


Clasificador de Madurez de Frutas en Tiempo Real Mediante Transfer Learning con ResNet50

1st Carbajal Urquiaga, Jose Antonio

Escuela de Informática

Universidad Nacional de Trujillo

Trujillo, Perú

 <https://orcid.org/0000-0002-1244-4480>
jcarbajalu@unitru.edu.pe

Resumen—La clasificación manual de la madurez de frutas en regiones agrícolas como La Libertad, Perú, resulta en un proceso tedioso e ineficiente. Este trabajo propone un clasificador automático en tiempo real utilizando ResNet50 y Transfer Learning para abordar este problema. Para asegurar la robustez del sistema en condiciones no controladas, se implementó una estrategia de Data Augmentation. El modelo, entrenado con el dataset público RawRipe, se integró en una aplicación de escritorio con inferencia vía webcam, ofreciendo una solución de bajo costo para estandarizar la calidad y optimizar la cadena post-cosecha en el sector agrícola.

Index Terms—redes neuronales convolucionales, transfer learning, visión por computadora, clasificación de frutas, madurez de frutas, ResNet50.

I. INTRODUCCIÓN

La agricultura es una de las actividades económicas más importantes a nivel mundial, siendo la base de la seguridad alimentaria y el sustento de miles de millones de personas. Sin embargo, la eficiencia en la cadena de valor post-cosecha sigue siendo un gran desafío, especialmente en las etapas de clasificación y selección de productos. Los métodos manuales, aún predominantes en muchas regiones, son blanco de subjetividad y lentitud, lo que resulta en pérdidas significativas. En este contexto, la Inteligencia Artificial (IA) y la visión por computadora ofrecen buenas soluciones. Dentro de este dominio, el aprendizaje profundo (Deep Learning) se ha consolidado como la tecnología más potente, gracias a su capacidad para aprender características complejas directamente de las imágenes. El presente artículo detalla el diseño y desarrollo de un sistema automático basado en una Red Neuronal Convolucional (CNN), potenciado mediante *Transfer Learning* y *Data Augmentation*, para clasificar la madurez de frutas en tiempo real.

Este trabajo se centra en la realidad problemática de la agricultura en la región de La Libertad, Perú. Esta región es una de las principales protagonistas del dinamismo agrícola nacional, sector que acumuló un crecimiento del 4,9 % en el año 2024 (Ministerio de Desarrollo Agrario y Riego (MIDAGRI), 2025). El Padrón Nacional de Sectores Estadísticos cuantifica esta realidad, detallando miles de hectáreas dedicadas a la producción agrícola en zonas como Virú y Ascope (Ministerio de Desarrollo Agrario y Riego (MIDAGRI), 2024). La Libertad es un motor clave para el país, representando

el 13,97 % del Valor Bruto de la Producción agropecuaria a nivel nacional y experimentando un crecimiento anual del 11,3 %. Este dinamismo se refleja en cultivos frutícolas de alto valor; por ejemplo, solo en el mes de diciembre de 2024, la producción de mango en Piura y La Libertad experimentó un crecimiento notable del 464 % respecto al año anterior. Este auge productivo, si bien es positivo, ejerce una presión enorme sobre los procesos post-cosecha, los cuales, en su mayoría, siguen siendo manuales. La clasificación de frutas depende de la percepción visual y la experiencia de operarios, un método subjetivo afectado por la fatiga que conduce a inconsistencias de calidad y pérdidas económicas, limitando la competitividad de los productores liberteños.

Frente a esto, la implementación de un sistema automatizado ofrece beneficios concretos y estratégicos. La importancia de este proyecto se debe a su capacidad para reducir costos operativos al disminuir la dependencia de la mano de obra, y aumentar la eficiencia al procesar frutas a una velocidad muy superior a la humana. Fundamentalmente, elimina la subjetividad, garantizando un criterio de clasificación uniforme que mejora la calidad del producto final y fortalece su posición en el mercado. Una clasificación precisa y rápida permite, además, una gestión óptima de los lotes, minimizando las pérdidas post-cosecha por deterioro y facilitando una monitorización en tiempo real para la toma de decisiones informadas en la cadena de suministro.

La literatura científica reciente muestra el potencial de la IA para resolver este problema, abarcando desde la precisión de la clasificación hasta la robustez en entornos no controlados. Por ejemplo, Behera et al. (2021) buscaron crear un sistema no destructivo para la clasificación de madurez de papayas, motivados por la naturaleza destructiva y lenta de la inspección manual. Su enfoque comparó técnicas de machine learning clásico (utilizando descriptores de características como HOG, LBP y GLCM con clasificadores SVM y KNN) frente a siete modelos de *Transfer Learning* (incluyendo ResNet, AlexNet y VGG). Sobre un dataset de 300 imágenes de papayas en tres estados de madurez, sus hallazgos fueron concluyentes: el modelo VGG19 alcanzó una precisión del 100 % en solo 1 minuto y 52 segundos de entrenamiento, superando significativamente a un método previo del 94.7 % y estableciendo la superioridad del aprendizaje profundo. Por su parte, Aherwadi et al. (2022)

se centraron en el desafío de la generalización del modelo para la clasificación de madurez y calidad de bananas, un fruto con alta variabilidad visual. Su principal contribución fue el uso de un agresivo *Data Augmentation* para expandir su dataset de 2,100 a 18,900 imágenes, simulando rotaciones, recortes y cambios de iluminación. Al comparar una CNN propia con AlexNet, demostraron que la aumentación de datos elevó la precisión del modelo AlexNet del 81.75 % al 99.44 %, concluyendo que esta técnica es crucial para la robustez y sugiriendo su aplicación futura a otras frutas como el mango. En una línea similar de enfrentar la escasez de datos, Siricharoen et al. (2023) abordaron la clasificación de piñas en escenas complejas y desordenadas. Su innovadora solución fue una técnica de "muestreo multi-objeto" para generar un dataset sintético pero realista a partir de un número limitado de imágenes con fondo blanco. Utilizando un modelo de segmentación de instancias (Mask R-CNN con backbone ResNet-101), lograron un mAP (mean Average Precision) de 86.7 %, demostrando que es posible entrenar modelos robustos para entornos complejos partiendo de datos iniciales limitados. Finalmente, el trabajo de Moya et al. (2024) se enfocó en la detección de ajíes directamente en el campo, un escenario con alta oclusión por hojas y condiciones de luz no controladas. Para ello, crearon un nuevo dataset con imágenes realistas y utilizaron el detector de objetos YOLOv5. Su sistema logró una precisión de clasificación casi perfecta del 99.9 % para diferenciar ajíes rojos de verdes, y una precisión de detección (localización) del 84.4 %, validando la viabilidad de los detectores modernos en escenarios agrícolas no ideales.

En respuesta a esta problemática y basándose en el estado del arte, este proyecto busca desarrollar una herramienta funcional y automatizada. Por lo tanto, el objetivo general de este trabajo es desarrollar e implementar un sistema de visión por computadora, basado en el modelo de aprendizaje profundo ResNet50, para la clasificación automática y en tiempo real del estado de madurez de diversas frutas. Para alcanzarlo, se plantearon los siguientes objetivos específicos: primero, adaptar y entrenar un modelo ResNet50 pre-entrenado mediante *transfer learning* utilizando el dataset público Raw-Ripe; segundo, diseñar e implementar una estrategia de *data augmentation* avanzada para mejorar la robustez del modelo frente a variaciones de iluminación, oclusión y perspectiva; y finalmente, desarrollar una aplicación de escritorio que integre el modelo entrenado para proporcionar una herramienta de clasificación práctica y de fácil uso a través de una cámara web.

II. MATERIALES Y MÉTODOS

El desarrollo de este proyecto se basa en un conjunto de herramientas de software de código abierto, un dataset público, y una metodología de evaluación.

II-A. Dataset: RawRipe

Para el entrenamiento y la evaluación del modelo, se utilizó el dataset público **RawRipe**, presentado por Jerripothula et al. (2021). Este conjunto de datos fue seleccionado por

ser específico para la tarea de reconocimiento de madurez, proporcionando una base estandarizada para el desarrollo y la eventual comparación de resultados. Contiene imágenes de 10 tipos de frutas en dos estados: inmaduro (Raw) y maduro (Ripe). Para este proyecto, las clases se combinaron en un formato multiclase, como `Apple_Raw` y `Apple_Ripe`, para que el modelo aprenda a identificar simultáneamente el tipo de fruta y su estado. Se eligieron solo 5 tipos de fruta.

II-B. Hardware y Software

El entrenamiento y la inferencia se realizaron en un entorno local con las siguientes especificaciones:

- **Procesador:** AMD Ryzen 5 6600H con gráficos integrados.
- **Lenguaje de programación:** Python 3.
- **Framework de Deep Learning:** TensorFlow (con su API de alto nivel Keras).
- **Librerías de Visión por Computadora:** OpenCV y Pillow (PIL).
- **Librerías de Análisis:** NumPy, Scikit-learn, Matplotlib y Seaborn.

II-C. Arquitectura del Modelo: ResNet50

Se seleccionó la arquitectura **ResNet50** (He et al., 2016) por su excelente equilibrio entre profundidad (y, por tanto, capacidad de aprendizaje de características complejas) y eficiencia computacional. ResNet50 introduce conexiones residuales que permiten un flujo de gradiente más efectivo, facilitando el entrenamiento de redes profundas sin sufrir el problema de la degradación.

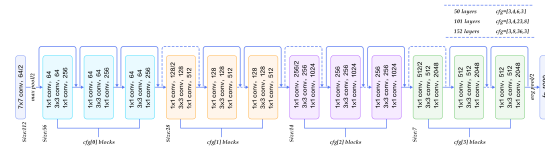


Figura 1. Arquitectura ResNet50. (He et al., 2016)

II-D. Arquitectura del Modelo y Transfer Learning

Se seleccionó la arquitectura ResNet50 (He et al., 2016) y se aplicó *Transfer Learning* en modo de extracción de características. Este enfoque consiste en:

1. Cargar el modelo ResNet50 pre-entrenado en ImageNet, pero sin su capa de clasificación final (`include_top=False`).
2. Congelar los pesos de todas las capas convolucionales (`base_model.trainable = False`) para preservar el conocimiento de características de bajo nivel ya aprendido.
3. Añadir una nueva "cabeza" de clasificación adaptada a nuestro problema, compuesta por una capa `GlobalAveragePooling2D` y una capa `Dense` final con activación `softmax`.
4. Entrenar únicamente los pesos de esta nueva cabeza, lo que resulta en un proceso computacionalmente eficiente y efectivo para la especialización del modelo.

II-E. Preprocesamiento y Aumento de Datos

Se definió una *pipeline* de aumento de datos (*Data Augmentation*). Sobre el conjunto de entrenamiento, se aplicaron las siguientes transformaciones en secuencia:

1. **Mejora de Contraste con CLAHE:** Primero, se aplica el algoritmo *Contrast Limited Adaptive Histogram Equalization* (CLAHE). Este método mejora el contraste local de la imagen, haciéndola más robusta a variaciones de iluminación sin amplificar el ruido excesivamente. La implementación convierte la imagen a espacio de color LAB y aplica CLAHE sobre el canal de luminosidad (L).
2. **Aumento de Datos (Data Augmentation):** A continuación, se aplica un conjunto de transformaciones aleatorias para aumentar la diversidad del dataset y mejorar la generalización del modelo:
 - **RandomFlip:** Volteo horizontal y vertical.
 - **RandomRotation:** Giros con un factor de hasta 0.35.
 - **RandomZoom:** Zoom con un factor de hasta 0.2.
 - **GaussianNoise:** Adición de ruido gaussiano (desviación estándar de 0.35) para simular las condiciones de captura con cámaras de menor calidad, como las de laptops, y mejorar la robustez en entornos reales.
3. **Normalización:** Finalmente, se utiliza la función `preprocess_input` específica de ResNet50 para escalar los valores de los píxeles al rango que el modelo preentrenado espera.

II-F. Métricas de Evaluación

Para una evaluación completa del rendimiento del modelo, se utilizaron las siguientes métricas, donde TP (True Positive), TN (True Negative), FP (False Positive) y FN (False Negative) representan los resultados de la clasificación para una clase específica.

- **Pérdida (Loss):** Se utilizó *Sparse Categorical Crossentropy* como función de pérdida, que mide qué tan alejadas están las predicciones del modelo de los valores reales.
- **Precisión General (Accuracy):** Proporción de predicciones correctas sobre el total.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (1)$$

- **Precisión por Clase (Precision):** De todas las predicciones para una clase, ¿cuántas fueron correctas? Es crucial para medir la fiabilidad de las predicciones positivas.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (2)$$

- **Sensibilidad (Recall):** De todas las instancias reales de una clase, ¿cuántas se identificaron correctamente? Mide la capacidad del modelo para encontrar todas las muestras relevantes.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (3)$$

- **Puntuación F1 (F1-Score):** Media armónica de Precisión y Recall, proporcionando una sola métrica que equilibra ambas. Es especialmente útil cuando hay un desequilibrio de clases.

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (4)$$

- **Matriz de Confusión:** Una tabla que visualiza el rendimiento detallado del clasificador. Las filas representan las clases reales y las columnas las clases predichas. Permite identificar fácilmente qué clases son confundidas entre sí por el modelo.

II-G. Técnicas de Análisis

El análisis del rendimiento del modelo se realizó mediante dos enfoques complementarios:

1. **Análisis Cuantitativo:** Se monitorizaron las curvas de aprendizaje, graficando la precisión y la pérdida tanto para el conjunto de entrenamiento como para el de validación a lo largo de las épocas. Esta técnica es fundamental para diagnosticar el comportamiento del entrenamiento, identificar problemas como el sobreajuste (sobreentrenamiento) o el subajuste, y verificar la convergencia del modelo.
2. **Análisis Cualitativo:** Se realizó una inspección visual de las predicciones del modelo en imágenes individuales y en el flujo de video en tiempo real. Este análisis permite identificar los escenarios donde el modelo funciona bien y aquellos en los que comete errores, proporcionando información valiosa sobre su robustez frente a desafíos como oclusiones, fondos complejos o iluminaciones adversas.

II-H. Estrategia de Validación

Para asegurar que el modelo desarrollado generalice correctamente a datos nuevos y no simplemente memorice el conjunto de entrenamiento, se implementó una estrategia de validación de tipo **hold-out**. El dataset RawRipe se dividió en dos subconjuntos mutuamente exclusivos:

- **Conjunto de Entrenamiento (80 %):** Utilizado para el ajuste de los pesos del modelo. Sobre este conjunto se aplicó la estrategia de aumento de datos intensivo.
- **Conjunto de Validación (20 %):** Utilizado para evaluar el rendimiento del modelo al final de cada época. Este conjunto no se utiliza para el entrenamiento y sirve como una proxy de datos no vistos.

El criterio para la selección del mejor modelo fue la máxima precisión alcanzada en el conjunto de validación. Al final de cada época, si la precisión en este conjunto superaba la mejor precisión registrada hasta el momento, se guardaba una copia de los pesos del modelo. Este enfoque garantiza que el modelo final sea el que mejor generaliza, previniendo el sobreajuste.

III. DESARROLLO DEL PROYECTO

III-A. Análisis inicial

El presente proyecto satisface la necesidad de crear un modelo de clasificación que mejore la velocidad, disminuya los costos y los errores que puede tener un trabajador humano en la clasificación de maduración de frutas en empresas agrícolas o agroindustrias. Múltiples investigaciones demostraron que la visión computacional es una herramienta muy poderosa en este tipo de problemas.

Antes de la implementación: Antes del desarrollo del sistema, la clasificación de maduración de frutas en negocios agrícolas y agroindustriales se realizaba de forma manual, teniendo que contar con la inspección realizada por personal. Este proceso traía consigo una gran cantidad de tiempo, fatiga por parte de los trabajadores que daría paso a cometer errores en clasificar y, por consiguiente, causaría pérdidas.



Figura 2. Revisión manual de maduración de frutas.

Luego de la implementación: Luego del desarrollo del sistema y su implementación, sucedería una automatización del proceso de clasificación de maduración; esto ayudaría a identificar rápidamente si una fruta está inmadura o madura. También se obtendría una reducción significativa del error; por consiguiente, una mejora en el proceso de calidad.



Figura 3. Implementación de un sistema de VC clasificador de frutos usando un carro controlado remotamente.

Alcance:

1. **Automatización de la Inspección Visual:** Reemplaza la inspección manual, que es lenta y propensa a errores, por

un proceso automatizado. Mejorando de forma directa la eficiencia y la precisión en la clasificación de frutas.

2. **Clasificación Multiclase para una Evaluación Integral:** El sistema está diseñado para determinar simultáneamente tanto el tipo de fruta (manzana, plátano, fresa, etc.) como su estado de madurez (inmaduro o maduro). Esta capacidad permite una categorización completa y detallada del producto en una sola pasada.
3. **Uso de Tecnologías de Aprendizaje Profundo para Alta Precisión:** Se emplea un modelo de Red Neuronal Convolutiva (CNN) basado en la arquitectura ResNet50 con Transfer Learning. Esta elección tecnológica busca alcanzar un alto rendimiento en la clasificación, aprovechando modelos ya probados en tareas complejas de visión por computadora.
4. **Optimización de la Cadena de Suministro y Reducción de Costos:** Al automatizar la clasificación, el sistema tiene como objetivo optimizar el tiempo de selección y la gestión de los lotes de frutas. Esto busca reducir los costos operativos asociados a la mano de obra y minimizar las pérdidas económicas causadas por el deterioro de la fruta mal clasificada.
5. **Desarrollo de Prototipos Funcionales para Validación:** El proyecto no se limita a un modelo teórico, sino que entrega dos aplicaciones de escritorio funcionales (para imágenes estáticas y para video en tiempo real). Estas herramientas sirven como prototipos para demostrar la viabilidad y la facilidad de uso del sistema en escenarios prácticos.

Limitaciones:

1. **Dependencia de la Calidad del Dataset:** El rendimiento del modelo está directamente ligado a las características del dataset *RawRipe*. Este dataset contiene muy pocos ejemplos y hay imágenes que causan ciertas dudas a pesar de ser un dataset de un artículo científico publicado en la IEEE. Lo ideal sería tener acceso a un entorno de campo de alguna empresa o negocio agroindustrial, pero por motivos de estudio, en este artículo resulta práctico para calcular su rendimiento.
2. **Discretización de la Madurez:** El modelo solo clasifica las frutas en dos categorías binarias: inmaduro o "maduro". No tiene la capacidad de identificar estados intermedios (como "semi-maduro"), que son de gran importancia comercial, esta es otra limitación de usar el dataset elegido.
3. **Dificultad en conseguir datasets de artículos científicos de renombre:** La mayoría de artículos científicos relacionados usan datasets privados, es muy difícil encontrar datasets públicos para este tipo de problema, lo cual recorta la disponibilidad y elección.
4. **Generalización a Nuevas Frutas:** El modelo está especializado en las 5 clases de frutas para las que fue entrenado. No puede clasificar un tipo de fruta que no ha visto previamente (por ejemplo, peras o uvas). Para ello, sería necesario ampliar el dataset y reentrenar el

modelo.

III-B. Diseño de alto nivel

El sistema sigue una arquitectura modular, como se muestra en la Figura 4.

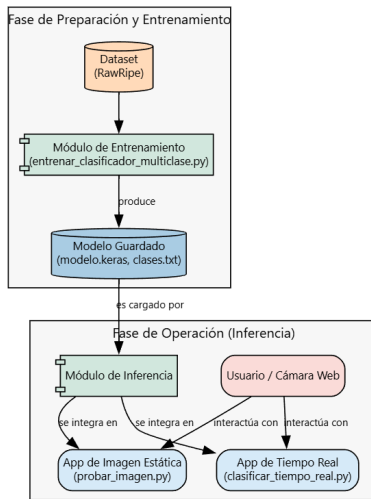


Figura 4. Diagrama de la arquitectura de alto nivel del sistema.

- **Módulo de Datos:** Carga, preprocesa (con CLAHE) y aumenta el dataset usando 'tf.data'.
- **Módulo de Entrenamiento:** Configura y entrena el modelo ResNet50 con Keras.
- **Módulo de Inferencia:** Carga el modelo '.keras' para predicciones.
- **Módulo de UI:** Gestiona la interacción con el usuario (Tkinter y OpenCV).

III-C. Diseño detallado

A continuación, se detallan los flujos de trabajo clave.

III-C1. Módulo de Entrenamiento: Implementado en `entrenar_clasificador_multiclase.py`. El proceso se visualiza en la Figura 5.

Flujo del Módulo de Entrenamiento (`entrenar_clasificador_multiclase.py`)

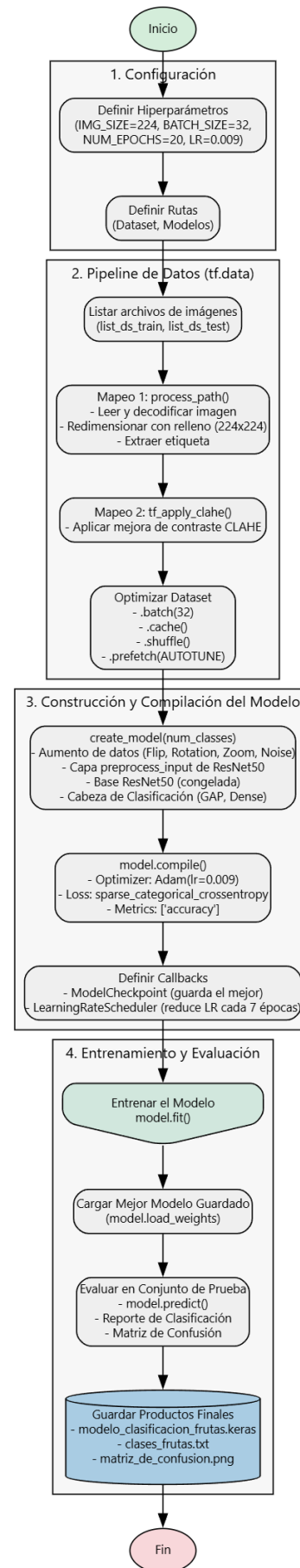


Figura 5. Diagrama detallado del proceso de entrenamiento.

1. **Configuración de Hiperparámetros:** Se definen los valores clave: `IMG_SIZE` (224), `BATCH_SIZE` (32), `NUM_EPOCHS` (20), y una tasa de aprendizaje inicial `LEARNING_RATE` (0.009).
2. **Compilación y Callbacks:** Se utiliza el optimizador Adam y la pérdida `SparseCategoricalCrossentropy`. Se configuran dos *callbacks* clave:

- `ModelCheckpoint`: Para guardar el mejor modelo basado en `val_accuracy`.
- `LearningRateScheduler`: Para reducir la tasa de aprendizaje en un factor de 0.1 cada 7 épocas, permitiendo un ajuste más fino de los pesos en las etapas finales del entrenamiento.

3. **Ciclo de Entrenamiento y Evaluación:** Se invoca `model.fit` para ejecutar el entrenamiento. Al finalizar, el script carga el mejor modelo guardado y genera un reporte de clasificación y una matriz de confusión para una evaluación detallada del rendimiento en el conjunto de prueba.

Flujo de Inferencia de Imagen Estática (probar_imagen.py)

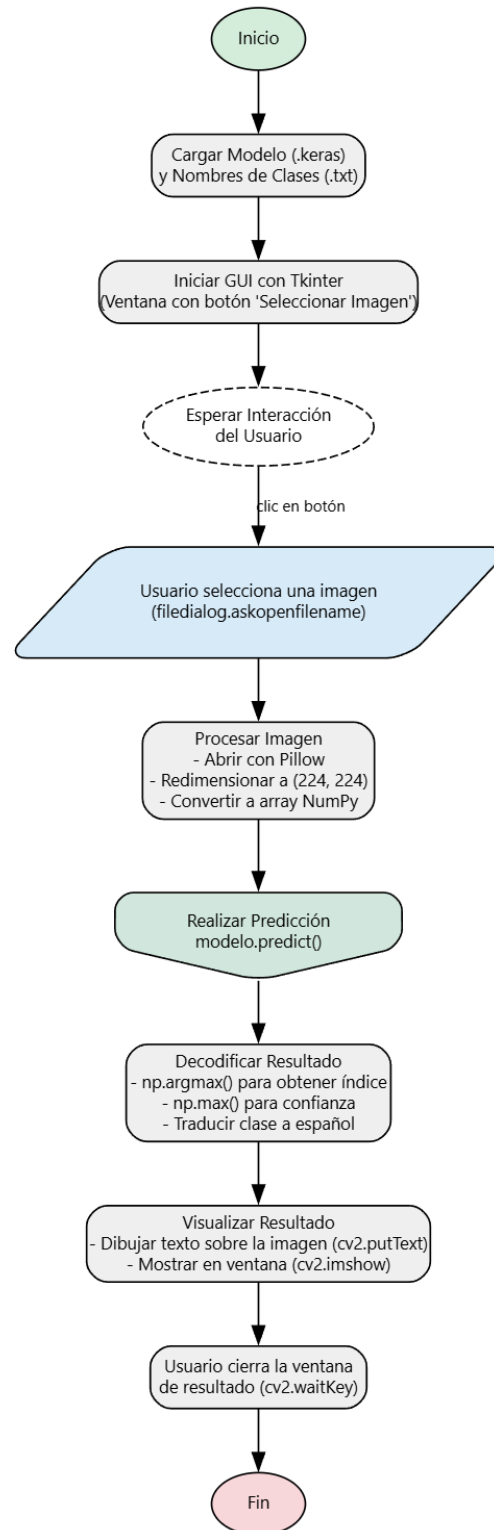


Figura 6. Diagrama detallado de la inferencia con imágenes estáticas.

III-C2. Módulo de Inferencia con imagen estática: Implementado en `clasificar_tiempo_probarimagen.py`. El flujo de trabajo se detalla en la Figura 6.

III-C3. Módulo de Inferencia en Tiempo Real: Implementado en `clasificar_tiempo_real.py`. El flujo de trabajo se detalla en la Figura 7.

Flujo de Inferencia en Tiempo Real (clasificar_tiempo_real.py)

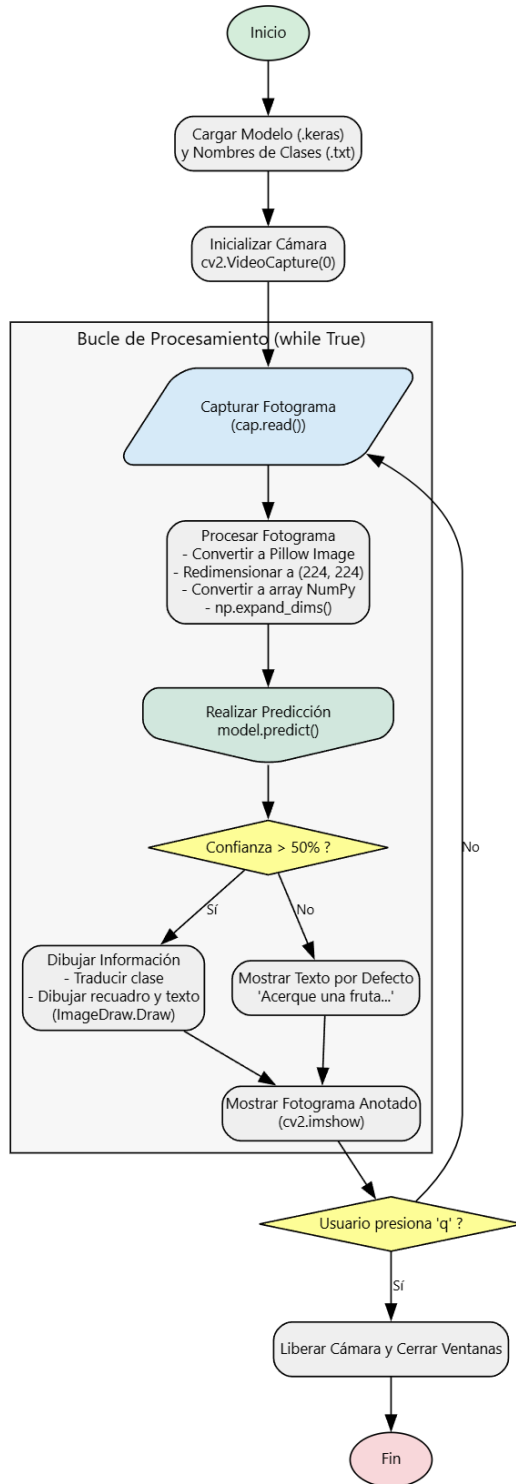


Figura 7. Diagrama detallado de la inferencia en tiempo real.

1. **Inicialización:** Se carga el modelo Keras, la lista de clases, se inicializa la cámara con OpenCV y una fuente TrueType con Pillow.
2. **Bucle de Procesamiento:** Se captura un fotograma, se convierte a formato Pillow, se redimensiona a 224x224

y se prepara para el modelo.

3. **Predicción y Visualización:** Se invoca `model.predict()`. Si la confianza supera el 50 %, se dibuja un recuadro y el texto de la predicción sobre el fotograma original usando Pillow para mayor calidad visual.
4. **Salida:** El fotograma anotado se muestra en una ventana de OpenCV. El bucle termina al presionar la tecla `q`.

IV. ANÁLISIS DE RESULTADOS

El entrenamiento del modelo se completó en un tiempo de 8 minutos y 19 segundos sobre el hardware especificado. El modelo final guardado, correspondiente al de mayor precisión en el conjunto de validación, alcanzó un valor de **0.9115**. La estructura final del modelo, como se detalla en la Tabla I, confirma la estrategia de Transfer Learning: de un total de 23,608,202 parámetros, solo se entrenaron 20,490 (0.087 %), correspondientes a la cabeza de clasificación añadida.

Cuadro I
RESUMEN DE LA ARQUITECTURA DEL MODELO

Layer (type)	Output Shape	Param #	Connected to
input_layer (InputLayer)	(None, 224, 224, 3)	0	-
data_augmentation (sequential)	(None, 224, 224, 3)	0	input_layer[0][0]
get_item (getitem)	(None, 224, 224)	0	data_augmentation[0][0]
get_item_1 (getitem)	(None, 224, 224)	0	data_augmentation[0][0]
get_item_2 (getitem)	(None, 224, 224)	0	data_augmentation[0][0]
stack (Stack)	(None, 224, 224, 3)	0	get_item[0][0], get_item_1[0][0], get_item_2[0][0]
add (Add)	(None, 224, 224, 3)	0	stack[0][0]
resnet50 (Functional)	(None, 7, 7, 2048)	23,587,712	add[0][0]
global_average_pooling2d (GlobalAveragePooling2D)	(None, 2048)	0	resnet50[0][0]
dense (Dense)	(None, 10)	20,490	global_average_pooling2d[0][0]

Total params: 23,608,202 (90.06 MB)
Trainable params: 20,490 (80.04 KB)
Non-trainable params: 23,587,712 (89.98 MB)

Al evaluar el mejor modelo sobre el conjunto de prueba, se obtuvo una precisión general (accuracy) del **91 %**. El reporte de clasificación detallado, mostrado en la Figura 8, permite un análisis más profundo del rendimiento por clase.

Figura 8. Reporte de Clasificación del Modelo

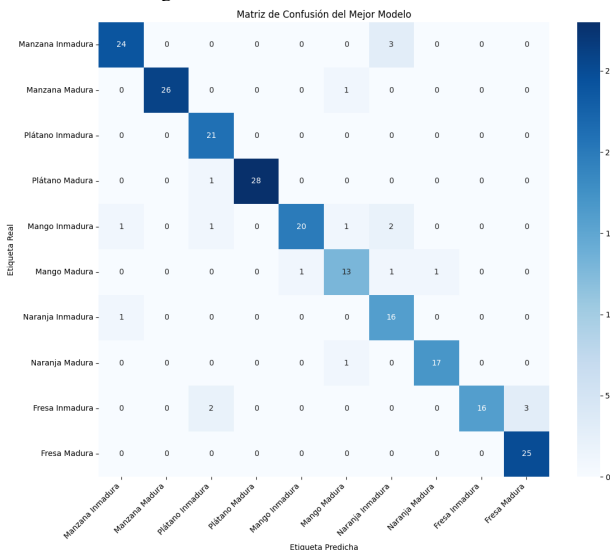
--- Reporte de Clasificación ---				
	precision	recall	f1-score	support
Manzana Inmadura	0.92	0.89	0.91	27
Manzana Madura	1.00	0.96	0.98	27
Plátano Inmadura	0.84	1.00	0.91	21
Plátano Madura	1.00	0.97	0.98	29
Mango Inmadura	0.95	0.80	0.87	25
Mango Madura	0.81	0.81	0.81	16
Naranja Inmadura	0.73	0.94	0.82	17
Naranja Madura	0.94	0.94	0.94	18
Fresa Inmadura	1.00	0.76	0.86	21
Fresa Madura	0.89	1.00	0.94	25
accuracy			0.91	226
macro avg	0.91	0.91	0.90	226
weighted avg	0.92	0.91	0.91	226

Se observa un rendimiento alto en varias clases. Por ejemplo, "Manzana Madura" y "Plátano Madura" obtuvieron una precisión de 1.00, indicando que todas las predicciones para estas clases fueron correctas. De igual forma, "Fresa Madura" alcanzó un recall de 1.00, lo que significa que el modelo identificó correctamente todas las fresas maduras del conjunto de prueba.

Sin embargo, el reporte también señala áreas de mejora. La clase "Naranja Inmadura" presenta la precisión más baja (0.73), mientras que "Fresa Inmadura" tiene el recall más bajo (0.76). Esto sugiere que el modelo tiene dificultades para identificar correctamente estas clases específicas.

La matriz de confusión en la Figura 9 visualiza estos resultados. La diagonal principal es fuerte, lo que confirma el buen rendimiento general. Los errores de clasificación (valores fuera de la diagonal) proporcionan información sobre las confusiones del modelo.

Figura 9. Matriz de Confusión del Modelo



Las principales confusiones observadas son:

- 3 instancias de "Manzana Inmadura" fueron clasificadas erróneamente como "Naranja Inmadura".
- 3 instancias de "Fresa Inmadura" fueron clasificadas como "Fresa Madura".
- La clase "Mango Inmaduro" muestra la mayor dispersión de errores, siendo confundida con "Manzana Inmadura"(1), "Plátano Inmaduro"(1), "Mango Maduro"(1) y "Naranja Madura"(2).

V. DISCUSIÓN

Los resultados obtenidos muestran que el modelo propuesto es capaz de clasificar el tipo y la madurez de las frutas con una precisión general alta (91 %). El uso de Transfer Learning con ResNet50 fue una estrategia efectiva, ya que permitió aprovechar una base de conocimiento visual sólida, entrenando un número muy pequeño de parámetros. Esto explica el corto tiempo de entrenamiento (aproximadamente 8 minutos) en un equipo sin GPU dedicada.

La aplicación de CLAHE y Data Augmentation, especialmente la adición de ruido gaussiano, parece haber contribuido a la buena generalización del modelo. Esto se refleja en el alto rendimiento para la mayoría de las clases, como las manzanas y plátanos, que tienen formas y colores relativamente consistentes.

Las dificultades de clasificación observadas en la matriz de confusión pueden tener varias causas. La confusión entre "Manzana Inmadura" y "Naranja Inmadura" puede deberse a similitudes en forma y coloración en sus etapas tempranas de madurez dentro del dataset utilizado. De igual forma, la menor efectividad con "Mango Inmaduro" puede estar relacionada con la alta variabilidad en la forma y el color de los mangos.

La confusión entre estados de madurez de la misma fruta, como en el caso de las fresas, es un desafío esperado, ya que la transición de inmadura a madura es un proceso gradual con características visuales que pueden solaparse. Estos errores específicos no invalidan el modelo, sino que señalan los límites de un sistema basado únicamente en características visuales capturadas en un dataset específico.

VI. CONCLUSIONES

Con base en el desarrollo y los resultados de este trabajo, se concluye lo siguiente:

1. Se desarrolló con éxito un sistema de clasificación de madurez de frutas utilizando Transfer Learning con la arquitectura ResNet50, alcanzando una precisión general del 91 % en el conjunto de datos de prueba.
2. La combinación de preprocesamiento con CLAHE y una estrategia de aumento de datos que incluyó transformaciones geométricas y ruido gaussiano fue efectiva para entrenar un modelo robusto en un tiempo computacional reducido y sin hardware especializado.
3. El modelo demostró un alto rendimiento para la identificación de frutas con características bien definidas (manzanas, plátanos), pero presentó dificultades para

diferenciar entre clases con alta variabilidad visual (manzanos) o con características similares en estado inmaduro (manzanas y naranjas).

4. El sistema desarrollado representa una solución funcional y de bajo costo para la automatización del control de calidad en el sector agrícola. Como trabajo futuro, se recomienda ampliar el dataset con más ejemplos de las clases problemáticas e incluir estados de madurez intermedios.

REFERENCIAS

- Aherwadi, N., Mittal, U., Singla, J., Jhanjhi, N. Z., Yassine, A., and Hossain, M. S. (2022). Prediction of fruit maturity, quality, and its life using deep learning algorithms. *Electronics*, 11(24):4100.
- Behera, S. K., Rath, A. K., and Sethy, P. K. (2021). Maturity status classification of papaya fruits based on machine learning and transfer learning approach. *Information Processing in Agriculture*, 8(2):244–250.
- He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.
- Jerripothula, K. R., Shukla, S. K., Jain, S., and Singh, S. (2021). Fruit maturity recognition from agricultural, market and automation perspectives. In *2021 IEEE 11th International Conference on System Engineering and Technology (ICSET)*, pages 250–255. IEEE.
- Ministerio de Desarrollo Agrario y Riego (MIDAGRI) (2024). Padrón nacional de sectores estadísticos (se) 2024 (marzo 2024). Technical report, Dirección de Estadística e Información Agraria - DEIA. Padrón Nacional de Datos Abiertos del Estado Peruano.
- Ministerio de Desarrollo Agrario y Riego (MIDAGRI) (2025). Valor bruto de la producción agropecuaria: Diciembre 2024. Technical report, Dirección General de Estadística, Seguimiento y Evaluación de Políticas (DGESEP). Informe Mensual.
- Moya, V., Quito, A., Pilco, A., Vásquez, J. P., and Vargas, C. (2024). Crop detection and maturity classification using a yolov5-based image analysis. *Emerging Science Journal*, 8(2):496–514.
- Siricharoen, P., Yomsatieankul, W., and Bunsri, T. (2023). Fruit maturity grading framework for small dataset using single image multi-object sampling and mask r-cnn. *Smart Agricultural Technology*, 3:100130.