado de Maduración
Imagen 01	0	HALF RIPE
Imagen 01	1	HALF RIPE
Imagen 01	2	HALF RIPE
Imagen 01	3	HALF RIPE
Imagen 02	0	HALF RIPE
Imagen 02	1	RIPE
Imagen 02	2	RIPE
Imagen 02	3	HALF RIPE

Fuente: Elaboración propia.

Durante el proceso de entrenamiento del modelo CNN, se registraron las curvas de pérdida (loss) correspondientes tanto al conjunto de entrenamiento como al de validación, con el propósito de monitorear la estabilidad del aprendizaje y prevenir el sobreajuste. Dichas gráficas permitieron evaluar visualmente la convergencia del modelo y validar los parámetros de entrenamiento seleccionados. Por motivos de síntesis, las representaciones gráficas se presentan en el Anexo 3 para el modelo con Data Augmantation, y en el Anexo 4 para el modelo sin Data Augmentation, cabe recalcar que el modelo One Shot no cuenta con esta métrica por la naturaleza del modelo.

El desarrollo y la toma de datos para cada uno de los modelos se llevó acabo en un periodo de 3 meses y desarrollo siguiendo la ahora presentada estructura:

En el desarrollo de software y modelos de inteligencia artificial, se opto por la

metodología SCRUM la cual en investigaciones recientes demuestran la aplicación de la metodología Scrum en el desarrollo de modelos de aprendizaje automático (machine learning) y aprendizaje profundo (deep learning) en diversos dominios. (Urbina Cienfuegos & Bravo Rivas, 2025).

Historias de Usuario

Tabla 5: Historias de Usuario.

ID	User Story	Objective	Priority	Sprint
HU-01	Como desarrollador, quiero configurar el entorno de TensorFlow/Keras para garantizar compatibilidad con el modelo de embeddings.	Instalar Python 3.9, TensorFlow 2.10 y librerías auxiliares.	High	Sprint 1
HU-02	Como equipo, necesitamos un dataset inicial de imágenes de tomates (verde, madurando, maduro) para pruebas	Recolectar al menos 76 imágenes y etiquetarlas manualmente	High	Sprint 1

HU-03	Como investigador, necesito los 3 modelos para comparar métricas	Desarrollar 3 modelos de clasificación	High	Sprint 2
HU-04	Como investigador, necesito los tomates en solitario para aplicar los modelos de clasificación.	Desarrollar un modelo de selección de tomates.	High	Sprint 3
HU-05	Como desarrollador, necesito implementar un script que aplique los modelos desarrollados.	Implementar script para procesar las imágenes recolectadas.	High	Sprint 4
HU-06	Como equipo, necesitamos evaluar el modelo con métricas (precisión, recall) para garantizar confiabilidad.	Generar reporte de métricas con al menos 76 imágenes de test	Medium	Sprint 5
HU-07	Como investigador, necesito generar	Implementar script que guarda las	Medium	Sprint 4

	imágenes etiquetadas para presentar en la investigación	imágenes con las anotaciones de los modelos desarrollados		

Fuente: Elaboración propia.

Product Backlog

Tabla 6: Product Backblog del proyecto.

Sprint	Tarea	**Responsable**	**Prioridad**	**Fecha** Inicio	**Fecha de Fin**
1	Configuración de entorno de desarrollo	Andre Sanchez	Alta	6 de junio de 2025	20 de junio de 2025
	Recolección de imágenes de tomates	David Anaya	Alta		
2	Desarrollo de los 3 modelos de Clasificación.	Andre y David	Alta	27 de junio de 2025	11 de julio de 2025
3	Desarrollo del modelo de selección.	Andre	Alta	14 de Julio de 2025	25 de julio de 2025
4	Implementar script para procesar las imágenes de Tomates	Andre Sanchez	Alta	28 de julio de 2025	8 de agosto de 2025

	Desarrollar e implementar script para guardar las fotos procesadas	David Anaya	Alta		
5	Generar reporte de métricas con las imágenes test	David Anaya	Alta	11 de agosto de 2025	15 de agosto de 2025

Fuente: Elaboración propia.

Sprint Review Meeting

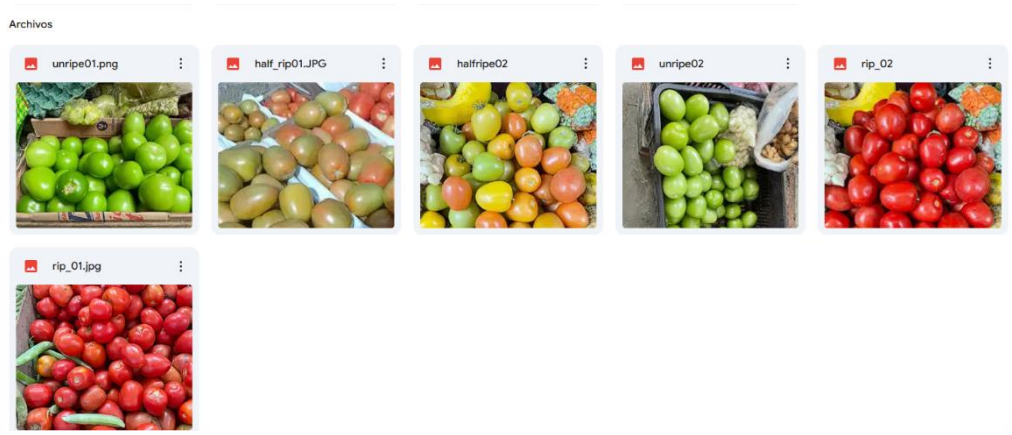
Tabla 7: Review Meeting 1

Review Meeting		1
Requerimiento	Configuración de entorno de desarrollo	Preparación de entorno de desarrollo

Instalación de las librerías pertinentes y sus dependencias:

```
Downloading nvidia_cublas_cu12-12.4.5.8-py3-none-manylinux2014_x86_64.whl (363.4 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 363.4/363.4 MB 4.5 MB/s eta 0:00:00
Downloading nvidia_cuda_cupti_cu12-12.4.127-py3-none-manylinux2014_x86_64.whl (13.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 13.8/13.8 MB 78.8 MB/s eta 0:00:00
Downloading nvidia_cuda_nvrtc_cu12-12.4.127-py3-none-manylinux2014_x86_64.whl (24.6 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 24.6/24.6 MB 67.9 MB/s eta 0:00:00
Downloading nvidia_cuda_runtime_cu12-12.4.127-py3-none-manylinux2014_x86_64.whl (883 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 883.7/883.7 kB 31.0 MB/s eta 0:00:00
Downloading nvidia_cudnn_cu12-9.1.0.70-py3-none-manylinux2014_x86_64.whl (664.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 664.8/664.8 MB 2.6 MB/s eta 0:00:00
Downloading nvidia_cufft_cu12-11.2.1.3-py3-none-manylinux2014_x86_64.whl (211.5 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 211.5/211.5 MB 5.5 MB/s eta 0:00:00
Downloading nvidia_curand_cu12-10.3.5.147-py3-none-manylinux2014_x86_64.whl (56.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 56.3/56.3 MB 12.0 MB/s eta 0:00:00
Downloading nvidia_cusolver_cu12-11.6.1.9-py3-none-manylinux2014_x86_64.whl (127.9 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 127.9/127.9 MB 6.6 MB/s eta 0:00:00
Downloading nvidia_cusparse_cu12-12.3.1.170-py3-none-manylinux2014_x86_64.whl (207.5 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 207.5/207.5 MB 6.2 MB/s eta 0:00:00
Downloading nvidia_nccl_cu12-2.21.5-py3-none-manylinux2014_x86_64.whl (188.7 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 188.7/188.7 MB 6.1 MB/s eta 0:00:00
Downloading nvidia_nvjitlink_cu12-12.4.127-py3-none-manylinux2014_x86_64.whl (21.1 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 21.1/21.1 MB 42.6 MB/s eta 0:00:00
Downloading ultralytics_thop-2.0.15-py3-none-any.whl (28 kB)
Installing collected packages: nvidia-nvjitlink-cu12, nvidia-nccl-cu12, nvidia-curand-cu12, nvidia-cufft
  Attempting uninstall: nvidia-nvjitlink-cu12
    Found existing installation: nvidia-nvjitlink-cu12 12.5.82
    Uninstalling nvidia-nvjitlink-cu12-12.5.82:
      Successfully uninstalled nvidia-nvjitlink-cu12-12.5.82
  Attempting uninstall: nvidia-nccl-cu12
    Found existing installation: nvidia-nccl-cu12 2.23.4
    Uninstalling nvidia-nccl-cu12-2.23.4:
      Successfully uninstalled nvidia-nccl-cu12-2.23.4
  Attempting uninstall: nvidia-curand-cu12
    Found existing installation: nvidia-curand-cu12 10.3.6.82
    Uninstalling nvidia-curand-cu12-10.3.6.82:
      Successfully uninstalled nvidia-curand-cu12-10.3.6.82
  Attempting uninstall: nvidia-cufft-cu12
    Found existing installation: nvidia-cufft-cu12 11.2.3.61
    Uninstalling nvidia-cufft-cu12-11.2.3.61:
      Successfully uninstalled nvidia-cufft-cu12-11.2.3.61
```

Carga de las imágenes recolectadas en el entorno de trabajo.



Aprobación	Descripción	**Aprobado**	**No Aprobado**
	Configuración de entorno de desarrollo	**x**	

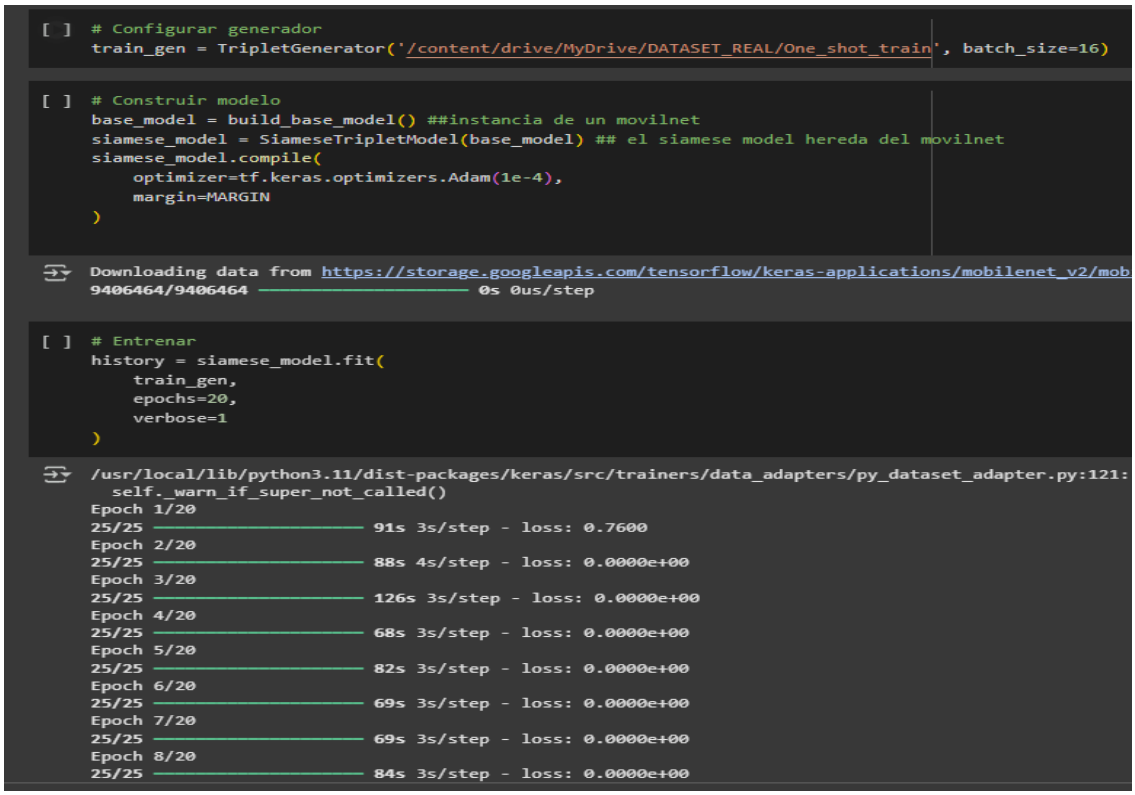
	Recolección de imágenes de tomates	**x**	
Comentario	**---**		

Fuente: Elaboración propia.

Tabla 8: Review Meeting 2

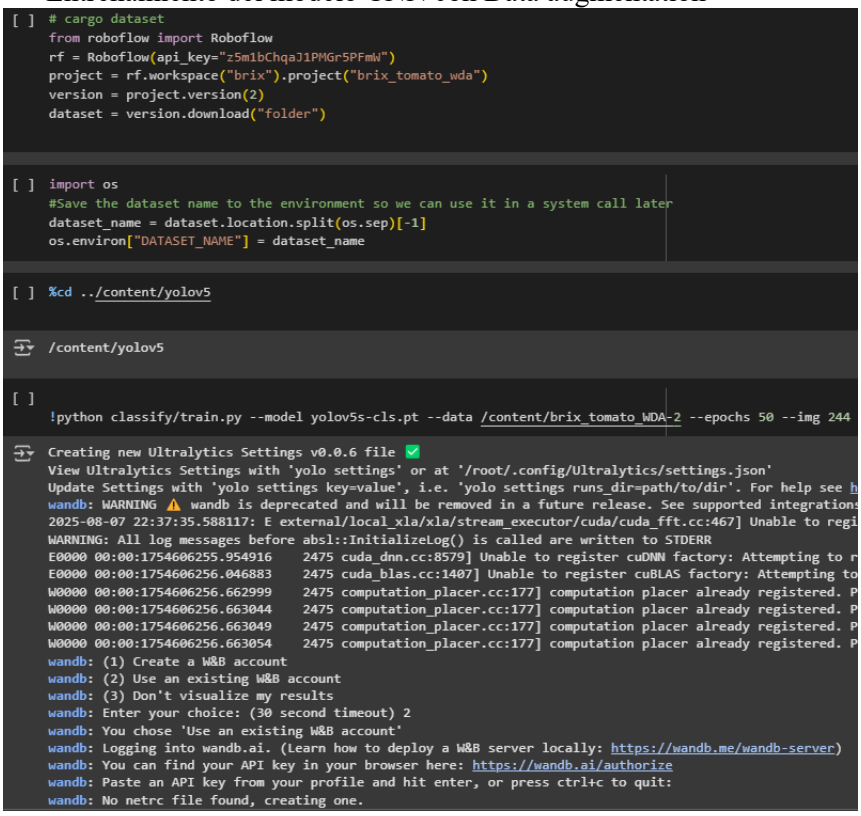
Review Meeting	2	
Requerimiento	Desarrollo de los 3 modelos de Clasificación	

Entrenamiento del modelo One-shot



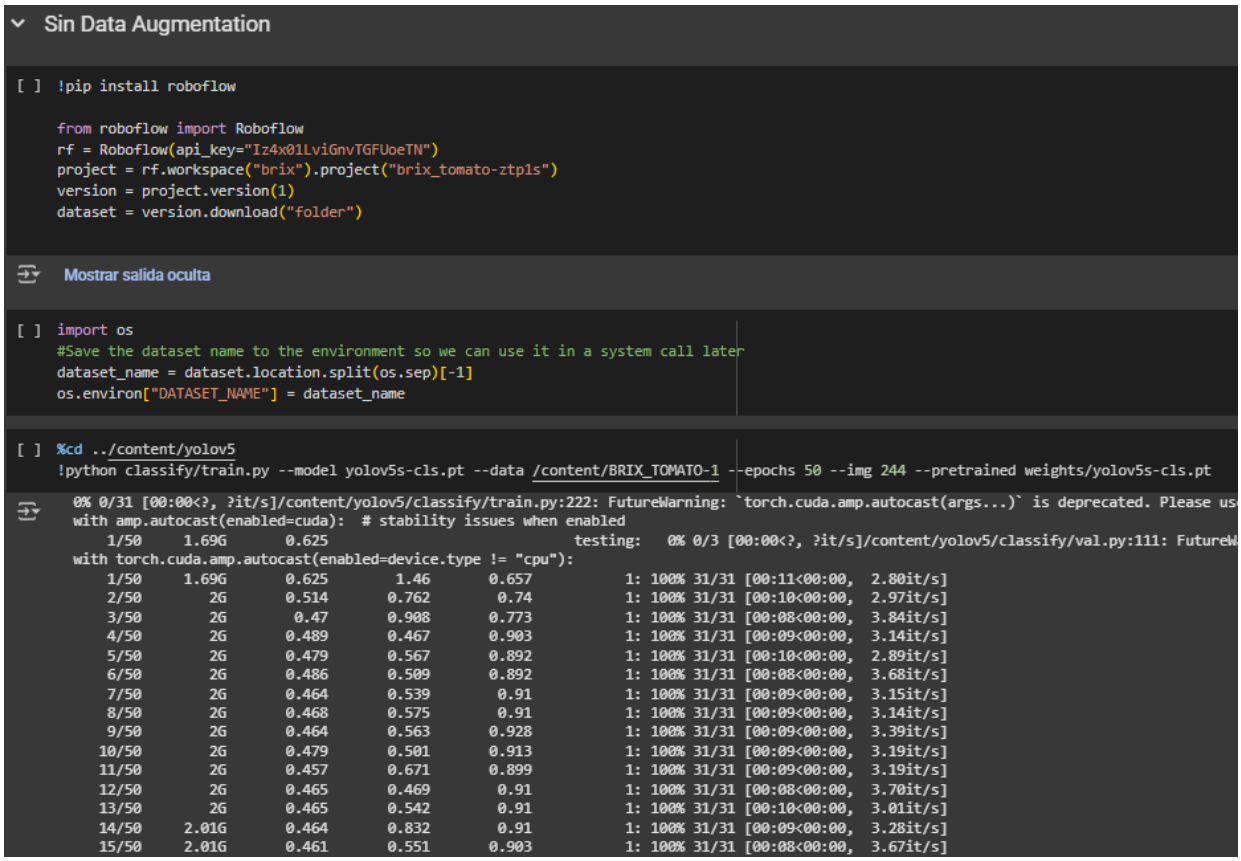
```
[ ]  # Configurar generador
     train_gen = TripletGenerator('/content/drive/MyDrive/DATASET_REAL/One_shot_train', batch_size=16)

[ ]  # Construir modelo
     base_model = build_base_model() ##instancia de un movilnet
     siamese_model = SiameseTripletModel(base_model) ## el siamese model hereda del movilnet
     siamese_model.compile(
         optimizer=tf.keras.optimizers.Adam(1e-4),
         margin=MARGIN
     )

     Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/mobilenet_v2/mob
     9406464/9406464 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step

[ ]  # Entrenar
     history = siamese_model.fit(
         train_gen,
         epochs=20,
         verbose=1
     )

     /usr/local/lib/python3.11/dist-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:121:
       self._warn_if_super_not_called()
     Epoch 1/20
     25/25 ━━━━━━━━━━━━━━━━━━━━ 91s 3s/step - loss: 0.7600
     Epoch 2/20
     25/25 ━━━━━━━━━━━━━━━━━━━━ 88s 4s/step - loss: 0.0000e+00
     Epoch 3/20
     25/25 ━━━━━━━━━━━━━━━━━━━━ 126s 3s/step - loss: 0.0000e+00
     Epoch 4/20
     25/25 ━━━━━━━━━━━━━━━━━━━━ 68s 3s/step - loss: 0.0000e+00
     Epoch 5/20
     25/25 ━━━━━━━━━━━━━━━━━━━━ 82s 3s/step - loss: 0.0000e+00
     Epoch 6/20
     25/25 ━━━━━━━━━━━━━━━━━━━━ 69s 3s/step - loss: 0.0000e+00
     Epoch 7/20
     25/25 ━━━━━━━━━━━━━━━━━━━━ 69s 3s/step - loss: 0.0000e+00
     Epoch 8/20
     25/25 ━━━━━━━━━━━━━━━━━━━━ 84s 3s/step - loss: 0.0000e+00
```

Entrenamiento del modelo CNN con Data augmentation



```
[ ] # cargo dataset
    from roboflow import Roboflow
    rf = Roboflow(api_key="z5m1bChqaJ1PMGr5PFmW")
    project = rf.workspace("brix").project("brix_tomato_wda")
    version = project.version(2)
    dataset = version.download("folder")

[ ] import os
    #Save the dataset name to the environment so we can use it in a system call later
    dataset_name = dataset.location.split(os.sep)[-1]
    os.environ["DATASET_NAME"] = dataset_name

[ ] %cd ../content/yolov5

    /content/yolov5

[ ]
    !python classify/train.py --model yolov5s-cls.pt --data /content/brix_tomato_WDA-2 --epochs 50 --img 244

    Creating new Ultralytics Settings v0.0.6 file
    View Ultralytics Settings with 'yolo settings' or at '/root/.config/Ultralytics/settings.json'
    Update Settings with 'yolo settings key=value', i.e. 'yolo settings runs_dir=path/to/dir'. For help see h
    wandb: WARNING ⚠ wandb is deprecated and will be removed in a future release. See supported integrations
    2025-08-07 22:37:35.588117: E external/local_xla/xla/stream_executor/cuda/cuda_fft.cc:467] Unable to regi
    WARNING: All log messages before absl::InitializeLog() is called are written to STDERR
    E0000 00:00:1754606255.954916    2475 cuda_dnn.cc:8579] Unable to register cuDNN factory: Attempting to r
    E0000 00:00:1754606256.046883    2475 cuda_blas.cc:1407] Unable to register cuBLAS factory: Attempting to
    W0000 00:00:1754606256.662999    2475 computation_placer.cc:177] computation placer already registered. P
    W0000 00:00:1754606256.663044    2475 computation_placer.cc:177] computation placer already registered. P
    W0000 00:00:1754606256.663049    2475 computation_placer.cc:177] computation placer already registered. P
    W0000 00:00:1754606256.663054    2475 computation_placer.cc:177] computation placer already registered. P
    wandb: (1) Create a W&B account
    wandb: (2) Use an existing W&B account
    wandb: (3) Don't visualize my results
    wandb: Enter your choice: (30 second timeout) 2
    wandb: You chose 'Use an existing W&B account'
    wandb: Logging into wandb.ai. (Learn how to deploy a W&B server locally: https://wandb.me/wandb-server)
    wandb: You can find your API key in your browser here: https://wandb.ai/authorize
    wandb: Paste an API key from your profile and hit enter, or press ctrl+c to quit:
    wandb: No netrc file found, creating one.
```

Entrenamiento de Modelo sin Data augmentation



Sin Data Augmentation

```
[ ] !pip install roboflow

    from roboflow import Roboflow
    rf = Roboflow(api_key="Iz4x01LviGnvTGFUoeTN")
    project = rf.workspace("brix").project("brix_tomato-ztp1s")
    version = project.version(1)
    dataset = version.download("folder")
```

Mostrar salida oculta

```
[ ] import os
    #Save the dataset name to the environment so we can use it in a system call later
    dataset_name = dataset.location.split(os.sep)[-1]
    os.environ["DATASET_NAME"] = dataset_name
```

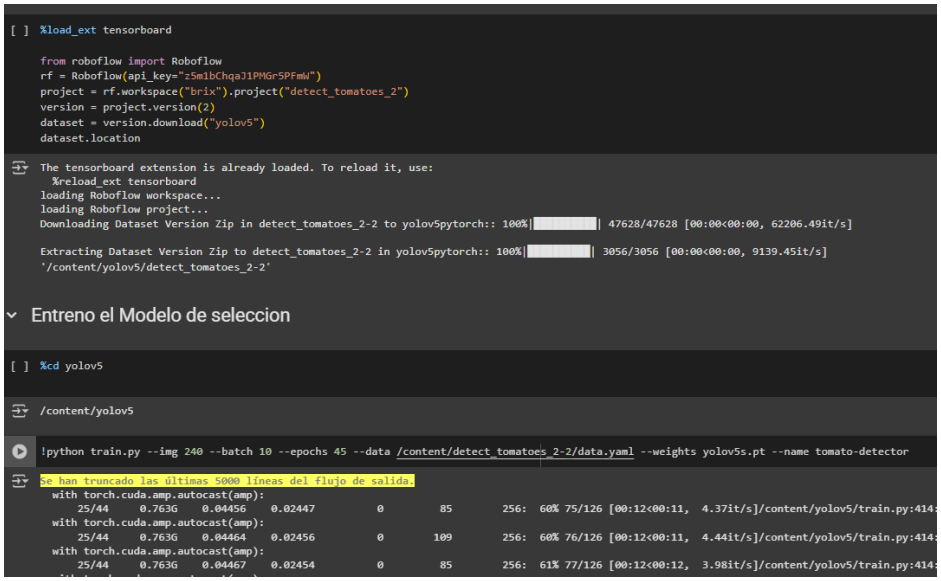
```
[ ] %cd ../content/yolov5
    !python classify/train.py --model yolov5s-cls.pt --data /content/BRIX_TOMATO-1 --epochs 50 --img 244 --pretrained weights/yolov5s-cls.pt
```

```
  0% 0/31 [00:00<?, ?it/s]/content/yolov5/classify/train.py:222: FutureWarning: `torch.cuda.amp.autocast(args...)` is deprecated. Please us
  with amp.autocast(enabled=cuda):  # stability issues when enabled
      1/50     1.69G      0.625                              testing:   0% 0/3 [00:00<?, ?it/s]/content/yolov5/classify/val.py:111: FutureW
  with torch.cuda.amp.autocast(enabled=device.type != "cpu"):
      1/50     1.69G      0.625       1.46      0.657          1: 100% 31/31 [00:11<00:00,  2.80it/s]
      2/50       2G      0.514      0.762       0.74          1: 100% 31/31 [00:10<00:00,  2.97it/s]
      3/50       2G       0.47      0.908      0.773          1: 100% 31/31 [00:08<00:00,  3.84it/s]
      4/50       2G      0.489      0.467      0.903          1: 100% 31/31 [00:09<00:00,  3.14it/s]
      5/50       2G      0.479      0.567      0.892          1: 100% 31/31 [00:10<00:00,  2.89it/s]
      6/50       2G      0.486      0.509      0.892          1: 100% 31/31 [00:08<00:00,  3.68it/s]
      7/50       2G      0.464      0.539       0.91          1: 100% 31/31 [00:09<00:00,  3.15it/s]
      8/50       2G      0.468      0.575       0.91          1: 100% 31/31 [00:09<00:00,  3.14it/s]
      9/50       2G      0.464      0.563      0.928          1: 100% 31/31 [00:09<00:00,  3.39it/s]
     10/50       2G      0.479      0.501      0.913          1: 100% 31/31 [00:09<00:00,  3.19it/s]
     11/50       2G      0.457      0.671      0.899          1: 100% 31/31 [00:09<00:00,  3.19it/s]
     12/50       2G      0.465      0.469       0.91          1: 100% 31/31 [00:08<00:00,  3.70it/s]
     13/50       2G      0.465      0.542       0.91          1: 100% 31/31 [00:10<00:00,  3.01it/s]
     14/50     2.01G      0.464      0.832       0.91          1: 100% 31/31 [00:09<00:00,  3.28it/s]
     15/50     2.01G      0.461      0.551      0.903          1: 100% 31/31 [00:08<00:00,  3.67it/s]
```

	Descripción	**Aprobado**	**No Aprobado**
Aprobación	Desarrollo del Modelo OneShot	**x**	
	Desarrollo del Modelo sin Data Augmentatio	**x**	
	Desarrollo del Modelo con Data Augmentatio	**x**	
Comentario	**---**		

Fuente: Elaboración propia.

Tabla 9: Review Meeting 3

<table>
<tr><td>Review Meeting</td><td colspan="2">3</td><td></td></tr>
<tr><td>Requerimiento</td><td colspan="2">Desarrollo del modelo de selección</td><td></td></tr>
<tr><td colspan="3">Desarrollo del modelo de selección
 
</td><td></td></tr>
<tr><td></td><td>Descripción</td><td>Aprobado</td><td>No Aprobado</td></tr>
<tr><td>Aprobación</td><td>Desarrollo del modelo de selección</td><td>x</td><td></td></tr>
<tr><td>Comentario</td><td colspan="2">---</td><td></td></tr>
</table>

Fuente: Elaboración propia.

Tabla 10: Review Meeting 4

Review Meeting	4	
Requerimiento	Etiquetado y Guardado de Tomates	

Etiquetado de cada tomate en cada foto

```
print(f"Se encontraron {len(image_files)} imágenes en '{real_dataset_path}'")
all_detections = []
# Itera a través de cada archivo de imagen
for image_file in image_files:
    image_path = os.path.join(real_dataset_path, image_file)
    print(f"\nProcesando imagen: {image_file}")

    # Carga la imagen
    img = cv2.imread(image_path)
    new_dimensions = (640, 480)

    # Redimensionar la imagen
    # cv2.INTER_AREA es bueno para reducir el tamaño de la imagen
    resized_img = cv2.resize(img, new_dimensions, interpolation=cv2.INTER_AREA)
    resized_img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    if img is None:
        print(f"Advertencia: No se pudo cargar la imagen {image_file}. Saltando.")
        continue
    # Paso 3: Realizar la detección de objetos
    results = detection_model(resized_img_rgb)
    detections = results.pandas().xyxy[0][results.pandas().xyxy[0]['confidence'] > 0.65]
    detections['image_path'] = image_path
    detections['ripeness'] = pd.NA
    detections['index_img'] = pd.NA
    detections['shape']  = [list(img.shape[0:2])] * len(detections)


    print(f"Se detectaron {len(detections)} objetos en la imagen.")

    # Paso 4: Recortar los tomates individuales
    cropped_tomatoes = []
    for index, row in detections.iterrows():
        x1, y1, x2, y2 = int(row['xmin']), int(row['ymin']), int(row['xmax']), int(row['ymax'])

        x1 = max(0, x1)
        y1 = max(0, y1)
        x2 = min(resized_img_rgb.shape[1], x2)
        y2 = min(resized_img_rgb.shape[0], y2)

        cropped_img = resized_img_rgb[y1:y2, x1:x2]
        cropped_tomatoes.append(cropped_img)

    print(f"Se recortaron {len(cropped_tomatoes)} tomates.")
  # Paso 5: Clasificar los tomates recortados
    print("Clasificando tomates recortados...")
    classified_results = []
    for i, cropped_tomato in enumerate(cropped_tomatoes):
        # Preprocesar la imagen recortada para el modelo de clasificación
        processed_tomato_r = cv2.resize(cropped_tomato, (224, 224))
        #processed_tomato_rgb = cv2.cvtColor(processed_tomato_r, cv2.COLOR_BGR2RGB)
        img_tensor = torch.from_numpy(processed_tomato_r).permute(2, 0, 1).unsqueeze(0).float() / 255.0  # Normalizar si es necesario
        processed_tomato = img_tensor.to(classification_device) # Mover al dispositivo

        # Realizar la predicción con el modelo de clasificación
        with torch.no_grad():
            outputs = classification_model(processed_tomato)
            cat_t = torch.argmax(outputs).item()
        detections.loc[i,'ripeness'] = cat_t
        detections.loc[i,'index_img'] = i
    all_detections.append(detections)

total_detections = pd.concat(all_detections,axis=0)
print("Clasificación completada.")
```

Dataframe con los datos recolectados

	xmin	ymin	xmax	ymax	confidence	...	name	image_path	ripeness	index_img	shape
0	1214.213135	2007.612549	1718.482910	2389.535156	0.904477	...	tomato	/content/drive/MyDrive/DATASET_REAL/test_T/rip...	2	0	[2393, 3024]
1	296.562378	1398.179810	745.339478	1795.436523	0.884920	...	tomato	/content/drive/MyDrive/DATASET_REAL/test_T/rip...	2	1	[2393, 3024]
2	2547.638184	1094.130859	3011.664795	1465.792236	0.881841	...	tomato	/content/drive/MyDrive/DATASET_REAL/test_T/rip...	2	2	[2393, 3024]
3	1077.215088	1426.894897	1529.780029	1816.491699	0.880871	...	tomato	/content/drive/MyDrive/DATASET_REAL/test_T/rip...	2	3	[2393, 3024]
4	2450.061523	634.953979	2922.225830	1045.477661	0.872232	...	tomato	/content/drive/MyDrive/DATASET_REAL/test_T/rip...	2	4	[2393, 3024]
...	...	...	...	...	...	...	...	...	...	...	...
20	288.498260	340.301300	427.333252	474.961029	0.771378	...	tomato	/content/drive/MyDrive/DATASET_REAL/test_T/unr...	1	20	[864, 1152]
21	98.937607	576.929138	290.563049	673.577209	0.752812	...	tomato	/content/drive/MyDrive/DATASET_REAL/test_T/unr...	1	21	[864, 1152]
22	649.087585	150.809875	779.069824	288.562988	0.747423	...	tomato	/content/drive/MyDrive/DATASET_REAL/test_T/unr...	1	22	[864, 1152]
23	418.353973	107.198257	543.275330	230.646744	0.680759	...	tomato	/content/drive/MyDrive/DATASET_REAL/test_T/unr...	1	23	[864, 1152]
24	646.563599	241.023438	783.794617	399.518616	0.657681	...	tomato	/content/drive/MyDrive/DATASET_REAL/test_T/unr...	1	24	[864, 1152]

181 rows × 11 columns

Guardado de cada tomate detectado y etiquetado

```
[ ]     Dibuja los bounding boxes y la clasificación de madurez en las imágenes originales
        y guarda las imágenes resultantes.

        Args:
            df (pd.DataFrame): DataFrame con los resultados de detección y clasificación,
                               incluyendo 'xmin_original', 'ymin_original', 'xmax_original',
                               'ymax_original', 'ripeness', 'image_path'.
            output_dir (str): Directorio donde se guardarán las imágenes de salida.
        """
        if not os.path.exists(output_dir):
            os.makedirs(output_dir)

        # Mapeo de índice de madurez a etiqueta (ajusta según tu modelo de clasificación)
        ripeness_labels = {
            0: 'Green',
            1: 'Half-Ripened',
            2: 'Fully-Ripened'
        }

        # Itera sobre cada imagen única en el DataFrame
        for image_path in df['image_path'].unique():
            # Carga la imagen original
            img_original = cv2.imread(image_path)

            if img_original is None:
                print(f"Advertencia: No se pudo cargar la imagen {image_path}. Saltando.")
                continue

            # Filtra las detecciones para la imagen actual
            detections_for_image = df[df['image_path'] == image_path]

            # Dibuja los bounding boxes y etiquetas
            for index, row in detections_for_image.iterrows():
                x1, y1, x2, y2 = int(row['xmin']), int(row['ymin']), int(row['xmax']), int(row['ymax'])
                ripeness_index = row['ripeness']
                img_index = row['index_img']
                label = ripeness_labels.get(ripeness_index, 'Unknown') # Obtiene la etiqueta o 'Unknown' si el índice no está en el mapeo

                # Define el color del bounding box basado en la madurez (puedes personalizar esto)
                if ripeness_index == 0: # Green
                    color = (0, 255, 0)
                elif ripeness_index == 1: # Half-Ripened
                    color = (0, 255, 255) # Yellow
                elif ripeness_index == 2: # Fully-Ripened
                    color = (0, 0, 255) # Red
                else:
                    color = (255, 0, 0) # Blue for unknown

                # Dibuja el rectángulo
                cv2.rectangle(img_original, (x1, y1), (x2, y2), color, 2)

                # Dibuja el texto con la etiqueta
                text = f"{label}: {img_index:.2f}"
                font = cv2.FONT_HERSHEY_SIMPLEX
                font_scale = 0.8
                thickness = 2
                text_size, _ = cv2.getTextSize(label, font, font_scale, thickness)
                text_x = x1
                text_y = y1 - 10 if y1 - 10 > 10 else y1 + 10 # Posiciona el texto encima o debajo del bbox

                cv2.putText(img_original, text, (text_x, text_y), font, font_scale, color, thickness, cv2.LINE_AA)

            # Guarda la imagen resultante
            output_path = os.path.join(output_dir, os.path.basename(image_path))
            cv2.imwrite(output_path, img_original)
            print(f"Imagen guardada: {output_path}")
```

Imagen etiquetada

	Descripción	**Aprobado**	**No Aprobado**
Aprobación	Etiquetado de cada tomate en cada foto	**x**	
	Guardado de cada tomate detectado y etiquetado	**x**	
Comentario	**---**		

Fuente: Elaboración propia.

Tabla 11: Review Meeting 5

Review Meeting	5	
Requerimiento	Generar reporte de	Preparación de entorno de desarrollo

	métricas con las imágenes test	

Script que calcula las métricas

```python
from sklearn.metrics import f1_score, precision_score, recall_score, accuracy_score
import pandas as pd
import numpy as np

# Assuming 'total_detections' DataFrame with 'true_ripeness' and 'ripeness' columns exists.

# Group by image path and calculate metrics for each image (fold)
metrics_by_image = {}
for image_path, group in total_detections.groupby('image_path'):
    # Ensure true_labels and predicted_labels are integers
    true_labels = group['true_ripeness'].astype(int)
    predicted_labels = group['ripeness'].astype(int)

    # Calculate metrics
    accuracy = accuracy_score(true_labels, predicted_labels)
    precision = precision_score(true_labels, predicted_labels, average='weighted', zero_division=0) # Use weighted average for multi-class
    recall = recall_score(true_labels, predicted_labels, average='weighted', zero_division=0) # Use weighted average for multi-class
    f1 = f1_score(true_labels, predicted_labels, average='weighted', zero_division=0) # Use weighted average for multi-class

    metrics_by_image[image_path] = {
        'accuracy': accuracy,
        'precision': precision,
        'recall': recall,
        'f1_score': f1
    }

# a dataframe para mejor vizualisacion
metrics_df = pd.DataFrame.from_dict(metrics_by_image, orient='index')

print("Metrics per Image (Fold):")
display(metrics_df)


overall_accuracy = metrics_df['accuracy'].mean()
print(f"\nOverall Mean Accuracy across images: {overall_accuracy:.4f}")

overall_precision = metrics_df['precision'].mean()
overall_recall = metrics_df['recall'].mean()
overall_f1 = metrics_df['f1_score'].mean()
```

Aprobación	Descripción	**Aprobado**	**No Aprobado**
	Script que calcula las metricas por modelo	**x**	
Comentario	**---**		

Fuente: Elaboración propia.

La recopilación de la información se llevó a cabo luego de que todos los modelos y partes necesarias dentro de la estructura de código fue desarrollada, cabe recalcar que la base

de los modelos clasificadores es el modelo selector. Para recolectar las métricas de los modelos se llevó acabo los siguientes pasos:

1. Se cargaron los modelos a utilizar (3 de clasificación, 1 de selección)
2. El modelo selector separa cada imagen grande y pequeñas imágenes de resolución estándar.
3. Estas imágenes son ingeridas por los modelos clasificadores.
4. Los datos se almacenan en un dataframe.
5. Se usa el dataframe para dibujar sobre las imágenes originales y guardarlas.
6. Se añade la columna de las clases reales de cada tomate.
7. Se calculan las métricas para cada cluster (imagen).

El entrenamiento de los respectivos modelos se llevó a cabo previo a la evaluación de los mismos para con el conjunto de imágenes recolectadas, para ello se utilizó un dataset de autoría propia obtenido de seleccionar y combinar los disponibles en los diversos repositorios de datos el cual fue almacenado en RoboFlow para mayor facilidad de conexión con el entorno de desarrollo, así mismo las diferentes técnicas de data augmentation aplicadas al dataset fueron las siguientes:

1. Rotation: Between -15° and +15°
2. Shear: ±10° Horizontal, ±10° Vertical
3. Blur: Up to 2.5px
4. Noise: Up to 0.73% of pixels

Finalmente, se procedió a realizar cálculos matemáticos para poder obtener los valores de las distintas dimensiones ya mencionadas en objetivos.

$$Precision = \frac{VP}{VP + FP}$$

Ecuación 1: Fórmula para determinar la dimensión Precisión

En la que:

- **VP (Verdaderos Positivos):** Número de casos correctamente clasificados como positivos.
- **FP (Falsos Positivos):** Número de casos incorrectamente clasificados como positivos.

$$Recall = \frac{VP}{VP + FN}$$

Ecuación 2: Fórmula para determinar la dimensión Recall

En la que:

- **FN (Falsos Negativos):** Número de casos incorrectamente clasificados como negativos.

$$Accuracy = \frac{VP + VN}{VP + VN + FP + FN}$$

Ecuación 3: Fórmula para determinar la dimensión Accuracy

En la que:

- **VN (Verdaderos Negativos):** Número de casos correctamente clasificados como negativos.

$$F1 - Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Ecuación 4: Fórmula para determinar la dimensión F1-Score

2.5. Aspectos éticos

En el desarrollo de esta investigación comparativa de modelos de aprendizaje automático, se han adoptado principios éticos fundamentales para garantizar la integridad científica, la transparencia y el respeto a los derechos de propiedad intelectual. Estas consideraciones son esenciales para mantener la credibilidad académica y social del estudio.

1. **Origen y uso ético de datos**:

   Los conjuntos de imágenes utilizados provienen exclusivamente de repositorios públicos con licencias *Creative Commons* que permiten uso académico. Se verificó la ausencia de

elementos sensibles (rostros humanos, ubicaciones identificables) en todas las imágenes para garantizar privacidad.

2. **Transparencia en resultados**:

   Los resultados de la comparación entre modelos (CNN base, data augmentation, one-shot) se presentan sin manipulación. Todas las métricas (precisión, recall, F1-Score) reflejan exactamente los experimentos realizados, incluyendo limitaciones observadas durante las pruebas.

3. **Atribución rigurosa de fuentes**:

   Todas las herramientas, códigos y datasets externos se citaron siguiendo estándares APA . Se reconocen contribuciones previas en clasificación de imágenes agrícolas y se utilizaron verificadores de plagio (*Turnitin*) para garantizar originalidad en el análisis.

CAPÍTULO III: RESULTADOS

En el presente capitulo se mostrará los rendimientos de cada uno de los modelos evaluados para cada una de las métricas de interés, alcanzados a partir de la inferencia de cada uno de los modelos sobre el conjunto de imágenes de tomates recolectados de la agroindustria peruana, 2025

Las métricas de exactitud (accuracy)", "precisión (precision)", "sensibilidad (recall)" y "puntaje F1 (F1-score) se calcularon utilizando el parámetro average='macro' de la librería scikit-learn. Esto significa que cada clase (unripe, half ripe y ripe) contribuye de manera equitativa al promedio, sin importar el número de muestras por clase. De esta manera se evita el sesgo que podría introducir un desbalance en el dataset.

Con el fin de evaluar la robustez de los resultados, se calcularon intervalos de confianza al 95% para cada métrica mediante remuestreo bootstrap. La Tabla 12 muestra el valor promedio y el rango de variación esperado para cada modelo.

Tabla 12: Mediana e intervalos de confianza por métrica.

Modelo	**accuracy**	**precision**	**recall**	**f1_score**
CNN Base	0.630 [0.564, 0.696]	0.753[0.694, 0.804]	0.684 [0.631, 0.735]	0.618 [0.548, 0.689]
Data Augmented	0.924 [0.884, 0.961]	0.903 [0.852, 0.945]	0.915 [0.865, 0.958]	0.907 [0.859, 0.950]
One Shoot	0.553 [0.486, 0.624]	0.678 [0.620, 0.734]	0.592 [0.527, 0.654]	0.551 [0.480, 0.613]

Fuente: Elaboración propia

3.1 Análisis de la dimensión Exactitud(accuracy)

Tabla 13: Resultados de cada batch en la dimensión Exactitud

	accuracy_DA	accuracy_NDA	accuracy_OS
0	0.862069	0.758621	0.793103
1	0.709677	0.612903	0.354839
2	0.969697	0.424242	0.727273
3	1.000000	0.147059	0.588235
4	1.000000	1.000000	0.320000
5	1.000000	1.000000	0.482759
Media	0.923574	0.657138	0.544368

Fuente: Elaboración propia

Sobre estos resultados se plantearon las siguientes hipótesis:

H_0: La hipótesis nula de la prueba de Friedman establece que los tres modelos de clasificación no difieren en su desempeño. Es decir, las distribuciones (o medianas) de las exactitudes obtenidas por cada modelo son iguales y cualquier variación observada se debe al azar.

H_i: La hipótesis de investigación afirma que al menos uno de los modelos presenta un rendimiento distinto. En otras palabras, no todos los modelos tienen la misma exactitud y existe al menos una diferencia significativa entre ellos.

Figura 5

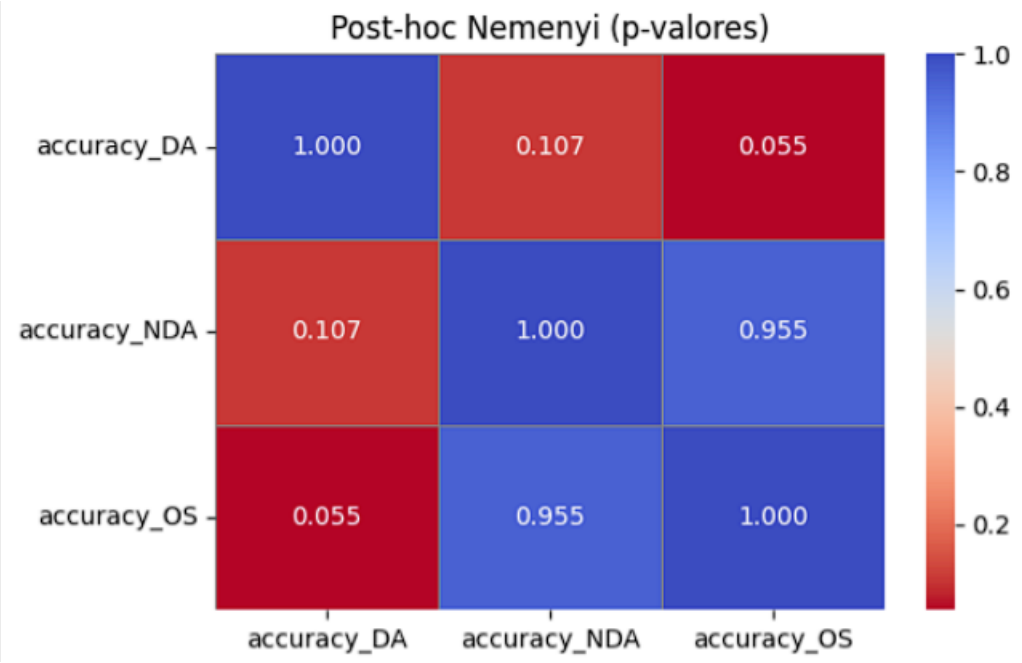
Prueba no paramétrica de Friedman para la dimensión exactitud(accuracy)

```
Metrica: accuracy
Estadístico Chi-cuadrado de Friedman: 6.909090909090904
p-valor: 0.03160166548535303
Hay diferencias significativas entre los métodos (rechazamos H0)
Kendall's W (manual): 0.5278
Interpretación: Acuerdo fuerte
```

Nota. Prueba Friedman para el accuracy en Python con la librería Scipy. Elaboración propia.

Figura 6

Heatmap para la dimensión exactitud(accuracy) en la prueba de Nemenyi



Nota. Elaboración propia.

Tras el estudio, se obtuvo el estadístico chi-cuadrado = 6.909 y un p-value = 0.0316. Dado que este valor es menor que el nivel de significancia $\alpha = 0.05$, se rechaza la hipótesis nula (H_0) y se acepta la hipótesis de investigación, concluyendo que existen diferencias significativas entre los modelos evaluados. Adicionalmente, se calculó el coeficiente de concordancia de Kendall ($W = 0.53$), el cual indica un nivel de acuerdo moderado en los rankings generados por los modelos, reforzando la validez de los resultados obtenidos. A raíz de ello, en la prueba post-hoc de Nemenyi se construyó la matriz de p-values cruzada para todos los modelos, evidenciando las comparaciones pareadas y señalando dónde se presentan diferencias estadísticamente significativas en los valores de exactitud alcanzados por cada modelo.

3.2 Análisis de la dimensión precisión

Tabla 14: Resultados de cada batch en la dimensión precisión

	precisión_DA	precisión_NDA	precisión_OS
0	1.00000	0.959248	0.859195

1	0.80117	0.457478	0.246882
2	1.00000	1.000000	1.000000
3	1.00000	1.000000	1.000000
4	1.00000	1.000000	1.000000
5	1.00000	1.000000	1.000000
Media	0.966862	0.902788	0.851013

Fuente: Elaboración propia

Sobre estos resultados se plantearon las siguientes hipótesis:

H_0: La hipótesis nula de la prueba de Friedman establece que los tres modelos de clasificación no difieren en su desempeño. Es decir, las distribuciones (o medianas) de las precisiones obtenidas por cada modelo son iguales y cualquier variación observada se debe al azar.

H_i: La hipótesis investigación afirma que al menos uno de los modelos presenta un rendimiento distinto. En otras palabras, no todos los modelos tienen la misma precisión y existe al menos una diferencia significativa entre ellos.

Figura 7

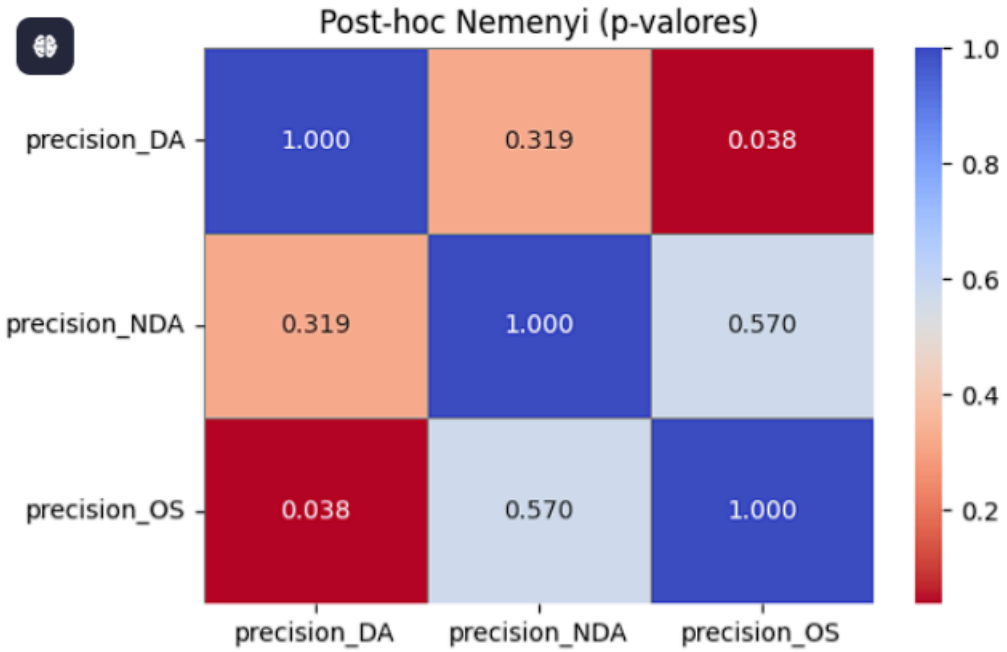
Prueba no paramétrica de Friedman para la dimensión precisión

```
Metrica: precision
Estadístico Chi-cuadrado de Friedman: 7.299999999999994
p-valor: 0.025991128778755427
Hay diferencias significativas entre los métodos (rechazamos H0)
Kendall's W (manual): 0.5069
Interpretación: Acuerdo fuerte
```

Nota. Prueba Friedman para el precision en Python con la librería Scipy. Elaboración propia.

Figura 8

Heatmap para la dimensión precisión en la prueba de Nemenyi



Nota. Elaboración propia.

Tras el estudio, se obtuvo el estadístico chi-cuadrado = 7.29 y un p-value = 0.026. Dado que este valor es menor que el nivel de significancia $\alpha = 0.05$, se rechaza la hipótesis nula (H_0) y se acepta la hipótesis de investigación, concluyendo que existen diferencias significativas entre los modelos evaluados. Adicionalmente, se calculó el coeficiente de concordancia de Kendall ($W = 0.51$), el cual indica un nivel de acuerdo moderado en los rankings generados por los modelos, reforzando la validez de los resultados obtenidos. A raíz de ello, en la prueba post-hoc de Nemenyi se construyó la matriz de p-values cruzada para todos los modelos, evidenciando las comparaciones pareadas y señalando dónde se presentan diferencias estadísticamente significativas en los valores de exactitud alcanzados por cada modelo.

3.3 Análisis de la dimensión sensibilidad (recall)

Tabla 15:Resultados de cada batch en la dimensión sensibilidad

	recall_DA	recall_NDA	recall_OS
0	0.615385	0.380342	0.294872
1	0.683333	0.577778	0.316667

2	0.484848	0.141414	0.363636
3	1.000000	0.073529	0.294118
4	1.000000	1.000000	0.160000
5	1.000000	1.000000	0.241379
Media	0.797261	0.528844	0.278445

Fuente: Elaboración propia

Sobre estos resultados se plantearon las siguientes hipótesis:

H_0: La hipótesis nula de la prueba de Friedman establece que los tres modelos de clasificación no difieren en su desempeño. Es decir, las distribuciones (o medianas) de las sensibilidades obtenidas por cada modelo son iguales y cualquier variación observada se debe al azar.

H_i: La hipótesis de investigación afirma que al menos uno de los modelos presenta un rendimiento distinto. En otras palabras, no todos los modelos tienen la misma sensibilidad y existe al menos una diferencia significativa entre ellos.

Figura 9

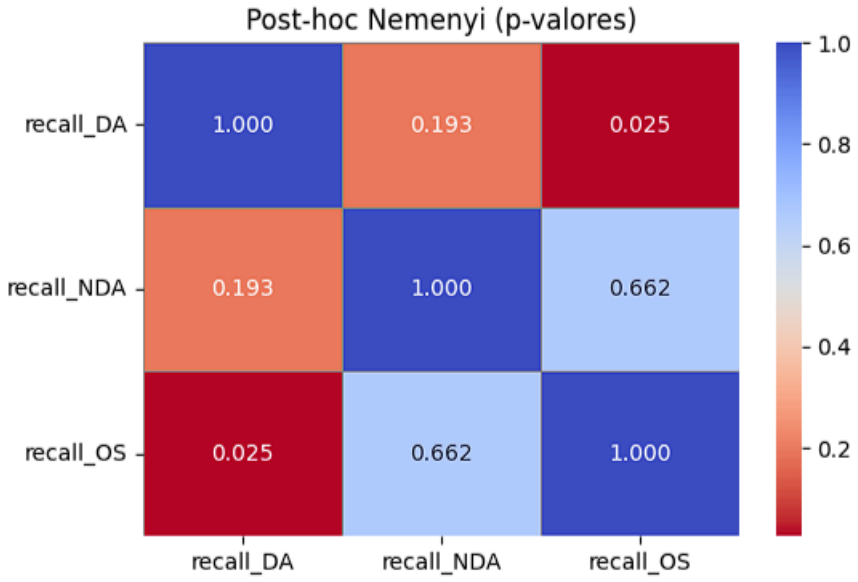
Prueba no paramétrica de Friedman para la dimensión sensibilidad(recall)

```
Metrica: recall
Estadístico Chi-cuadrado de Friedman: 7.636363636363637
p-valor: 0.021967705889435495
Hay diferencias significativas entre los métodos (rechazamos H0)
Kendall's W (manual): 0.5833
Interpretación: Acuerdo fuerte
```

Nota. Prueba Friedman para la sensibilidad en Python con la librería Scipy. Elaboración propia.

Figura 10

Heatmap para la dimensión sensibilidad(recall) en la prueba de Nemenyi



Nota. Elaboración propia.

Tras el estudio, se obtuvo el estadístico chi-cuadrado = 7.64 y un p-value = 0.022. Dado que este valor es menor que el nivel de significancia $\alpha = 0.05$, se rechaza la hipótesis nula (H_0) y se acepta la hipótesis de investigación, concluyendo que existen diferencias significativas entre los modelos evaluados. Adicionalmente, se calculó el coeficiente de concordancia de Kendall (W = 0.58), el cual indica un nivel de acuerdo moderado en los rankings generados por los modelos, reforzando la validez de los resultados obtenidos. A raíz de ello, en la prueba post-hoc de Nemenyi se construyó la matriz de p-values cruzada para todos los modelos, evidenciando las comparaciones pareadas y señalando dónde se presentan diferencias estadísticamente significativas en los valores de exactitud alcanzados por cada modelo.

3.4 Análisis de la dimensión F1-Score

Tabla 16: Resultados de cada batch en la dimensión f1-score

	f1_score_DA	f1_score_NDA	f1_score_OS
0	0.638889	0.458333	0.306667
1	0.683928	0.494048	0.245495
2	0.492308	0.198582	0.421053
3	1.000000	0.128205	0.370370
4	1.000000	1.000000	0.242424
5	1.000000	1.000000	0.325581
Media	0.802521	0.546528	0.318598

Fuente: Elaboración propia

Sobre estos resultados se plantearon las siguientes hipótesis:

H_0: La hipótesis nula de la prueba de Friedman establece que los tres modelos de clasificación no difieren en su desempeño. Es decir, las distribuciones (o medianas) de los f1-score obtenidas por cada modelo son iguales y cualquier variación observada se debe al azar.

H_i: La hipótesis de investigación afirma que al menos uno de los modelos presenta un rendimiento distinto. En otras palabras, no todos los modelos tienen el misma f1-score y existe al menos una diferencia significativa entre ellos.

Figura 11

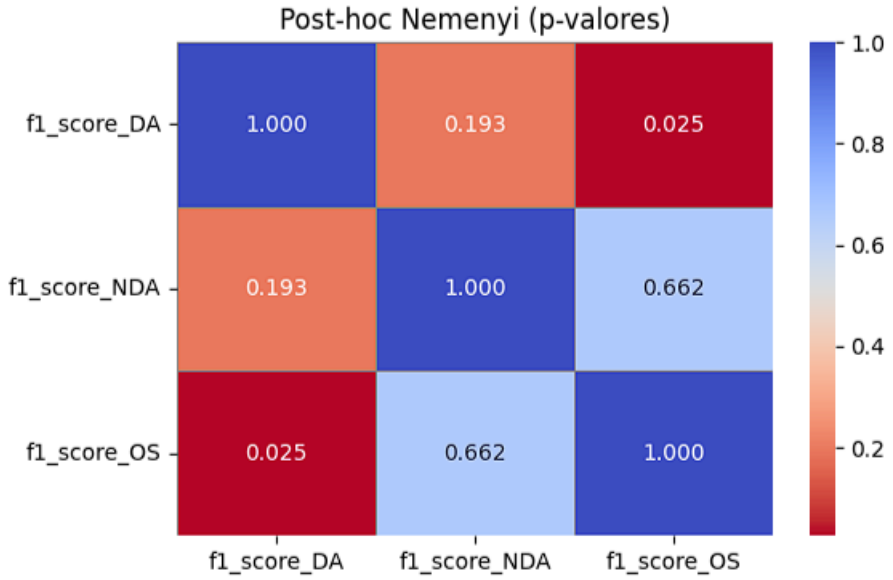
Prueba no paramétrica de Friedman para la dimensión F1-score

```
Metrica: f1_score
Estadístico Chi-cuadrado de Friedman: 7.636363636363637
p-valor: 0.021967705889435495
Hay diferencias significativas entre los métodos (rechazamos H0)
Kendall's W (manual): 0.5833
Interpretación: Acuerdo fuerte
```

Nota. Prueba Friedman para la F1-score en Python con la librería Scipy. Elaboración propia.

Figura 12

Heatmap para la dimensión F1-score en la prueba de Nemenyi



Nota. Elaboración propia.

Tras el estudio, se obtuvo el estadístico chi-cuadrado = 7.64 y un p-value = 0.022. Dado que este valor es menor que el nivel de significancia $\alpha = 0.05$, se rechaza la hipótesis nula (H_0) y se acepta la hipótesis de investigación, concluyendo que existen diferencias significativas entre los modelos evaluados. Adicionalmente, se calculó el coeficiente de concordancia de Kendall (W = 0.58), el cual indica un nivel de acuerdo moderado en los rankings generados por los modelos, reforzando la validez de los resultados obtenidos. A raíz de ello, en la prueba post-

hoc de Nemenyi se construyó la matriz de p-values cruzada para todos los modelos, evidenciando las comparaciones pareadas y señalando dónde se presentan diferencias estadísticamente significativas en los valores de exactitud alcanzados por cada modelo.

3.5 Análisis de la dimensión Tiempo

Tabla 17: Resultados de cada batch en la dimensión Tiempo en segundos por cada tomate

	time_DA	time_NDA	time_OS
0	0.040907	0.039538	0.130882
1	0.039252	0.036197	0.125526
2	0.039006	0.036118	0.255855
3	0.040171	0.036021	0.143553
4	0.040388	0.051156	0.159268
5	0.039983	0.052117	0.173538
Media	0.039951	0.041858	0.164770

Fuente: Elaboración propia

Sobre estos resultados se plantearon las siguientes hipótesis:

H_0: La hipótesis nula de la prueba de Friedman establece que los tres modelos de clasificación no difieren en su desempeño. Es decir, las distribuciones (o medianas) de los tiempos obtenidas por cada modelo son iguales y cualquier variación observada se debe al azar.

H_i: La hipótesis de investigación afirma que al menos uno de los modelos presenta un rendimiento distinto. En otras palabras, no todos los modelos tienen el mismo tiempo, al menos una diferencia significativa entre ellos.

Figura 13

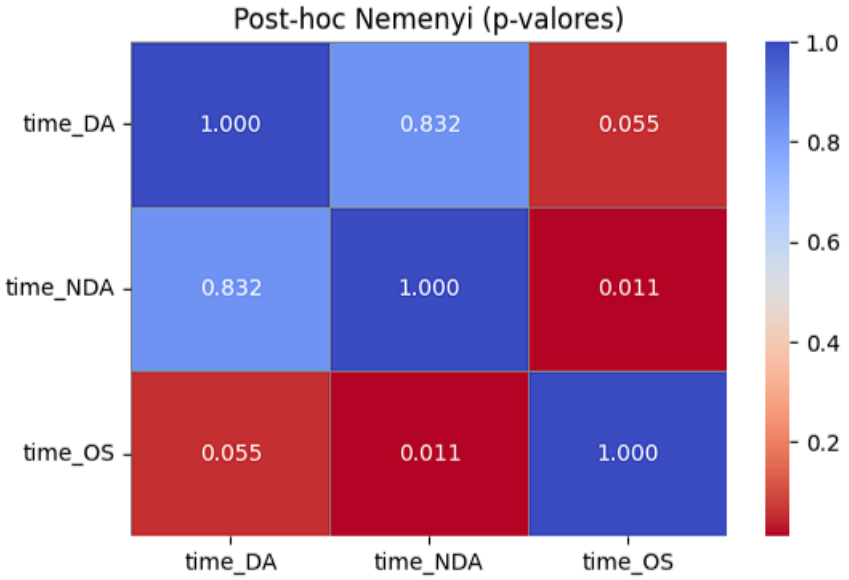
Prueba no paramétrica de Friedman para la dimensión tiempo

```
Metrica: time
Estadístico Chi-cuadrado de Friedman: 9.333333333333329
p-valor: 0.00940356255149523
Hay diferencias significativas entre los métodos (rechazamos H0)
Kendall's W (manual): 0.7778
Interpretación: Acuerdo muy fuerte
```

Nota. Prueba Friedman para el tiempo en Python con la librería Scipy. Elaboración propia.

Figura 14

Heatmap para la dimensión tiempo en la prueba de Nemenyi



Nota. Elaboración propia.

Tras el estudio, se obtuvo el estadístico chi-cuadrado = 9.33 y un p-value = 0.0094. Dado que este valor es menor que el nivel de significancia $\alpha = 0.05$, se rechaza la hipótesis nula (H_0) y se acepta la hipótesis de investigación, concluyendo que existen diferencias significativas entre los modelos evaluados. Adicionalmente, se calculó el coeficiente de concordancia de Kendall ($W = 0.778$), el cual indica un nivel de acuerdo moderado en los rankings generados por los modelos, reforzando la validez de los resultados obtenidos. A raíz de ello, en la prueba post-hoc de Nemenyi se construyó la matriz de p-values cruzada para todos los modelos, evidenciando las comparaciones pareadas y señalando dónde se presentan diferencias estadísticamente significativas en los valores de exactitud alcanzados por cada modelo.

CAPÍTULO IV: DISCUSIÓN Y CONCLUSIONES

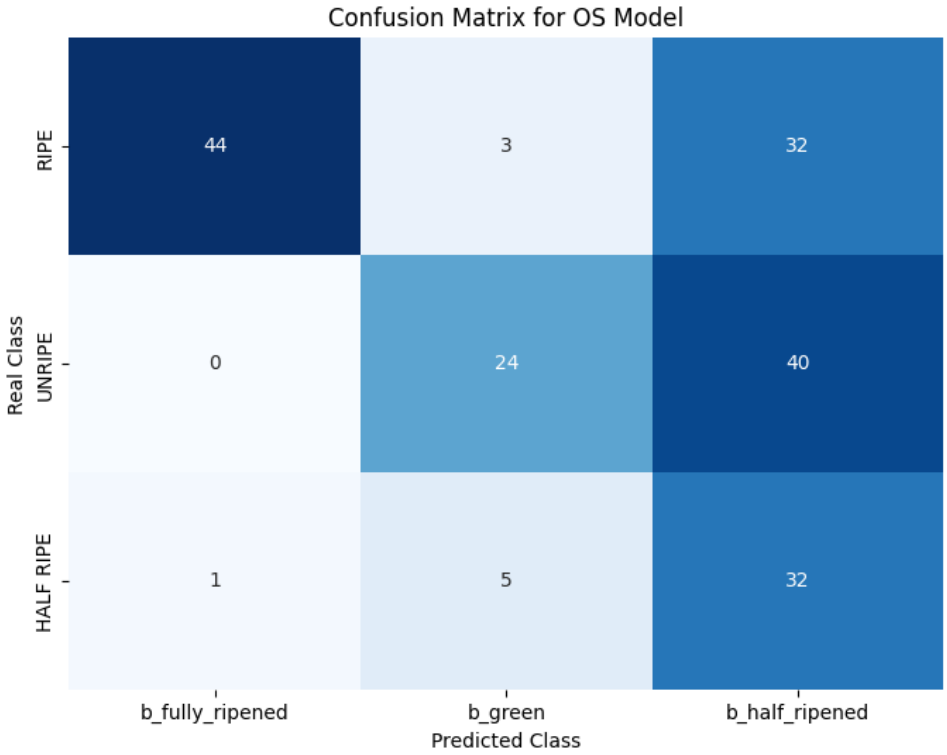
4.1 Discusión

Para empezar, se debe tener en cuenta que el presente estudio comparó como los diferentes sets de datos con los que se entrenaron los modelos rinden en entornos reales, en ese sentido los data sets se pueden intercambiar a cualquier fruto que respete las características de clasificación para modelos de visión artificial (clases marcadas por cambios visuales notables en el fruto). Como se puede observar en la revisión sistemática realizada por Injante et al. (2025) titulada "Procesamiento de imágenes para la detección de la madurez del fruto: una revisión sistemática" en donde analiza 15 estudios de entre 2019-2024, donde evidencia que las manzanas son las más estudiadas seguidas de las naranjas. Así mismo como, en el estudio de Sa et al. (2016) DeepFruits: A fruit detection system using deep neural networks. En donde se obtiene un F1-Score de 0.807 a 0.838 en la detección de madurez en papayas, estos resultados concuerdan con lo mostrados en el presente estudio en donde la media de F1-Score fue de 0.803 para el modelo con DA. Con lo cual de querer aplicarse el estudio en un tipo diferente de fruto bastaría con consolidar un dataset con las características necesarias como el usado en el presente estudio.

Como segundo punto no podemos pasar por alto las evidentes diferencias entre los modelos, las cuales son mas evidentes al comparar con DA con el modelo One-shot véase la Figura 9, los cuales trataremos de explicar a continuación a partir de las matrices de confusión de cada modelo.

Figura 15

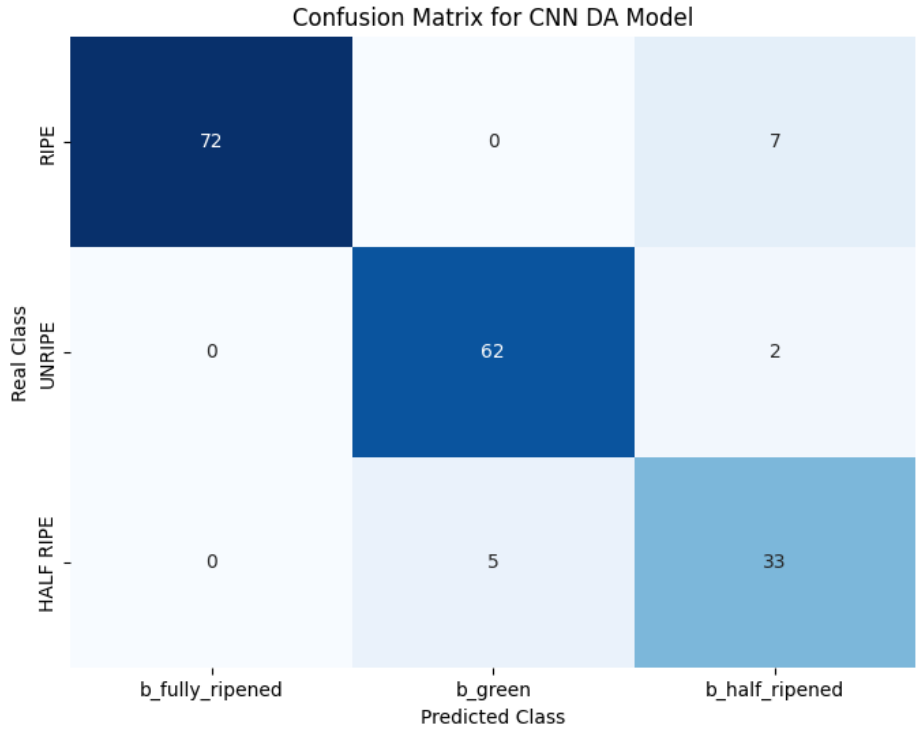
Matriz de confusión para el modelo One-Shot



Nota. Elaboración propia.

Figura 16

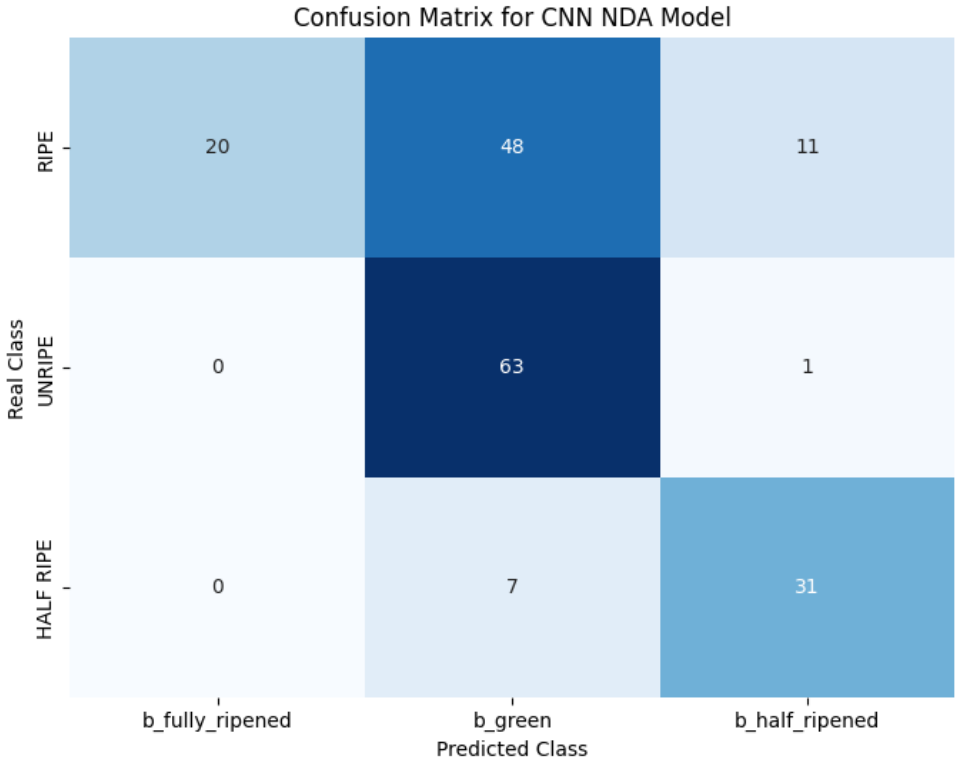
Matriz de confusión para el modelo CNN con Data Augmatation



Nota. Elaboración propia.

Figura 17

Matriz de confusión para el modelo CNN sin Data Augmatation



Nota. Elaboración propia.

Podemos observar como la cantidad de etiquetas erróneas es consistente en la esquina superior derecha, es decir los modelos presentan problemas para clasificar clases intermedias de manera consistente como lo menciona Puyal et al. (2015) ilustra la complejidad de clasificar múltiples clases intermedias en aplicaciones médicas, donde distinguir entre cinco categorías diferentes presenta desafíos similares a los encontrados en la detección de estados intermedios de madurez.

Con lo cual hace sentido que en el Modelo One-Shot el problema con la clase intermedia resulte más marcado dada la naturaleza de embedding usada para predecir las clases.

Por otro lado, si hablamos del modelo sin DA es víctima de lo que describe Buda et al. (2017) Un problema fundamental en CNN es el desbalance de clases dado que el desequilibrio de clases tiene efectos perjudiciales en el rendimiento de clasificación. Dado que una parte del Data Augmentation consiste en balancear las clases, es totalmente esperable que un modelo con las clases desbalanceadas tenga este tipo de rendimiento.

En cuanto a cada una de las métricas. Las pruebas de Friedman negaron la hipótesis nula en cada una de ellas, así mismo las diferencias entre las métricas medida por el p-value en el test post-hoc de Nemenyi fueron desde 0.001 como mínimo hasta 0.83 como máximo, como se describe a continuación:

Para la dimensión de exactitud (accuracy), la diferencia más significativa —con el p-valor más bajo— se observó al comparar el modelo con aumento de datos frente al modelo one-shot: 92,3 % frente a 54 % (p = 0,055), lo que supone una mejora absoluta de 38,3 %. Frente a Chuquimarca Jiménez et al. (2025), que reportaron un 91,7 % de exactitud en la madurez de bananas utilizando datasets totalmente sintéticos, nuestro enfoque basado en técnicas de aumento de datos reales supera dicha referencia en 0,5 puntos porcentuales. Aunque la mejora en exactitud es moderada, la naturaleza del aumento difiere: ellos emplean datos completamente sintéticos, mientras que nosotros aplicamos transformaciones sobre muestras reales.

Para la dimensión de precisión, la diferencia más significativa —con el p-valor más bajo— se observó entre el modelo con aumento de datos y el modelo one-shot: 96,7 % frente a 85 % (p = 0,038), lo que supone una mejora absoluta de 11,7 %. Zahrotul Ilmi Wijayanti (2024) reportó un 97 % de precisión en la clasificación binaria de la madurez de fresas. No obstante, su estudio se limita a una tarea binaria, mientras que la nuestra emplea un esquema multiclase, lo que eleva la complejidad de la clasificación

Para la dimensión de F1-score, la comparación entre el modelo con aumento de datos y el modelo one-shot —con el p-valor más bajo— arrojó un 80 % frente a 32 % (p = 0,025), lo que supone una mejora absoluta de 48 puntos porcentuales. Yashu et al. (2024) reportaron un F1-score del 87 % en la clasificación binaria de la madurez de bananas; sin embargo, su estudio combina Random Forest y CNN, mientras que la nuestra emplea exclusivamente una CNN multiclase convencional, lo que podría explicar su ligera ventaja en rendimiento.

Por lo tanto, se afirma que los estudios citados con anterioridad proporcionaron una base sólida que respaldó los resultados alcanzados de la presente investigación y reforzaron la idea de que el desarrollo de un modelo CNN para la detección del estado de madurez del tomate en comparación con modelos existentes mediante métricas de precisión y F1-Score, para la agroindustria del Norte peruano, 2025 presenta mejoras significativas.

En términos de implicancias practicas este trabajo es extrapolable a diversos frutos como ya se describió línea arriba, en consecuencia, tiene un impacto importante en el marco de conocimiento del sector. la implementación de un modelo CNN con Data Augmentation para la detección del estado de madurez del tomate tiene un mejor desempeño que los modelos con los que se comparó, esto debido a que el data augmentation no solo aumenta el conjunto de entrenamiento y crea un modelo mas robusto, sino que también balancea las clases y evitar un overfitting a causa de un desbalance de clases, característica bastante común en modelo de multiclase como este. Tener esto en cuenta, no solo para entrenar modelos CNN como comúnmente se hace nos abre un horizonte en el cual la importancia del dataset cobra mayor relevancia y nos permite explorar modelos con mayor potencial para datasets integros.

Es importante señalar que aspectos como los costos de implementación, la integración del sistema en planta y los requerimientos específicos de hardware escapan al alcance de la presente investigación. Este trabajo se centra en la comparación del desempeño de distintos modelos de Deep Learning y en los conjuntos de datos empleados para su entrenamiento, con el propósito de contribuir al conocimiento en un área aún poco explorada dentro de la clasificación automatizada de frutos. En cuanto a los aspectos técnicos de infraestructura o integración industrial, existe abundante literatura especializada que aborda dichos temas con mayor profundidad y precisión, por lo que se consideró más pertinente mantener el enfoque en la evaluación comparativa de los modelos propuestos.

4.2 Conclusiones

Al término de este estudio, se alcanzaron las siguientes conclusiones en correspondencia con los objetivos planteados.

En primer lugar, se diseñó y desarrolló un modelo CNN optimizado para la clasificación del estado de madurez del tomate, adaptado a las condiciones particulares de la agroindustria peruana. El diseño del modelo integró estrategias de aumento de datos, normalización y regularización, que permitieron mejorar la capacidad de generalización ante la variabilidad lumínica y las diferencias morfológicas de los frutos. Estas configuraciones resultaron determinantes para obtener un modelo robusto y funcional en entornos de producción reales.

Asimismo, el modelo CNN fue implementado exitosamente utilizando un conjunto de imágenes de tomates en tres estados de madurez. Se aplicaron técnicas de segmentación, redimensionamiento y aumento de datos que incrementaron la representatividad del conjunto de entrenamiento. Estas acciones contribuyeron a reducir el sobreajuste y estabilizar el

aprendizaje, lo cual se evidenció en la convergencia de los errores de entrenamiento y validación a lo largo de las iteraciones.

En cuanto a la evaluación del rendimiento, la validación cruzada de seis particiones y las pruebas externas demostraron un desempeño promedio de 0.9668 en precisión y 0.8025 en F1-score, superando ampliamente a los modelos sin aumento de datos (0.9028 y 0.5465) y al modelo One-Shot (0.8510 y 0.3186). Estos resultados confirman la efectividad de las estrategias de optimización y evidencian la capacidad del modelo para clasificar con alta fiabilidad los distintos estados de madurez del tomate.

Finalmente, el análisis comparativo de las métricas de rendimiento, realizado mediante la prueba de Friedman y el test post-hoc de Nemenyi, reveló diferencias estadísticamente significativas entre los modelos evaluados ($p < 0.021$). Los resultados demostraron que el modelo CNN con aumento de datos supera de manera consistente a las alternativas sin aumento y One-Shot, evidenciando su superioridad estadística y robustez ante la variabilidad de las muestras. Asimismo, el desempeño obtenido se encuentra dentro o por encima de los valores reportados en la literatura, validando la eficacia del enfoque propuesto y su contribución al avance del conocimiento en la aplicación de redes neuronales convolucionales dentro de la agroindustria

4.3 Recomendaciones

Considerando la relevancia de este estudio y su relación con los hallazgos alcanzados. Esta indagación abre la posibilidad de continuar investigando acerca del desarrollo de modelos CNN para la detección del estado de madurez del tomate, comparándolos con otros modelos siendo que de manera general siempre obedecen las métricas de precisión y F1-score, todo esto

dentro del marco de la agroindustria del norte peruano. Por ello se plantean algunas sugerencias a tener en cuenta para posteriores investigaciones:

- Ampliar el entrenamiento de modelos basados en embeddings.

  Se sugiere evaluar el desempeño del modelo One-shot embedding utilizando conjuntos de datos de tamaño y diversidad comparables a los empleados en las CNN. De esta manera, podría explorarse su verdadero potencial, dado que su rendimiento podría incrementarse significativamente con mayor cantidad de ejemplos representativos por clase.Analizar el adicionamiento de una Adaptive Equalization (Ecualización adaptativa) para un modelo mas robusto ante los cambios de iluminación.

- Incorporar técnicas de ecualización adaptativa.

  Se recomienda incluir procesos de Adaptive Histogram Equalization en la etapa de preprocesamiento de imágenes, con el fin de mejorar la diferenciación entre estados intermedios de madurez. Esta técnica podría incrementar la robustez del modelo ante variaciones de iluminación, propias de los entornos agrícolas reales.

- Implementar el modelo dentro de un sistema de aprendizaje continuo y automatizado.

  Se recomienda integrar el modelo CNN desarrollado en un entorno de producción que permita la automatización del proceso de clasificación, captura de nuevas imágenes y almacenamiento de predicciones. Este sistema debe incorporar mecanismos de monitoring y feedback loops, tal como lo sugiere el enfoque de Machine Learning in Production, para detectar degradaciones en el rendimiento y actualizar el modelo mediante reentrenamiento periódico. De esta forma, se garantiza la sostenibilidad,

escalabilidad y adaptabilidad del modelo ante cambios en las condiciones reales de operación. Asimismo, su estructura puede adaptarse para la clasificación de otros frutos que expresen su madurez mediante características visuales observables.

Referencias

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., & Zheng, X. (2016). TensorFlow: A system for large-scale machine learning. Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation (OSDI '16), 265-283. USENIX Association. https://arxiv.org/abs/1605.08695

Aghaebrahimian, A., & Cieliebak, M. (2019). Hyperparameter Tuning for Deep Learning in Natural Language Processing. *4th Swiss Text Analytics Conference (SwissText 2019), Winterthur, June 18-19 2019*. 10.21256/zhaw-18993

Aherwadi, N., Mittal, U., Singla, J., Jhanjhi, N. Z., Yassine, A., & Hossain, M. S. (2022*).* Prediction of Fruit Maturity, Quality, and Its Life Using Deep Learning Algorithms. *Electronics (Switzerland)*, 11(24), 1-13. https://doi.org/10.3390/electronics11244100

Arévalo Zenteno, Ma. D., Ruiz Castilla, J. S., & Ayala de la Vega, J. (2021). Clasificación de frutos del durazno en maduros, no maduros y dañados hacia la cosecha automatizada. *CIBA Revista Iberoamericana de Las Ciencias Biológicas y Agropecuarias, 10*(19), 39–53. https://doi.org/10.23913/ciba.v10i19.107

Atencio, R., & Cruz, E. (2022). Situación actual de la aplicación y potenciales usos de la visión artificial en la entomología agrícola en Panamá. *I+D Tecnológico*, *18*(2), 124-135. https://doi.org/10.33412/idt.v18.2.3741

Bala, M.M., Bai, K.J., Kattubadi, S., Pulipati, G., & Balguri, A. (2024). Deep Learning Transformations in Content-Based Image Retrieval: Exploring the Visual Geometry Group (VGG16) Model. *2024 IEEE 4th International Conference on ICT in Business Industry & Government (ICTBIG)*, 1-4. 10.59544/HYIS2213/ICRCCT24P62

Barkley, F. A. (1949). Un Esbozo de Clasificación de los Organismos. *Revista Facultad Nacional de Agronomía Medellín, 10*(34), 83–103. https://revistas.unal.edu.co/index.php/refame/article/view/29794

Berlanga-Silvente, V., & Rubio-Hurtado, M.-J. (2012). Classificació de proves no paramètriques. Com aplicar-les en SPSS. REIRE Revista d'Innovació I Recerca En Educació, 5(2), 101–113. https://doi.org/10.1344/reire2012.5.2528

Buda, M., Maki, A., & Mazurowski, M. A. (2018). A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks, 106,* 249–259 https://doi.org/10.1016/j.neunet.2018.07.011

Bhattacharjee, Rohan & Ghosh, Debjyoti & Mazumder, Abhirup. (2021). A review on hyper-parameter optimisation by deep learning experiments. *Journal of Mathematical Sciences & Computational Mathematics*, 2, 532-541. 10.15864/jmscm.2407

Campos Ferreira, U.E., & González Camacho, J.M. (2021). CLASIFICADOR DE RED NEURONAL CONVOLUCIONAL PARA IDENTIFICAR ENFERMEDADES DEL FRUTO DE AGUACATE (Persea americana Mill.) A PARTIR DE IMÁGENES DIGITALES. *Agrociencia*. 10.47163/agrociencia.v55i8.2662

Cárdenas, L., Pipino, A., Donadio, M. T., & Picota, P. (2024). Modelo de un sistema de alerta basado en visión artificial para la prevención de aglomeraciones en el transporte público. *Visión Antataura, 8*(2), 93–107. https://doi.org/10.48204/j.vian.v8n2.a6571

Castillo, J., Domínguez, J., García, M. M., Marín, G., Olórtegui, D., & Ynocente, C. (2022). Determinación de metales pesados en tomate (Solanum lycopersicum) y su riesgo para la salud humana. *Ciencia e Investigación, 25*(1), 17–22. https://doi.org/10.15381/ci.v25i1.23470

Cervera, E. (2020). GPU-Accelerated Vision for Robots: Improving System Throughput Using OpenCV and CUDA. *IEEE Robotics & Automation Magazine, 27*, 151-158. 10.1109/MRA.2020.2977601

Choi, K., Lee, G., Han, Y. J., & Bunn, J. M. (1995). Tomato maturity evaluation using color image analysis. *Transactions of the ASAE, 38*(1), 171–176. https://doi.org/10.13031/2013.27827

Chuang, Y., Zhang, S., Zhao, X.: Deep learning-based panoptic segmentation: Recent advances and perspectives. IET Image Process. 17, 2807–2828 (2023). https://doi.org/10.1049/ipr2.12853

Chucos Baquerizo, N., & Vega Ventocilla, E. J. (2022). Evaluación de algoritmos de machine learning en la clasificación de imágenes satelitales multiespectrales, caso: Amazonia Peruana. *Ciencia Latina Revista Científica Multidisciplinar*, *6*(1), 4946-4963. https://doi.org/10.37811/cl_rcm.v6i1.1843

Culjak, I., Abram, D., Pribanić, T., Džapo, H., & Cifrek, M. (2012, mayo 21-25). A brief introduction to OpenCV. *En 2012 Proceedings of the 35th International Convention MIPRO* (pp. 1725–1730). IEEE. https://www.researchgate.net/publication/261424692_A_brief_introduction_to_OpenCV

Chuquimarca, L., Vintimilla, B., & Velastin, S. (2025). Banana ripeness level classification using a simple cnn model trained with real and synthetic datasets https://arxiv.org/pdf/2504.08568

Da Silva, I. T., & Freixo, A. A. (2020). Ensino de botânica e classificação biológica em uma escola família agrícola: Diálogo de saberes no campo. *Ensaio: Pesquisa em Educação em Ciências*, *22*, e21781. https://doi.org/10.1590/21172020210122

Du, L. (2019). Shallow Deep Learning: Embedding Verbatim K-Means in Deep Neural Networks. *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, 194-197. 10.1109/ICTAI.2019.00035

Figueroa, A.R. (2011). Inferencia abductiva basada en modelos. Una relación entre lógica y cognición. *Critica-revista Hispanoamericana De Filosofía, 43*, 3-29.

https://doi.org/10.22201/iifs.18704905e.2011.786

Gama Campillo, L. M. (2024). ¿Por qué clasificar? *Taxonomía folk como ejemplo de los inicios. Kuxulkab', 30*(66), 37–41. https://doi.org/10.19136/kuxulkab.a30n66.5981

Garcés Cadena, A. A., Menéndez Granizo, O. A., Córdova, E. P., & Prado Romo, A. J. (2023). Clasificación de calidad de manzana para monitoreo de cosechabilidad utilizando visión por computador y algoritmos de aprendizaje profundo. *Ingeniare. Revista chilena de ingeniería*, *31*(15), 1-18. https://dx.doi.org/10.4067/s0718-33052023000100215

Guillén, G. (2019). Digital image processing with Python and OpenCV. En *Sensor Projects with Raspberry Pi: Internet of Things and Digital Image Processing* (pp. 97–140). Apress. https://doi.org/10.1007/978-1-4842-5299-4_5

Guo, Y., Zhang, J., Su, P., Hou, G. H., & Deng, F. Y. (2020). The study of locating diseased leaves based on RPN in complex environment. *Journal of Physics: Conference Series*, *1651*(1), 012089. 10.1088/1742-6596/1651/1/012089

Huang, S., Kang, Z., Xu, Z. (2020). Deep K-Means: A Simple and Effective Method for Data Clustering. In: Zhang, H., Zhang, Z., Wu, Z., Hao, T. (eds) Neural Computing for Advanced Applications. NCAA 2020. Communications in Computer and Information Science, vol 1265. Springer, Singapore. https://doi.org/10.1007/978-981-15-7670-6_23

Huang, S.-C., & Le, T.-H. (2021). Front matter. *En Principles and labs for deep learning*. *Elsevier*. https://doi.org/10.1016/B978-0-323-90198-7.09993-6

Hu, C. H., Shi, Z. F., Wei, H. L., Hu, X. D., Xie, Y. N., & Li, P. P. (2022). Automatic detection of pecan fruits based on Faster RCNN with FPN in orchard. *International Journal of Agricultural and Biological Engineering*, *15*(6), 189–196. https://doi.org/10.25165/j.ijabe.20221506.7241

Hu, H. M., Kaizu, Y., Zhang, H. D., Xu, Y. W., Imou, K., Li, M., Huang, J. J., & Dai, S. (2022). Recognition and localization of strawberries from 3D binocular cameras for a

strawberry picking robot using coupled YOLO/Mask R-CNN. *International Journal of Agricultural and Biological Engineering, 15*(6), 175–179. https://doi.org/10.25165/j.ijabe.20221506.7306

Injante, R., Rios-Trigoso, G., Ramírez-Shupingahua, S., & Tejada Shupingahua, K. (2025). Procesamiento de imágenes para la detección de la madurez del fruto: una revisión sistemática. Revista Científica de Ingeniería, Diseño y Arquitectura Contemporánea. 10.37711/idac.2025.2.1.3

Kanna, K., Kumaraperumal, R., Pazhanivelan, P., Jagadeeswaran, R., & Prabu, P.C. (2024). YOLO deep learning algorithm for object detection in agriculture: a review. *Journal of Agricultural Engineering*, 15(1641), 1-15. 10.4081/jae.2024.1641

Komaravalli, B., Ravi, K.J., & Srinidhi, P. (2024). Optimized Low Weight CNN Concatenation Using Deep Curve Estimation for Low Light Image Enhancement. *2024 First International Conference on Pioneering Developments in Computer Science & Digital Technologies (IC2SDT)*, 1-6. 10.1109/INOCON57975.2023.10101064

Lempitsky, V.S., Kohli, P., Rother, C., & Sharp, T. (2009). Image segmentation with a bounding box prior. *2009 IEEE 12th International Conference on Computer Vision*, 277-284. 10.1109/ICCV.2009.5459262

Li, C., Guo, C., & Loy, C.C. (2021). Learning to Enhance Low-Light Image via Zero-Reference Deep Curve Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 44*, 4225-4238. https://arxiv.org/abs/2103.00860

Lin, T., Dollár, P., Girshick, R.B., He, K., Hariharan, B., & Belongie, S.J. (2016). Feature Pyramid Networks for Object Detection. *2017 IEEE Conference on Computer Vision and Pattern Recognition* (CVPR), 936-944. 10.1109/CVPR.2017.106

Londoño, M. (2008). Cosecha y manejo poscosecha. Recuperado de:

http://hdl.handle.net/20.500.12324/13465.

López-Barrios, J. D., Escobedo Cabello, J. A., Gómez-Espinosa, A., & Montoya-Cavero, L. E. (2023). Green Sweet Pepper Fruit and Peduncle Detection Using Mask R-CNN in Greenhouses. *Applied Sciences, 13*(10), 1-18. https://doi.org/10.3390/app13106296

Mei, X., Xie, F., & Jiang, Z. (2016). Uneven illumination removal based on fully convolutional network for dermoscopy images. *2016 13th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, 243-247. 10.1088/1742-6596/2700/1/012003

Mota-Delfin, C., Juárez-González, C., & Olguín-Rojas, J. C. (2018). Clasificación de manzanas utilizando visión artificial y redes neuronales artificiales. *Ingeniería Y Región*, *20*(1), 52–57. https://doi.org/10.25054/22161325.1917

Montaño-Roldan, V. L., Borda Luna, B. E., Ortiz Porras, J. E., & Guerrero Moncada, C. R. (2023). Incidence of pesticides on the quality of Solanum Lycopersicum tomato in the supply of retail markets in the city of Lima, Peru. Sapienza: International Journal of Interdisciplinary Studies, 4(3), e23037. https://doi.org/10.51798/sijis.v4i3.692

Moreira, G., Magalhães, S. A., Pinho, T., dos Santos, F. N., & Cunha, M. (2022). Benchmark of Deep Learning and a Proposed HSV Colour Space Models for the Detection and Classification of Greenhouse Tomato. *Agronomy, 12*(2), 1-23. https://doi.org/10.3390/agronomy12020356

Nana Hermana, A., Rosmala, D., & Gustiana Husada, M. (2021, 8 de junio). Transfer Learning for Classification of Fruit Ripeness Using VGG16. ICCMB '21: Proceedings of the 2021 4th International Conference on Computers in Management and Business, 139–146. https://doi.org/10.1145/3450588.3450943

Numata, K., & Itagaki, K. (2011). Estudio sobre el caso de la producción creciente del

tomate en los desiertos mediante el sistema agrario con poco insumo: Desafíos en la zona costera del Perú. *Ciencia y Desarrollo*, *14*(02), 27-37. http://dx.doi.org/10.21503/cyd.v14i0.1141

Núñez-Colín, Carlos. (2018). Análisis de varianza no paramétrica: un punto de vista a favor para utilizarla. Acta Agrícola y Pecuaria. 4. 69-79. https://www.researchgate.net/publication/330133050_Analisis_de_varianza_no_parametrica_un_punto_de_vista_a_favor_para_utilizarla

Osco-Mamani, E., Santana-Carbajal, O., Chaparro-Cruz, I., Ochoa-Donoso, D., & Alcazar-Alay, S. (2025). The Detection and Counting of Olive Tree Fruits Using Deep Learning Models in Tacna, Perú. AI, 6(25), 1-18. https://doi.org/10.3390/ai6020025

Puyal, J.G., Sánchez, B.R., & Avila, M.T. (2015). Detección y Clasificación de Tejidos Anómalos en Mamografías Digitales Mediante Redes Neuronales Convolucionales. https://www.semanticscholar.org/paper/Detección-y-Clasificación-de-Tejidos-Anómalos-en-Puyal-Sánchez/4d58257ac3e6e27424876c14e9c2b00d4f231eb0

Pirinen, A., & Sminchisescu, C. (2018). Deep Reinforcement Learning of Region Proposal Networks for Object Detection. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6945-6954. https://www.researchgate.net/publication/329962462_Deep_Reinforcement_Learning_of_Region_Proposal_Networks_for_Object_Detection

Puga-Muima, A. M., & Chalco-Sandoval, W. R. (2021). Manejo poscosecha de granadilla en la parroquia Yangana, cantón y provincia de Loja. *Revista Investigación Agraria*, 3(1), 7-16. https://doi.org/10.47840/ReInA.3.1.1059

Purwono, P., Ma'arif, A., Rahmaniar, W., Fathurrahman, H. I. K., Frisky, A., & ul Haq, Q. M. (2023). *Understanding of convolutional neural network (CNN): A review*. *International Journal of Robotics and Control Systems*, 2(4), 739–748 10.31763/ijrcs.v2i4.888

Sa, I., Ge, Z., Dayoub, F., Upcroft, B., Perez, T., & McCool, C. (2016). DeepFruits: A fruit detection system using deep neural networks. In *Proceedings of the Italian National Conference on Sensors*. https://doi.org/10.3390/s16081222

Santa Maria Pinedo, J. C. ., Ríos López, C. A., Rodríguez Grández, C., & García Estrella, C. W. (2021). Recognition of image patterns through an artificial vision system in MATLAB. *Revista Científica De Sistemas E Informática*, 1(2), 15–26. https://doi.org/10.51252/rcsi.v1i2.131

Siwei, H., & Liu, B. (2021). Review of Bounding Box Algorithm Based on 3D Point Cloud. *International Journal of Advanced Network, Monitoring and Controls, 6*, 18 - 23. 10.21307/ijanmc-2021-003

Stojanovic, B., Marques, O., Neskovic, A., & Puzović, S. (2016). Fingerprint ROI segmentation based on deep learning. *2016 24th Telecommunications Forum (TELFOR)*, 1-4. 10.1109/TELFOR.2016.7818799

Suguna A., Deepa S., Jyothi Swaroop K. C., Jyothsna V., Kiran B. K. (2024). Ml Driven Multimodal Skin Disease Detection and Monitoring. *International Conference on Recent Trends in Computing & Communication Technologies (ICRCCT'2K24)*. DOI:10.59544/HYIS2213/ICRCCT24P62

Torres Puente, V. M. (2019). Tiempo, clima y los fenómenos atmosféricos: desde torbellinos hasta cambio climático. *Revista Digital Universitaria, 20*(1), 1-13. https://doi.org/10.22201/codeic.16076079e.2019.v20n1.a3

Trigo Riveros, R. J., y Fernández Chávez, C. M. (2024). Percepción de compradores de tomates en mercados de la Paz respecto a contaminación por agroquímicos. *Apthapi*, *10*(3), 2780-2787. ISSN 2519-9382. https://doi.org/10.53287/dity3044yb36y

Urbina Cienfuegos, S. M., & Bravo Rivas, J. J. (2025). Artificial Intelligence: Machine

Learning for Early Detection of Pests and Diseases in Basic Crops, Nicaragua. INGENIO, 8(1), 24–34. https://doi.org/10.29166/ingenio.v8i1.7221

Velazco, J. (2012). Características del empleo agrícola en el Perú. En C. Garavito & I. Muñoz (Eds.), *Empleo y protección social* (pp. 160–211). Pontificia Universidad Católica del Perú. Fondo Editorial. https://doi.org/10.18800/978-612-4146-17-6.005

Villaseñor-Aguilar, M. J., Bravo-Sánchez, M. G., Padilla-Medina, J. A., Vázquez-Vera, J. L., Guevara-González, R. G., García-Rodríguez, F. J., & Barranco-Gutiérrez, A. I. (2020). A maturity estimation of bell pepper (Capsicum annuum L.) by artificial vision system for quality control. *Applied Sciences, 10*(15), 1-18. https://doi.org/10.3390/app10155097

Wang, W., Zheng, V. W., Yu, H., & Miao, Ch. (2019, 16 de junio). A Survey of Zero-Shot Learning: Settings, Methods, and Applications. *ACM Trans. Intell. Syst. Technol*, *10*(2), Article 13, 1-37 pages. https://doi.org/10.1145/3293318

Wijayanti, Z.I. (2024). Penerapan Teknologi CNN Dalam Proses Pendeteksi Kematangan Buah Stroberi. Uranus : Jurnal Ilmiah Teknik Elektro, Sains dan Informatika. https://doi.org/10.61132/uranus.v2i3.192

Yadav, R.K., Moriwal, R., & Sakhare, A.V. (2024). Analysis of Computational Model for Detection and Recognition of Human Activity Using Deep Learning. *2024 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*, 1-7. 10.1109/Confluence51648.2021.9377114

Yashu, Kukreja, V., Srivastava, P., & Garg, A. (2024). Fruitful Fusion: CNN-Random Forest Synergy in Banana Ripeness Detection. 2024 IEEE International Conference on Information Technology, Electronics and Intelligent Communication Systems (ICITEICS), 1-6. https://doi.org/10.1109/ICITEICS61368.2024.10625429

ANEXOS

ANEXO nro. 1. Matriz de consistencia

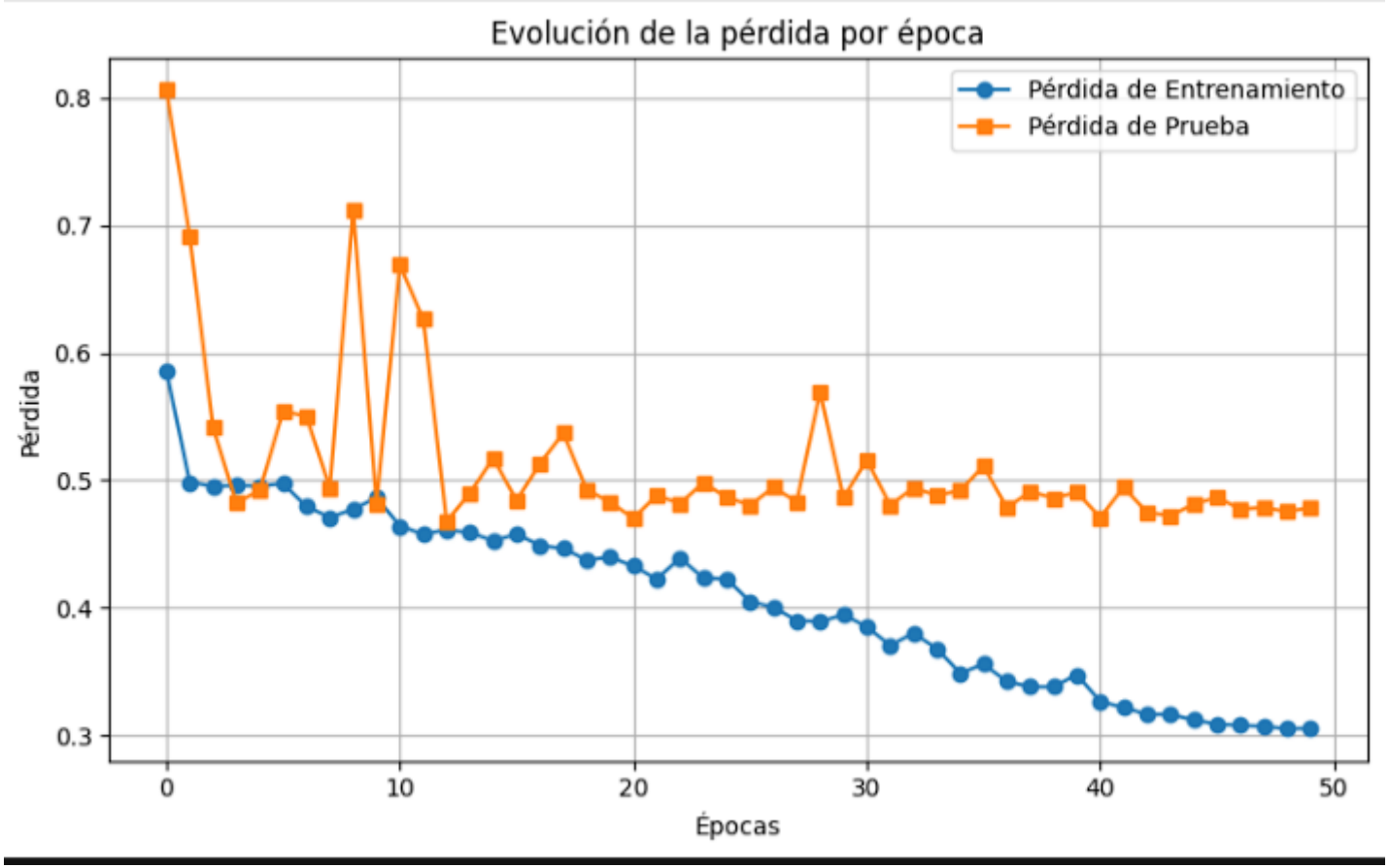
<table>
<tr><th colspan="5">DESARROLLO DE UN MODELO CNN PARA LA DETECCIÓN DEL ESTADO DE MADUREZ DEL TOMATE Y SU COMPARACIÓN CON MODELOS EXISTENTES MEDIANTE MÉTRICAS DE PRECISIÓN Y F1-SCORE, PARA LA AGROINDUSTRIA PERUANA, 2025</th></tr>
<tr><th>PROBLEMA</th><th>HIPÓTESIS</th><th>OBJETIVO GENERAL</th><th>VARIABLE INDEPENDIENTE</th><th>METODOLOGÍA</th></tr>
<tr><td rowspan="5">¿En qué medida un modelo CNN desarrollado para la detección del estado de madurez del tomate mejora las métricas de precisión y F1-score en comparación con los modelos one-shot y sin data augmentation, para</td><td rowspan="5">El modelo CNN desarrollado para la detección del estado de madurez del tomate mejora significativamente las métricas de precisión y F1-score en comparación con los modelos one-shot y sin data augmentation en la agroindustria peruana, 2025..</td><td rowspan="5">Determinar en qué medida un modelo CNN desarrollado para la detección del estado de madurez del tomate mejora las métricas de precisión y F1-score en comparación con los modelos one-shot y sin data augmentation, para su aplicación en la agroindustria peruana, 2025.</td><td rowspan="5">Procesamiento de imágenes con Red Neuronal Convolucional (CNN)</td><td>Diseño</td></tr>
<tr><td>$GX_1O\ \ GX_2O\ \ GX_3O$
Donde:
G=Conjunto de 6 folds (imágenes con ~20 tomates cada una).
X1=Modelo CNN entrenado con dataset tipo A.
X2= Modelo CNN entrenado con dataset tipo B.
X3 = Modelo one-shot basado en embeddings.
O: Medición del desempeño de cada modelo (precisión y F1-score) sobre cada fold.</td></tr>
<tr><td>Población</td></tr>
<tr><td>Todos los tomates dentro de las fotos recolectadas de pertenecientes a la agroindustria</td></tr>
</table>

su aplicación en la agroindustria peruana, 2025?				peruana.

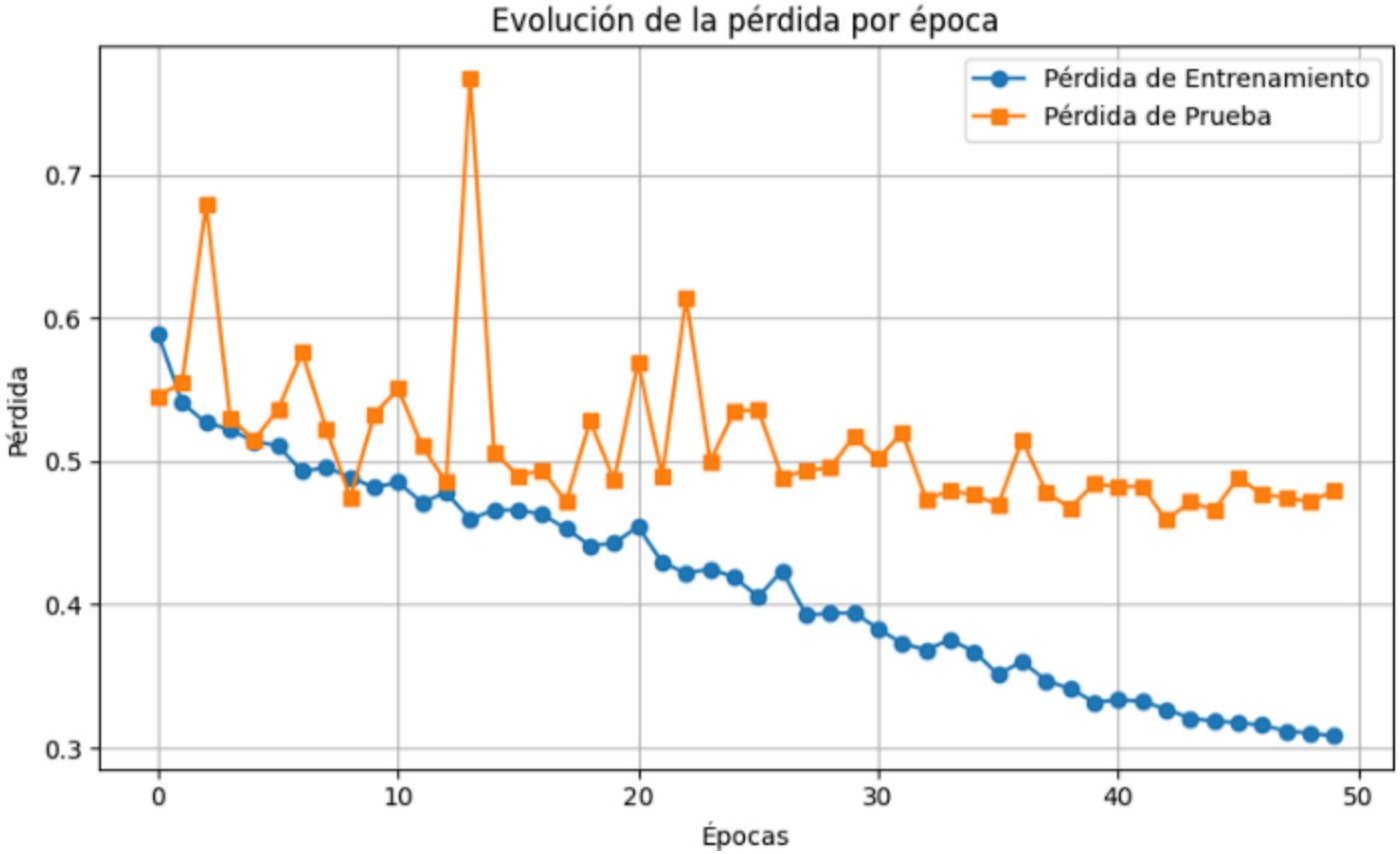
ANEXO nro. 2. Matriz de operacionalización

VARIABLE INDEPENDIENTE	**DEFINICIÓN CONCEPTUAL**	**DEFINICIÓN OPERACIONAL**	**DIMENSIONES**	**INDICADORES**
Procesamiento de imágenes con Red Neuronal Convolucional (CNN)	*Utilización de técnicas de visión por computadora para analizar y comprender imágenes capturadas por una cámara (Al-Hassani et al., 2013)*	*Un modelo que recibe imágenes de tomates y devuelve una clasificación en 3 niveles de madurez, medido con métricas de desempeño*	*Rendimiento*	*Tolerancia a fallas*
				Disponibilidad
			Confiabilidad	*Redundancia*
				Estabilidad
VARIABLE DEPENDIENTE	**DEFINICIÓN CONCEPTUAL**	**DEFINICIÓN OPERACIONAL**	**DIMENSIONES**	**INDICADORES**
Clasificación de tomates	*Selección de tomates según su apariencia y calidad (Pradeep, 2014).*	*La discriminación de tomates será medida según el indiciador 0 que indica madures, 1 que indica que esta verde, y 2 que indica que está madurando.*	*Exactitud*	*Verdaderos Positivos*
				Falsos Negativos
			Precisión	*Verdaderos Negativos*
				Falsos Positivos
			F1-Score	*Resultado Alcanzado*
				Resultado Previsto

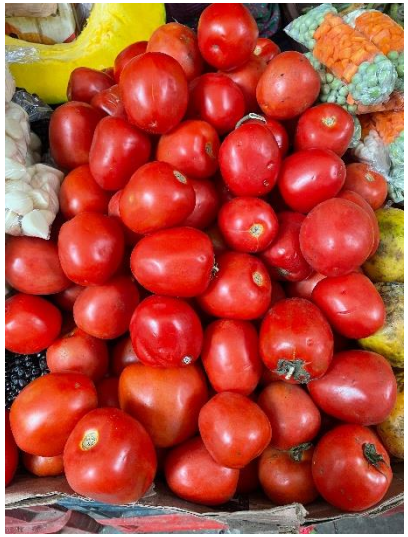
ANEXO nro. 3. Métricas de Perdida para el Modelo con DA



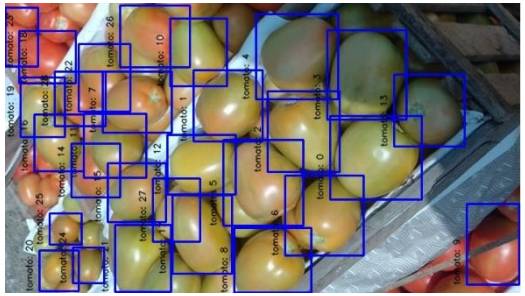
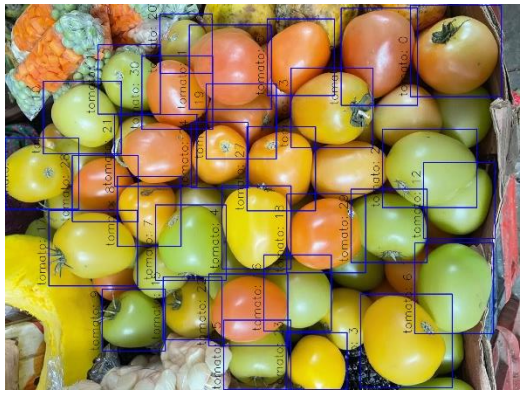
ANEXO nro. 4. Métricas de Perdida para el Modelo sin DA



ANEXO nro. 5. Imágenes por clasificar en pruebas reales.

Imagen 01 	**Imagen 02** 	**Imagen 03** 
Imagen 04 	**Imagen 05** 	**Imagen 06** 

ANEXO nro. 6. Imágenes con los tomates seleccionados

Imagen 01 	**Imagen 02** 	**Imagen 03** 
Imagen 04 	**Imagen 05** 	**Imagen 06** 

ANEXO nro. 7. Enlace al notebook

https://colab.research.google.com/drive/1Jhgh6HUqPPgrAksYIftn-c_wbpmfqLxl?usp=sharing