Universidad Autónoma de Querétaro
Facultad de Ingeniería

Título del tema de tesis registrado

Sistema de alerta del estado de maduración de alimentos frescos dentro de un refrigerador utilizando Inteligencia Artificial

Tesis

Que como parte de los requisitos para
obtener el Grado de

Maestro en Ciencias en Inteligencia Artificial

Presenta
Enoc Tapia Méndez

Dirigido por:
Dr. Luis Alberto Morales Hernández

Querétaro, Qro. julio 2023

Universidad Autónoma de Querétaro
Facultad de Ingeniería
Maestría en Ciencias en Inteligencia Artificial

Título del tema de tesis registrado

Sistema de alerta del estado de maduración de alimentos frescos dentro de un refrigerador utilizando Inteligencia Artificial

Tesis

Que como parte de los requisitos para
obtener el Grado de

Maestro en Ciencias en Inteligencia Artificial

Presenta
Enoc Tapia Méndez

Dirigido por:
Dr. Luis Alberto Morales Hernández

Dr. Luis Alberto Morales Hernández
Presidente
Dr. Irving Armando Cruz Albarrán
Secretario
Dr. Saúl Tovar Arriaga
Vocal
Dr. Andras Tacaks
Suplente
M.C. Luis Ballesteros Martínez
Suplente

Querétaro, Qro. julio 2023

Dedicado a mis padres,
Efraín Tapia y Ma. Josefina Méndez

Sin darnos cuenta, hoy tenemos logradas cosas que años atrás fueron el *"algún día"*

AGRADECIMIENTOS

Mi primer agradecimiento, se dirige a quien con su gracia y favor inmerecido me ha permitido concluir esta etapa, a Dios.

A mis padres Efraín Tapia Pérez y Ma. Josefina Méndez Baeza, por su apoyo incondicional, por darme las palabras cuando sentía que todo se venía abajo, por ser el motor que genera esperanza dentro de mí.

A mis hermanos Zuri y Josué, por ser mi ejemplo por seguir, por tolerar mis malos ratos o bajo presión y en general a toda mi familia, ya que sin todos ellos, esto no sería posible.

Un grato agradecimiento a mi equipo de asesores de tesis por parte de la Universidad Autónoma de Querétaro. Dr. Luis A. Morales Hernández, gracias por aceptar dirigir la investigación, por su noble entrega y apoyo, por su dedicación y motivación para enseñarme lo que se hace con la ciencia. Dr. Irving A. Cruz Albarrán, gracias por su orientación y demostración de pasión hacia la investigación. Dr. Saúl Tovar Arriaga, gracias por la confianza y tan grande esfuerzo que realiza por divulgar temas de Inteligencia Artificial. Dr. Andras Tacaks, gracias por cada uno de sus conocimientos transmitidos y por su singular paciencia en ello.

A los profesores e investigadores de la Maestría en Ciencias en Inteligencia Artificial.

Un agradecimiento especial al M.C. Luis Ballesteros Martínez, por guiarme en el proceso de aprendizaje laboral y personal, por hacerme notar mis áreas de oportunidad y sobre todo por la gran amistad; es un gusto hablar sobre ciencia contigo. Al M.C. César Gutiérrez Pérez, gracias por la oportunidad para desarrollar el proyecto, por la paciencia y la transmisión de conocimientos, y que con ello nació una buena amistad.

Al centro de tecnología y proyectos de Mabe (TyP), por la colaboración en el desarrollo del proyecto de investigación.

A la Universidad Autónoma de Querétaro y al CONAHCYT por la beca otorgada para mis estudios de posgrado, con número de CVU 1144309.

ÍNDICE GENERAL

ÍNDICE DE CUADROS

ÍNDICE DE FIGURAS

RESUMEN

A medida que las frutas y verduras crecen, así como maduran, experimentan un fenómeno fisiológico en el que se producen cambios visibles en su apariencia externa, además en su contenido nutricional. El objetivo de la investigación es crear e implementar un sistema inteligente que utilice Inteligencia Artificial para determinar el estado de maduración de frutas y verduras. La metodología utilizada corresponde para el diseño de modelos a través de aprendizaje profundo y consta de una serie de pasos que son indispensables para generación de modelos de aprendizaje profundo, donde se inicia con la etapa del análisis y limpieza de datos, seguido del entrenamiento del modelo, para finalmente realizar la validación de las métricas obtenidas. Los métodos utilizados son clasificación de imágenes, un primer modelo para clasificación de 32 clases de frutas y verduras, en el cual se obtuvo una precisión de 98% y una exactitud de 97.86%. El segundo modelo realiza la clasificación de 12 estados de maduración, obteniendo una precisión de 97% y una exactitud de 96.67%. El método de detección de objetos predice 39 clases de frutas, verduras y estados de maduración, para el cual se obtienen métricas de 72.4% para precisión, 75.2% para mAP@50 y 63.4% para mAP@50-95. A su vez, se realiza la implementación de los modelos en hardware, brindando un tiempo de inferencia para el modelo de detección de objetos de aproximadamente 10 segundos por muestra.

Palabras clave: Aprendizaje profundo, frutas, maduración, verduras.

ABSTRACT

As fruits and vegetables grow also ripen, they undergo a physiological phenomenon in which visible changes occur in their external appearance as well as their nutritional content. The objective of the research is to create and implement an intelligent system that uses Artificial Intelligence to determine the ripening state of fruits and vegetables. The methodology used corresponds to the design of models through deep learning and consists of a series of steps that are essential for the generation of deep learning models, where it begins with the stage of analysis and data cleaning, followed by model training, to finally perform the validation of the metrics obtained. The methods used are image classification, a first model for classification of 32 kinds of fruits and vegetables, in which an accuracy of 98% and an accuracy of 97.86% were obtained. The second model performs the classification of 12 states of maturation, obtaining an accuracy of 97% and an accuracy of 96.67%. The object detection method predicts 39 kinds of fruits, vegetables and ripening states, for which metrics of 72.4% for accuracy, 75.2% for mAP@50 and 63.4% for mAP@50-95 are obtained. In turn, the implementation of the models in hardware is carried out, providing an inference time for the object detection model of approximately 10 seconds per sample.

Keywords: Deep learning, fruits, ripeness, vegetables.

CAPÍTULO I. MARCO DE REFERENCIA

1.1 INTRODUCCIÓN

La detección del estado de maduración en frutas y verduras se ha convertido en una tarea importante para la sociedad, pero se ha tratado de un problema que se ha tratado de resolver durante un largo tiempo. Conocer el estado de maduración de los alimentos frescos se ha convertido en una tarea que se lleva a cabo día con día, pero que algunas veces no se logra entender este proceso o en cuál etapa es la ideal para poder consumirla.

El ser humano está dotado de cinco sentidos o sensores principales que ayudan a la orientación, los cuales son: Gusto, olfato, tacto, vista y oído. Estos sentidos intervienen en la vida diaria y se relacionan entre sí, brindando como resultado una salida que se redirecciona al cerebro, logrando una generalización respecto a un problema en específico.

Empíricamente a través de los años, el humano determina el estado de maduración de frutas y verduras mediante cuatro sentidos: Gusto, olfato, tacto y vista. Generalmente el orden sobre cómo se utilizan los sentidos para poder determinar un estado de maduración o sí el alimento se encuentra en mal estado dependen de algunos factores, pero la mayoría de las ocasiones se realiza de la siguiente forma, siguiendo un orden sucesivo, ocurriendo de la siguiente forma: primeramente interviene la visita donde se observan características tales como el color, enseguida interviene el tacto donde se localizan características como textura y consistencia, en tercer lugar se utiliza el olfato, en donde se determina a través del olor expedido y como último caso el gusto, aunque suele considerarse no tan común, se toma la determinación de usar este sentido cuando no es posible identificar el estado de madurez a través de los sentidos anteriormente mencionados.

El estado de maduración de la mayoría de las frutas y verduras se puede determinar a través de la variable del color, siendo esta una de las que se puede identificar mediante técnicas de Inteligencia Artificial.

A medida que se realizan avances dentro de la ciencia y la tecnología, la inteligencia artificial ha apostado por intentar imitar el pensamiento humano y llevar a cabo tareas por medio de algoritmos sofisticados que los humanos pueden realizar. Esto es posible gracias a las investigaciones que se han llevado a cabo desde hace algunos años, particular en las redes neuronales artificiales.

La inteligencia artificial es capaz de realizar un sin número de generalizaciones y resolver problemas que para el humano serían complicadas de realizar. Si se realiza una comparativa entre los sentidos humanos y se tratara de localizar un sensor

industrial que pudiera realizar la función de estos, se encontrarían para la vista, cámaras, para el olfato, sensores de olfato o narices electrónicas, por mencionar.

De las grandes ventajas que se obtienen a partir del diseño e implementación de un algoritmo inteligente por medio de técnicas de inteligencia artificial, es que se pueden acoplar sensores que emulen la capacidad de los sentidos humanos y estos mismos tener la capacidad de alimentar al algoritmo para obtener un resultado o a su vez, alimentar al sistema para obtener los resultados esperados.

Dado que el procesamiento de los datos de tipo imágenes suele ser de forma muy compleja y consumo de grandes recursos computacionales, es que nace el Aprendizaje Profundo o por su acrónimo en inglés Deep Learning. Esta técnica emplea redes neuronales artificiales profundas, en las cuales toda la información es procesada de tal forma que pueda predecir dada una entrada a la red.

En las últimas décadas, el desperdicio de alimentos se ha convertido en una problemática a nivel mundial y que ha causado gran preocupación debido a que los índices aumentan día con día. La inteligencia artificial puede ayudar a resolver esta problemática haciendo uso de la identificación de frutas y verduras, que generalmente se encuentran dentro del refrigerador y una vez realizado esto, predecir su estado de maduración.

1.2 PLANTEAMIENTO DEL PROBLEMA

Tanto las frutas como las verduras a medida que ocurre su etapa de crecimiento y maduración presentan un fenómeno fisiológico, en el que suceden cambios que son visibles tanto en el exterior como en el interior.

Dicha maduración fisiológica ocurre antes del desarrollo completo y que después de la cosecha debe sobrevivir sin ayuda de la planta, de la mano de sus propios sustratos acumulados. En esta etapa intervienen procesos físicos, químicos, bioquímicos y biológicos que tienen que ver con la maduración de la fruta o verdura.

Cuando el alimento fresco alcanza las etapas de maduración y desarrollo, se producen cambios visibles como la modificación del color, debido al cambio de contenido en clorofila, carotenoides y la acumulación de flavonoides. Además, se presenta la modificación de la textura que lo provoca la alteración del turgor celular, así como la modificación de azúcares, ácidos orgánicos y algunos compuestos volátiles [1].

Generalmente para determinar de forma exacta el estado de maduración se utilizan métodos invasivos o destructivos, en los que se requiere dañar la fruta o verdura para extraer su estado de maduración exacto.

La estimación del estado de maduración presenta todo un reto, dado que no es un proceso sencillo debido a todos los procesos que intervienen en el mismo, así como

que cada clase de frutas o verduras presenta variaciones distintas a medida que ocurre el proceso de maduración.

La Inteligencia Artificial puede ayudar a solucionar el problema, implementando técnicas de redes neuronales artificiales para la predicción de frutas y verduras, así como la detección de su estado de maduración, ya que arrojaría información actual sobre las clases que se tienen y su estado de maduración a forma de notificación y recordatorio para el usuario.

1.3 JUSTIFICACIÓN

En el mundo se desperdicia aproximadamente el 34% de los alimentos frescos, específicamente en México se calcula que es el 37%, lo que interpretado en otras cifras representa 10 millones 431 mil toneladas de alimentos al año. Con esa cifra, se podrían alimentar aproximadamente a 7 millones de personas. Esto también representa un impacto económico, dado que se estima que genera pérdidas de alrededor de 490 mil millones de pesos y hablando ambientalmente, se generan 36 millones de toneladas de Dióxido de Carbono, lo que equivale a la emisión anual de casi 16 millones de vehículos. La Inteligencia Artificial demuestra grandes potenciales en la solución de problemas como el descrito anteriormente, se considera que puede aportar a la solución dadas sus capacidades que imitan los sentidos y procesos cognitivos de los seres humanos.

1.4 HIPÓTESIS

Mediante el desarrollo e implementación de un sistema, utilizando inteligencia artificial con técnicas de visión por computadora, se puede lograr la detección de frutas y verduras, así como clasificar la etapa de maduración en la que se encuentran.

1.5 OBJETIVOS

1.5.1 Objetivo general

Diseñar e implementar un sistema inteligente para la obtención del estado de maduración en alimentos, específicamente frutas y verduras, a través de la utilización de inteligencia artificial.

1.5.2 Objetivos específicos

a) Explorar e identificar las tecnologías utilizadas en la clasificación de imágenes para implementarse en la inferencia de frutas y verduras, así como crear o mejorar un modelo de red neuronal que sea capaz de arrojar la mayor precisión posible.

b) Identificar y aplicar una técnica de visión artificial que sea capaz de identificar en tiempo real las frutas y verduras.
c) Analizar y evaluar las posibilidades de implementar el sistema inteligente de detección del grado de maduración de frutas y verduras.
d) Implementar un sistema basado en inteligencia artificial capaz de identificar un par de frutas y verduras, así como la etapa del proceso de maduración en el que se encuentren y enviar al usuario los datos obtenidos.

CAPÍTULO II. MARCO TEÓRICO

2.1 ESTADO DEL ARTE

Dentro de las principales áreas de investigación acerca del monitoreo del estado de maduración y lo relacionado con refrigeradores inteligentes se encuentran 5 propuestas que conducen a la exploración de dichas tecnologías.

El monitoreo y gestión remota se refiere a que los refrigeradores pueden conectarse a internet, lo que permite a los usuarios monitorear y gestionar su contenido desde cualquier lugar y en cualquier momento a través de una aplicación móvil. Además, algunos modelos pueden incluso enviar alertas y notificaciones al usuario si detectan algún problema o cambio en la temperatura o en el contenido del refrigerador.

El control de temperatura sugiere que los refrigeradores utilicen sensores y algoritmos de aprendizaje automático para controlar la temperatura interna del refrigerador y ajustarla de manera automática y precisa. También pueden permitir a los usuarios establecer diferentes zonas de temperatura para diferentes tipos de alimentos.

Para el control de inventario, se investiga la posibilidad de que los refrigeradores estén equipados con cámaras internas y sistemas de reconocimiento de voz que permita a los usuarios gestionar el inventario de alimentos y bebidas almacenados en el refrigerador. Algunos modelos también pueden generar listas de compras automáticamente en función del contenido del refrigerador.

Además, se encuentran las sugerencias de recetas que pueden ofrecer sugerencias de recetas en función del contenido del refrigerador y de las preferencias del usuario.

Un área de investigación con tendencia futurista refiere al ahorro energético, donde los refrigeradores pueden incorporar tecnologías de eficiencia energética, como iluminación LED y compresores de velocidad variable, para reducir el consumo de energía.

La búsqueda por la conservación de los alimentos y la prolongación de su estado de consumo ideal trajo consigo la invención de la tecnología de enfriamiento, mejor conocido en la actualidad como la refrigeración. A partir de esto, han existido modificaciones que van desde el diseño mecánico hasta el mejoramiento en los procesos de enfriamiento.

En los años 90 y principios de la década del 2000, se popularizó la idea de hacer un electrodoméstico inteligente y que fuera capaz de conectarse a internet. Posteriormente a esta tecnología se le asignó el nombre de Internet de las cosas o por su acrónimo en inglés Internet of Things (IoT).

En el año 2000, se lanza el primer refrigerador conectado a internet, siendo la marca LG la responsable de esto. Este refrigerador era capaz de detectar los tipos de productos que se encontraban almacenados y realizar un seguimiento de estos productos a través del escaneo de códigos de barras o RFID. Cabe señalar que el producto no tuvo el alcance suficiente para considerarse un éxito y mantenerse dentro del sector comercial [2].

Se han llevado a cabo investigaciones acerca de la predicción del estado de los alimentos, que bien pueden ser de la clase frescos o no frescos. Dentro de la clase de alimentos frescos se pueden encontrar frutas y verduras, mientras que en la clase de no frescos se localizan todos aquellos que vienen empaquetados o que son preparados en casa.

Dentro de las investigaciones, se pueden encontrar desde aquellos que solamente se realiza la parte teórica hasta aquellos en los que se lleva a cabo un prototipado.

Se han creado aplicaciones que ayudan a la detección de la caducidad de alimentos a través de sensores. Esta propuesta está basada en la integración de sensores con la incorporación de IoT, a través del cual envía al usuario una alerta sobre los alimentos que se encuentran dentro del refrigerador y el tiempo que llevan dentro [3].

Existe otro sistema basado en IoT, el cual consta de tres partes principales que son el módulo de detección, el módulo de control y el módulo de transmisión. El módulo de detección consta de una celda de carga y un sensor de olores, mientras que el módulo de control consta de Arduino UNO y una fuente de alimentación y, por último, pero no menos importante, el módulo de transmisión consta de un módulo LCD y un módulo Wi-Fi. Estos módulos trabajan juntos para determinar el estado del contenido dentro del refrigerador y notifican al usuario sobre la condición y la cantidad de alimentos a través de un SMS o un correo electrónico [4].

En el artículo de B. Rahnemoonfar y T. Sheppard, presenta un método de detección y segmentación de frutas automatizado utilizando técnicas de aprendizaje profundo para mapear inventarios de huertos.

El estudio propone un enfoque utilizando una red neuronal convolucional (CNN) para detectar y segmentar automáticamente las frutas en las imágenes capturadas en los huertos. Para lograr esto, se recopilaron imágenes de huertos de manzanas y peras y se las etiquetó manualmente para su uso en el entrenamiento del modelo.

Los autores describen cómo se utilizó la arquitectura de la red neuronal convolucional Mask R-CNN, que es una variante de las redes neuronales convolucionales, para segmentar las frutas de los árboles en las imágenes de los huertos. Luego, se utilizó una técnica de mapeo de huertos de frutas para generar

mapas de inventario de los huertos, utilizando las imágenes segmentadas para estimar el número de frutas en cada árbol.

Los resultados del estudio muestran que el método propuesto puede detectar y segmentar con precisión las frutas en las imágenes de los huertos. Además, el método propuesto demostró ser escalable y se pudo aplicar con éxito a diferentes variedades de frutas [5].

El estudio realizado por N. N. K. Balasooriya y G. M. N. P. Gunathilake, presenta un enfoque para detectar y clasificar el estado de madurez de la fruta de papaya utilizando técnicas de visión por computadora y aprendizaje automático.

El estudio describe cómo se recopilaron imágenes de frutas de papaya de diferentes estados de madurez y se las procesó utilizando técnicas de visión por computadora para extraer características de las imágenes. Luego, se utilizó un modelo de aprendizaje automático basado en Support Vector Machine (SVM) para clasificar las frutas en diferentes categorías de madurez.

Los autores muestran que el enfoque propuesto puede detectar con precisión el estado de madurez de la fruta de papaya con una precisión promedio del 97,5%. Además, se observó que el modelo SVM es capaz de clasificar las frutas de papaya en diferentes categorías de madurez con una precisión promedio del 95,6%.

El artículo también presenta un análisis comparativo de diferentes métodos de extracción de características y algoritmos de clasificación utilizados en el estudio y se concluye que el método propuesto con SVM tuvo el mejor rendimiento en términos de precisión y eficiencia [6].

Además, se presenta un método para detectar el estado de madurez de los plátanos utilizando técnicas de aprendizaje automático. El estudio describe cómo se recopilaron imágenes de plátanos en diferentes estados de madurez y se extrajeron características de estas imágenes utilizando técnicas de procesamiento de imágenes. A continuación, se utilizó un modelo de aprendizaje automático basado en la red neuronal convolucional (CNN) para clasificar los plátanos en diferentes categorías de madurez.

Los autores muestran que el enfoque propuesto puede detectar con precisión el estado de madurez de los plátanos con una precisión promedio del 97,21%. Además, se observó que el modelo de CNN es capaz de clasificar los plátanos en diferentes categorías de madurez con una precisión promedio del 97,44%.

También presenta un análisis comparativo de diferentes métodos de extracción de características y algoritmos de clasificación utilizados en el estudio, y se concluye que el método propuesto con CNN tuvo el mejor rendimiento en términos de precisión y eficiencia [7].

2.2 ANTECEDENTES

2.2.1 Redes neuronales artificiales

Las redes neuronales artificiales o ANN por sus siglas en inglés son una forma de inteligencia artificial que trata de imitar el funcionamiento de las redes neuronales biológicas. Además se consideran un tipo de modelo de aprendizaje máquina que son diseñadas para simular el comportamiento del cerebro humano. Las redes neuronales artificiales tienen la habilidad de aprender dada una serie de datos y a partir de estos realizar predicciones.

El sistema nervioso del ser humano contiene células, que son conocidas como neuronas. Las neuronas están conectadas entre sí mediante axones y dendritas, y las conexiones de regiones entre axones y dendritas son conocidas como sinapsis. La fuerza de las conexiones sinápticas a menudo cambia en respuesta a estímulos externos. Este cambio es cómo se lleva a cabo el aprendizaje en los organismos vivos.

El mecanismo biológico es simulado en las redes neuronales artificiales, las cuales contienen unidades computacionales que se conocen como neuronas, dichas unidades computacionales están conectadas a través de pesos

La estructura básica de una red neuronal artificial consiste en tres capas: capa de entrada, una más capas ocultas y capa de salida. Cada una de estas capas contiene neuronas que están conectadas a neuronas en capas adyacentes mediante conexiones de pesos. La capa de entrada recibe los datos, que son procesados por las neuronas en las capas ocultas, antes de pasar los resultados a la capa de salida.

La capa intermedia u oculta, realiza la mayor parte del cálculo computacional en el modelo y es responsable de aprender las representaciones y características que son más relevantes en la tarea que lleva a cabo. El número de capas y el número de neuronas en cada capa son hiperparámetros que deben ser adecuados durante el proceso de entrenamiento para optimizar el rendimiento del modelo.

La función de activación es una función no lineal que introduce la no linealidad al modelo, permitiendo que pueda aprender patrones más complejos y relaciones en los datos. Existen distintas funciones de activación que pueden ser utilizadas en las redes neuronales artificiales, como por ejemplo: Sigmoid, Tanh, ReLU, Softmax, entre otras.

Las neuronas están organizadas en cada capa por nodos, en los cuales se realiza un cálculo dependiendo los datos de entrada. Enseguida, los resultados son pasados a la siguiente capa de neuronas con conexiones de pesos. Los pesos son ajustados durante el proceso de entrenamiento, permitiendo a la red aprender de los datos de entrada y mejore su rendimiento.

El proceso de entrenamiento de una red neuronal artificial involucra la alimentación con una cantidad grande de datos como entrada, así como realizando el ajuste de pesos en las conexiones entre neuronas para minimizar la diferencia entre la salida de la red y la salida deseada. Este proceso típicamente se lleva a cabo mediante un método llamado backpropagation, en el cual se calcula el error del gradiente con respecto a cada peso en la red y ajustando los pesos respectivamente.

Existen distintas técnicas que se utilizan para incrementar el rendimiento de una red neuronal artificial mientras se realiza el proceso de entrenamiento. Una de las técnicas es la de regularización, la cual agrega penalizaciones a la función de pérdida para evitar que la red se sobreajuste a los datos de entrenamiento. Otra técnica usada es el dropout, que consiste en apagar o desactivar algunas neuronas a fin de prevenir el sobre aprendizaje de la red [8].

2.2.2 Aprendizaje profundo

También conocido por su acrónimo en inglés Deep Learning, es una topología utilizada en Inteligencia artificial y se basa en redes neuronales artificiales con aprendizaje representativo. Este tipo de aprendizaje puede ser supervisado, semi supervisado o no supervisado. El término de aprendizaje profundo se refiere al uso de múltiples capas en una red.

Los métodos de aprendizaje profundo son aprendizajes representativos con múltiples niveles de capas de representación, obteniendo módulos por composición simple pero no lineal que transforman la representación a un nivel a una alta representación o complejidad mayor. Con la composición de algunas transformaciones, funciones muy complejas pueden ser aprendidas. Para tareas de clasificación, capas elevadas de representación amplifican aspectos de la entrada que son importantes para la discriminación y eliminar variaciones irrelevantes.

Generalmente, los datos de entrada son de tipo imagen y se presentan en un arreglo de pixeles. Las características aprendidas en la primera capa de representación representan la presencia o ausencia de contornos en una orientación en particular y localización de la imagen. La segunda capa típicamente detecta características particulares de los bordes, ya sea que existan variaciones pequeñas en las posiciones de los bordes. La tercera capa ensambla los patrones en una combinación más extensa que corresponde a partes similares de los objetos. El aspecto básico del aprendizaje profundo es que estas capas de características no son diseñadas por el programador, sino que son aprendidas de los datos utilizando un procedimiento general de aprendizaje [9].

2.2.3 Aprendizaje supervisado

Este tipo de aprendizaje corresponde a algoritmos que aprenden para asociar entradas con las salidas del modelo, dado un conjunto de datos de entrenamiento [9]. Estos algoritmos son utilizados cuando se desea predecir una salida cercana a un valor de entrada y cuando se tienen ejemplos pares de entradas y salidas. El objetivo es realizar predicciones exactas para nuevos datos jamás vistos por el modelo.

Existen dos clases generales de algoritmos de aprendizaje supervisado llamados, clasificación y regresión. Para los algoritmos de clasificación, el objetivo es predecir la clase mediante una etiqueta, las cuales son elegidas de una lista de posibilidades anteriormente predefinidas. Los algoritmos de regresión tienen la capacidad de predecir un número continuo o un número de punto flotante, hablando en términos de programación y un número real, en términos matemáticos [10].

Específicamente existen algoritmos totalmente dedicados al aprendizaje supervisado, entre los que se encuentran: K-nearest neighbors, Árboles de decisión, modelos lineales, redes Bayesianas, Máquinas de soporte vectorial (SVM), redes neuronales artificiales, entre otros.

2.2.4 Redes neuronales convolucionales

También conocidas como redes convolucionales o por sus siglas en inglés CNN, son redes neuronales para procesar datos que se conocen. Son capaces de incluir datos de series de tiempo, que pueden ser tomados como un vector de una dimensión, así como datos de tipo imagen que se pueden representar en vectores pixeles de dos dimensiones. El nombre de redes convolucionales proviene de una red que utiliza convoluciones matemáticas como método para generar resultados. Las redes neuronales convolucionales son simplemente redes neuronales que utilizan convoluciones en lugar de una matriz de multiplicación en alguna de sus capas.

La arquitectura general de una red neuronal convolucional se compone de la siguiente forma:

1. Capa convolucional
2. Capa de agrupación
3. Capa completamente conectada

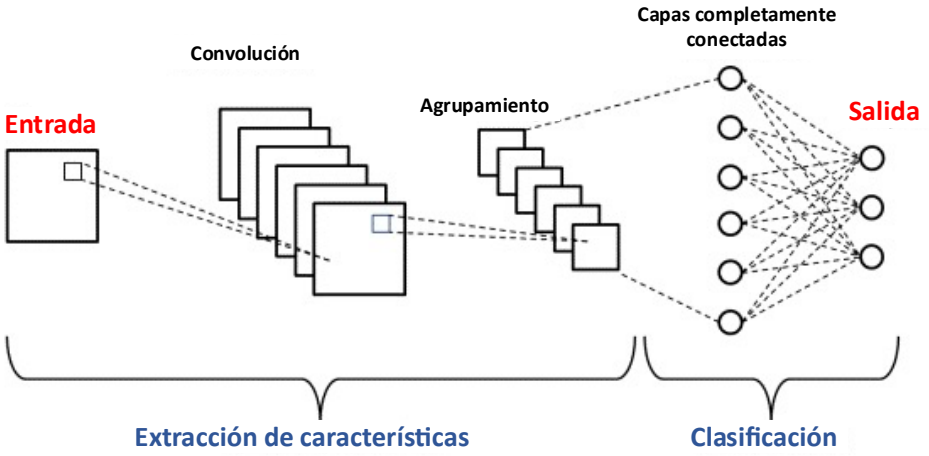


Figura 1. Esquema general de una red neuronal convolucional

La capa de convolución extrae las características de la entrada, que corresponde a la base de datos. Corresponde a una serie de operaciones matemáticas, donde la convolución toma el dato de entrada y el kernel, que es una matriz pequeña de valores, en donde este último realiza un barrido sobre la entrada. El kernel es posicionado sobre cada pixel de los datos de entrada y se calcula el producto escalar de los valores superpuestos. El valor resultante es posicionado en la salida en la posición correspondiente al centro del kernel. La ecuación matemática que representa la convolución es como se muestra a continuación.

$$y_n = \sum_{n=-\infty}^{\infty} (x_k * h_{n-k})$$

Donde x representa el dato de entrada, h el kernel e y la salida.

La salida de esta capa corresponde a un mapa de características que contiene la información de cada imagen, así como de las esquinas y contornos. El mapa de características alimenta a otras capas para que pueda producirse el proceso de extracción de características con otras imágenes de entrada. La capa de convolución pasa el resultado a la siguiente capa, una vez que ha aplicado la operación de convolución en la entrada.

La capa de agrupación o por su acrónimo en inglés Pooling, es la capa seguida de la capa de convolución. El objetivo de esta capa es decrecer el tamaño del mapa de características obtenido previamente para reducir el costo computacional. Esto se genera a partir de la reducción de conexiones entre capas, independientemente la función que ejerza cada una de ellas. En general, reúne y reduce las características generadas a partir de la capa de convolución.

La capa completamente conectada se compone de los pesos y bias generados, los que conduce a la conexión entre neuronas de dos capas distintas. Generalmente estas capas se localizan antes de la capa de salida y generan la última capa de la arquitectura de la red neuronal.

2.2.5 Visión por computadora

La visión por computadora es un área de la inteligencia artificial que tiene como propósito habilitar las máquinas para interpretar y analizar datos visuales obtenidos del mundo real, que pueden ser imágenes y videos. Dentro de las aplicaciones de la visión por computadora, se encuentran la detección de objetos, clasificación de imágenes, así como el reconocimiento facial.

Esta área utiliza algoritmos con distinto enfoque que pueden ser el procesamiento de imágenes, extracción de características, reconocimiento de objetos y aprendizaje máquina. Las técnicas de procesamiento de imágenes involucran la transformación de imágenes en información más sencilla de analizar como el filtrado o suavizado. La extracción de características identifica características importantes de una imagen, como los bordes y contornos. El objetivo de la detección de objetos identifica y clasifica objetos de una imagen o video, utilizando técnicas como las redes neuronales convolucionales (CNN), redes neuronales convolucionales basadas en regiones (R-CNN) y You Only Look Once (YOLO).

CAPÍTULO III. METODOLOGÍA

3.1 MATERIALES

El desarrollo del proyecto incluyó la parte de software y hardware, por lo tanto los materiales involucran ambas partes. Cabe señalar que primeramente se diseñó la parte del software y se validó, para enseguida pasar a la parte del diseño de hardware.

3.1.1 Base de datos

Se utilizaron distintas bases de datos ya creadas y con un fin específico, así como la creación de una base de datos para resolver el problema planteado. Para esto, las primeras bases de datos utilizadas se refieren a la clasificación de imágenes, mientras que la creada propiamente refiere a la detección de objetos. En el Cuadro 1 se muestran las bases de datos utilizadas.

Cuadro 1. Bases de datos

No.	Nombre	Número de imágenes	Número de clases
1	Fruits 360 [11]	90483	131
2	Fruits and vegetables [12]	3272	36
3	Fresh and stale [13]	9600	12
4	Elaboración propia	3900	39

3.1.1 Software

El desarrollo de los algoritmos inteligentes se programó en el lenguaje de programación Python. El entorno de programación se realizó en Jupyter Notebook y los entrenamientos de los algoritmos se llevó a cabo tanto en una CPU como en la plataforma Google Colab. Entre las librerías de acceso libre utilizadas se encuentran: Tensorflow, Pytorch, Sklearn, Pandas, Numpy, Matplotlib, entre otras.

Lo correspondiente para la implementación en hardware, se utilizó el sistema operativo Raspbian, lenguaje de programación en Python y las librerías mencionadas anteriormente.

3.1.2 Hardware

Para la realización de los entrenamientos de los algoritmos de clasificación, así como para inferencia se utilizó una CPU Apple M1 con 8 núcleos y 16 GB de memoria RAM.

La implementación del sistema se realizó en una Raspberry Pi 4 con 8 GB de memoria RAM, además de un módulo de cámara Raspberry Pi Camera, así como otros dispositivos electrónicos como cables, botones, entre otros.

3.2 METODOLOGÍA

3.2.1 Design thiking

Como parte del desarrollo metodológico, se tomó la metodología llamada Design Thinking, la cual contiene 6 etapas. Se le realizó una modificación, ya que se agrega la etapa donde interviene el modelo de Inteligencia Artificial como un proceso iterativo. En la Figura 2 se puede observar lo antes mencionado.

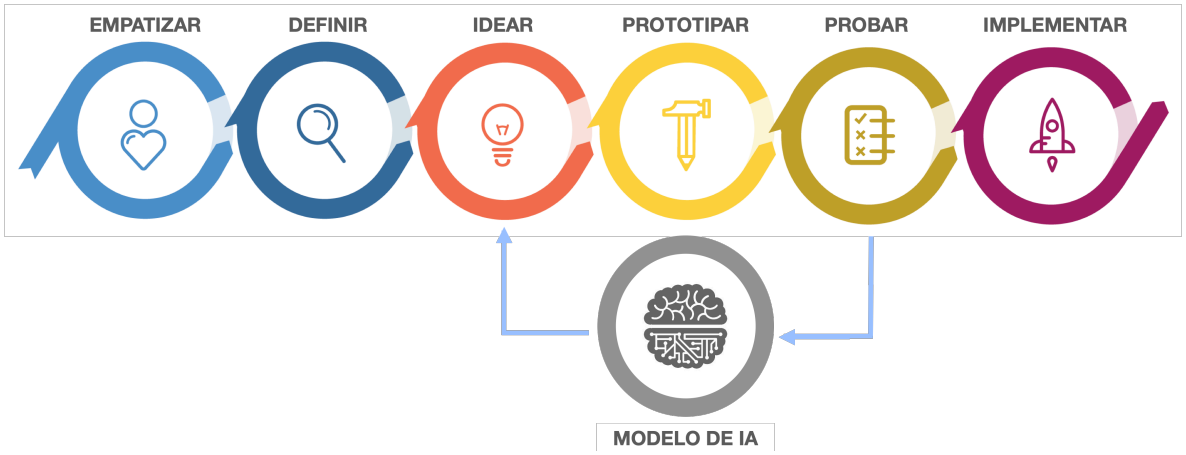


Figura 2. Metodología propuesta

En el Cuadro 2, se muestra la descripción de cada etapa de la metodología.

Cuadro 2. Etapas de la metodología Design Thinking

Etapa	Descripción
Empatizar	En este primer paso, se busca comprender las necesidades, deseos y perspectivas de las personas. Esto implica realizar investigaciones, observar el comportamiento e interactuar con los usuarios a través de entrevistas, encuestas u otros métodos para obtener información. El objetivo es identificar las necesidades de los usuarios.
Definir	Se sintetizan los hallazgos de la investigación de la etapa anterior e identificar los desafíos y oportunidades clave que deben abordarse. Se define el problema, que es una ideación clara y concisa que captura la esencia del problema a resolver. Este paso ayuda a enmarcar el problema de una manera que sea centrada en el usuario.

Idear	Se generan ideas y posibles soluciones al problema. Es una fase creativa y en la que se crean ideas y conceptos que podrían abordar las necesidades del usuario identificadas en los pasos anteriores.
Prototipar	Se generan representaciones de baja fidelidad de las posibles soluciones identificadas en la etapa anterior. El objetivo es crear prototipos tangibles que puedan ser probados y evaluados en el siguiente paso.
Probar	La etapa de prueba es donde se evalúan y refinan los prototipos creados en el paso anterior. Se busca una relación con los usuarios y las partes interesadas para recopilar comentarios y conocimientos sobre los prototipos, utilizando esta información para clarificar las soluciones e iterar en los diseños. Este paso lleva por objetivo validar suposiciones, descubrir nuevos conocimientos e identificar oportunidades de mejora.
Implementar	El paso final es donde se implementa la solución que se ha desarrollado a través de la metodología de Design Thinking. Esto puede implicar la creación de un producto o servicio final, el lanzamiento de un nuevo programa o la implementación de un proceso o cambio de sistema. También se considerará cómo comunicar la solución a las partes interesadas y garantizar que la solución sea sostenible y escalable en el tiempo.

Como se menciona en la parte del planteamiento del problema, en secciones anteriores, la necesidad identificada es el desperdicio de alimentos. A partir de esto, se realiza la propuesta de una metodología que resuelva el problema, pero recurriendo a la utilización de técnicas de Inteligencia Artificial.

3.2.2 Metodología modelo de Inteligencia Artificial

La solución a este problema se resuelve con la implementación de un sistema con Inteligencia Artificial, por lo que la metodología de Inteligencia Artificial propuesta se observa en la Figura 3, donde se inicia con la adquisición de una base de datos, seguido de un preprocesamiento, un análisis, la etapa de entrenamiento del algoritmo, una etapa de validación y finalmente la obtención de resultados a partir de inferencias.

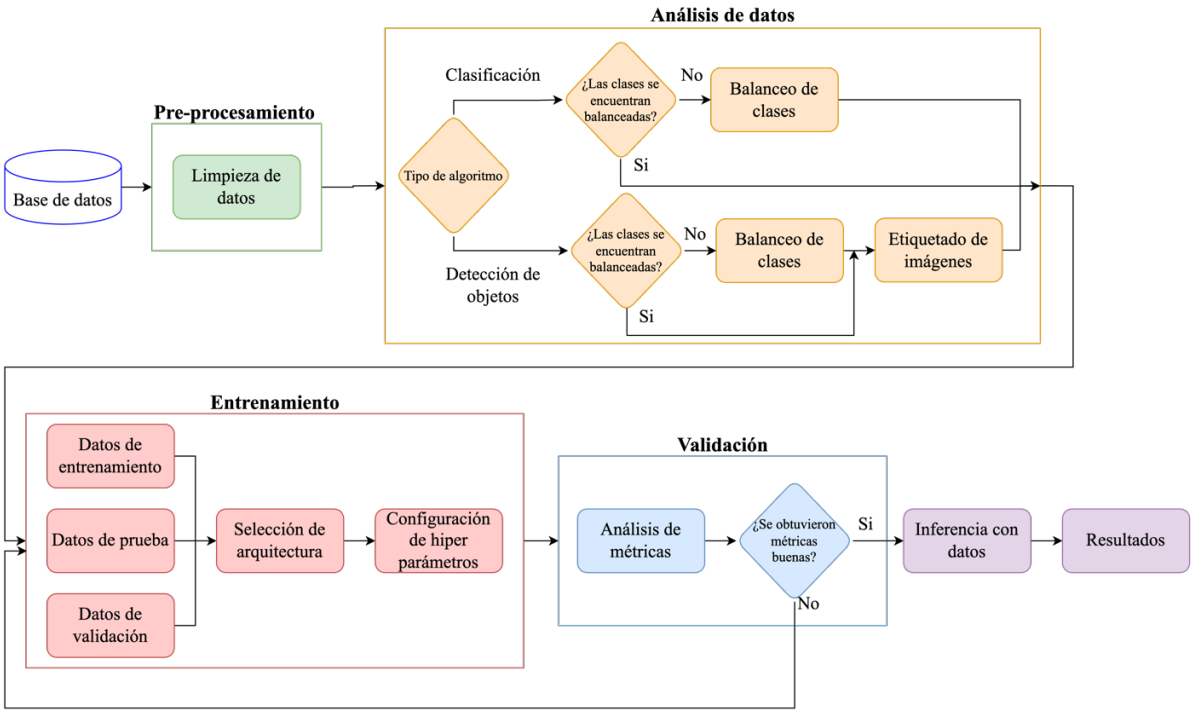


Figura 3. Metodología de IA propuesta

3.2.3 Adquisición de datos

3.2.3.1 Base de datos

Como puede observarse en la tabla 1, el número de clases y el número de imágenes varía con respecto a cada una. Las bases de datos 1, 2 y 3, se utilizaron para clasificación de imágenes, mientras que la 4 se utilizó para detección de objetos.

La base de datos uno contiene 131 clases de frutas y verduras. Dicha base de datos se encuentra dividida en datos de entrenamiento y validación, con 100 y 10 imágenes por cada clase, respectivamente. Cabe señalar que existen clases que corresponden a la misma fruta o verdura, pero que son de variedad distinta, por ejemplo para manzana se encuentran divididas por variedades correspondiente a su color y así sucesivamente. El tamaño de las imágenes es de 100x100 pixeles.

La base de datos dos se compone de 36 clases de frutas y verduras, lo cual es una representa una cantidad mucho menor en comparación a la base de datos mencionada anteriormente. El tamaño de las imágenes no es uniforme, lo que significa que se encuentran imágenes a lo largo de la base de datos de distintos tamaños. Se encuentra dividida en datos de entrenamiento, prueba y validación, de los cuales la distribución es de 90, 10 y 10 imágenes, respectivamente.

Para la base de datos tres se incluyen ya los estados de maduración de algunas frutas y verduras que corresponden a Mango, Papa, Jitomate, Manzana, Plátano y Naranja. Cada clase tiene dos estados de maduración: Fresco o sobre maduro, lo que

lleva a un total de 12 clases. Al igual que en la base de datos dos, los tamaños de las imágenes son aleatorios. No se localiza ninguna división de datos en la misma.

La última base de datos que corresponde al número cuatro se construyó de forma propia extrayendo datos de la red de forma aleatoria y sin sufrir ninguna transformación. Cuenta con un total de 3900 imágenes, sin un tamaño uniforme. La base de datos cuenta con clases de frutas y verduras, pero que algunas de ellas cuentan con la asignación de su estado de maduración, que corresponde a buen estado o mal estado y otras clases contienen tres estados de maduración que corresponden a: Sin madurar, buen estado y mal estado. Lo comentado anteriormente se puede observar en el Cuadro 3.

Cuadro 3. Estados de maduración por clase para base de datos número 4

Clases	**Estados de maduración**		
	Sin madurar	**Buen estado**	**Mal estado**
Cebolla		X	X
Plátano	X	X	X
Chile		X	X
Limón		X	X
Mango	X	X	X
Manzana		X	X
Naranja		X	X
Papa		X	X
Zanahoria		X	X
Jitomate	X	X	X

3.2.4 Pre-procesamiento de datos

Dentro del Machine Learning la etapa de pre-procesamiento en una etapa crítica y fundamental para que el algoritmo pueda entrenar de forma adecuada y a su vez obtener buenos resultados. Este proceso implica la limpieza, transformación y normalización de los datos de entrada antes de ser utilizados en un modelo de aprendizaje automático. El objetivo es mejorar la calidad de los datos y hacer que los datos sean más adecuados para su uso. Como se sabe, la mayoría de los algoritmos de Machine Learning utilizan datos tabulares, mientras que en Deep Learning son datos de tipo imagen, aun así se debe realizar este paso para tener datos de entrada de calidad. Para el caso en específico, se analizaron todos los datos y se eliminaron aquellos que eran erróneos.

3.2.4.1 Limpieza de datos

Esta etapa corresponde al proceso de identificar, corregir o eliminar datos que se encuentren incompletos, incorrectos, inconsistentes o irrelevantes. Así como se lleva a cabo en Machine Learning este proceso, se realiza de forma similar en Deep Learning, donde lo primero que se realizó fue la identificación de datos atípicos. Para el caso de estudio se observaron las imágenes una por una y se procedió a eliminar aquellas que no representaban valor o que no seguían un patrón de acuerdo con la clase. En la Figura 4 se muestra un ejemplo.

a) Dato atípico b) Dato típico

Figura 4. Ejemplo de limpieza de datos

Como se observa en la figura anterior, a) representa un dato atípico, ya que se identifica como un valor extremo o fuera de rango. Esto se llevó a cabo mediante la comparación con valores reales que deberían representar a la clase de interés, en este caso Papa. Se busca que las imágenes representen la naturaleza de la clase, es decir que muestre la fruta o verdura sin preparar. En comparación, la figura b) representa un dato que es típico o en otras palabras, que representa un dato real para los fines de la base de datos.

Al localizar los datos atípicos para las bases de datos 1, 2 y 3, estos se eliminaron para garantizar que la limpieza de datos estuviera correcta y que el modelo de aprendizaje pudiera generalizar de una forma correcta.

Además de este proceso se implementaron la detección de duplicados y la corrección de errores de formato. Para la detección de duplicados, se identificaron aquellos datos que eran iguales y se procedió a su eliminación, mientras que para la corrección de errores de formato, algunos de los datos contenían un formato incorrecto y el modelo de aprendizaje no lo admitía. Entre los formatos erróneos localizados se encuentran .gif, .tiff, entre otros. Los datos con los formatos mencionados anteriormente se cambiaron por formatos que el modelo de aprendizaje admite, como por ejemplo .png, .jpeg, entre otros.

3.2.5 Análisis de datos

Este proceso consiste en examinar, transformar o modelar los datos con el fin de comprobar sí se puede trabajar con ellos en un modelo de aprendizaje automático. Esto involucra la utilización de distintas técnicas, herramientas y metodologías para el análisis de bases de datos, así como identificar patrones, tendencias y correlaciones.

3.2.5.1 Balanceo de clases

Esta técnica utilizada en Machine Learning, se refiere a la detección de no balanceo de clases en la base de datos. Esto ocurre cuando la distribución en las clases no es igual y puede afectar al rendimiento del modelo.

Para el caso en específico, tanto para las bases de datos 2 y 3 no fue necesario realizar un balance de clases, ya que cada una de ellas contenía el mismo número de imágenes.

En la base de datos de creación propia, que corresponde al número 4, sí fue necesario el balanceo de clases, ya que al crear cada clase de forma manual, no se contabilizó con exactitud el número de imágenes con las cuales contaba cada clase.

3.2.5.2 Etiquetado de imágenes

En el aprendizaje supervisado se requiere de la intervención humana para que el algoritmo de predicción pueda generalizar de forma correcta. La primera técnica implementada es la de clasificación de imágenes. Para esto, no se requiere del etiquetado de imágenes, sólo se deben agrupar las clases en carpetas, que a su vez deben estar nombradas correctamente.

El segundo método implementado es la detección de objetos. Esta técnica requiere del etiquetado de las imágenes, es decir que sobre cada imagen se debe seleccionar la región de interés y asignar el nombre de la clase. Algunas imágenes contenidas en la base de datos, contienen una o más clases, por lo que se requirió de un etiquetado multiclase. A cada clase generada, se le asigna un nombre para que a la hora de realizar el entrenamiento, el algoritmo sepa identificar de forma correcta a cuál clase corresponde.

Para realizar el etiquetado sobre las imágenes se usó el software LabelImg, en cual se realizó el etiquetado de forma manual sobre las 3,900 imágenes contenidas en la base de datos. En la Figura 5 se muestra el multi etiquetado sobre una imagen que contiene 3 atributos de la clase Papa. Se generaron aproximadamente 10,000 etiquetas sobre la base de datos.

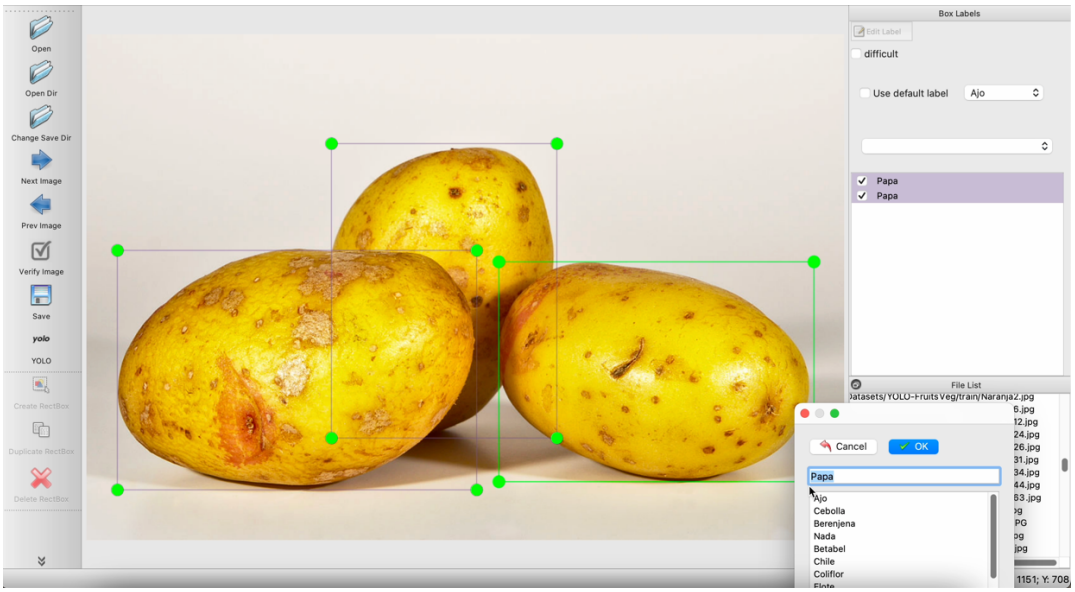


Figura 5. Etiquetado sobre imágenes

3.2.6 Entrenamiento

3.2.6.1 Datos de entrenamiento, prueba y validación

La partición de datos de entrada en datos de entrenamiento, prueba y/o validación es necesario para que el algoritmo sea capaz de entrenar y conforme pasa el tiempo, pueda realizar comparaciones para comprobar que está generalizando de una forma correcta. No siempre se utiliza la partición en los tres grupos mencionados anteriormente. Esta división en tres partes generalmente se utiliza para darle robustez al sistema y que pueda ser validado.

Para el caso específico de ambos métodos, tanto para clasificación como para detección de objetos se realiza una partición de datos en 80-10-10, que corresponde a datos de entrenamiento, prueba y validación, respectivamente. En otras palabras, es el porcentaje de datos utilizados, el 80% de los datos totales corresponde a los datos de entrenamiento, un 10% es asignado a los datos de prueba y el 10% restante a datos de validación. La partición de datos se realiza de forma aleatoria, es decir que sí se realizan n número de entrenamientos, los datos asignados a cada grupo siempre serán distintos.

La partición de datos de forma aleatoria se realiza con el fin de que se entreguen métricas confiables al finalizar un entrenamiento, ya que al no hacerse esto, las métricas obtenidas podrían resultar engañosas, dado que se utilizarían los mismos datos siempre.

El conjunto de datos de entrenamiento es el más grande de los tres conjuntos, y se utiliza para entrenar los parámetros del modelo minimizando la función de pérdida. La función de pérdida mide qué tan bien el modelo predice la salida correcta.

Los datos de validación se utilizan para evaluar el modelo durante el entrenamiento para tomar decisiones sobre hiperparámetros, como la tasa de aprendizaje, el número de capas y el tamaño del lote. La pérdida del conjunto de datos de validación se utiliza para determinar si el modelo está sobre ajustado o sub ajustado. Si el rendimiento del modelo en el conjunto de validación no mejora después de un cierto número de épocas, el proceso de entrenamiento se puede detener antes para evitar el sobreajuste.

El conjunto de datos de pruebas se utiliza para evaluar el rendimiento final del modelo después de que haya sido entrenado y ajustado. Los datos de prueba deben ser un conjunto de datos completamente independiente que el modelo no haya visto durante el entrenamiento o la validación. El rendimiento del modelo en el conjunto de datos de prueba es una estimación imparcial de qué tan bien generaliza el modelo a datos nuevos e invisibles.

Aunado a esto, la partición de datos de forma aleatoria también es utilizada generalmente para la validación por medio del método de K-Fold Cross Validation, donde se puede comprobar el correcto entrenamiento de la red neuronal.

3.2.6.2 Selección de arquitectura

Los modelos pre entrenados en Aprendizaje Profundo son aquellos que fueron entrenados con bases de datos sumamente grandes. Generalmente, estos modelos aprenden patrones complejos y características que pueden ser utilizadas para múltiples tareas sin la intervención de una programación compleja. Dichos modelos pueden ser utilizados en tareas como el reconocimiento de imágenes, procesamiento del lenguaje natural, entre otras.

Para utilizar los modelos pre entrenados, la mayoría de las ocasiones se recurre al Transfer Learning, que en pocas palabras es la utilización del conocimiento generado en los entrenamientos y que comúnmente se conocen como pesos.

Entre los modelos pre entrenados más comunes y utilizados, se encuentran VGG16, ResNet e Inception, todos ellos entrenados con la base de datos ImageNet que contiene millones de imágenes para la tarea de clasificación de imágenes.

En el caso de estudio de la clasificación de frutas y verduras, se recurrió a la utilización de los modelos InceptionV3, MobileNetV1 y MobileNetV2. En total se realizaron 5 entrenamientos distintos, en los cuales se utilizaba un modelo pre entrenado y una configuración de hiperparámetros.

Los modelos pre entrenados de Inception corresponden a una arquitectura de red neuronal convolucional creada por Google, cuyo atributo es capturar y procesar información a múltiples escalas y resoluciones de manera simultánea, por lo que se permite una extracción de características efectiva y eficiente. Su principio de funcionamiento es realizar convoluciones de forma paralela con variaciones en los

tamaños de los filtros de los datos de entrada, lo que conlleva a extraer características en distintas escalas espaciales y con esto permitiendo al modelo aprender datos detallados y abstractos complejos. El funcionamiento de la arquitectura es como sigue:

1. Capas de entrada: Las capas de inicio corresponden a capas de convolución y de agrupación (Pooling) que se encargan de extraer características, como son la detección de bordes o equinas.
2. Módulos Inception: Están compuestos por ramas paralelas y cada una realiza una operación de convolución distinta, las cuales extraen características a diferentes escalas.
3. Clasificadores auxiliares: Generalmente este tipo de clasificación se realiza en etapas intermedias de la red y consisten en capas convolucionales que están completamente conectadas. Estas ayudan a generar una mejor estabilidad en el entrenamiento a través de la reducción del desvanecimiento del gradiente.
4. Capa de agrupamiento: En esta última etapa, se reducen los tamaños de los mapas de características, para obtenerlos en un tamaño general o fijo.
5. Capas completamente conectadas: Son las capas finales que se encargan de realizar la clasificación.

Para realizar un primer entrenamiento, se utilizó la arquitectura de InceptionV3, es decir la tercera versión, la cual presenta mejoras en rendimiento con respecto a las versiones anteriores.

En posteriores entrenamientos, se recurrió a los modelos de MobileNet en sus versiones uno y dos. Los modelos MobileNet son arquitecturas de redes neuronales convolucionales, diseñadas con fines para tareas relacionadas con visión por computadora, específicamente en dispositivos móviles y que cuentan con recursos computacionales limitados, por lo que su entrenamiento resulta de forma rápida.

La principal característica de MobileNet es la capacidad de obtener una alta eficiencia en cómputo y bajo consumo de recursos, pero no se sacrifican las métricas a evaluar. La técnica que se utiliza para obtener esto es conocida como Depthwise separable convolution. El funcionamiento de esta técnica es factorizando las operaciones de convolución en dos pasos, siendo una convolución de profundidad y una convolución punto a punto. Al realizar esto en dos etapas, se reduce considerablemente el costo computacional, permitiendo ser una red más liviana y con tiempos de inferencia acelerados.

Como se menciona anteriormente, las versiones del modelo empeladas son uno y dos por lo que sus mejoras tienen que ver con su programación interna, hablando específicamente de los procesos y las capas de la red. Ambas versiones utilizan convoluciones separables en profundidad, pero MobileNetV1 la aplica en cada capa,

mientras que MobileNetV2 aplica una versión mejorada conocida como Inverted residual, lo que genera el uso de bloques residuales para extraer características más complejas. Además, MobileNetv2 añade capas de atención a escala múltiple para extraer características a distinta escala y niveles de detalle. El funcionamiento de la red MobileNetV2 se describe a continuación:

1. Capas de entrada: Las capas de inicio corresponden a convoluciones y extracción de características.
2. Bloques residuales: Su función es permitir que la información no se pierda entre las capas y ayudan a combatir el problema del desvanecimiento del gradiente.
3. Convoluciones separables en profundidad: Dividen la operación de convolución estándar en dos pasos, lo que reduce la cantidad de operaciones y parámetros necesarios.
4. Capas de atención a escala múltiple: Se encargan de extraer las características de distinto nivel de complejidad por medio de módulos de expansión y contracción. Estas capas permiten que el modelo focalice en las características importantes.
5. Regularización: Se incluyen técnicas tales como regularización tipo L2 y regularización Dropout, que ayudan a reducir la complejidad del modelo y mejorar la capacidad de generalización.
6. Capa de clasificación: Es la capa final de la red que se encuentra totalmente conectada, toma las características de las capas anteriores y las mapea a las clases de salida correspondientes.

Para el modelo de clasificación de frutas y verduras, las capas de salida sufren modificaciones para determinar el resultado correcto, ya que al realizar aprendizaje por transferencia, las últimas capas y la capa de salida deben adaptarse a la naturaleza del problema a resolver. Las configuraciones realizadas dieron como resultado la adición de dos capas densas anteriores a la salida con la función de activación de ReLu y una capa densa como salida con el número de clases existentes con la función de activación de Softmax, lo que resultó en el modelo de entrenamiento final. La capa de salida densa contiene el número de clases, que para este caso es de 36.

El modelo para la determinación del estado de madurez presenta una configuración como el modelo de clasificación de frutas y verduras, manteniendo un aprendizaje de transferencia con los pesos de MobileNetV2, pero con modificaciones en las últimas capas y la salida. La antepenúltima capa es una normalización por lotes para normalizar el tamaño y ayudar a la salida, añadiendo como penúltima

capa, con un regularizador de kernel tipo l2, regularizador de activación l1 y un regularizador bias l1 con función de activación Relu. Cabe destacar que existe una capa antes de llegar a la capa de salida en la que las neuronas se apagan con un Dropout y así ayudan en el entrenamiento de la red. Finalmente, se presenta una capa de salida densa con el número de clases, que para este caso es 12, ya que las 6 clases de frutas y verduras tienen cada una dos estados de madurez.

3.2.6.3 Aumento de datos

Se recurrió al aumento de datos para expandir la base de datos y lograr que el rendimiento de los modelos mejorara. El aumento de datos realizado para la clasificación de imágenes se realiza para los datos de entrenamiento, validación y prueba. En el Cuadro 4 se muestran la configuración para cada uno de los conjuntos de datos.

Cuadro 4. Configuración para aumento de datos

Datos	Shear range	Color mode	Horizontal flip	Rotation range	Zoom range
Entrenamiento	0.2	RGB	Si	30	0.15
Validación	0.2	RGB	Si	30	0.15
Prueba	0.2	RGB	Si	No	No

Donde:

- Shear range es la intensidad de corte o ángulo de corte, generalmente en sentido contrario a las manecillas del reloj en grados.
- Color mode corresponde al espacio de color utilizado.
- Horizontal flip gira aleatoriamente las imágenes horizontalmente.
- Rotation range es el rango de grados para rotaciones de forma aleatoria.
- Zoom range genera un rango de acercamiento aleatorio.

Para el caso de estudio de la detección de objetos también se realiza un aumento de datos pero de forma lineal, es decir que sólo se realizan cambios en espacios de color, incorporación de ruido, entre otros. Los cambios en espacios de color para el aumento de datos son: Blur, Median Blur, Gray y CLAHE. Para cada una de estas configuraciones, los valores fueron asignados de forma automática por el algoritmo utilizado para realizar la detección de objetos, el cual corresponde a YOLO (You Only Look Once).

3.2.6.4 Configuración de hiperparámetros

Es de suma importancia ajustar los hiperparámetros para que el rendimiento del modelo sea satisfactorio. Desde luego, existen técnicas de optimización para que estos se ajusten de forma automática, pero en el caso de la investigación se realizó de forma manual y se iban configurando, dependiendo el resultado del anterior entrenamiento. Los hiperparámetros utilizados para el caso de clasificación de imágenes y una pequeña descripción de su funcionamiento se muestran a continuación:

- Arquitectura: Corresponde al modelo empleado para realizar el entrenamiento de la red neuronal, el número de capas ocultas, número de neuronas o unidades, tipo de capas, tamaño del kernel, entre otros.

  Para el caso de de estudio, sólo se agregaron capas en la salida, tal como se menciona en la sección de selección de arquitectura, ya que se emplearon modelos previamente diseñados. El tamaño del kernel escalecido para todos los experimentos fue de 3x3.
- Optimizador: Es el algoritmo que se utiliza para ajustar los parámetros del modelo durante el entrenamiento. En este caso, se utilizaron optimizadores Adam y Adamax.
- Learning rate: O tasa de aprendizaje, es el porcentaje al cual el modelo ajusta los pesos durante el entrenamiento. Para todos los experimentos, no se cambió la tasa de aprendizaje que venía por default con cada optimizador, el cual corresponde a 0.001.
- Loss function: O función de pérdida es una medida para evaluar qué tan bien un modelo generaliza, se calcula la diferencia entre las salidas predichas por el modelo y las salidas reales. En los experimentos se utiliza la función de pérdida Categorical Crossentropy, ya que aplica para problemas de clasificación multiclase.
- Batch size: O tamaño de lote, es el número de datos de entrada que se toman en cada iteración del proceso de optimización. Para el caso, se realizan expermientos con un tamaño de 32 o 16.
- Función de activación: Esta función ayuda a que el modelo no presente una linealidad y en este caso, para todos los experimentos se utiliza la función de activación ReLu.

En el Cuadro 5 se muestran las configuraciones de hiperparámetros para cada entrenamiento realizado para el caso de la clasificación de frutas y verduras.

Cuadro 5. Configuración de hiperparámetros clasificación de frutas y verduras

No.	Arquitectura	Optimizador	Tasa aprendizaje	Función pérdida	Tamaño lote
1	InceptionV2	Adam	0.001	Categorical Crossentropy	32
2	MobileNetV2	Adamax	0.001	Categorical Crossentropy	32
3	MobileNetV1	Adam	0.001	Categorical Crossentropy	32
4	MobileNetV2	Adamax	0.001	Categorical Crossentropy	16
5	InceptionV3	Adam	0.001	Categorical Crossentropy	16
6	MobileNetV2	Adam	0.001	Categorical Crossentropy	32

En Cuadro 6 puede apreciarse la configuración de hiperparámetros para cada entrenamiento, específicamente para la clasificación de estado de maduración.

Cuadro 6. Configuración hiperparámetros clasificación estados de maduración

No.	Arquitectura	Optimizador	Tasa aprendizaje	Función pérdida	Tamaño lote
1	InceptionV2	Adamax	0.001	Categorical Crossentropy	32
2	MobileNetV2	Adamax	0.001	Categorical Crossentropy	32
3	MobileNetV2	Adam	0.001	Categorical Crossentropy	32

Como se comenta en secciones anteriores, se implementa un caso de estudio que corresponde a la detección de objetos utilizando el modelo YOLO. El modelo cuenta con la mayoría de sus hiperparámetros asignados, pero se pueden modificar de acuerdo con la naturaleza del problema. En el Cuadro 7 se muestran las configuraciones para cada entrenamiento, señalando que para el experimento 1 se utiliza la versión 7 del modelo YOLO y para el restante la versión 8.

Cuadro 7. Configuración de hiperparámetros en detección de objetos

No.	Arquitectura	Pesos	Optimizador	Tasa aprendizaje	Tamaño lote
1	YOLOv7	yolov7E6	SGD	0.01	32
2	YOLOv8	yolov8m	SGD	0.01	32
3	YOLOv8	yolov8m	Adam	0.001	32
4	YOLOv8	yolov8m	SGD	0.01	16
5	YOLOv8	yolov8m	Adam	0.001	16
6	YOLOv8	yolov8x	SGD	0.01	16

3.2.7 Validación

En todo modelo se producen cuatro tipos de predicciones, con las cuales se pueden determinar las métricas de evaluación del modelo.

Sí la predicción positiva de una clase es correcta, se produce un verdadero positivo o true positive (TP), sí la predicción de un negativo es correcta, se genera un verdadero negativo o true negative (TN). Además, sí se predice un positivo en una clase negativa, se produce un falso positivo o false positive (FP), sí se predice un negativo cuando la clase es positiva, se obtiene un falso negativo o false negative (FN).

Para realizar la validación de los entrenamientos, es necesario recurrir a las métricas de evaluación utilizadas en Aprendizaje profundo. Las métricas empleadas son exactitud (accuracy), precisión (presicion), sensibilidad (recall), puntaje F1 (F1-score) y mean average precision (mAP), que aplica para detección de objetos. Cada una de estas métricas para medir la fiabilidad del modelo, se mencionan a continuación:

- Exactitud (Accuracy): Calcula el porcentaje de predicciones correctas contra el total de predicciones realizadas, sin embargo no es la más recomendable cuando se encuentra un desbalanceo de clases.

$$Exactitud = \frac{TP + TN}{TP + TN + FP + FN}$$

- Precisión (Precision): Mide la proporción de predicciones positivas verdaderas sobre todas las predicciones positivas, en otras palabras la capacidad del modelo para evitar falsos positivos. Es importante contar pocos falsos

positivos, ya que no se quiere una predicción errónea, tanto como en predicción de fruta y verdura, así como en su estado de maduración.

$$Presición = \frac{TP}{TP + FP}$$

- Sensibilidad (Recall): Genera el porcentaje de casos positivos que fueron correctamente clasificados por el modelo, en otras palabras identifica de todos los positivos reales el porcentaje de estos que puede ser clasificado correctamente por el modelo. Generalmente se utiliza cuando es necesario contar con una cantidad menor de falsos negativos.

$$Sensibilidad = \frac{TP}{TP + FN}$$

- Puntaje F1 (F1-score): Es la medida entre la precisión y la sensibilidad ponderada armónicamente.

$$Puntaje\ F1 = \frac{2TP}{2TP + FP + FN}$$

- Precisión promedio media (Mean average precision, mAP): Esta métrica considera la precisión promedio, que corresponde a AP para cada clase de objeto y luego calcula el promedio de todos los AP de las clases. El cálculo implica la contrucción de la curva de presición vs. sensibilidad para cada una de las clases. La precisión se calcula en varios puntos de recuperación, mientras se varía el umbral que determina sí una detección es considerada como verdadero positivo o falso positivo, enseguida se calcula el área debajo de la curva, que representa el AP para una clase en específico. El mAP se obtiene del promedio de los AP.

$$mAP = \frac{1}{n}\sum_{k=1}^{n} AP_k$$

Donde, n es el número de clases totales y k la clase actual.

$$AP = \sum \frac{TP}{Recall} * (Recall_i - Recall_{i-1})$$

Para el caso de los modelos de clasificación, se obtienen las métricas de exactitud, precisión, sensibilidad y puntuación F1, mientras que para el caso de la detección de objetos se calculan las métricas de precisión, sensibilidad, mean Average Precision en un nivel de 50% de confianza (mAP@50) y mean Average Precision en un nivel de entre 50-95% de confianza (mAP@50-95).

CAPÍTULO IV. RESULTADOS

En este capítulo se presentan los resultados experimentales obtenidos en los distintos entrenamientos de los modelos, tanto para clasificación de imágenes, como para detección de objetos. Además, se muestran resultados con respecto a la inferencia de datos con los mejores modelos y la implementación en hardware.

4.1 RESULTADOS EXPERIMENTALES

Existen varias formas sobre cómo visualizar los resultados obtenidos después del entrenamiento de un modelo. Una forma es a través de las métricas alcanzadas por el modelo, en otras palabras el rendimiento obtenido. En la sección anterior se mencionan las métricas y cómo obtenerlas, tanto para los algoritmos de clasificación, como para el algoritmo de detección de objetos.

En Machine Learning existen dos tipos sobre cómo presentar las métricas alcanzadas cuando se refiere a una clasificación multiclase. La primera se conoce como macro promedio y corresponde a que cada clase tiene el mismo peso, en otras palabras que todas las clases contribuyen de manera igual al promedio final. La segunda se conoce como promedio ponderado y toma en cuenta la contribución de cada clase ponderada por el número de ejemplos cada clase, se utiliza cuando existe un desequilibrio en el conjunto de datos.

Para el caso de la clasificación de frutas y verduras, se utiliza la media ponderada, ya que existe un pequeño desbalance en las clases. En concreto, se realizaron 6 experimentos, con 36 clases cada uno, de los cuales los resultados pueden apreciarse en el Cuadro 8.

Cuadro 8. Métricas de entrenamientos de clasificación de frutas y verduras

No.	Arquitectura	Épocas	Tiempo ejecución	Exactitud	Precisión	Sensibilidad	Puntaje F1
1	InceptionV2	24	112.7 min	87.01%	88%	87%	87%
2	MobileNetV2	5	87.5 min	95.72%	97%	96%	96%
3	MobileNetV1	36	50.2 min	96.11%	97%	96%	96%
4	MobileNetV2	50	106 min	94.20%	95%	94%	94%
5	InceptionV3	23	75.5 min	96.84%	97%	97%	97%
6	**MobileNetV2**	**7**	**17.9 min**	**97.86%**	**98%**	**98%**	**98%**

Se realizaron un total de tres experimentos para el caso de la clasificación de estados de maduración, con un total de 12 estados de maduración, correspondientes a 6 clases de frutas o verduras. Las métricas alcanzadas se muestran en el Cuadro 9.

Cuadro 9. Métricas de entrenamientos clasificación de estados de maduración

No.	Arquitectura	Épocas	Tiempo ejecución	Exactitud	Precisión	Sensibilidad	Puntaje F1
1	InceptionV2	9	6.06 min	95.10%	95%	95%	95%
2	MobileNetV2	103	90.20 min	95.69%	96%	95%	95%
3	**MobileNetV2**	**12**	**29.61 min**	**96.67%**	**97%**	**97%**	**97%**

Como se menciona en secciones anteriores, se utilizan los métodos de clasificación de imágenes y detección de objetos, para lo cual se recurre a la implementación de los modelos YOLOv7 y YOLOv8, de los cuales se muestran los resultados en el Cuadro 10.

Cuadro 10. Métricas obtenidas en entrenamientos de detección de objetos

No.	Arq.	Épocas	Tiempo ejecución (hrs)	Exactitud	Precisión	Sensibilidad	Puntaje F1
1	YOLOv7	300	14.32	0.711	0.764	0.718	0.606
2	YOLOv8	24	2.47	0.726	0.754	0.749	0.619
3	YOLOv8	100	9.83	0.728	0.679	0.723	0.608
4	YOLOv8	64	6.55	0.711	0.749	0.746	0.627
5	YOLOv8	105	12.04	0.707	0.76	0.751	0.627
6	**YOLOv8**	**78**	**9.23**	**0.724**	**0.741**	**0.752**	**0.634**

4.2 INFERENCIA CON DATOS

Para corroborar que el sistema funciona correctamente y determinar donde pueden existir fallas o errores durante el entrenamiento, interviene la inferencia con datos. Este método consiste en mandar llamar los pesos del modelo y realizar una predicción con datos reales y que el sistema no ha visto o no se tomaron en cuenta

para el entrenamiento. De esta forma se corrobora que el sistema generalizó de manera correcta y que es capaz de predecir sobre cualquier ambiente en el que sea sometido. En la Figura 6 se muestran 6 inferencias realizadas con los modelos de clasificación, tanto para frutas y verduras, así como para definir el estado de maduración. Primeramente se despliega el resultado de la predicción de la fruta o verdura y sí esta cuenta con la asignación de estado de maduración, se predice arrojando en qué etapa de maduración se encuentra. Sí es caso contrario, sólo se despliega en la parte inferior de la imagen el tipo de fruta o verdura.

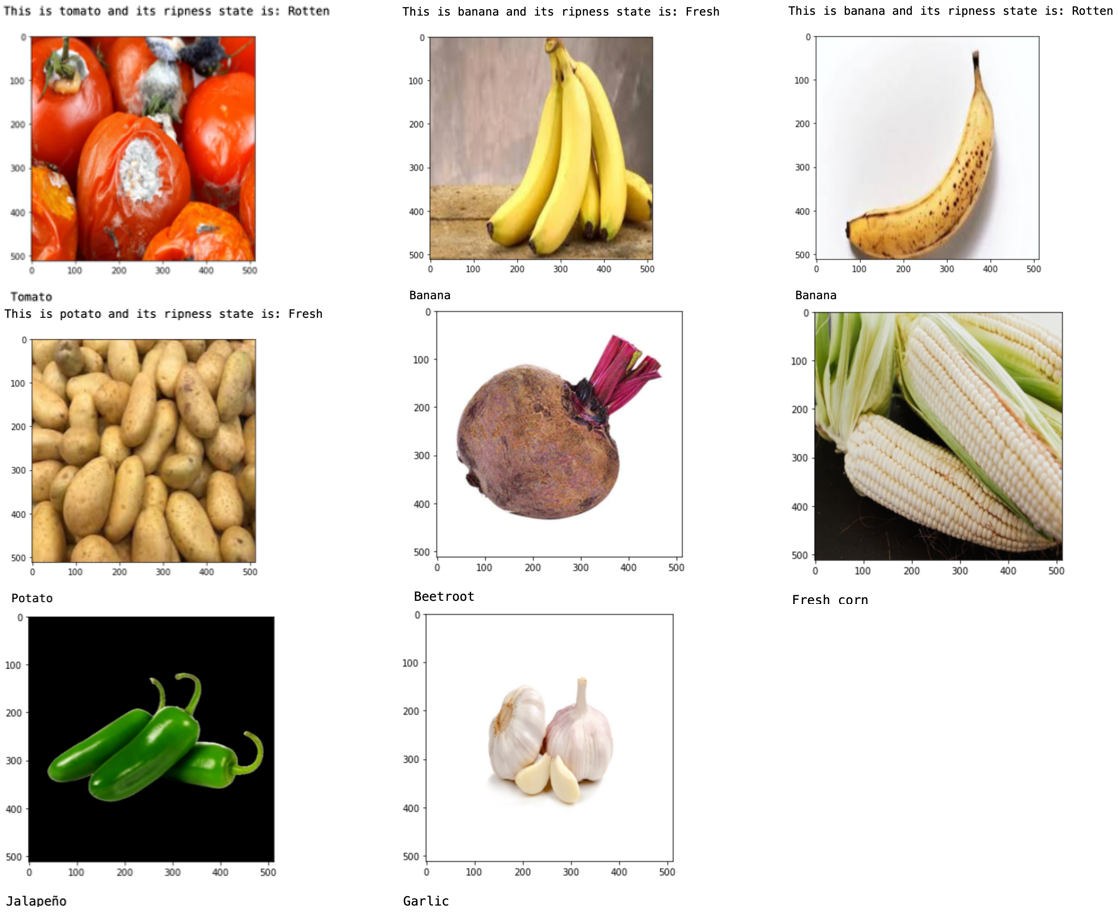
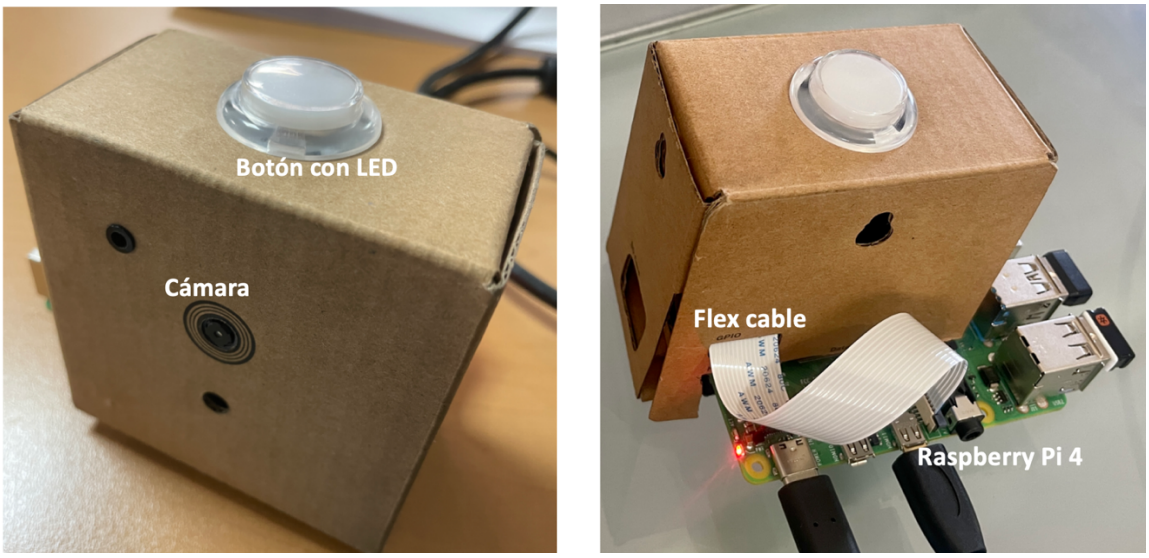


Figura 6. Resultados de inferencia con datos

4.3 IMPLEMENTACIÓN EN HARDWARE

Como parte de la implementación del software en el hardware, se realizaron modificaciones del kit AIY Vision de Google, el cual en su estructura inicial cuenta con una serie de componentes que se mencionan a continuación: Vision Bonnet, Raspberry Pi Zero, Raspberry Pi Camera V2, botón, arnés, buzzer, LED, microSD, caja de cámara, entre otros.

Por razones de compatibilidad entre el software requerido de la Raspberry Pi Zero y el modelo de YOLO v8, no fue posible implementarse en la tarjeta antes mencionada. Para realizar esto, se requirió de la tarjeta de adquisición de datos Raspberry Pi 4 con la configuración más alta disponible, además se realiza un acondicionamiento del material con el que contaba en un inicio el kit, solamente utilizando la cámara, botón y microSD. En la Figura 7 se muestra el prototipo funcional.



a) Vista frontal a) Vista posterior

Figura 7. Prototipo final

Cabe señalar que el dispositivo debe estar conectado a una pantalla externa a través de una conexión micro HDMI para que la información pueda ser visualizada, así como la conexión de dispositivos como un ratón y un teclado para manipular el software.

El funcionamiento del prototipo es como se enlista a continuación:

1. Conectar dispositivos y encender tarjeta electrónica
2. Ejecutar el código de programación
3. Enfocar la cámara hacia el objetivo y presionar botón para realizar captura
4. Esperar tiempo de procesamiento de datos
5. Despliegue de resultados

Con respecto al tiempo de inferencia de datos, el tiempo ronda los 10 segundos, aproximadamente; esto desde que se presiona el botón para capturar la imagen, hasta que se despliega el resultado.

CAPÍTULO V. DISCUSIÓN DE RESULTADOS

En este capítulo se discute el rendimiento del modelo a través de los resultados obtenidos. Se analizan las métricas alcanzadas, las matrices de confusión de los mejores modelos, así como las gráficas generadas a través de los entrenamientos. Se muestran los resultados del modelo con el mejor rendimiento para cada tarea, siendo para clasificación de imágenes los modelos de clasificación de frutas y verduras, así como la predicción de estados de maduración y para detección de objetos el modelo general.

5.1 MÉTODO 1: CLASIFICACIÓN DE IMÁGENES

5.1.1 Clasificación de frutas y verduras

Las métricas obtenidas por el mejor modelo de clasificación de frutas y verduras son como se muestran a continuación:

- Exactitud (Accuracy) = 97.86%
- Precisión (Precision) = 98%
- Sensibilidad (Recall) = 98%
- Puntaje F1(F1-score) = 98%

A raíz de los resultados anteriores, se puede decir que el modelo clasifica correctamente 97.86% de los datos de prueba, en cuanto a la precisión, el 98% de las instancias que fueron clasificadas como positivas son actualmente positivas, con respecto a la sensibilidad, el 98% de las instancias positivas fueron clasificadas como positivas y el puntaje F1, muestra que el rendimiento en general del modelo es del 98%.

Además, con respecto a las métricas se determina que el modelo es muy apto en la correcta clasificación para los datos de prueba, en la identificación de instancias positivas, identificando todas las instancias positivas y otorga un buen rendimiento en general.

De los problemas más comunes en el área de Machine Learning y Deep Learning, con respecto al entrenamiento, se conocen como sobre ajuste (overfitting) o sub ajuste (under fitting). Este tipo de problemas sólo se pueden identificar visualizando las gráficas de exactitud y pérdida, generadas durante el entrenamiento del modelo.

El sobre ajuste se produce cuando el modelo aprende de forma exacta los datos de entrada y no es apto para realizar una generalización con datos nuevos; en otras palabras replica la información, en lugar de aprenderla.

El sub ajuste ocurre cuando el modelo no aprende a partir de los datos de entrada, lo que produce que no sea apto para realizar predicciones acertadas.

Ambos fenómenos se pueden corregir observando la complejidad del modelo, analizando a detalle los datos de entrada, agregando o quitando técnicas de regularización, entre otros. Sin duda alguna todo esto representa un reto, ya que se debe ir haciendo de forma manual hasta evitar que se produzcan los fenómenos antes mencionados, sí es que se producen.

Una técnica para evitar estos fenómenos es implementar un Early Stopping, que en otras palabras significa detener el entrenamiento del modelo en determinado momento, sí es que la exactitud o la pérdida no presentan cambios. Se puede monitorear la exactitud de los datos de entrenamiento o validación, así como la pérdida en datos de entrenamiento o validación. Generalmente se recomienda monitorear la pérdida en los datos de validación durante el rendimiento.

Para este criterio de paro se debe configurar un parámetro de paciencia, el cual es el número de épocas en el que el rendimiento del modelo debe mejorar en el conjunto de datos antes de que se detenga el entrenamiento. Es decir, sí la paciencia equivale a dos épocas, sí el modelo al paso de dos épocas no presenta mejora se detiene y guarda el modelo antes de las dos últimas épocas.

En el caso de estudio, para el modelo de clasificación de frutas y verduras, se configuró una paciencia de dos épocas. El modelo requirió un total de 7 épocas, pero el mejor rendimiento lo alcanzó en la época 5. En la Figura 8 se muestra el rendimiento del modelo con respecto a la exactitud.

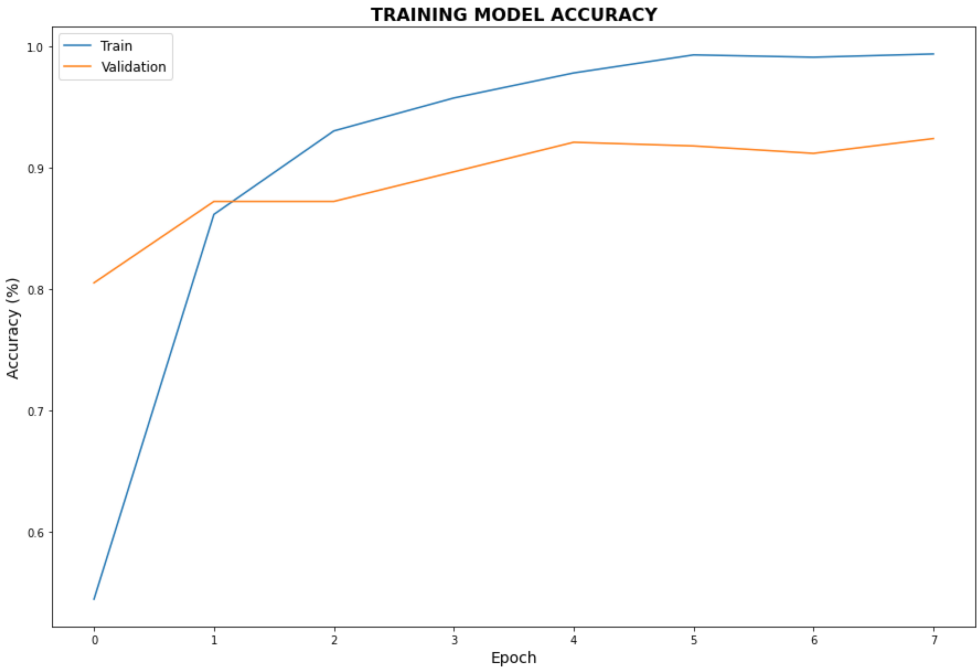


Figura 8. Gráfica de exactitud modelo de clasificación de frutas y verduras

Cabe señalar que en la figura anterior pude observarse que después de la época 5, tanto la curva de entrenamiento como la de validación, siguen un patrón de incremento en cuanto al nivel de exactitud. Esto se puede interpretar como que el modelo sigue aprendiendo, pero al analizar la gráfica de pérdida, los datos confirman lo contrario. En la Figura 9 se observa la gráfica de pérdida producida durante el entrenamiento del modelo.

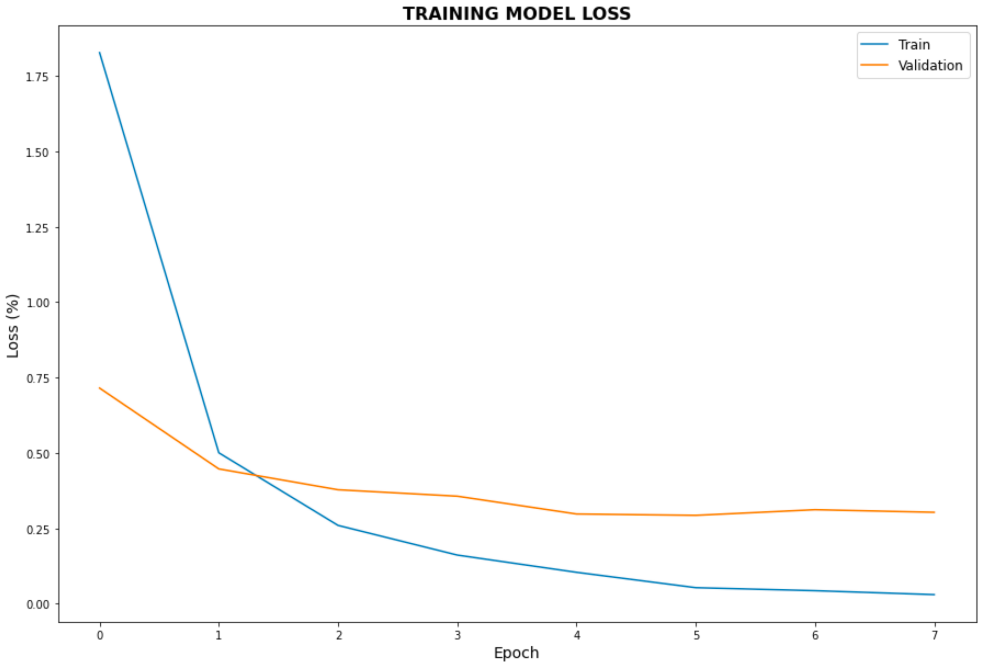


Figura 9. Gráfica de pérdida modelo de clasificación de frutas y verduras

Lo mostrado en la figura anterior, muestra como a partir de la época 5, la curva de entrenamiento sigue decreciendo, pero caso contrario ocurre con la curva de validación, lo que demuestra por qué el modelo se detuvo antes de entrar en un sobre ajuste.

Otro método para validar los modelos es a través de matrices de confusión. Una matriz de confusión es una tabla utilizada para medir el rendimiento de modelos de clasificación, así como para detectar las clases donde el modelo no generaliza de forma correcta. En la Figura 10 se muestra la matriz de confusión generada para la clasificación de frutas y verduras.

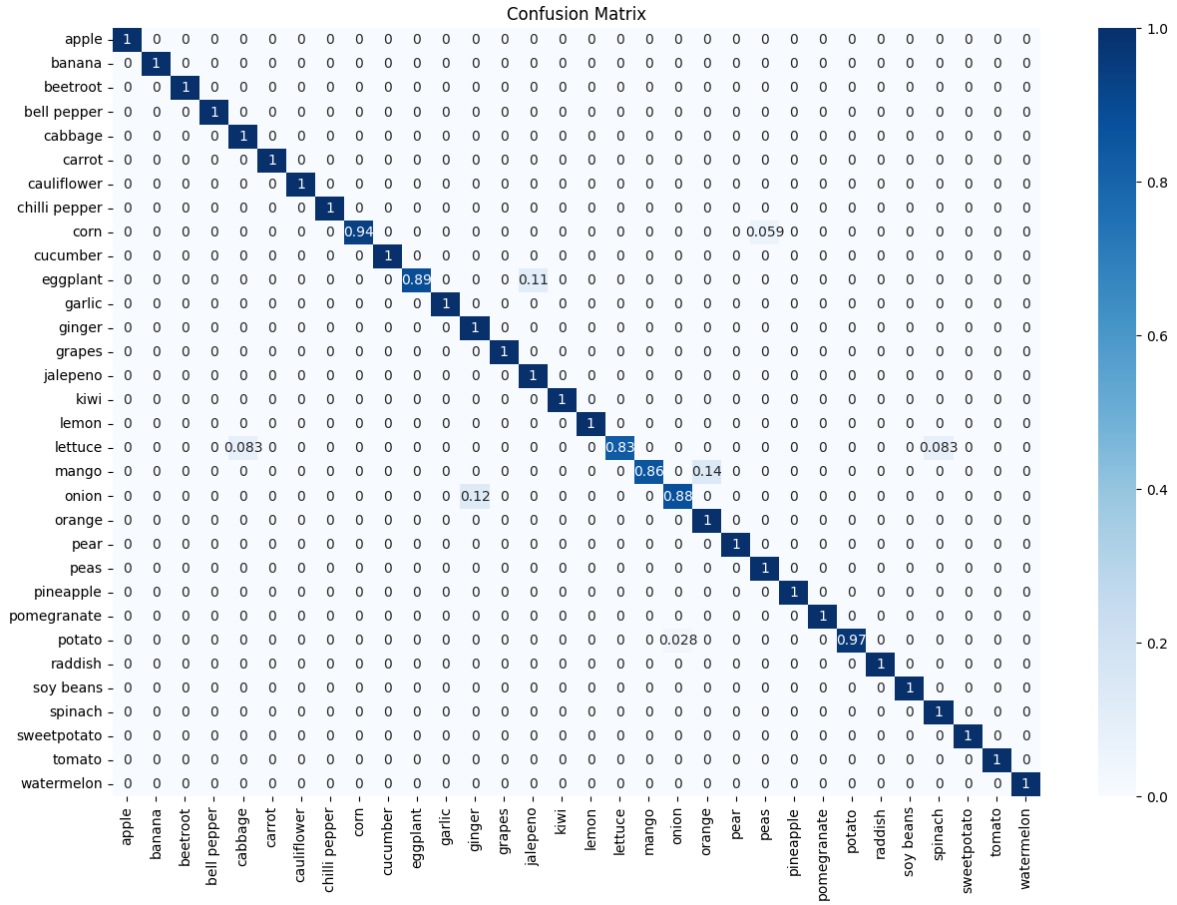


Figura 10. Matriz de confusión modelo de clasificación de frutas y verduras

En una matriz de confusión, el eje horizontal representa los valores verdaderos y el eje vertical los valores de predicción. Para el caso mostrado en la figura anterior, todas las clases están normalizadas en un rango de cero a uno, donde uno representa el 100% de predicciones correctas para la clase, esto se puede observar en la diagonal principal de la matriz, donde lo ideal sería que todos los elementos tuvieran asignado un uno.

Como puede observarse, la clasificación se realiza adecuadamente para la mayoría de las clases a reserva de 6 de ellas, donde puede deberse a la confusión entre clases, esto debido a la similitud de algunas frutas y verduras.

5.1.2 Clasificación de estados de maduración

El rendimiento del mejor modelo de estados de maduración es como se muestran a continuación:

- Exactitud (Accuracy) = 96.66%
- Precisión (Precision) = 97%
- Sensibilidad (Recall) = 97%
- Puntaje F1(F1-score) = 97%

Con las métricas del modelo mostradas anteriormente, es probable que el modelo funcione de forma similar en el siguiente conjunto de datos, además que el modelo sea capaz de generalizar ante la entrada de nuevos datos y es posible que pueda realizar predicciones precisas.

Además, el 96.66% de exactitud indica que la mayoría de las predicciones realizadas por el modelo son correctas, con la precisión se asume que cuando el modelo predice un resultado positivo lo hace el 97% de las veces correctamente, la sensibilidad indica que el modelo identifica correctamente el 97% de los casos reales positivos y con el puntaje obtenido de 97% se indica un buen balance entre la precisión y la sensibilidad, lo que sugiere que el rendimiento del modelo es bueno.

En todos los casos, siempre es importante considerar la distribución de clases, ya que esto ayuda a una mejora comprensión de las métricas de rendimiento obtenidas.

Al igual que en el modelo anterior, como criterio de paro se utiliza una paciencia de dos épocas, monitoreando la pérdida con respecto a los datos de validación, por lo que el modelo guardado es el generado en la época 10. Dicha información se puede apreciar en la Figura 11.

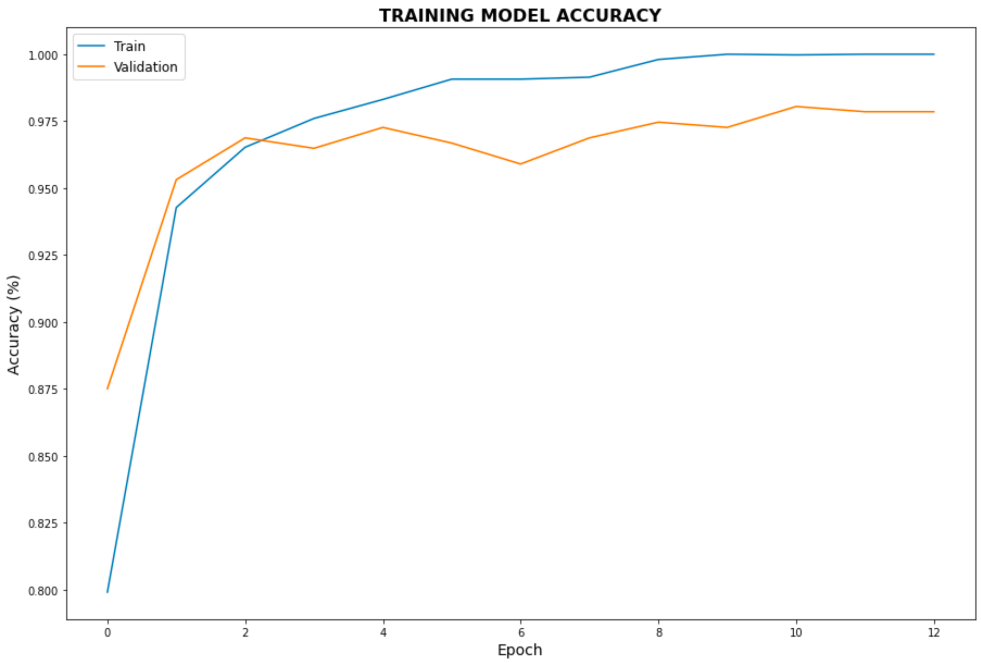


Figura 11. Gráfica de exactitud modelo de estados de maduración

En la figura anterior se muestra como el entrenamiento del modelo, tanto en datos de entrenamiento y validación aumentan con el paso de las épocas, pero al llegar a la época 10 se mantiene sin mejora, lo que podría producir un sobre ajuste, es por ello que el modelo termina el proceso de entrenamiento en la época 12.

En la Figura 12 se aprecia la gráfica de pérdida producida a lo largo de los entrenamientos, de la cual se puede asumir y dado que se configuró un Early Stopping de 2 épocas, como el modelo dejó de entrenar en la época 10. Se aprecia gráficamente como las curvas de entrenamiento y validación a partir de la época antes mencionada no generan una mejora, sino al contrario se observa que la curva de validación aumenta en lugar de decrecer y la curva de entrenamiento se mantiene constante, lo que se interpreta como que el modelo dejó de aprender y podría desencadenar en un sobre aprendizaje. El modelo final se guardó con los datos producidos hasta la época 10.

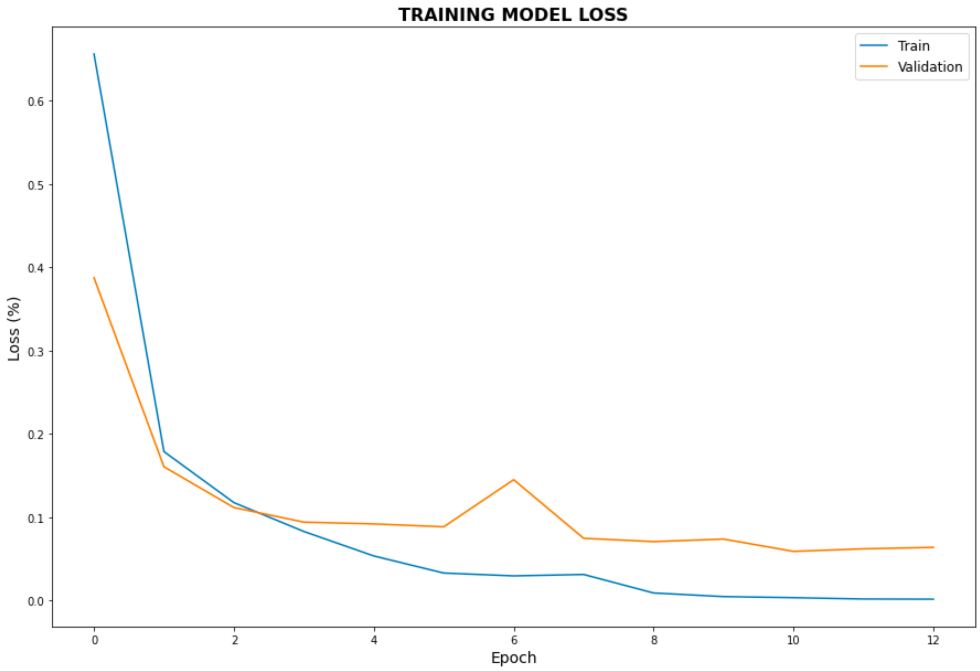


Figura 12. Gráfica de pérdida modelo de estados de maduración

El otro método que se menciona anteriormente para analizar sí el entrenamiento de una red neuronal artificial se llevó a cabo de forma correcta, así como identificar las áreas de oportunidad para mejorar el rendimiento del modelo, es mediante el análisis de la matriz de confusión.

Para el caso en concreto del modelo para clasificación de estados de maduración, la matriz de confusión se puede observar en la Figura 13.

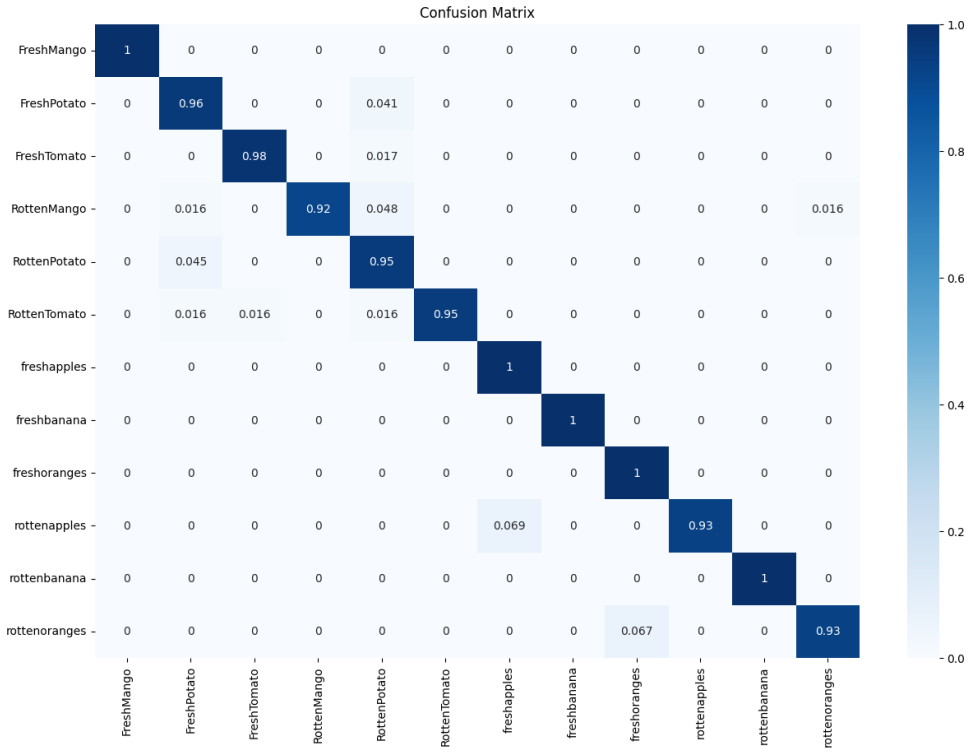


Figura 13. Matriz de confusión modelo de estados de maduración

En la figura anterior se observa en el eje horizontal los valores verdaderos o reales, mientras que en el eje vertical los valores de predicción. Realizando un análisis a detalle sobre la matriz de confusión generada por el modelo a partir del entrenamiento, es observa que del total de las 12 clases, 5 generan el 100% de las predicciones correctas; esto se determina observando la diagonal principal. El valor máximo que se puede determinar es uno y el meno es cero.

Lo ideal sería que en la diagonal principal se observaran sólo números uno y el tono del color más oscuro posible. Dado que esto no se produce, en 7 de las clases el modelo no generaliza de forma precisa, pero esas clases se encuentran por encima del 93% de correcta predicción, con lo que se puede asumir que el modelo predice de forma correcta.

Para corregir este tipo de problemas y que en la diagonal principal pueda presentarse el porcentaje más alto posible por clase, se podría observar en la base de datos e identificar sí existen datos que pudieran estar causando el problema o incluso cambiar la configuración de los hiperparámetos para el entrenamiento. Otra de las soluciones es realizar un aumento de datos más grande sobre la base de datos, incluso producir ruido sobre las imágenes para que el modelo generalice de mejor forma.

5.2 MÉTODO 2: DETECCIÓN DE OBJETOS

El segundo método implementado y como se comenta en secciones anteriores, trata sobre la detección de objetos. Para este caso en particular, el mejor resultado se obtuvo utilizando el algoritmo YOLO v8. A continuación se presentan las métricas del rendimiento del modelo:

- Precisión (Precision) = 72.4%
- Sensitividad (Recall) = 74.1%
- mAP@50 (Mean Average Precision) = 75.2%
- mAP@50-95 (Mean Average Precision) = 63.4%

Cabe señalar que en los algoritmos de detección de objetos, la métrica que más importa es Mean Average Precision (mAP). Esta métrica se divide en una detección a 50% de confianza o umbral de detección y de 50-95%. En otras palabras, mAP@50 representa la capacidad del modelo para detectar objetos grandes y mAP@50-95, la capacidad para detectar objetos pequeños.

Analizando las métricas obtenidas, de todos los objetos detectados por el modelo, el 72.4% son predicciones correctas, indicando que el modelo es capaz de minimizar los falsos positivos en una medida razonable. Se pueden identificar un 74.1% de todos los objetos positivos reales en las imágenes, lo que quiere decir que el modelo tiene una buena capacidad para detectar objetos. Con respecto a los umbrales de precisión, el 75.2% indica que el modelo fue capaz de obtener un buen nivel de precisión y sensitividad, así como una buena respuesta a localizar objetos de tamaño grande, mientras que el 63.4% indica que el rendimiento del modelo disminuye a medida que aumenta el umbral de IoU, por lo que se asume que el modelo pudiera tener algunas dificultades para localizar objetos pequeños.

En general, los algoritmos de detección de objetos, específicamente los de YOLO, generan una serie de gráficas al finalizar el entrenamiento para poder visualizar cómo se desarrolló el entrenamiento, así como analizar sí pudiera existir un sub o sobre ajuste. En la Figura 14 se muestran las gráficas del entrenamiento.

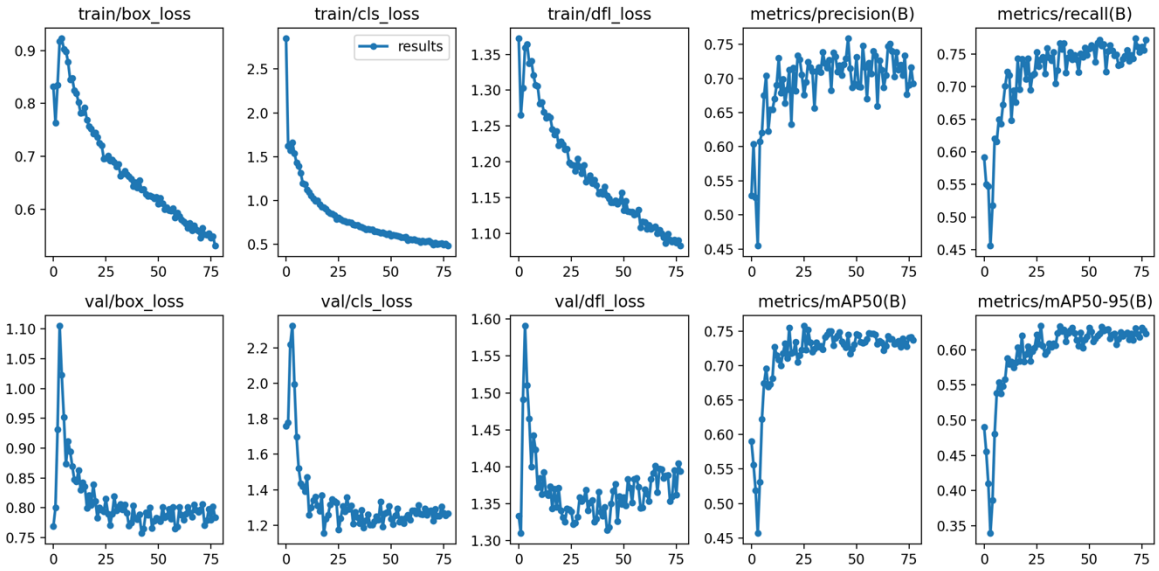


Figura 14. Gráfica detección de objetos frutas, verduras y estados de maduración

La figura anterior muestra en las tres primeras columnas, en la parte superior las curvas generadas para los datos de entrenamiento y en la parte inferior lo respectivo para los datos de validación. En las columnas 4 y 5 muestra en rendimiento en general del modelo.

En la primera columna, box_loss refiere a los valores de pérdida para las predicciones del cuadro delimitador que se producen en los datos de validación y entrenamiento, en otras palabras mide el error entre la predicción de cuadros delimitadores y los cuadros delimitadores reales. Para realizar un análisis sobre el gráfico antes mencionado, se debe observar la tendencia producida por las curvas, notando que este siempre debe ser de forma descendente conforma transcurren las épocas. Para el caso en específico, se observa la tendencia de disminución del error.

La segunda columna, cls_loss o class loss representa los valores de pérdida en las predicciones de clasificación para los datos de entrenamiento y validación, lo que quiere decir que mide el error entre las probabilidades de las clases predichas y las etiquetas de las clases verdaderas. La tendencia de este gráfico debe ser en forma descendente, así como se muestra para el caso de análisis.

Para la tercera columna dfl_loss o detection focal loss, muestran valores de pérdida para la detección de pérdida focal en los datos de entrenamiento o validación, lo que en otras palabras se puede interpretar como un tipo de función de pérdida que es usada en detección de objetos y fue diseñada para focalizar en ejemplos difíciles, los cuales son muestras donde el modelo puede cometer errores.

Para el caso de estos tres tipos de gráficas de pérdida mencionadas anteriormente, se puede identificar en particular para la gráfica val/dfl_loss, que se

localiza en la parte inferior de la tercera columna, cómo se produce un fenómeno donde la curva, a partir de la época 28 incrementa con el paso del tiempo. Este fenómeno se conoce como sobre ajuste u overfitting, que se ha mencionado en secciones más anteriores, por lo que el modelo se guardó en la época antes mencionada, con el fin de evitar el sobre ajuste y la no correcta generalización de este.

En el caso de las métricas restantes, como son precisión, sensitividad (recall), mAP@50 y mAP@50-95, la tendencia de las curvas debe presentarse de forma ascendente, caso contrario a las curvas de pérdida. Para este caso en particular, se observa dicha tendencia, pero a partir de la época 28, las curvas se mantienen prácticamente en línea recta con la horizontal, lo que quiere decir que el modelo ha dejado de entrenar y probablemente ha comenzado a replicar los datos, caso que no se quiere en un modelo de aprendizaje máquina.

Sumado a todo esto, es posible analizar el potencial del modelo a través de la matriz de confusión, la cual es mostrada en la Figura 15 indica todas las clases utilizadas en el entrenamiento del modelo.

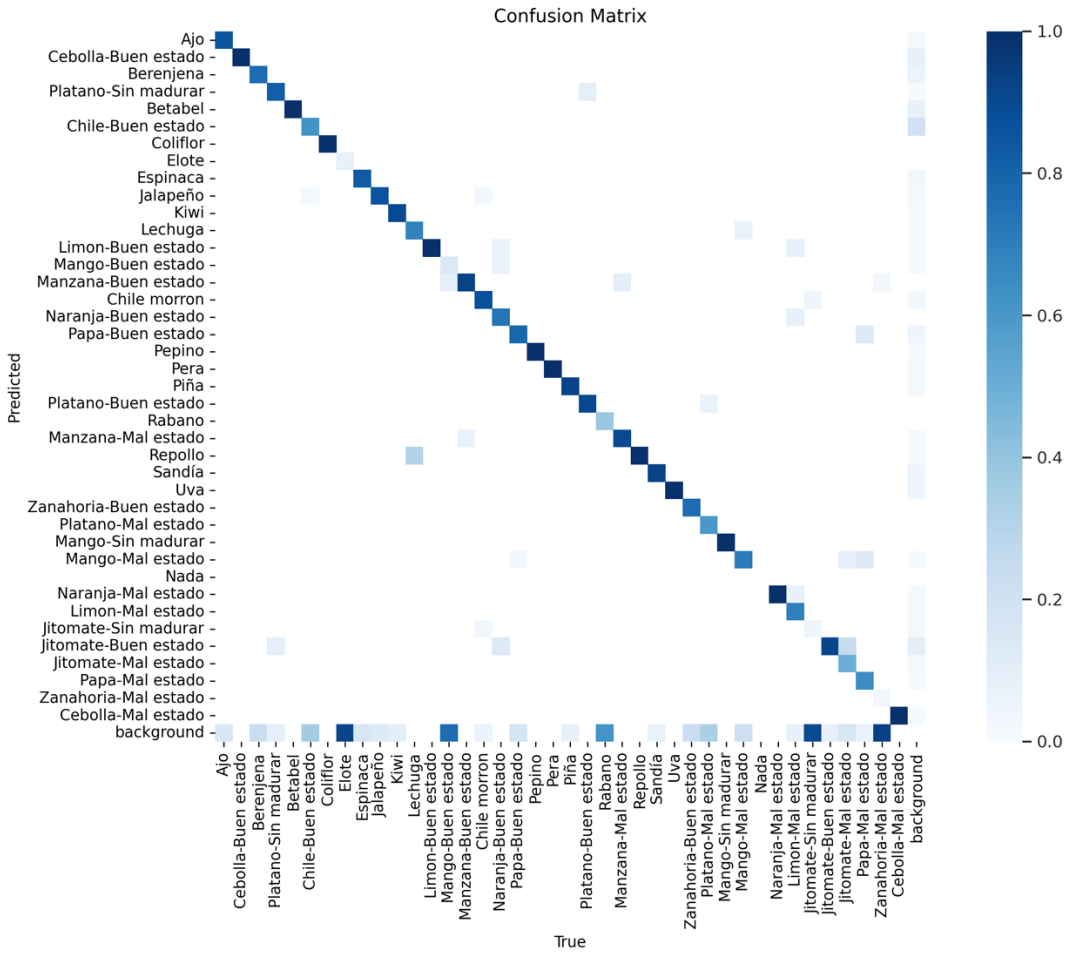


Figura 15. Matriz de confusión en detección de objetos

A diferencia de las matrices de confusión presentadas para el método 1 referente a la clasificación de imágenes, en la figura anterior se puede localizar en una de las clases, la cual se reconoce como background, que refiere a una clase sin etiquetas establecidas, pero que si cuenta con imágenes dentro de la base de datos. En otras palabras se refiere a la clase con imágenes de fondo y que contiene objetos que no forman parte de las clases establecidas. Para este caso, como las clases refieren a frutas y verduras, se asignaron para background imágenes de paisajes y generalmente deben ser un 10% del total de la base de datos con etiquetas.

Existen una serie de factores que a considerar cuando se interpreta la clase "background" en una matriz de confusión, los cuales son como se muestran a continuación:

- Número de objetos de fondo: Esta variable depende del tamaño de la imagen y el número de objetos en la imagen.

- Relación entre objetos de fondo y objetos de interés: Esto se puede utilizar para comprender la calidad del modelo en cuanto a la capacidad de distinguir entre los objetos de interés y de fondo.
- Distribución de los objetos de fondo: Se utiliza para comprender en dónde el modelo presenta dificultades para diferenciar entre objetos de interés y objetos de fondo.

El número de objetos que se clasifican en la matriz de confusión para la clase "background", se pueden utilizar para comprender qué tan bien el modelo es capaz de distinguir entre objetos de interés y objetos de fondo. En el caso de estudio presentado, una gran cantidad de objetos de fondo indica que el modelo pudiera no distinguir entre objetos de interés y los objetos de fondo, siendo las clases de Elote, Jitomate sin madurar y Zanahoria en mal estado, donde se genera un alto porcentaje de confusión en el modelo.

En el Cuadro 11, se presenta una comparación entre los resultados obtenidos en el estado del arte, mostrando el objetivo del algoritmo, las clases, modelos utilizados y la exactitud reportada.

Cuadro 11. Comparación de resultados con estado del arte

Referencia	**Objetivo**	**Clases**	**Modelo**	**Exactitud**
[14]	Maduración	Manzana Jonagold	SVM	90.30%
[15]	Maduración	Manzanas Golden	PCA	90.50%
[16]	Clasificación	10	CNN	93.00%
[17]	Clasificación	18	CNN	94.94%
[18]	Maduración	Manzanas	SVM	95.94%
[19]	Maduración	Uvas	Discriminantes	96.00%
[20]	Maduración	Jitomates	ANN	96.47%
[21]	Clasificación	Limones	CNN	97.30%
[22]	Clasificación	131	CNN	99.00%
Propio	**Clasificación**	**32**	**CNN**	**97.86%**
Propio	**Maduración**	**12**	**CNN**	**96.67%**

CAPÍTULO VI. CONCLUSIONES Y PROSPECTIVAS

Dada la presente investigación realizada, se exploraron e identificaron los distintos métodos utilizados en Inteligencia Artificial en la clasificación de imágenes y detección de objetos, identificando a las redes neuronales convolucionales y el algoritmo YOLO.

Una vez identificados dichos métodos, se aplicaron para su utilización como una herramienta de visión por computadora y a su vez, se analizaron y evaluaron los métodos mencionados antes para posteriormente evaluar sí era posible la implementación en hardware, resultando como factible. Lo que contribuyó al desarrollo de un sistema basado en Inteligencia Artificial capaz de identificar estados de maduración de frutas y verduras.

Los resultados mostrados, conllevan a la conclusión de que es posible predecir estados de maduración en frutas y verduras, existiendo la posibilidad de mejora de estos algoritmos. Dichas mejoras podrían consistir en agregar más clases de frutas y verduras, así como estados de maduración. Esto podría ser a través de la creación de una base de datos con mayor número de imágenes y clases.

Las métricas obtenidas en los distintos experimentos muestran el correcto funcionamiento de la red neuronal artificial, donde al ser corroboradas a través de la inferencia de datos, se puede comprobar lo anteriormente dicho.

La metodología implementada para Inteligencia Artificial podría considerarse como la adecuada para diseñar e implementar sistemas de Machine Learning, Deep Learning, entre otros, ya que provee los pasos e iteraciones necesarias para la obtención de buenos resultados. Sin duda alguna, la metodología se debe ajustar dependiendo la naturaleza del problema y los resultados esperados.

Como prospectivas se tienen, que el rendimiento de los modelos se podría mejorar realizando cambios a los hiperparámetros o incluso cambiar el modelo utilizado tanto para clasificación de imágenes, como para detección de objetos.

Además, el sistema inteligente podría ser rediseñado tanto mecánicamente como electrónicamente para que este pueda cumplir con la ergonomía para ser ingresado en un refrigerador como tecnología de acoplamiento.

REFERENCIAS

[1] M. E. Martínez-González, R. Balois Morales, I. Alia-Tejacal, M. A. Cortes-Cruz, Y. A. Palomino-Hermosillo, and G. G. López-Gúzman, "Postcosecha de frutos: maduración y cambios bioquímicos," *Rev. Mex. Ciencias Agrícolas*, no. 19, pp. 4075–4087, 2017, doi: 10.29312/remexca.v0i19.674.

[2] A. S. Shweta, "Intelligent refrigerator using ARTIFICIAL INTELLIGENCE," *Proc. 2017 11th Int. Conf. Intell. Syst. Control. ISCO 2017*, pp. 464–468, 2017, doi: 10.1109/ISCO.2017.7856036.

[3] K. Montalvo, N. Arias, R. Cueva, K. Enriquez, D. Pallo, and G. Guerrero, "productos alimenticios refrigerados mediante sensores de alerta Mobile application for the prediction of the sensors," *2018 13th Iber. Conf. Inf. Syst. Technol.*, pp. 1–7, 2018.

[4] H. Nasir, W. B. W. Aziz, F. Ali, K. Kadir, and S. Khan, "The Implementation of IoT Based Smart Refrigerator System," *2018 2nd Int. Conf. Smart Sensors Appl. ICSSA 2018*, pp. 48–52, 2018, doi: 10.1109/ICSSA.2018.8535867.

[5] H. Kang and C. Chen, "Fruit detection and segmentation for apple harvesting using visual sensor in orchards," *Sensors (Switzerland)*, vol. 19, no. 20, 2019, doi: 10.3390/s19204599.

[6] S. Kumari, A. Kumar, and P. Kumar, "Maturity status classification of papaya fruits based on machine learning and transfer learning approach," *Inf. Process. Agric.*, vol. 8, no. 2, pp. 244–250, 2021, doi: 10.1016/j.inpa.2020.05.003.

[7] M. S. Hossain, "Prediction of Fruit Maturity , Quality , and Its Life Using Deep Learning Algorithms," 2022.

[8] C. C. Aggarwal, *Neural Networks and Deep Learning*. 2018.

[9] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015, doi: 10.1038/nature14539.

[10] C. A. Müller and S. Guido, *Introduction to Machine Learning with Python*, 1st ed. California: O'Reilly Media, 2019.

[11] M. Horea and O. Mihai, "Fruit recognition from images using deep learning," *Acta Univ. Sapientiae*, vol. 10, pp. 26–42, 2018.

[12] S. Kritik, "Fruits and Vegetables Image Recognition Dataset," *Kaggle 2020*.

[13] R. R. Potdar, "Fresh and Stale Images of Fruits and Vegetables," *Kaggle*, 2022.

[14] D. Unay and B. Gosselin, "Automatic defect segmentation of 'Jonagold'

apples on multi-spectral images: A comparative study," *Postharvest Biol. Technol.*, vol. 42, no. 3, pp. 271–279, Dec. 2006, doi: 10.1016/J.POSTHARVBIO.2006.06.010.

[15] B. Zhu, L. Jiang, Y. Luo, and Y. Tao, "Gabor feature-based apple quality inspection using kernel principal component analysis," *J. Food Eng.*, vol. 81, no. 4, pp. 741–749, Aug. 2007, doi: 10.1016/J.JFOODENG.2007.01.008.

[16] R. K. N, R. K. S, N. C, C. H. K, and S. C, "Deep Features Based Approach for Fruit Disease Detection and Classification," *Int. J. Comput. Sci. Eng.*, vol. 7, no. 4, pp. 810–817, Apr. 2019, doi: 10.26438/ijcse/v7i4.810817.

[17] Y.-D. Zhang *et al.*, "Image based fruit category classification by 13-layer deep convolutional neural network and data augmentation," *Multimed. Tools Appl.*, vol. 78, no. 3, pp. 3613–3632, Feb. 2019, doi: 10.1007/s11042-017-5243-3.

[18] S. R. Dubey and A. S. Jalal, "Apple disease classification using color, texture and shape features from images," *Signal, Image Video Process.*, vol. 10, no. 5, pp. 819–826, Jul. 2016, doi: 10.1007/s11760-015-0821-1.

[19] D. G. Kim, T. F. Burks, J. Qin, and D. M. Bulanon, "Classification of grapefruit peel diseases using color texture feature analysis," *Int. J. Agric. Biol. Eng.*, vol. 2, no. 3, pp. 41–50, 2009, doi: 10.3965/j.issn.1934-6344.2009.03.041-050.

[20] M. P. Arakeri and Lakshmana, "Computer Vision Based Fruit Grading System for Quality Evaluation of Tomato in Agriculture industry," *Procedia Comput. Sci.*, vol. 79, pp. 426–433, Jan. 2016, doi: 10.1016/J.PROCS.2016.03.055.

[21] A. Jahanbakhshi, M. Momeny, M. Mahmoudi, and Y. D. Zhang, "Classification of sour lemons based on apparent defects using stochastic pooling mechanism in deep convolutional neural networks," *Sci. Hortic. (Amsterdam).*, vol. 263, p. 109133, Mar. 2020, doi: 10.1016/J.SCIENTA.2019.109133.

[22] S. Sakib, Z. Ashrafi, and M. A. B. Siddique, "Implementation of Fruits Recognition Classifier using Convolutional Neural Network Algorithm for Observation of Accuracies for Various Hidden Layers," Apr. 2019, [Online]. Available: http://arxiv.org/abs/1904.00783.