UNIVERSIDAD TÉCNICA DE MACHALA
FACULTAD DE INGENIERIA CIVIL

MAESTRÍA EN INGENIERIA EN SOFTWARE

PROCESAMIENTO DE IMÁGENES PARA LA CLASIFICACIÓN DE GRANOS DE CAFÉ POR FORMA Y COLOR

ING. ESTEBAN FABRICIO GONZABAY JIMENEZ

PROPUESTA METODOLÓGICA Y TECNOLÓGICA AVANZADA

TUTOR: ING. CARLOS ALBERTO CALDERON CORDOVA, MGTR.
COTUTOR: ING. NANCY MAGALY LOJA MORA, MGTR.

MACHALA
2023

DEDICATORIA

Dedico este trabajo principalmente a mi madre Olguita Jimenez y a mi padre Javier Gonzabay, porque sé que sin ellos no pude haber logrado nada en la vida. Ellos con su ejemplo y determinación me han convertido en un hombre de bien y me han demostrado que todo se puede con dedicación, diciplina y esfuerzo.

Dedico este trabajo a mis hermanos mayores Melissa y William los cuales desde pequeño me han enseñado el valor de la responsabilidad y el respeto con todo lo que haga en mi vida profesional.

Dedico este trabajo a mi familia y amigos que me quieren y me acompañan a lo largo de los años.

AGRADECIMIENTO

Agradezco principalmente a Dios por este gran logro, porque el me dio la inteligencia y capacidad necesaria para alcanzar esta meta. Sé que sin su mano poderosa no podría haber escrito todo lo que pude escribir a lo largo de este trabajo.

A mi tutor Carlos Calderón Córdova quien con paciencia y ejemplo me ha ayudado a completar este trabajo de posgrado, pero sobre todo que me ha dado la oportunidad de ser su aprendiz.

Agradezco a mis profesores que me han convertido en un profesional, que con conocimiento y experiencia me han ayudado a ser mejor desde la época escolar hasta la universitaria.

RESPONSABILIDAD DE AUTORÍA

Yo, Ing. Esteban Fabricio Gonzabay Jimenez con c.c. 0706282373, declaro que el trabajo de “Procesamiento de imágenes para la clasificación de granos de café por forma y color”, en opción al título de Magister en ingeniería en software, es original y auténtico; cuyo contenido: conceptos, definiciones, datos empíricos, criterios, comentarios y resultados son de mi exclusiva responsabilidad.

Firmado electrónicamente por:
ESTEBAN FABRICIO
GONZABAY JIMENEZ

Ing. Esteban Fabricio Gonzabay Jimenez
C.C. 0706282373

Machala, 07/05/2023

REPORTE DE SIMILITUD TURNITIN

Clasificación Café

INFORME DE ORIGINALIDAD

8%	6%	2%	1%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE

FUENTES PRIMARIAS

1	e-archivo.uc3m.es Fuente de Internet	<1%
2	raw.githubusercontent.com Fuente de Internet	<1%
3	repositorio.uchile.cl Fuente de Internet	<1%
4	Monares Zabaleta Carlos Alberto. "Interpretación del lenguaje de señas utilizando redes neuronales", TESIUNAM, 2017 Publicación	<1%
5	icc2.act.uji.es Fuente de Internet	<1%
6	revistas.univalle.edu Fuente de Internet	<1%
7	www.scribd.com Fuente de Internet	<1%
8	Montalvo Lezama Berenice. "Clasificación multi-etiqueta de videos cortos usando	<1%

CERTIFICACIÓN DEL TUTOR

Yo, Ing. Carlos Alberto Calderón Córdova con C.C. 1104014038, tutor del trabajo de "Procesamiento de imágenes para la clasificación de granos de café por forma y color", en opción al título de Magister en ingeniería en software, ha sido revisado, enmarcado en los procedimientos científicos, técnicos, metodológicos y administrativos establecidos por el Centro de Posgrado de la Universidad Técnica de Machala (UTMACH), razón por la cual doy fe de los méritos suficientes para que sea presentado a evaluación.

Ing. Carlos Alberto Calderón Córdova
C.C. 1104014038

Machala, 07/05/2023

CESIÓN DE DERECHOS DE AUTORÍA

Yo, ESTEBAN FABRICIO GONZABAY JIMENEZ con C.I. 0706282373 autor del trabajo de titulación "PROCESAMIENTO DE IMÁGENES PARA LA CLASIFICACIÓN DE GRANOS DE CAFÉ POR FORMA Y COLOR", en opción al título de Magister en Software, declaro bajo juramento que:

- El trabajo aquí descrito es de mi autoría, que no ha sido presentado previamente para ningún grado o calificación profesional. En consecuencia, asumo la responsabilidad frente a cualquier reclamo o demanda por parte de terceros de manera exclusiva.
- Cede a la Universidad Técnica de Machala de forma exclusiva con referencia a la obra en formato digital los derechos de:
    a. Incorporar la mencionada obra en el repositorio institucional para su demostración a nivel mundial, respetando lo establecido por la Licencia Creative Commons Atribution-NoCommercial – Compartir Igual 4.0 Internacional (CC BY NCSA 4.0); la Ley de Propiedad Intelectual del Estado Ecuatoriano y el Reglamento Institucional.
    b. Adecuarla a cualquier formato o tecnología de uso en INTERNET, así como correspondiéndome como Autor la responsabilidad de velar por dichas adaptaciones con la finalidad de que no se desnaturalice el contenido o sentido de la misma.

ESTEBAN FABRICIO GONZABAY JIMENEZ

C.I. 0706282373

Machala, 07/05/2023

RESUMEN

El café es un producto muy valorado dentro de la sociedad ecuatoriana, así como también lo es a nivel mundial. Parte del café producido en Ecuador es exportado, es así que para que los precios internacionales del producto sigan siendo altos, sea necesario que el café exportado esté debidamente seleccionado. Actualmente en muchas asociaciones de productores cafetaleros realizan esta actividad de manera manual mediante inspección visual, en base a ello el objetivo que aborda el presente proyecto es el desarrollo de un algoritmo de procesamiento de imágenes basado en Redes Neuronales Convolucionales (CNN) para la clasificación de granos de café, en donde, la respuesta del sistema corresponde a la clasificación de granos en buen estado y de granos en mal estado en base a irregularidades en la forma y en el color del grano de café. El tipo de CNN utilizada es la MobileNet, la cual se basa en una arquitectura optimizada en las convoluciones para generar redes neuronales profundas con menor número de parámetros y en consecuencia con bajo requerimiento computacional y con baja latencia. Para la evaluación de comportamiento del algoritmo de clasificación de granos de café se realizó el análisis comparativo del rendimiento para las arquitecturas MobileNetV1 y MobilNetV2. Los parámetros de comparación utilizados fueron evaluados en la respuesta del diagrama de precisión basándose en las pérdidas del sistema clasificador. Como resultados de la comparación de las arquitecturas, se logró una precisión del 99.18% y una validación del 99.20% para el modelo MobileNetV2; mientras que para el modelo MobileNetV1 la precisión alcanzó el 88.72 y la validación alcanzó el 81.25%. En consecuencia, el desempeño del modelo basado en MobileNetV2 es considerado como la opción elegida para la clasificación de granos de café.

Palabras claves

Inteligencia artificial, redes neuronales convolucionales, clasificación de imágenes, café en grano.

ABSTRACT

The present project focuses on the development of an image processing algorithm based on Convolutional Neural Networks (CNN) for the classification of coffee beans, aiming to distinguish between beans in good condition and those with defects based on irregularities in shape and color. The chosen CNN model is MobileNet, which is optimized for convolutional operations to create deep neural networks with fewer parameters, resulting in low computational requirements and low latency. The manual visual inspection currently performed by many coffee producer associations for proper selection of exported coffee motivated the objective of this project. As coffee is highly valued both in Ecuadorian society and globally, ensuring high international prices for the exported product requires careful selection. To evaluate the performance of the coffee bean classification algorithm, a comparative analysis was conducted using the MobileNetV1 and MobileNetV2 architectures, comparing loss statistics. The results of the architecture comparison demonstrated a precision of 99.18% and validation accuracy of 99.20% for the MobileNetV2 model, while the MobileNetV1 model achieved a precision of 88.72% and validation accuracy of 81.25%. Consequently, the performance of the MobileNetV2-based model was considered the optimal choice for coffee bean classification.

Keywords:

Artificial intelligence, convolutional neural networks, image classification, coffee beans.

ÍNDICE

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

INTRODUCCIÓN

La automatización de las industrias es un hecho dentro de la economía mundial, los avances tecnológicos hacen que la calidad de los productos agrícolas se comercialicen de manera internacional , cumpliendo estándares de comercialización globales [1],[2] y [3]. El café es un producto muy valorado dentro de la sociedad ecuatoriana, así como también lo es a nivel mundial [4]. En consecuencia, la labor de los agricultores en la selección cuidadosa de los granos de café óptimos es crucial para garantizar la producción de un alimento de alta calidad, rico en nutrientes. Además de conservar el sabor peculiar del puro café; suele ser un reto para el empleado por la forma manual en que se realiza tal selección [5]. En la actualidad, la clasificación de granos de café en la agricultura ecuatoriana es realizado de manera manual y en muchos casos industrial [2]. Las industrias se ven afectadas por las grandes sumas de dinero que tienen que pagar para conseguir dichas herramientas para la clasificación de café. En este escenario, se ha demostrado que la clasificación manual del café es una de las formas más seguras para reservar la calidad del producto final. Esto se debe a que el proceso de selección manual permite una evaluación detallada de factores como el tamaño, forma y humedad de los granos, lo que garantiza que solo los mejores granos sean seleccionados para su procesamiento. De esta manera, se asegura que la calidad del café sea óptima y se logre obtener un producto final de alta calidad [2].

Vale destacar que este proyecto está destinado a aplicarse en la ciudad de Loja - Ecuador, con el propósito de obtener mejoras en el proceso de selección de los granos de café conservando una buena calidad del producto. El consumo de café dentro de esta área ha sido de alrededor de 164,9 millones de sacos durante el año 2020 y 2021[4]. Por lo tanto, la población de estudio estará conformada por los granos de café cultivados en dicha región.

Para la recolección de datos, se utilizarán técnicas de muestreo aleatorio y se realizará un análisis detallado de los granos seleccionados [6]. El tipo de diseño de investigación será experimental [7], ya que se implementará en un brazo mecánico equipado con una cámara para la percepción de objetos y se evaluará la eficacia del algoritmo de clasificación basado en redes neuronales convolucionales. Además, se realizarán pruebas comparativas con métodos manuales e industriales existentes. Este enfoque permitirá obtener datos precisos y relevantes para la evaluación del sistema automatizado de clasificación.

La calidad es un factor fundamental a considerar al clasificar los granos de café [8]. Actualmente, los procesos de clasificación industrial no garantizan una efectividad del 100% en la calidad de clasificación. Por lo que los varios estudios recientes [8] y [9], han propuesto y dado a conocer que la manipulación de datos e información de una imagen digital resulta eficiente mediante la utilización del aprendizaje automático y la visión artificial. En este punto, se ha propuesto en [5] y [10] tecnologías computacionales innovadoras para la clasificación de productos agrícolas, tales como: granos, semillas, entre otros. En algunos de los casos se utiliza la visión artificial para la selección de tierras fértiles para la siembra de plantas de café [11].

En la actualidad existen una gran cantidad de algoritmos que se usan para la detección y clasificación de objetos, formas y tamaños. Entre estos, se referencia el algoritmo "random forest", el cual permite la detección de objetos en una imagen digital por medio de sus pixeles y patrones; de esta manera, se puede identificar y categorizar lo más importante de una imagen [10] y [12]. Inclusive, se presenta otro algoritmo denominado Support Vector Machine, que funciona para entrenar los datos basándose en una sola tarea y se utiliza para la identificación de patrones [13], este algoritmo se propone como una herramienta poderosa para la detección de objetos; sin embargo, se ven en gran desventaja para proyectos que conlleven el uso de multitareas. En virtud de subsanar esta desventaja, se incluye indagaciones en otros tipos de alternativas para mejorar el referido algoritmo.

El alcance de este proyecto se denotará en la ciencia de la experimentación y utilización de redes neuronales convolucionales (CNN), a partir de sus principales características. Las redes neuronales convolucionales funcionan como un sistema de manipulación de datos, principalmente son usadas para proyectos de visión artificial [14], línea de investigación que refiere este proyecto.

Cabe destacar que en la actualidad existen algoritmos que utilizan redes neuronales (NN) para la detección de imágenes e incluso para clasificación de granos de café [1].

La utilización de técnicas basadas en inteligencia artificial [15], visión artificial [5] y aprendizaje automático [16], hacen que el uso de un modelo de redes neuronales convolucionales sea la opción más adecuada para la detección de defectos en granos de café por la forma de su arquitectura, funciones de pertenencia y segmentaciones [2] y [3].

La clasificación precisa de granos de café es de suma importancia en la industria agrícola. Por lo tanto, el desarrollo de una máquina capaz de separarlos con precisión representaría una gran ventaja en términos de calidad y costos de producción [5],[8] y [17]. Las enfermedades que se pueden encontrar en los granos de café son variadas y estas

enfermedades se pueden confundir a veces en granos sanos, como lo menciona [8], dado que la detección de granos de café de calidad óptima es en muchos casos una tarea prácticamente imposible, se puede asumir que la realidad de dicha actividad presenta dificultades significativas.

Por lo tanto, el desarrollo de un algoritmo capaz de cumplir con la demanda de un 90% de eficacia se ve reflejada en el número de pruebas que se pueda realizar al producto. Se pretende utilizar un algoritmo capaz de segmentar por partes la imagen digital, para llegar a tener una visión de los objetos que se encuentran frente a la cámara del sistema robótico [5].

El objetivo de este proyecto es desarrollar un algoritmo que pueda utilizarse en un brazo robótico y, mediante una cámara, percibir de manera clara la información de los objetos destinados a la clasificación, ya sea granos de café con defectos o sin defectos. Se busca que este algoritmo pueda identificar correctamente los granos de café, mejorando así la calidad de los productos derivados del café. Para lograrlo, se propone implementar un proceso rápido y rentable que permita obtener granos limpios y de calidad para su exportación.

Para llevar a cabo este objetivo, se propone diseñar un sistema de clasificación de granos de café basado en cálculos predictivos y entrenamiento de datos, tal como se menciona en [8]. Este sistema será capaz de procesar y detectar defectos en un conjunto de granos de café. Una vez que el sistema detecte los granos, el brazo robótico se encargará de separar los granos con defectos y garantizar la inclusión en el proceso de selección final de aquellos granos que cumplan con los estándares de calidad requeridos.

El aprendizaje de datos previamente entrenado funciona de una manera eficiente, tal que el método de identificación que se utiliza es K-Nearst Neighbor, utilizado en el procesamiento y detección de fallos de imágenes digitales en el área de la medicina [6] y [7]. Las técnicas que se utilizan para el análisis de imágenes RGB (Red, Green, Blue) se estructuran en 3 fases fundamentales para el procesamiento de la información; umbralización, limpieza y filtrado [3] y [8]. La umbralización trata la parte de la segmentación de información, de tal manera que convierte la imagen a escalas de grises para así la imagen categorizarse en dos estados o niveles, blanco o negro. Así mismo, en la limpieza se intenta limitar el ruido de la imagen a tal magnitud que se elimine la información que está por demás en la data que se va a almacenar. Y, por último, el filtrado intenta clasificar los objetos por orden geométrico [3].

Este proyecto posee una gran importancia y representa un aporte significativo a la investigación en el ámbito de la clasificación de productos agrícolas. La implementación de tecnologías como la inteligencia artificial, la visión artificial y el aprendizaje automático en la clasificación de granos de café permitirá mejorar la calidad del producto, optimizar los procesos de producción y reducir los costos asociados. Además, se espera obtener resultados que contribuyan al avance de la ciencia y a la aplicación práctica de estos avances tecnológicos en la industria agrícola. Este proyecto representa un paso hacia la automatización y la mejora continua en la selección de los granos de café, proporcionando una solución eficiente y precisa para garantizar la calidad del producto final.

FORMULACIÓN DEL PROBLEMA

¿Cómo desarrollar un algoritmo para un sistema de clasificación de imágenes reduciendo asi la dependencia de los procesos manuales en la clasificación de granos de café?

Sistematización del problema

- ¿Cuáles son los métodos actuales utilizados en la clasificación manual de granos de café?
- ¿Qué características y criterios deben considerarse en el desarrollo de un algoritmo de clasificación de imágenes eficiente y preciso para los granos de café?
- ¿Qué tecnologías, como la inteligencia artificial y la visión artificial, son más adecuadas para el desarrollo del algoritmo de clasificación de imágenes?
- ¿Cuáles son los beneficios y ventajas de implementar un algoritmo de clasificación de imágenes en la industria de los granos de café?

OBJETIVOS

Objetivo general

Desarrollar y evaluar algoritmos de procesamiento de imágenes para la identificación y clasificación de granos de café por forma y color.

Objetivos específicos

- Realizar la investigación bibliográfica y trabajos relacionados con respecto a algoritmos de procesamiento de imágenes para identificación y clasificación de semillas de café.

- Diseñar y desarrollar un algoritmo basado en redes neuronales convolucionales (CNN) para la detección y clasificación precisa de granos de café.
- Evaluar y comparar la eficacia y precisión del sistema automatizado de clasificación de granos de café.
- Realizar pruebas y ajustes en el sistema para lograr una eficiencia mínima del 90% en la clasificación de los granos de café.

CAPÍTULO 1: MARCO TEÓRICO

1.1. Antecedentes históricos

Existen varios estudios que comprueban la validez y la utilidad que tienen las redes neuronales convolucionales en proyectos de visión artificial.

Un estudio que se realizó en Colombia en el 2006 para lograr clasificar granos de café por medio de un FPGA (Field-Programable Gate Array) [2], que funciona como un hardware donde se puede montar la programación para la clasificación de café. En el artículo [2], deduce que se logró una efectividad de entre el 89.5% y 91.7%, utilizando un modelo bidimensional por capas y pesos enteros, y una estructura de percepción multicapa, el cual funciona con un algoritmo de aprendizaje diseñado en C++.

En este proyecto existe una gran cantidad de información que ingresa al modelo [9], por lo tanto el modelo tendrá que manipular esa información de manera correcta y precisa, por consiguiente las redes neuronales convolucionales deben cumplir con los parámetros necesarios para la identificación de datos [18].

En la ciudad de Genova - Italia se llevó a cabo un estudio para hallar fallos en los granos de café verdes que contengan algún tipo de impurezas. Se propuso un identificador multivariante para imágenes hiper - espectrales en la región del infrarrojo más cercano (HSI – NIR), este sistema permite adquirir espectros de un área completa de entre varios cuerpos [19]. Propone que para validar la calidad de los granos de café primero se debe identificar qué tipos de enfermedades pueden afectar al grano en cuestión, usando este problema como punto de partida para la creación del sistema que permita la clasificación de los granos de café. Este análisis ha identificado diversas categorías de disfunciones que pueden afectar a los granos de café, ocasionando que se asemejen visualmente a granos de café normales. Algunos ejemplos de estas disfunciones incluyen:

- Fragmentos de granos que se maltratan al momento de la transportación por los mismos mecanismos que se utilizan para la cosecha
- Granos que se ven afectados por la deshidratación
- Granos que tienen defectos internos como moho o suciedad
- Granos negros que se los derivan como granos muertos por la misma descomposición natural del producto.

Para el analizar los resultados, este proyecto usa granos que aparentemente no tienen defectos, que son aprobados para el consumo humano y que supuestamente están listos para la comercialización [19], sin embargo, el sistema presenta buenos resultados.

El uso de un sistema electrónico para la obtención de datos en imágenes digitales como lo propone[8], el mismo que menciona que evaluando las características principales de una imagen como: pixeles, patrones de bits y descriptores espectrales [18] (información de una imagen digital), llevan a detectar los defectos que puede tener un grano de café [2]. En el estudio realizado en Brasil [8], se utilizó algoritmos de código abierto como lo son SVM (support vector machine) [8], RF (random forest) [12] y [20] y CNN (convolution neuronal network) [21], en donde se especificó que estos algoritmos eran eficientes al momento de utilizarlos para procesamiento de imágenes enfocados en el plano de la agronomía para la detección y clasificación de objetos [10] y [18].

Un estudio reciente realizado en Indonesia, donde implementaron CNN para la clasificación de granos de café mediante un sistema móvil [15], el cual se enfoca en un pre entrenamiento de datos que mediante el uso de una arquitectura VGG-16 (segmentación semántica) para poder ubicar posibles defectos en granos de café [10]. Los autores de este estudio obtuvieron resultados de precisión de entre el 58.7% hasta 66.8%, sin embargo, paralelamente a este estudio aplicaron un modelo denominado ResNet – 152 [10], este modelo tiene como referencia la segmentación semántica por capas que significa crear un mapa de características que luego son utilizadas para una agrupación piramidal de datos del sistema y de esta manera obtener información a diferentes escalas, estableciendo que el procesamiento de pixeles de una imagen sea más conveniente [15]. Este último estudio tuvo entre 62.3% - 73.3% de precisión en la evaluación de datos.

Existe un algoritmo que se utiliza para lograr rapidez en el procesamiento de imágenes denominado “You Only Look Once” (YOLO), el cual pretende detectar objetos mediante un solo escaneo de información. YOLO ha sido utilizado para la detección de placas de automóviles en movimiento [22], y para el mapeo de tierras y granos de café cosechados por maquinaria pesada [5].

El algoritmo YOLO es una técnica de procesamiento de imágenes que permite realizar un análisis cualitativo de las mismas [5]. Este análisis resulta muy útil para mejorar la clasificación de parámetros por etiquetas en imágenes que contienen datos relevantes [18] y [16]. Al aplicar este enfoque, se puede obtener un modelo predictivo que facilita el proceso de entrenamiento de datos [1] y [11]. En este sentido, aplicación del algoritmo

YOLO se ha convertido en una herramienta muy valiosa para el desarrollo de proyectos de inteligencia artificial.

Los estudios revisados (ilustración 1) respaldan la validez y utilidad de las redes neuronales convolucionales en proyectos de visión artificial relacionados con la clasificación de granos de café y la detección de defectos.

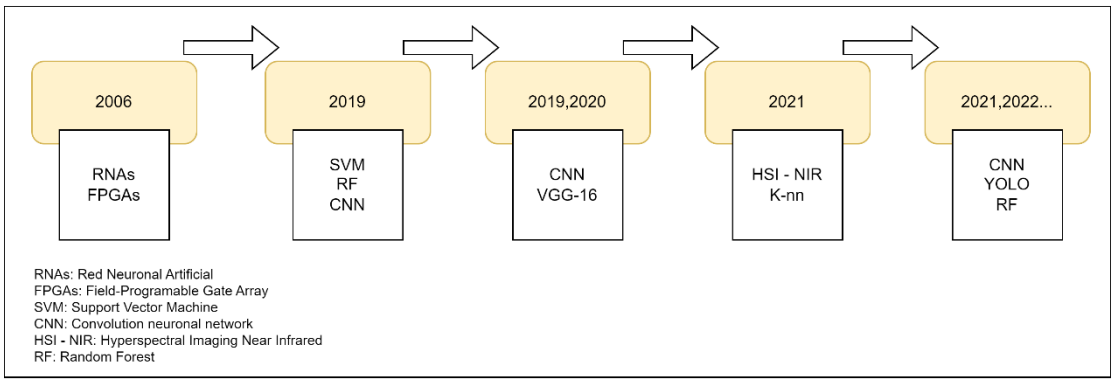


Ilustración 1. Línea de tiempo de antecedentes históricos

En general, Se ha observado que estos estudios han empleado diferentes enfoques y técnicas, como el uso de FPGA, imágenes hiperespectrales y algoritmos como SVM, RF, CNN y YOLO.

Estos hallazgos sustentan la noción de que las redes neuronales convolucionales son herramientas poderosas para abordar desafíos de visión artificial relacionados con la clasificación de granos de café. La habilidad de estas redes para analizar imágenes y extraer características relevantes ha demostrado ser efectiva en la detección de defectos y la clasificación precisa de los granos.

1.2. Antecedentes contextuales sobre granos de café

El grano de café es extraído de un arbusto llamado cafeto que usualmente nace en climas tropicales y en lugares donde suelen abundar las estaciones de primavera o verano. Necesitan un cuidado delicado al momento de cultivar esta planta, porque se tiene que mantener en una temperatura adecuada para no causar ningún tipo de daño en la planta principal. Un cafeto tiene una vida comercial de entre 20 a 25 años, dependiendo de las condiciones o la manera de cultivo. Un arbusto joven puede comenzar a producir frutos a partir del primer año y continuar año tras año produciendo granos, hasta llegar a su máxima productividad entre los 6 y 8 años [23].

Ecuador es considerado como un país con gran capacidad de producción de café por la variedad de ecosistemas y el lugar geográfico en el que está ubicado [24].

1.2.1. Proceso de obtención del grano

La recolección de granos de café es un proceso que se realiza de forma manual y selectiva, tomando las cerezas de café maduras y evitando dañar las ramas, lo que lleva a la manipulación de la misma rama por varias ocasiones para de esta manera obtener una gran variedad de café. Cabe destacar que, si las ramas del árbol de café se manipulan de una mala manera, esto puede afectar negativamente la productividad de estas.

Existe una manera alternativa de cosechar esta planta que consiste en que la persona hace caer los granos de café sobre una lona amplia y luego seleccionar manualmente las cerezas que tengan las mejores condiciones.

1.2.2. Fases para la obtención de granos de café

Para el proceso de obtención de grano previo a la comercialización del producto [25], se realiza lo siguiente:

1. Recolección: La recolección se realiza a mano por agricultores, de esta manera se mantiene una calidad en el producto.
2. Cosecha: Se deben recoger en su totalidad los frutos maduros.
3. Despulpar: Los productores usan una despulpadora para quitar la corteza del grano, usualmente para este proceso utilizan el agua como principal recurso.
4. Fermentación: Este proceso tiene un tiempo de entre 18 a 24 horas dentro de tanques de agua, de esta manera la capa viscosa del grano se descompone.
5. Lavado: Se realiza en los mismos tanques de agua del paso anterior donde se busca eliminar el mucílago.
6. Remoción del mucílago: El propósito de este proceso es lograr la eliminación total de la capa viscosa que recubre los granos de café. Para lograr esto, se realiza una fermentación en agua que permite disolver la capa viscosa, la cual luego es eliminada mediante un proceso de lavado. La duración de este proceso puede variar según el clima, y puede tardar entre 18 y 30 horas en completarse.
7. Secado: El grano se somete a un proceso de secado al sol, lo que resulta en un nivel de humedad del 12% en el grano.
8. Trilla: En este paso se elimina el recubrimiento y dejando el grano verde listo para ser clasificado por maquinas seleccionadoras.

1.2.3. Enfermedades que pueden afectar a los granos de café

Para lograr que la clasificación de granos de café sea óptima, se debe analizar los tipos de datos con los que se va a trabajar. En este caso se necesita saber cuáles son las razones por cuales los granos de café pueden bajar la calidad del producto final [8], es por esto que en la ilustración 2 se muestran las posibles enfermedades que pueden afectar directamente a los granos de café en diferentes circunstancias.

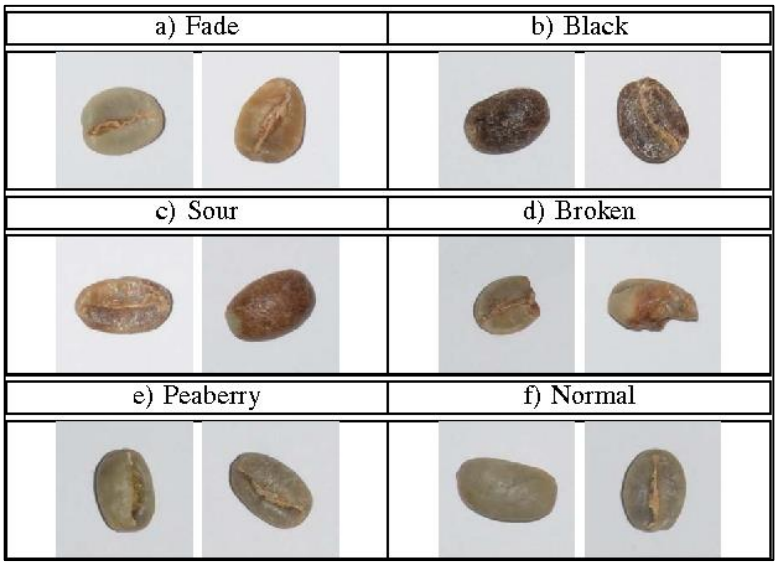


Ilustración 2. Enfermedades de los granos de café

El objetivo principal de este proyecto de investigación es utilizar un sistema de entrenamiento de datos basado en redes neuronales convolucionales para clasificar granos de café en función de la presencia o ausencia de defectos [19]. Para ello, se utilizará un conjunto de imágenes que contienen granos de café con y sin defectos, el cual servirá para que el algoritmo aprenda de estos datos [12]. De esta manera, se espera que el modelo de clasificación resultante sea capaz de detectar las enfermedades más comunes que pueden afectar a los granos de café y, por lo tanto, contribuir a mejorar la calidad de producto final. Es importante destacar que contar con un mayor número de imágenes en el conjunto de datos de entrada contribuye significativamente a la obtención de un clasificador más confiable [26].

1.3. Antecedentes conceptuales sobre inteligencia artificial

Hace tiempo que la inteligencia artificial dejó de lado el fantasma de la ciencia ficción y comenzó a apegarse más a la vida rutinaria de las personas [26]. Los avances en inteligencia artificial y la industria 4.0 están transformando la funcionalidad de los

dispositivos tecnológicos actuales. Esto hace que se creen dispositivos más sofisticados y eficientes, como asistentes virtuales, robots y vehículos autónomos.

Además, la automatización y la digitalización en la fabricación de productos, a través del uso de la web 4.0 y tecnologías de la Industria 4.0, han reducido costos y aumentado la calidad de los productos y servicios [27].

El modo en como las industrias prevén el futuro de estas es gracias a los avances que ha habido por la llegada de la inteligencia artificial como principal recurso para desarrollar nuevos productos [28].

La definición de inteligencia artificial puede ser un tema enigmático debido a la variedad de definiciones existentes [24]. No obstante, investigadores la definen como un conjunto de algoritmos capaces de dotar a las máquinas de comprensión y capacidad de aprendizaje, para tomar decisiones al igual que lo haría un ser humano. La implementación adecuada de esta tecnología puede permitir que las máquinas trabajen de manera continua sin fatiga [6] y realicen un análisis de metadatos más rápido y eficiente [7] y [19].

1.3.1. Machine Learning o Aprendizaje automático

El aprendizaje automático es una disciplina en el campo de la inteligencia artificial, que, a través de algoritmos [9], las computadoras pueden identificar patrones en gran cantidad de datos y hacer predicciones (análisis predictivo) [14],[17] y [29]. Este aprendizaje permite al ordenador realizar tareas específicas de forma autónoma, es decir, sin necesidad de ser programador [16]. Debido a las capacidades del aprendizaje automático los sistemas que dependen de la inteligencia artificial, para funcionar, puede realizar tareas que antes eran imposibles para las máquinas e incluso tareas que resultan difíciles o peligrosas para humanos [21] y [30].

El aprendizaje automático o machine learning (ML) emplea algoritmos para aprender patrones a partir de datos [12] y [26]. Por ejemplo, el comportamiento de las sugerencias de búsqueda en temas similares a las búsquedas recientes puede ser entendido fácilmente gracias al estudio de patrones de la persona que utiliza el navegador, entrenando así el navegador para sugerir temas similares en el futuro. Esta misma dinámica se aplica en aplicaciones de música, videos y otros ámbitos, demostrando la omnipresencia del machine learning [9].

1.3.2. Visión Artificial

La visión artificial o visión por computadora es un campo de la informática y la ingeniería eléctrica que integra mecanismos para adquirir, procesar, analizar y comprender contenido digital como imágenes (comunes, infrarrojas, entre otros) y video [5]. Un sistema de visión por computadora se desarrolla para aceptar varios tipos de datos en sus variables de entrada, como una secuencia de imágenes o videos que se pueden transmitir desde múltiples fuentes para su correspondiente procesamiento y transformación en información relevante para ser utilizada en la toma de decisiones [7].

1.3.3. Red Neuronal

Dentro de un enfoque biológico, una red neuronal es una parte del cerebro que sirve para tomar decisiones en base a un estímulo [31]. El contexto de esto es que dentro de un cerebro humano existen miles de millones de neuronas que controlan las acciones que se toman día a día para cualquier evento o acción que se realiza, e incluso con este concepto, se puede predecir eventos futuros tomando en cuenta los recuerdos de eventos que sucedieron en el pasado [32].

Una analogía entre la definición biológica de una red neuronal y su contraparte tecnológica es que ambas constan de un conjunto de neuronas conectadas. Sin embargo, en el contexto de una red neuronal artificial [14], se utilizan capas de entrada, capas de procesamiento y capas de salida que están conectadas en una estructura de múltiples capas. De esta forma, las señales de entrada se procesan a través de la red neuronal y se generan las respuestas de salida correspondientes.

1.3.3.1. Red neuronal artificial

Las redes neuronales artificiales son algoritmos que funcionan como un modelo de programación para resolver un problema [33]. Se pueden utilizar para cualquier entorno en donde se necesite procesar una gran cantidad de datos y resolver problemáticas basadas en un sistema de decisiones[34] y [13]. Las unidades básicas que contemplan este sistema son neuronas, que a menudo se organizan en capas.

Una red neuronal bien configurada cuenta con las siguientes características principales:

- Capa de entrada: Información que ingresa al modelo para un posterior entrenamiento.

- Capas ocultas: Algoritmos de procesamiento de datos, donde se consideran las funciones de activación, entrenamiento de datos, aprendizaje automático, etc.
- Capas de salida: Respuesta final del sistema

Para entender las redes neuronales convolucionales se puede revisar el anexo A.

1.3.3.2. Composición de redes neuronales artificiales

Las redes neuronales artificiales son una composición de un conjunto amplio de neuronas organizadas en capas, que de esta manera crean un modelo o un sistema de predicciones a base de condiciones y entrenamientos complejos.

Las redes neuronales artificiales hacen que un sistema de aprendizaje automático tenga un grado de validez clara al momento de dar una respuesta [21] y [35].

Para la representación gráfica de un modelo de redes neuronales se utiliza círculos que representan las neuronas de información que se interconectan con flechas que quieren dar a entender la manera en cómo las neuronas interactúan entre ellas, dando origen a un gráfico que relaciona todas las neuronas del modelo basándose en las características programadas para brindar una respuesta o solución en sus neuronas de salida.

1.3.3.3. Representación de un modelo de redes neuronales

Cada círculo representa una neurona del cerebro humano: El círculo de entrada representa a la acción que se va a tomar, el círculo verde representa los factores de cambio o las características que puede pertenecer a tal acción y el círculo de salida es el resultado del modelo, tal y como se muestra en la ilustración 3 que presenta una red neuronal básica con 3 neuronas de entrada y 2 neuronas de salida, con una capa oculta.

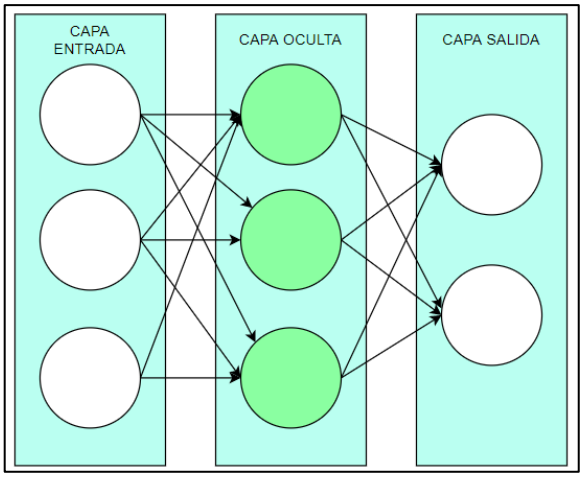


Ilustración 3. Red neuronal con 3 entradas y 2 salidas

Cuando se analiza un gran número de neuronas, estas se pueden clasificar por un modelo de capas, las cuales sirven para tener una representación de las entradas del sistema, las características a calificar y la salida (capas de entrada, capas ocultas y capas de salida). Cabe destacar que pueden existir múltiples capas ocultas, esto depende de objetivo que se quiera lograr y el grado de error que se desee evitar [8].

La complicación de este modelo de redes neuronales radica en el gran número de neuronas de entrada que se pueda tener, por lo que se deberá ingresar varias características de cambio acorde al número de soluciones que se requiera interpretar. En un sistema de reconocimiento de imágenes las entradas vienen a ser el número de pixeles que tiene una imagen (sea a color o a escala de grises), las características que calificarán a estos pixeles vienen a ser la tasa de cambio que los pixeles puedan tener, aquí aparece el patrón RGB de una imagen que por lo general se lo califica en base a 3 colores (Red, Green, Blue) y el resultado se verá acorde a lo que se quiere encontrar en una imagen [2] y [8].

Para poder interpretar correctamente los resultados de un modelo de redes neuronales, se requiere de un algoritmo capaz de entrenarlo. Este algoritmo debe ser capaz de ubicar todas las posibles combinaciones que el modelo pueda tener [21]. Es aquí donde entra en juego el aprendizaje automático, ya que una vez que el modelo encuentra la respuesta correcta en la salida del sistema, el modelo estará totalmente entrenado.

Las capas también se vuelven complejas. Se pueden encontrar varias capas intermedias con varias neuronas cada una, llegando a lo que llaman una "red neuronal profunda o capa profunda". La idea es que más capas con más neuronas por capa, pueden mejorar las predicciones en conjuntos de datos más complejos. Desde una perspectiva visual y matemática, las múltiples capas y unidades de funcionamiento de la red neuronal afectan el modelo como tal.

El uso de las redes neuronales no siempre da con la respuesta correcta la primera vez entrenada, porque para poder encontrar la respuesta más valida se deberá hacer cambios en las capas existentes [8] o ingresar más capas ocultas, aumentando así el nivel de confiabilidad de respuesta [19].

1.3.3.4. Redes neuronales convolucionales

Las redes neuronales convolucionales (CNN) son una evolución de las redes neuronales artificiales. Estas redes cuentan con una o varias capas ocultas que funcionan como filtros de información, mejorando la eficiencia en el procesamiento de datos de entrada y dando una respuesta válida para un sistema de clasificación. Un ejemplo de aplicación de CNN

es la identificación de objetos en imágenes de alta resolución, donde la CNN puede analizar una imagen de 250 x 250 pixeles y extraer características específicas del objeto para una mejor identificación. De esta forma la capa oculta (convolucional) realiza un proceso de filtrado por medio de matrices (2x2, 3x3, entre otros), que involucra detectar los pixeles en blanco o que no tengan datos importantes para luego reducir la cantidad de información que va a ingresar al sistema de clasificación, un ejemplo de un sistema de redes neuronales se muestra en la ilustración 4. Las redes neuronales convolucionales se usan con más frecuencia en reconocimiento de voz y procesamiento de imágenes [1],[30] y [35].

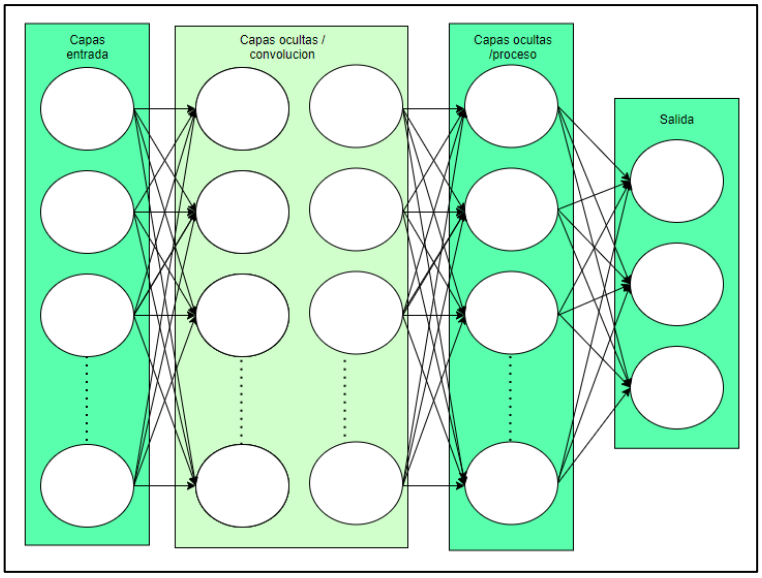


Ilustración 4. Red neuronal convolucional

1.3.4. Herramientas computacionales

La elección de las herramientas adecuadas es un paso crucial en cualquier proyecto de desarrollo de algoritmos [34],[36] y [15]. La selección de las herramientas correctas afecta directamente la calidad del trabajo y la eficiencia del proceso [12]. Por lo tanto, en esta sección se presentarán detalladamente las herramientas utilizadas en el algoritmo para garantizar su correcto funcionamiento. En la tabla 1 se definen los conceptos del software que se utiliza para diseñar un sistema de reconocimiento de imágenes.

Tabla 1. Tabla de definiciones de software y librerías usadas en el proyecto

Python	Python es un lenguaje de programación orientado a objetos que se destaca por su facilidad de comprensión tanto en lectura como en programación. Es especialmente adecuado para proyectos de desarrollo rápido de aplicaciones debido a su dinamismo y facilidad

	de manejo, lo que lo diferencia de otros lenguajes de programación más complejos. Además, cuenta con herramientas de código abierto que lo hacen relativamente sencillo de aprender y accesible a todas las personas.
Google Colaboratory	Google Colaboratory es un entorno web de desarrollo gratuito que se ofrece a través de la plataforma web de Google. Este servicio cloud se presenta como notebooks de Jupyter, lo que permite que los usuarios puedan acceder a herramientas como Scikit-learn, PyTorch, TensorFlow, Keras y OpenCV de manera gratuita. Además, Google Colaboratory es un entorno agradable para los desarrolladores que desean trabajar en proyectos de aprendizaje automático o aprendizaje profundo, ya que está basado en Python 2.7 y 3.6 y permite el uso gratuito de GPUs y TPUs de Google.
Keras	Keras es una biblioteca de código abierto (bajo licencia MIT) escrita en Python, basada principalmente en el trabajo de François Chollet, un desarrollador de Google, en el marco del proyecto ONEIROS. La primera versión de este software multiplataforma se lanzó el 28 de marzo de 2015. El objetivo de la biblioteca es acelerar la generación de redes neuronales. Debido a esto, Keras no funciona como un marco de configuración independiente, sino más bien como una interfaz intuitiva (API) que brindan acceso y desarrollo de varios marcos dentro del aprendizaje automático. Los marcos compatibles con Keras incluyen Theano, Microsoft Cognitive Toolkit (anteriormente CNTK) y TensorFlow.
TesorFlow	Desarrollada por Google Brain, TensorFlow es una biblioteca de código abierto para programación aritmética y aprendizaje automático a gran escala. Esta herramienta cuenta con una amplia variedad de modelos y algoritmos para Machine learning y Deep learning. Se utiliza mediante Python y permite crear modelos para el entrenamiento de datos y ejecutar redes neuronales de manera eficiente y escalable. Con TensorFlow, los desarrolladores pueden crear modelos complejos de aprendizaje automático y personalizarlos según sus necesidades. Además, esta biblioteca es altamente flexible

	y se puede utilizar en diversos proyectos, desde aplicaciones móviles hasta sistemas empresariales.
Numpy	NumPy es una de las bibliotecas de Python más utilizadas en el campo de la ciencia de datos y la inteligencia artificial debido a su capacidad para trabajar con grandes cantidades de datos de manera eficiente. Una de las características principales de NumPy es su capacidad para manejar arreglos multidimensionales o matrices, que permiten representar datos de manera más eficiente que las estructuras de datos tradicionales. Además, esta biblioteca también ofrece una gran cantidad de funciones y herramientas para el análisis y manipulación de datos, lo que la hace una herramienta muy útil para los investigadores y desarrolladores en este campo.
Matplotlib	Matplotlib es una de las librerías de Python más populares para la creación de gráficos y visualizaciones de datos en dos dimensiones. Esta herramienta ofrece una gran cantidad de opciones y funcionalidades para personalizar y diseñar gráficos de alta calidad, lo que la convierte en una opción muy versátil para presentaciones y análisis de datos en diversos campos, como la investigación científica, la ingeniería, el análisis financiero, la visualización de datos, entre otros. Además, su integración con otras librerías como NumPy y Pandas, permite la creación de gráficos a partir de grandes volúmenes de datos, lo que la hace una herramienta muy útil para el análisis y visualización de información compleja.
Optimizador "Adam"	Adam es un algoritmo de optimización basado en gradiente estocástico que se utiliza para minimizar la función de pérdida en el entrenamiento de redes neuronales. Utiliza una estimación adaptativa de los momentos de primer y segundo orden de la taza de cambio para ajustar la tasa de aprendizaje de forma individual para cada parámetro, lo que le permite adaptarse a diferentes velocidades de cambio en la función objetivo. Adam también utiliza un sesgo de momento corregido para evitar la propagación de la información de momento no deseada a través de diferentes iteraciones. En general,

	Adam es más rápido y preciso que otros optimizadores tradicionales como SGD.

Es importante tener en cuenta que los conceptos presentados en la tabla 1 pueden variar según el tipo de sistema de reconocimiento de imágenes que se utilice. Por lo tanto, es recomendable adaptar estos conceptos a las características específicas del sistema para lograr un mejor rendimiento y resultados precisos.

1.3.5. Tipo de muestreo

La recolección de datos para este proyecto se fundamenta en la utilización de bases de datos y conjuntos de datos públicos ampliamente reconocidos en el campo de la visión artificial. Estos recursos proporcionan imágenes etiquetadas y anotadas [5], lo cual facilita el entrenamiento y la evaluación de los algoritmos desarrollados. Esta elección metodológica se justifica en base a los siguientes fundamentos científicos:

- La disponibilidad de datos etiquetados y anotados en bases de datos públicas evita la necesidad de realizar una anotación manual exhaustiva de cada imagen. Este proceso puede ser laborioso y propenso a errores, por lo que contar con datos previamente etiquetados agiliza el desarrollo del proyecto [6].
- El uso de bases de datos públicas garantiza la reproducibilidad y comparabilidad de los resultados obtenidos. Al utilizar conjuntos de datos ampliamente reconocidos, se facilita la validación y verificación de los nuevos enfoques [35], así como la comparación con trabajos previos. Esto fortalece la base científica del proyecto y permite evaluar su rendimiento en relación con otras investigaciones.
- Al contar con una amplia variedad de imágenes, se obtiene una visión más completa y representativa de los objetos y situaciones que se desean analizar [37]. Esta diversidad contribuye a mejorar la generalización y capacidad de adaptación de los algoritmos a diferentes condiciones del grano de café [21].

El muestreo aleatorio simple se basa en el principio de equidad y evita sesgos de selección. Teniendo en cuenta que se tienen 4000 imágenes entre granos de café en buen estado y con defectos, se aplicó la fórmula de muestreo como se presenta en la formula I:

$$(I)\ n = \frac{\frac{z^2 * p * (1-p)}{e^2}}{1 + \left(\frac{z^2 * p * (1-p)}{e^2 N}\right)}$$

Donde:

- z: Nivel de confianza deseado.
- p: Proporción poblacional de interés.
- e: Margen de error aceptable.
- N: Tamaño de la población.

La técnica de muestreo utilizada en este proyecto fue el muestreo aleatorio simple, garantizando que cada imagen tenga la misma probabilidad de ser seleccionada, lo que evita sesgos de selección [34].

La fórmula del muestreo aleatorio simple, considerando los siguientes parámetros: un nivel de confianza del 99%, un margen de error máximo del 2% y una población total estimada de 4000 imágenes, se determinó que un conjunto de datos de 3000 imágenes sería adecuado para este proyecto.

Esta elección se justifica en base a la necesidad de obtener una muestra representativa de la población total de imágenes disponibles, asegurando al mismo tiempo un nivel de confianza estadísticamente significativo y un margen de error aceptable. Al seleccionar 3000 imágenes, se busca obtener un conjunto de datos lo suficientemente amplio como para abordar las diversas características y variaciones presentes en las imágenes y garantizar la validez y robustez de los resultados obtenidos.

CAPÍTULO 2: MATERIALES Y MÉTODOS

En este capítulo, se describe el marco metodológico utilizado en el presente estudio experimental, que tiene como objetivo clasificar imágenes de granos de café utilizando un modelo de red neuronal convolucional. Se abordarán aspectos como el tipo de investigación, la metodología empleada, la población y muestra del estudio, y la justificación de la selección del modelo MobilenetV2 para el entrenamiento.

2.1. Tipo de estudio o investigación realizada

El presente estudio se enmarca en una investigación de tipo experimental. Se ha seleccionado este enfoque debido a su capacidad para establecer relaciones de causalidad y permitir la manipulación controlada de variables. Además, a través de este diseño, se busca evaluar el rendimiento y la eficacia del modelo de red neuronal convolucional en la clasificación de imágenes de granos de café.

Según estudios realizados, han llegado a la conclusión de que la calidad de respuestas en un sistema de clasificación de imágenes, dependen de la cantidad de imágenes con el cual el modelo se entrena [7]. Y en caso de existir pocos datos para el entrenamiento de una red, se pueden manipular los mismos datos utilizando librerías especializadas (openCV) en dimensionamiento de imágenes, para de esta manera entrenar con datos más exactos una red neuronal.

Hay que tener en cuenta que este proyecto está destinado en la utilización de un modelo de red neuronal convolucional, para poder manipular imágenes de granos de café y así poder clasificarlos. Por lo tanto, las pruebas en el estudio de los resultados tendrán un punto fuerte para conocer la cantidad de épocas necesarias que se necesita en el entrenamiento de una red artificial y así finalmente conseguir un porcentaje aproximado al 90%.

Como referencia a la propuesta del estudio de [36], sugiere que se puede crear un algoritmo de visión artificial de bajo costo, implementando una arquitectura básica de clasificación de imágenes. Para comprender mejor el argumento anterior se puede observar la ilustración 5, donde muestra cómo se implementó el algoritmo de visión artificial en un diagrama de bloques para una mejor comprensión.

En el estudio realizado por el autor, se observó que tanto el sistema de clasificación manual como el sistema de clasificación artificial presentaron resultados similares, lo que sugiere que la implementación de un clasificador de imágenes para la clasificación de

granos de café pueden ser una opción beneficiosa y eficiente en términos de calidad del producto. Además, el uso de un sistema de clasificación de imágenes automatizado puede ahorrar tiempo y recursos, lo que puede resultar en una mayor productividad y rentabilidad en la industria cafetera.

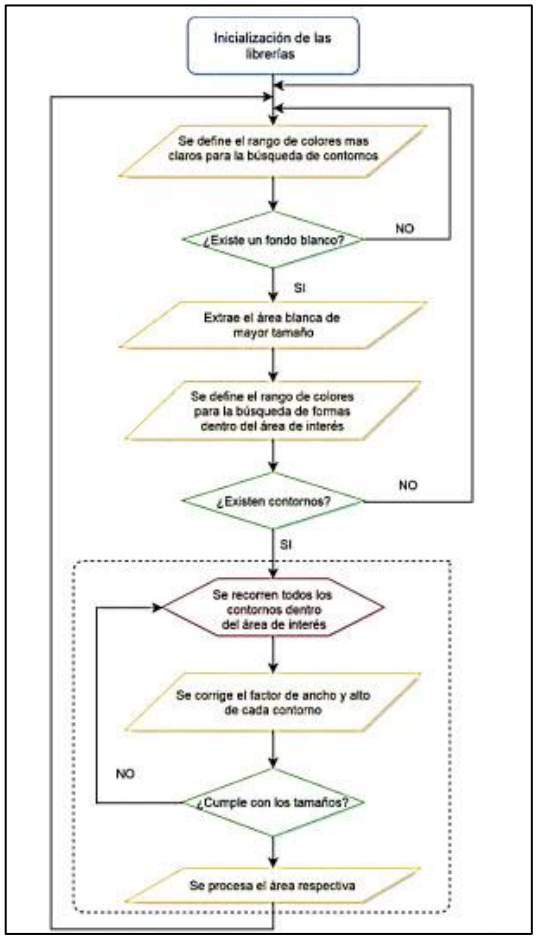


Ilustración 5. Diagrama de flujo de un sistema de clasificación de granos de café

2.2. Clasificación en los tipos de granos de café

En la industria del café, es importante contar con una selección de granos de alta calidad, ya que esto impacta directamente en la calidad del producto final. Los tres tipos de defectos más comunes que se han considerado en este estudio son el daño por insectos, los granos de tamaño irregular y los granos dañados mecánicamente. Estos defectos pueden ser identificados visualmente y son una indicación de la calidad de los granos. Por lo tanto, es importante tener un método confiable para detectar y clasificar estos defectos en los granos de café. En este sentido, el aprendizaje automático y las técnicas

de procesamiento de imágenes se han vuelto cada vez más relevantes en la industria del café para mejorar la calidad y la eficiencia del proceso de clasificación de granos. En la tabla 2 se presentan ejemplos de las imágenes que serán procesadas por el sistema para el entrenamiento del modelo de clasificación de granos de café.

Tabla 2. Tipos de granos de café, utilizados en el algoritmo

		
Grano de café bueno	Grano de café malo	Grano de café trizado o roto

Para el entrenamiento de los datos en este estudio, se han elegido conjuntos de imágenes (datasets) disponibles en un repositorio de datos, los cuales permitirán validarlos mediante un proceso estadístico utilizando la plataforma TensorFlow. Asimismo, se llevarán a cabo pruebas experimentales para obtener más información sobre los resultados obtenidos. Es importante destacar que el proceso de maduración del grano de café puede dar lugar a enfermedades, por lo que se busca que el sistema sea capaz de identificar estos defectos y clasificar los granos entre defectuosos y no defectuosos en función a las

enfermedades que puedan tener. Este enfoque permite asegurar una mejor calidad de los granos de café en la fase de producción y seleccionar solo aquellos que cumplan con los estándares de calidad establecidos en la industria.

En el estudio referenciado [34], se destacaron los defectos más comunes en los granos de café, por lo tanto a continuación se enumera algunas de las enfermedades:

1) Mantequillo: Grano de color amarillo traslucido.
2) Broca: Grano con pequeños orificios.
3) Reposado: Grano con alteraciones en su color normal, presenta colores que van desde el blanqueado, crema, amarillo hasta la carmelita.
4) Negro total o parcial: Grano con tonalidades entre marrón y negro.
5) Cardenillo: Grano atacado por hongos, recubierto de polvillo amarillo o amarillo rojizo.
6) Mordido: Grano con una herida o cortada y oxidado.
7) Vinagre: Grano con coloración desde el color crema al carmelita oscuro, película plateada puede tender a coloración pardo rojizas.
8) Inmaduro: Grano de color verdoso o gris claro, la cutícula no desprende.
9) Aplastado: Grano aplanado con fracturas parciales.
10) Flojo: Grano de color gris oscuro y blando.

Para complementar la información anterior, se puede revisar la ilustración 6 que muestra las enfermedades de los granos de café.

Ilustración 6. Enfermedades que pueden contraer los granos de café

2.3. Redes neuronales en la clasificación de imágenes

La red neuronal es una serie de algoritmos matemáticos que se utilizan para reconocer patrones complejos y procesar datos en paralelo como se referencia en [10]. Una red neuronal típica consta de varias capas, cada una compuesta por neuronas. Las neuronas reciben entradas, realizan cálculos y generan salidas a las neuronas de la capa siguiente.

En el caso específico de clasificación de imágenes de granos de café con defectos, la red neuronal utiliza técnicas de aprendizaje supervisado para aprender a identificar los diferentes tipos de defectos [5]. El aprendizaje supervisado implica el entrenamiento de la red con un conjunto de imágenes previamente etiquetadas con sus respectivas clases (granos defectuosos y granos sanos) [36].

El proceso de entrenamiento de una red neuronal implica la optimización de los pesos y sesgos de las neuronas para minimizar el error entre las salidas esperadas y las salidas reales [17] y [5]. Esto se logra mediante el uso de técnicas como la propagación hacia atrás del error (backpropagation), que ajusta los pesos y sesgos en función de la diferencia entre las salidas esperadas y las reales.

Una de las técnicas de clasificación de imágenes más comúnmente utilizadas en redes neuronales es la red neuronal convolucional (CNN) [38]. Esta técnica se basa en el uso de filtros convolucionales que aprenden a detectar características específicas de la imagen, como bordes, formas y patrones. Los resultados de los filtros se agrupan y alimentan a capas completamente conectadas que realizan la tarea de clasificación final. Según [14] la ecuación que se presenta a continuación, hace referencia a la implementación de redes neuronales convolucionales para el procesamiento de imágenes.

$$(1).\ \gamma = f(w * x + b)$$

Donde "ϒ" es la salida de la neurona, "x" son las entradas, "w" son los pesos de las conexiones entre las neuronas, "b" es el sesgo y "f" es una función de activación no lineal, como la función sigmoide.

Para lograr los resultados esperados, las redes neuronales utilizadas en la clasificación de imágenes, como las de granos de café con defectos, suelen contar con más capas que una red neuronal convencional. Cada capa funciona como un filtro de información, extrayendo características específicas de las imágenes con las que el modelo se entrena (revisar la ilustración 7). Es importante destacar que la cantidad de datos necesarios para este proceso puede ser considerable, ya que se requiere de una gran cantidad de imágenes etiquetadas para el entrenamiento supervisado de la red neuronal.

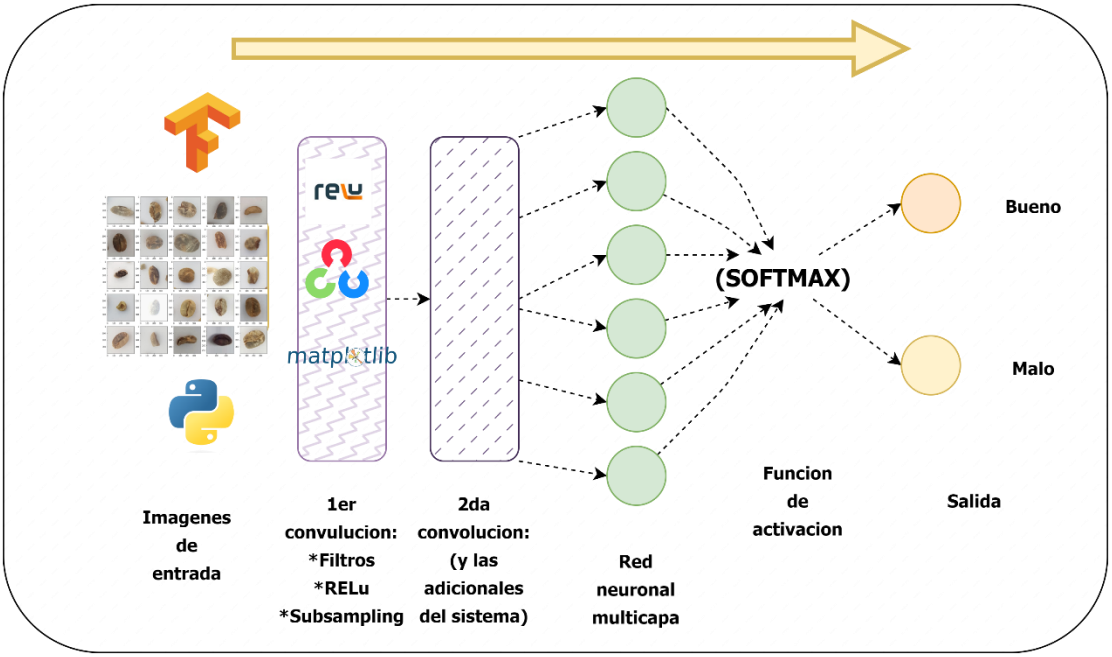


Ilustración 7. Arquitectura CNN

La arquitectura que se muestra en la ilustración 7 hace referencia a un modelo de clasificación de imágenes, donde las imágenes son granos de café en buen estado y granos

de café en mal estado. La imagen que ingresa al modelo pasa por dos capas convolucionales ocultas antes de llegar a la capa final de entrenamiento, donde la activación "softmax" decide a qué categoría pertenece la imagen.

2.3.1. ¿Cómo una imagen ingresa a un modelo de CNN?

La imagen a color cuenta con 3 canales de información, o también llamados conjuntos de colores, estos son: Red, Green, Blue [2].

La convolución es el bloque de construcción principal de CNN. El término convolución se refiere a la combinación matemática de dos funciones para producir una tercera función, es decir, combina dos conjuntos de información [14].

En el caso de imágenes, se utiliza la matriz de combinación de sus dimensiones (I), donde se multiplica el alto (H) por el ancho (W) de los pixeles de una imagen y la cantidad de canales de colores que se representan en un formato RGB (red, green, blue), de tal forma que se tiene:

$$(2).\ I \in \mathbb{R}^{HxWxC}$$

El banco de filtros o también denominado núcleo (K), se lo representa en la ecuación:

$$(3).\ K \in \mathbb{R}^{k_1 x k_2 x C x D}$$

en donde $k_1 x k_2$ se refiere a las dimensiones (renglones, columnas) de cada filtro, dando de esta manera un valor al sesgo para cada capa oculta.

$$(4).\ b \in \mathbb{R}^{D}$$

La salida del procesamiento de convolución es el indicado en la ecuación 5:

$$(5).\ (I * K)_{i,j} = \sum_{m=0}^{k_1-1} \sum_{m=0}^{k_1-1} \sum_{m=0}^{k_1-1} K_{m,n,p} \,.\, K_{i+m,j+n,c} + b$$

2.3.2. MobileNetV2

El objetivo de este proyecto con MobilenetV2, es utilizar su algoritmo de convolución para crear un modelo de red neuronal convolucional que funciona con el conjunto de imágenes que ingresarán a este programa, para así lograr resultados convenientes para la clasificación de imágenes basándose en granos de café.

Según [37], los datos y características matemáticas de la arquitectura del algoritmo de Mobile Net versión 2 se presentan en la tabla 3.

Tabla 3. Datos y características de la arquitectura MobilleNetV2 propuesto por [37]

Input	Operador	Output

$\boldsymbol{h} * \boldsymbol{w} * \boldsymbol{k}$	1*1 conv2d, ReLU6	$h * w * (k)$
$\boldsymbol{h} * \boldsymbol{w} * \boldsymbol{k}$	3*3 dwise s=s, ReLU6	$\frac{h}{s} * \frac{w}{s} * (tk)$
$\frac{\boldsymbol{h}}{\boldsymbol{s}} * \frac{\boldsymbol{w}}{\boldsymbol{s}} * \boldsymbol{tk}$	Linear 1*1 conv2d	$\frac{h}{s} * \frac{w}{s} * k'$

La arquitectura de MobileNetV2 consta de una capa inicial de convolución con 32 filtros, seguida de 19 capas residuales de cuello de botella. Para obtener una no linealidad robusta y eficiente en términos de computación, se utiliza la función de activación ReLU6 en lugar de la ReLU convencional. Además, el uso del kernel 3*3 es un estándar común en las redes neuronales modernas, ya que ha demostrado ser efectivo en la extracción de características de las imágenes de manera eficiente y precisa. Esta arquitectura ha demostrado ser altamente efectiva en aplicaciones de visión por computadora en dispositivos móviles y con recursos limitados [37].

2.3.1. Justificación de la selección del modelo MobileNetV2

La elección del modelo MobilenetV2 para el entrenamiento se basó en varios factores. En primer lugar, MobilenetV2 es conocido por su eficiencia computacional, lo que lo hace adecuado para aplicaciones en dispositivos con recursos limitados[38]. Dado que el procesamiento de imágenes en tiempo real es un requisito en este estudio, se consideró fundamental seleccionar un modelo que pudiera cumplir con esta demanda.

Además, MobilenetV2 ha demostrado un buen desempeño en tareas de clasificación de imágenes, especialmente en la detección de objetos y características visuales relevantes. Investigaciones previas han respaldado su eficacia y precisión en la clasificación de imágenes de objetos similares a los granos de café [35].

Asimismo, la arquitectura de MobilenetV2 se ha optimizado para extraer características significativas de las imágenes, incluso en situaciones donde los datos son limitados [37]. Dado que el presente estudio se enfrenta a la disponibilidad de una cantidad limitada de imágenes de granos de café, la capacidad de MobilenetV2 para aprovechar al máximo estos datos es un factor crucial para lograr un modelo de clasificación preciso y confiable.

2.4. Metodología

En este apartado, se presenta la metodología utilizada en el presente estudio, el cual se enfoca en la clasificación de imágenes de granos de café utilizando un modelo de red neuronal convolucional.

Para llevar a cabo este estudio, se recopilaron imágenes de granos de café de diversas fuentes, considerando características relevantes como variedades de café, condiciones de cultivo y estados de madurez. Se establecieron criterios de selección para garantizar la representatividad de la muestra y minimizar sesgos potenciales. La muestra final consta de 3000 imágenes de granos de café en diferentes estados.

2.4.1. Recolección de datos

Para garantizar la integridad y calidad de los datos utilizados en las pruebas y entrenamientos del algoritmo, se optó por utilizar imágenes que fueran de dominio público y que estuvieran disponibles en repositorios de imágenes con licencias abiertas y permisos claros para su uso. De esta manera, se asegura que los datos fueran confiables y no hubiera problemas de derechos de autor o licencias restrictivas.

Algunos de los datos utilizados en el sistema de reconocimiento de imágenes se pueden mostrar en la tabla 4, donde se podrá denotar imágenes de granos de café en buen estado y en mal estado [6].

Tabla 4. Dataset de imágenes de granos de café

Granos de café buenos

Granos de café defectuosos

Granos de café con imperfecciones y enfermedades

El proyecto utiliza un conjunto de datos que consta de 3000 imágenes de granos de café, cada imagen se ingresa al modelo con un tamaño de 350*350 pixeles, las cuales se dividirán en dos conjuntos: un 80% para entrenamiento y un 20% para validación [21]. El número de épocas utilizado en el entrenamiento se ajustará según los resultados obtenidos mediante pruebas estadísticas, con el objetivo de optimizar el rendimiento del modelo. Será necesario realizar pruebas con diferentes tamaños de muestra para determinar la cantidad óptima de imágenes requeridas para el entrenamiento.

2.4.2. Población y muestra

La elección de una muestra de 3000 imágenes de granos de café, dividida en un 80% para entrenamiento y un 20% para validación, se basa en consideraciones científicas y metodológicas que buscan garantizar la representatividad de los datos y la evaluación confiable del modelo de clasificación de imágenes.

En primer lugar, la muestra de 3000 imágenes se selecciona con el objetivo de abarcar una variedad suficiente de granos de café, considerando diferentes variedades, condiciones de cultivo y estados de madurez [39]. Una muestra de este tamaño permite

capturar la diversidad de características visuales presentes en los granos de café, lo que facilita el entrenamiento y la evaluación del modelo en una amplia gama de casos.

La división de la muestra en un 80% para entrenamiento y un 20% para validación sigue una práctica común en el campo del aprendizaje automático y la clasificación de imágenes [40]. Esta proporción proporciona una cantidad suficiente de datos para entrenar el modelo de manera efectiva, permitiendo que aprenda patrones y características relevantes de las imágenes. Al mismo tiempo, el conjunto de validación se reserva para evaluar el desempeño del modelo en datos no vistos durante el entrenamiento, lo que proporciona una medida confiable de su capacidad de generalización.

Además, el tamaño de imagen de 350x350 píxeles se selecciona teniendo en cuenta la resolución suficiente para capturar los detalles importantes de los granos de café, al mismo tiempo que se mantiene un equilibrio entre la calidad de los datos y la carga computacional requerida para el procesamiento y entrenamiento del modelo [41]. Esta resolución permite representar adecuadamente las características visuales de los granos de café y asegurar una clasificación precisa.

En conjunto, la elección de esta muestra de 3000 imágenes y la proporción de división de entrenamiento y validación del 80-20% se basan en principios metodológicos establecidos y buscan proporcionar una base sólida para el desarrollo y la evaluación del modelo de clasificación de imágenes de granos de café.

2.4.3. Evaluación y experimentación del modelo

La implementación de TensorBoard en el proyecto de clasificación de imágenes de granos de café ha sido una herramienta crucial para monitorear métricas en torno a la precisión de las funciones de pérdida en los entrenamientos del modelo de aprendizaje automático[42]. TensorBoard establece los gráficos estadísticos necesarios para comprender si el modelo está correctamente entrenado o si es necesario realizar más ciclos de entrenamiento, todo esto basándose en los datos de entrenamiento y validación de las redes profundas.

En el proyecto, se han utilizado diversas plataformas compatibles con TensorBoard para crear proyectos de visión artificial, como: Jupyter Notebook, Google Colab y PyCharm. Además, se han utilizado varias técnicas de procesamiento de datos (manipulación de imágenes con librerías como: openCV) para optimizar la precisión del modelo, incluyendo técnicas de aumento de datos para aumentar la cantidad de datos de entrenamiento disponibles.

Los resultados obtenidos a través de la implementación de TensorBoard han sido fundamentales para la evaluación y selección de un modelo adecuado para el proyecto de clasificación de imágenes de granos de café. Estos resultados se han presentado en forma de gráficos y tablas detallando las métricas de precisión y pérdida en cada ciclo de entrenamiento. La finalidad de esto es realizar una evaluación asertiva para el rendimiento del modelo.

CAPÍTULO 3: DESARROLLO Y RESULTADOS

En el presente capítulo se detalla paso a paso el proceso que se llevó a cabo para realizar el algoritmo de clasificación de imágenes.

3.1. Arquitectura del trabajo

Las capas ocultas del modelo de MobileNetV2 están entrenadas y listas para ser utilizadas. Sin embargo, no funcionan para cualquier sistema clasificador, por lo que se tienen que configurar aquellas capas densas de entrada y de salida tal y como se muestra en la ilustración 8.

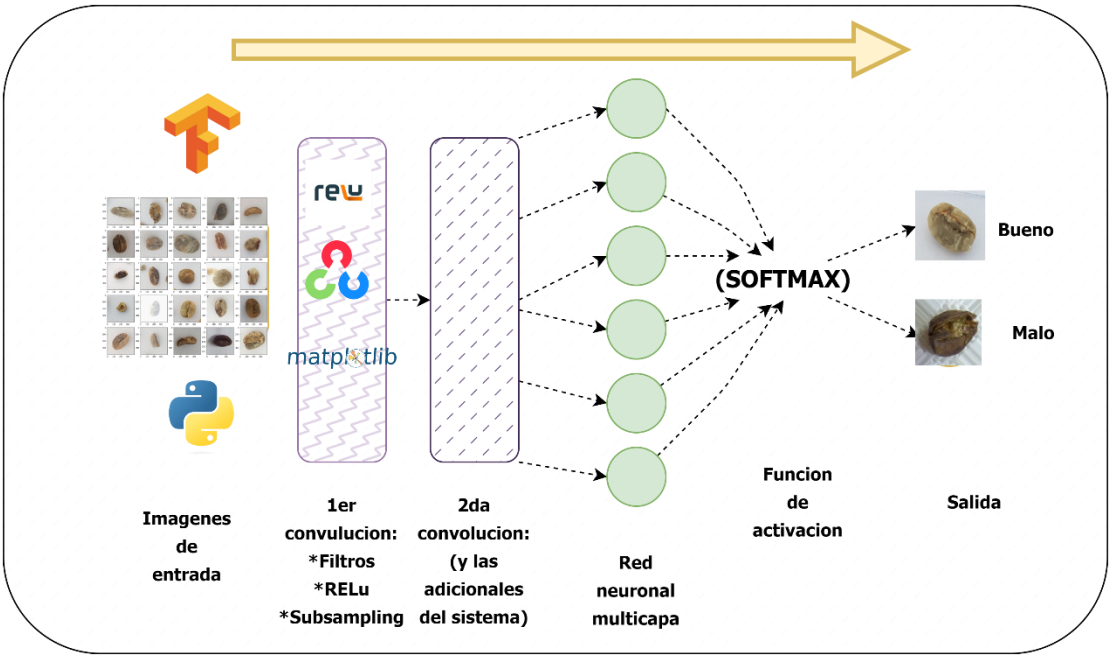


Ilustración 8. Representación gráfica del algoritmo MobilNetV2 en el proyecto

Se congelan (el termino congelar se utiliza para referirse a que se va a detener momentáneamente un funcionamiento) las capas ocultas del modelo y se ingresan los datos, una vez que se tienen listos los datos, descongelar el modelo y ponerlo a aprender. Esta la forma más práctica para poner en funcionamiento el propósito de este trabajo.

Una vez congeladas (ilustración 9) las capas densas, se pasa a la configuración de la salida y presentación de respuesta del sistema.

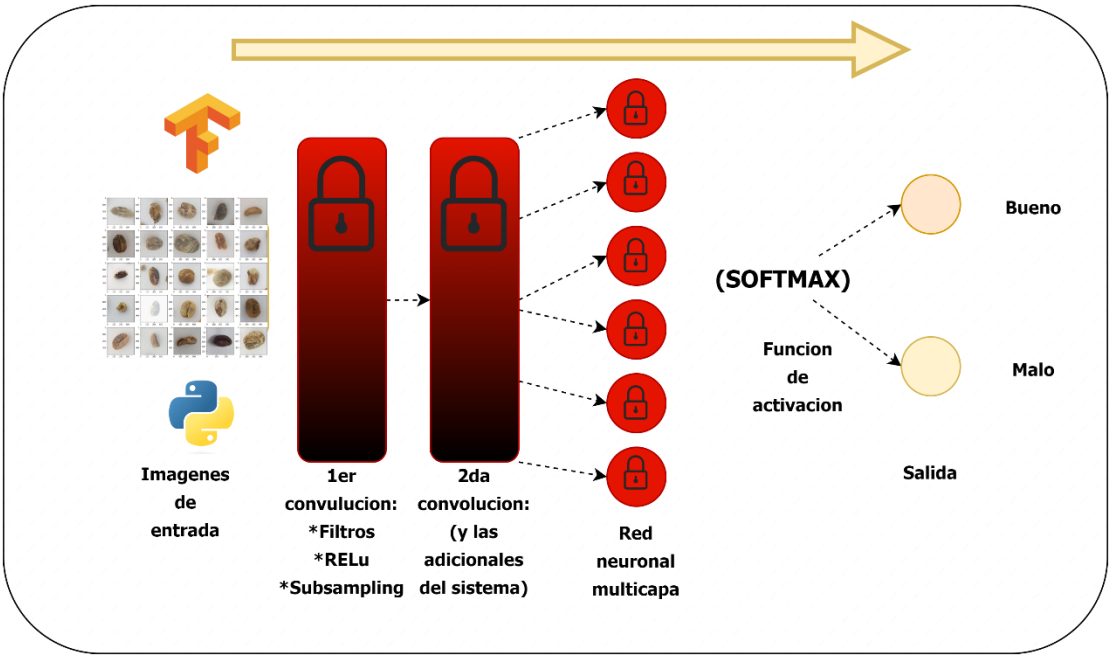


Ilustración 9. Arquitectura del sistema, con las capas ocultas bloqueadas

Una vez ingresada la data y configurado los parámetros de salida, se descongelan (ilustración 10) las capas ocultas y se pasa a entrenar el modelo completo.

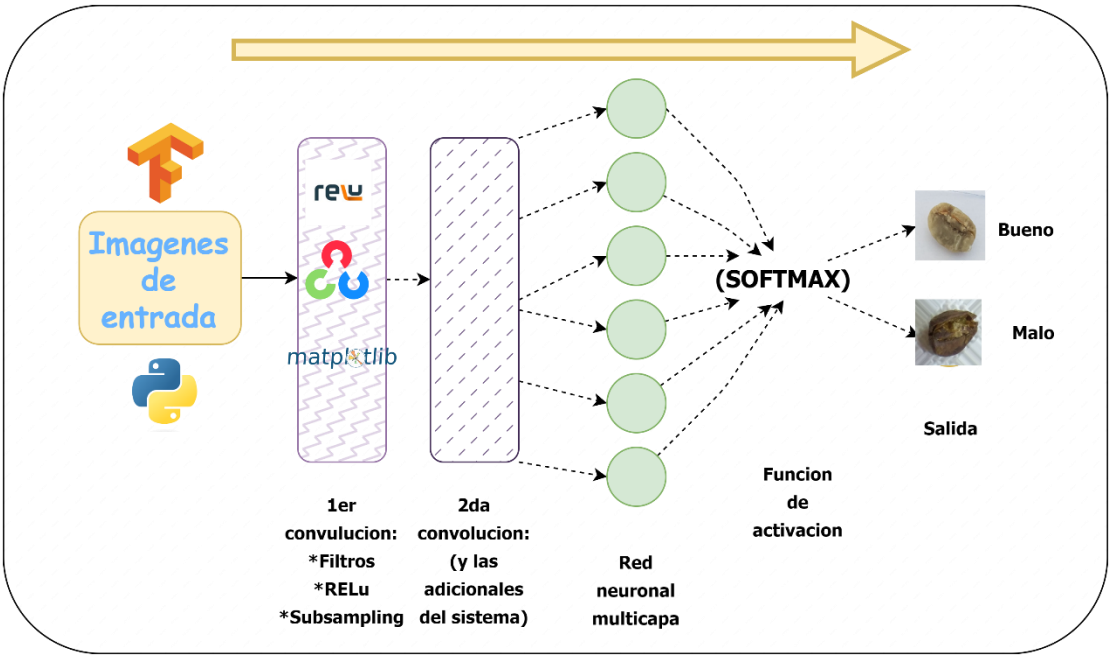


Ilustración 10. Arquitectura de un sistema de visión artificial clasificador de café

Para evaluar la precisión del modelo, se utilizará el diagrama estadístico de pérdidas generado por TensorBoard al final del entrenamiento, que permitirá identificar la convergencia del modelo y detectar posibles problemas de sobreajuste o sub-ajuste. Una vez que se alcance la precisión esperada y se compruebe la estabilidad del modelo, se

considerará que el modelo es funcional y estaría listo para ser implementado en otras aplicaciones o plataformas para realizar predicciones precisas y confiables. Es importante destacar que este proceso es fundamental en la creación de un modelo de aprendizaje automático exitoso y es necesario realizarlo de manera exhaustiva para garantizar su calidad y eficacia.

3.2. Desarrollo del sistema clasificador de imágenes

Para la implementación del algoritmo clasificador, se utilizó el entorno de trabajo Google Colab en el explorador debido a su facilidad de uso y la disponibilidad de recursos computacionales. Google Colab ofrece una infraestructura en la nube con procesamiento en GPU, lo que permite una mayor velocidad en el entrenamiento de modelos de aprendizaje profundo. Además, el entorno de trabajo es compatible con múltiples lenguajes de programación, incluyendo Python y sus principales bibliotecas para aprendizaje automático. La selección de Google Colab como entorno de trabajo para el proyecto se basó en su capacidad para ejecutar eficientemente el algoritmo clasificador y en su facilidad de integración con herramientas como TensorBoard y GitHub para el monitoreo y control del proceso de desarrollo (revisar anexo B).

3.2.1. Definición del dataset

El algoritmo del sistema se basa en el conjunto de imágenes utilizado para entrenar el modelo (revisar la ilustración 11), lo que es crucial para su eficacia y rendimiento. Para la recolección de imágenes, se utilizó un enfoque de búsqueda en línea utilizando motores de búsqueda de imágenes disponibles públicamente, junto con la recopilación de un conjunto de imágenes personalizado para la tarea específica. Para garantizar la calidad y diversidad del conjunto de imágenes, se aplicaron criterios rigurosos para seleccionar solo las imágenes relevantes y adecuadas. Esto permitió la creación de un conjunto de datos de alta calidad y variado, necesario para el entrenamiento efectivo del clasificador.

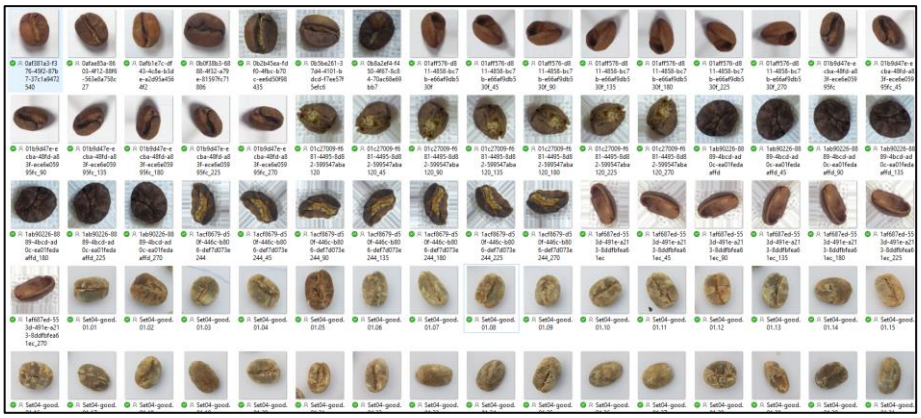


Ilustración 11. Imágenes aleatorias del dataset a utilizar

Es importante destacar que, en esta etapa de recolección de datos, el enfoque principal es adquirir una amplia variedad de imágenes para proporcionar una mayor diversidad de ejemplos al modelo. La selección de etiquetas y tamaños de las imágenes puede ser realizada posteriormente con el uso de bibliotecas de procesamiento de imágenes, como OpenCV y PIL, que permiten la manipulación y transformación de imágenes. Además, es importante tener en cuenta que la calidad de las imágenes también juega un papel fundamental en el entrenamiento del modelo, por lo que se debe asegurar que las imágenes recolectadas tengan una resolución adecuada y una iluminación uniforme para evitar la introducción de ruido y variabilidad en el modelo.

Una vez que se tienen todas las imágenes correspondientes al dataset, se procede a comprimir las imágenes en carpetas divididas por las clases que se van a utilizar en el modelo (ilustración 12). Es importante mencionar que el número de imágenes dentro de cada carpeta depende del tipo de grano de café, con algunas carpetas en donde su contenido era de 1200 imágenes y otras carpetas con 2911 imágenes. Sin embargo, para un procesamiento uniforme posterior en el código, se estandariza un número total de imágenes para cada clase. Es necesario destacar que todas las imágenes tienen una dimensión máxima de 600*600 pixeles para garantizar una eficiente manipulación y procesamiento de datos en el sistema.

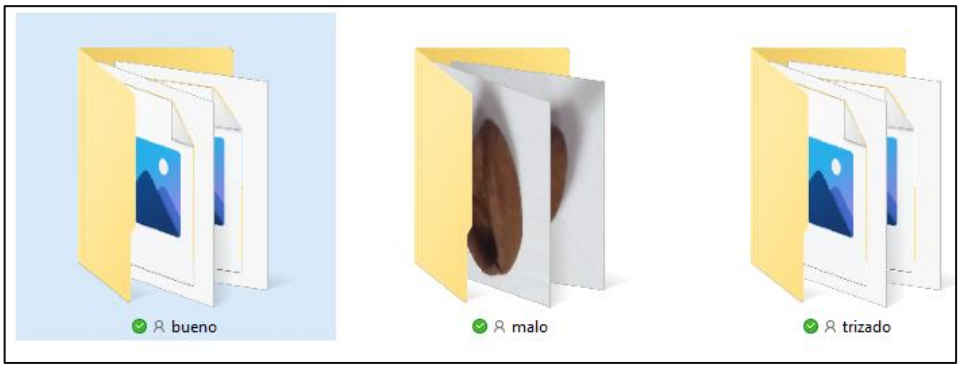


Ilustración 12. Carpetas contenedoras de imágenes

3.2.2. Importación de archivos

Los archivos se los importa al entorno de trabajo de la siguiente manera:

1. Se crean los directorios donde se va a subir las carpetas comprimidas (ilustración 13).

```
[ ]  #Crear las carpetas para subir las imagenes
     !mkdir bueno
     !mkdir malo
     !mkdir trizado
```

Ilustración 13. Comandos para crear carpetas que contendrán las imágenes de los granos de café

2. Se pasa a descomprimir los archivos zip, subidos anteriormente, en donde contienen las imágenes del proyecto (ilustración 14).

```
[ ]  %cd bueno
     !unzip bueno.zip
     %cd ..

     %cd malo
     !unzip malo.zip
     %cd ..

     %cd trizado
     !unzip trizado.zip
     %cd ..
```

Ilustración 14. Comandos para descomprimir carpetas contenedoras

3. Se revisa si se subieron correctamente las imágenes, al igual que conocer el número exacto de elementos en cada carpeta (ilustración 16).

```
[ ]  #Mostrar cuantas imagenes tengo de cada categoria
     !ls /content/bueno | wc -l #2149
     !ls /content/malo | wc -l #2149
     !ls /content/trizado | wc -l #2149

     2149
     2911
     1284
```

Ilustración 15. Comandos para conocer el número de elementos en una carpeta

4. Se realiza una verificación de que las imágenes han sido correctamente cargadas al código mediante la ejecución de los siguientes comandos (ilustración 16).

```
[ ]  #Mostrar algunas imagenes con pyplot
     import os
     import matplotlib.pyplot as plt
     import matplotlib.image as mpimg

     plt.figure(figsize=(15,15))

     carpeta = '/content/malo'
     imagenes = os.listdir(carpeta)

     for i, nombreimg in enumerate(imagenes[:25]):
       plt.subplot(5,5,i+1)
       imagen = mpimg.imread(carpeta + '/' + nombreimg)
       plt.imshow(imagen)
```

Ilustración 16. Uso de librerías matplotlib para visualizar imágenes cargadas

En la ilustración 17 se podrá notar algunas de las imágenes que contienen las carpetas cargadas.

Ilustración 17. Imágenes cargadas en el sistema antes del entrenamiento de datos

5. Como se pudo notar en los pasos anteriores, la creación de un dataset equilibrado es importante para evitar el sesgo del modelo en favor de alguna clase específica de imágenes. Para ello, se recomienda crear un dataset con un número de elementos equivalente entre las diferentes carpetas, para que el algoritmo trabaje de manera más eficiente y no se sature en la clasificación de una clase específica. Además, al contar con un dataset equilibrado se asegura una mejor generalización del modelo en la clasificación de nuevas imágenes (ver ilustración 18).

```
[ ]  #Crear carpetas para hacer el set de datos

     !mkdir dataset
     !mkdir dataset/bueno
     !mkdir dataset/malo
     !mkdir dataset/trizado
```

Ilustración 18. Comandos para crear nuevas carpetas contenedoras

Para lograr este paso se tiene que crear nuevas carpetas para los futuros archivos que van a ser seleccionados (ver ilustración 19).

```
#GRANOS BUENOS
import shutil
carpeta_fuente = '/content/bueno'
carpeta_destino = '/content/dataset/bueno'

imagenes = os.listdir(carpeta_fuente)

for i, nombreimg in enumerate(imagenes):
  if i < 1000:
    #Copia de la carpeta fuente a la destino
    shutil.copy(carpeta_fuente + '/' + nombreimg, carpeta_destino + '/' + nombreimg)
```

Ilustración 19. Comandos para copiar un cierto número de elementos, de una carpeta a otra (Imágenes granos buenos)

Se copian las imágenes necesarias, para este proyecto se utilizarán 1000 imágenes en cada contenedor del dataset.

Este conjunto de comandos se replica para todos los conjuntos de datos, granos con imperfecciones (ilustración 20) y granos trizados (ilustración 21).

```
#GRANOS MALOS
import shutil
carpeta_fuente = '/content/malo'
carpeta_destino = '/content/dataset/malo'

imagenes = os.listdir(carpeta_fuente)

for i, nombreimg in enumerate(imagenes):
  if i < 1000:
    #Copia de la carpeta fuente a la destino
    shutil.copy(carpeta_fuente + '/' + nombreimg, carpeta_destino + '/' + nombreimg)
```

Ilustración 20. Comandos para copiar un cierto número de elementos, de una carpeta a otra (Imágenes granos con defectos)

```
#GRANOS TRIZADOS
import shutil
carpeta_fuente = '/content/trizado'
carpeta_destino = '/content/dataset/trizado'

imagenes = os.listdir(carpeta_fuente)

for i, nombreimg in enumerate(imagenes):
  if i < 1000:
    #Copia de la carpeta fuente a la destino
    shutil.copy(carpeta_fuente + '/' + nombreimg, carpeta_destino + '/' + nombreimg)
```

Ilustración 21. Comandos para copiar un cierto número de elementos, de una carpeta a otra (Imágenes granos trizados)

6. En este paso, se verifica el estado de las nuevas carpetas creadas y se comprueba el número de elementos en cada una de ellas. Es importante destacar que se limitó el número de elementos a 1000 imágenes de granos de café en cada carpeta, con la finalidad de trabajar de manera más eficiente en la implementación del algoritmo (ver ilustración 22).

```
[ ]  #Mostrar cuantas imagenes tengo de cada categoria en el dataset
     !ls /content/dataset/bueno | wc -l #1000 elementos
     !ls /content/dataset/malo | wc -l #1000 elementos
     !ls /content/dataset/trizado | wc -l #1000 elementos
```

Ilustración 22. Comandos para revisar el número de elementos en las nuevas carpetas del dataset

De esta manera ya estaría listo el set de datos para la configuración de la red neuronal.

3.2.3. Aumento de datos

La técnica de aumento de datos en las redes neuronales convolucionales se utiliza cuando se dispone de un número limitado de imágenes para el entrenamiento y validación del algoritmo [42]. Es por esto, que resulta crucial aplicar este paso para mejorar la capacidad de modelar las imágenes, añadiéndoles características como forma, dimensión, rotación,

entre otras configuraciones. Para llevar a cabo esta tarea se utiliza la biblioteca de Tensor Flow, que cuenta con comandos como “ImageDataGenerator” (ilustración 23), el cual permite la manipulación de los datos y, de esta manera, incrementar el número de imágenes para el entrenamiento del algoritmo.

```
#Aumento de datos con ImageDataGenerator
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import numpy as np

#Crear el dataset generador
datagen = ImageDataGenerator(
    rescale=1. / 255,
    rotation_range = 30,
    width_shift_range = 0.25,
    height_shift_range = 0.25,
    shear_range = 15,
    zoom_range = [0.5, 1.5],
    validation_split=0.2 #20% para pruebas
)

#Generadores para sets de entrenamiento y pruebas
data_gen_entrenamiento = datagen.flow_from_directory('/content/dataset', target_size=(224,224),
                                                     batch_size=32, shuffle=True, subset='training')
data_gen_pruebas = datagen.flow_from_directory('/content/dataset', target_size=(224,224),
                                                     batch_size=32, shuffle=True, subset='validation')

#Imprimir 10 imagenes del generador de entrenamiento
for imagen, etiqueta in data_gen_entrenamiento:
  for i in range(25):
    plt.subplot(5,5,i+1)
    plt.xticks([])
    plt.yticks([])
    plt.imshow(imagen[i])
  break
plt.show()
```

Ilustración 23. Uso de las librerías numpy y los comandos de "ImageDataGenerator", para redimensionar las imágenes antes de su entrenamiento

En cuanto al resultado de la manipulación de imágenes, se puede observar los cambios realizados dentro del set de datos en la ilustración 23, donde se puede observar que el enfoque de los granos de café es diferente en comparación a los datos de la ilustración 16. En esta etapa, se han utilizado herramientas y técnicas de aumento de datos mediante el uso de la biblioteca "ImageDataGenerator" de TensorFlow. Estos cambios incluyen variaciones en la forma, dimensión, orientación y otros aspectos que permiten mejorar la calidad y la cantidad de datos de entrenamiento disponibles para el modelo de red neuronal convolucional. Este proceso es fundamental para lograr una mayor precisión y eficiencia en la clasificación de las imágenes de granos de café.

Ilustración 24. Imágenes alteradas con la función ImageDataGenerator

3.2.4. Importación del modelo MobileNetV2

Gracias a la plataforma de TensorFlow, se puede acceder a una gran variedad de modelos de redes neuronales convolucionales pre-entrenadas. Es necesario buscar el que mejor se adapte al proyecto y luego exportarlo al código de programación para su utilización (ver ilustración 25).

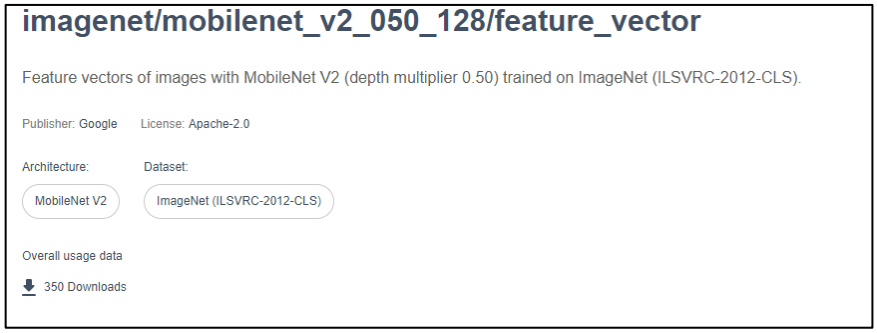


Ilustración 25. Captura del modelo descargable en la página TensorFlow hub

Se procede a llamar al modelo de MobileNetV2 desde su enlace, una vez que se importe las librerías de TensorFlow y TensorFlow hub (ver ilustración 26).

```
[ ]  import tensorflow as tf
     import tensorflow_hub as hub

     url = "https://tfhub.dev/google/tf2-preview/mobilenet_v2/feature_vector/4"
     mobilenetv2 = hub.KerasLayer(url, input_shape=(224,224,3))
```

Ilustración 26. Importación del modelo MobileNetV2

En este punto del proceso, se debe tener en cuenta la importancia de congelar las capas densas del modelo pre-entrenado (ilustración 27). Al congelar estas capas, se permite la manipulación de las capas de salida según sea necesario para el proyecto en cuestión. Esto es especialmente relevante en el contexto de redes neuronales convolucionales, donde se busca adaptar el modelo pre-entrenado a una tarea específica. La congelación de capas también es útil para evitar el sobreajuste, ya que se impide la actualización de los pesos y sesgos en estas capas durante el proceso de entrenamiento.

```
[ ]  #Congelar el modelo descargado
     mobilenetv2.trainable = False
```

Ilustración 27. Bloqueo de las capas ocultas del modelo MobileNetV2

3.2.5. Importación del modelo MobileNetV1

El algoritmo MobileNetV1 se presenta como una alternativa efectiva para el procesamiento de imágenes en aplicaciones de aprendizaje automático. Es un modelo de red neuronal convolucional de arquitectura liviana, diseñado específicamente para su implementación en dispositivos móviles con recursos limitados. El proceso de uso de este algoritmo implica la importación del dataset, lo que resulta en una representación numérica de las imágenes, y el entrenamiento del modelo mediante el ajuste de los pesos de las capas de la red neuronal. En comparación con su sucesor MobileNetv2, MobileNetV1 se enfoca en maximizar la eficiencia computacional a través de técnicas como la separación de convoluciones, la eliminación de conexiones redundantes, y el uso de filtros de tamaño reducido (ilustración 28).

```
[ ]  import tensorflow as tf
     import tensorflow_hub as hub

     url = "https://tfhub.dev/google/imagenet/mobilenet_v1_050_128/feature_vector/5"
     mobilenetv1 = hub.KerasLayer(url, input_shape=(224,224,3))
```

Ilustración 28. Importación del modelo MobileNetV1 en el sistema de clasificación de imágenes

3.2.6. Entrenamiento y muestra de resultados

A continuación, se detallarán las subsecciones que acompañarán en la explicación del proceso de aplicación del modelo MobilNetV1 en la implementación del sistema de clasificación de imágenes. Este modelo, basado en una arquitectura de red neuronal convolucional (CNN) de bajo costo computacional y alta eficiencia, será utilizado para obtener resultados precisos en la clasificación de los diferentes tipos de granos de café, considerando sus distintos grados de calidad y los posibles defectos presentes en los

mismos. Se explicará detalladamente cada paso del proceso, desde la carga y preprocesamiento de los datos, hasta la evaluación del rendimiento del modelo mediante métricas específicas como la precisión y la pérdida en el entrenamiento y validación.

3.2.6.1. Configuración de las capas de salida

En el proceso de entrenamiento de un algoritmo, es crucial la modificación de las capas de salida y la selección de la función adecuada para garantizar un resultado eficiente en el sistema clasificador.

Antes de proceder con el entrenamiento, es necesario hacer ajustes en estas áreas para lograr un proceso de selección efectivo. En este sentido, en el presente conjunto de procesos se abordará la modificación de las capas de salida y la selección de la función adecuada para el entrenamiento del algoritmo con el objetivo de optimizar los resultados obtenidos, como se muestra en la ilustración 29.

```
[ ]  modelo = tf.keras.Sequential([
         mobilenetv2,
         tf.keras.layers.Dense(3, activation='softmax')
     ])
```

Ilustración 29. Aplicación de la función de activación

Una vez configuradas las capas densas de entrada y salida y seleccionado el modelo a utilizar, se procede a entrenar los datos ingresados por épocas. Es importante tener en cuenta que en este punto el algoritmo se vuelve experimental, ya que el número de épocas puede ser modificado en función de las respuestas obtenidas. Si el algoritmo no alcanza los resultados deseados, se puede considerar la posibilidad de aumentar el número de épocas de entrenamiento para mejorar el rendimiento del modelo. Por tanto, la cantidad de épocas es un parámetro crítico que debe ser ajustado cuidadosamente para obtener el mejor resultado posible.

3.2.6.2. Optimizador Adam

Una vez que se han configurado las capas de entrada y salida del modelo y se ha seleccionado el modelo a utilizar, se procede a entrenar los datos ingresados. Es importante destacar que en este proceso se debe agregar un optimizador para la función de pérdida, y en este caso se ha optado por el optimizador Adam.

El optimizador Adam es ampliamente utilizado en la optimización de redes neuronales y se basa en la estimación adaptativa del momento de primer y segundo orden de los gradientes de la función de pérdida (ver ilustración 30). Esto lo hace especialmente útil

en problemas de aprendizaje profundo con grandes conjuntos de datos y múltiples parámetros.

```
[ ]  modelo.compile(
        optimizer='adam',
        loss='categorical_crossentropy',
        metrics=['accuracy']
    )
```

Ilustración 30. Comando para empezar con la compilación del modelo con el optimizador Adam

3.2.6.3. Entrenamiento del sistema

Durante el entrenamiento de un modelo de aprendizaje profundo, las épocas son un parámetro crítico que determina cuántas veces el modelo pasará por todo el conjunto de datos de entrenamiento. Cada época se divide en lotes (batch) de datos que se procesan en paralelo para actualizar los pesos de las capas del modelo, lo que permite que el modelo mejore su capacidad de predicción.

Encontrar el número óptimo de épocas es crucial en el entrenamiento de un modelo de aprendizaje profundo, ya que el número de épocas afecta directamente el rendimiento del modelo. Si se utilizan pocas épocas, el modelo no tendrá tiempo suficiente para aprender patrones complejos en los datos de entrenamiento, lo que resultará en un modelo subóptimo. Por otro lado, si se utilizan demasiadas épocas, el modelo puede sobre ajustarse a los datos de entrenamiento, lo que resulta en una disminución del rendimiento en datos no vistos.

Por lo tanto, el proceso de encontrar el número óptimo de épocas implica ajustar la cantidad de épocas y monitorear el rendimiento del modelo en un conjunto de datos de validación separado, lo que garantiza que el modelo sea capaz de generalizar bien en datos nuevos y desconocidos.

Para este proyecto en particular, se decidió utilizar un valor de 50 épocas para el entrenamiento del modelo de aprendizaje profundo (ver ilustración 31).

```
[ ]  EPOCAS = 50

    historial = modelo.fit(
        data_gen_entrenamiento, epochs=EPOCAS, batch_size=32,
        validation_data=data_gen_pruebas
    )
```

Ilustración 31. Comando para el entrenamiento por épocas del sistema

La elección de este valor se basó en la evaluación del rendimiento del modelo en un conjunto de datos de validación separado y en el equilibrio entre el tiempo de entrenamiento y la precisión en la predicción de nuevos datos. Cabe mencionar que la

elección del número de épocas puede variar acorde al proyecto y de los datos disponibles, además es importante realizar una evaluación adecuada del rendimiento del modelo en cada caso para encontrar el número óptimo de épocas (revisar la tabla 5).

Tabla 5. Entrenamiento por épocas del modelo

Epoch 1/50
50/50 [==============================] - 27s 513ms/step - loss: 0.3076 - accuracy: 0.8731 - val_loss: 0.1084 - val_accuracy: 0.9750
Epoch 2/50
50/50 [==============================] - 25s 493ms/step - loss: 0.0761 - accuracy: 0.9825 - val_loss: 0.0773 - val_accuracy: 0.9650
Epoch 3/50
50/50 [==============================] - 25s 508ms/step - loss: 0.0518 - accuracy: 0.9869 - val_loss: 0.0413 - val_accuracy: 0.9875
Epoch 4/50
50/50 [==============================] - 24s 490ms/step - loss: 0.0367 - accuracy: 0.9925 - val_loss: 0.0397 - val_accuracy: 0.9900
Epoch 5/50
50/50 [==============================] - 25s 491ms/step - loss: 0.0316 - accuracy: 0.9912 - val_loss: 0.0339 - val_accuracy: 0.9875
Epoch 6/50
50/50 [==============================] - 25s 506ms/step - loss: 0.0252 - accuracy: 0.9944 - val_loss: 0.0330 - val_accuracy: 0.9975
Epoch 7/50
50/50 [==============================] - 24s 488ms/step - loss: 0.0244 - accuracy: 0.9937 - val_loss: 0.0333 - val_accuracy: 0.9850
Epoch 8/50
50/50 [==============================] - 25s 492ms/step - loss: 0.0216 - accuracy: 0.9944 - val_loss: 0.0321 - val_accuracy: 0.9900
Epoch 9/50
50/50 [==============================] - 24s 489ms/step - loss: 0.0224 - accuracy: 0.9950 - val_loss: 0.0340 - val_accuracy: 0.9900
Epoch 10/50
50/50 [==============================] - 25s 505ms/step - loss: 0.0155 - accuracy: 0.9981 - val_loss: 0.0195 - val_accuracy: 0.9975
Epoch 11/50
50/50 [==============================] - 24s 489ms/step - loss: 0.0144 - accuracy: 0.9975 - val_loss: 0.0236 - val_accuracy: 0.9900
Epoch 12/50

```
50/50 [==============================] - 25s 492ms/step - loss: 0.0134 - accuracy: 0.9975 -
val_loss: 0.0204 - val_accuracy: 0.9900
Epoch 13/50
50/50 [==============================] - 25s 507ms/step - loss: 0.0115 - accuracy: 0.9987 -
val_loss: 0.0180 - val_accuracy: 0.9950
Epoch 14/50
50/50 [==============================] - 25s 493ms/step - loss: 0.0129 - accuracy: 0.9981 -
val_loss: 0.0165 - val_accuracy: 0.9950
Epoch 15/50
50/50 [==============================] - 25s 492ms/step - loss: 0.0185 - accuracy: 0.9962 -
val_loss: 0.0202 - val_accuracy: 0.9950
Epoch 16/50
50/50 [==============================] - 24s 488ms/step - loss: 0.0146 - accuracy: 0.9975 -
val_loss: 0.0135 - val_accuracy: 0.9950
Epoch 17/50
50/50 [==============================] - 25s 506ms/step - loss: 0.0147 - accuracy: 0.9962 -
val_loss: 0.0311 - val_accuracy: 0.9900
Epoch 18/50
50/50 [==============================] - 25s 493ms/step - loss: 0.0091 - accuracy: 0.9981 -
val_loss: 0.0239 - val_accuracy: 0.9925
Epoch 19/50
50/50 [==============================] - 24s 489ms/step - loss: 0.0115 - accuracy: 0.9975 -
val_loss: 0.0234 - val_accuracy: 0.9875
Epoch 20/50
50/50 [==============================] - 25s 508ms/step - loss: 0.0169 - accuracy: 0.9956 -
val_loss: 0.0193 - val_accuracy: 0.9950
Epoch 21/50
50/50 [==============================] - 24s 490ms/step - loss: 0.0100 - accuracy: 0.9962 -
val_loss: 0.0146 - val_accuracy: 0.9925
Epoch 22/50
50/50 [==============================] - 25s 492ms/step - loss: 0.0110 - accuracy: 0.9975 -
val_loss: 0.0261 - val_accuracy: 0.9900
Epoch 23/50
50/50 [==============================] - 24s 490ms/step - loss: 0.0103 - accuracy: 0.9969 -
val_loss: 0.0107 - val_accuracy: 0.9975
Epoch 24/50
50/50 [==============================] - 25s 510ms/step - loss: 0.0118 - accuracy: 0.9981 -
val_loss: 0.0113 - val_accuracy: 0.9975
Epoch 25/50
```

```
50/50 [==============================] - 25s 492ms/step - loss: 0.0068 - accuracy: 0.9987 -
val_loss: 0.0110 - val_accuracy: 0.9975
Epoch 26/50
50/50 [==============================] - 24s 489ms/step - loss: 0.0108 - accuracy: 0.9956 -
val_loss: 0.0157 - val_accuracy: 0.9950
Epoch 27/50
50/50 [==============================] - 25s 507ms/step - loss: 0.0102 - accuracy: 0.9956 -
val_loss: 0.0086 - val_accuracy: 0.9975
Epoch 28/50
50/50 [==============================] - 24s 491ms/step - loss: 0.0112 - accuracy: 0.9969 -
val_loss: 0.0114 - val_accuracy: 0.9950
Epoch 29/50
50/50 [==============================] - 24s 488ms/step - loss: 0.0062 - accuracy: 0.9981 -
val_loss: 0.0107 - val_accuracy: 0.9975
Epoch 30/50
50/50 [==============================] - 24s 489ms/step - loss: 0.0054 - accuracy: 0.9994 -
val_loss: 0.0091 - val_accuracy: 0.9950
Epoch 31/50
50/50 [==============================] - 25s 508ms/step - loss: 0.0088 - accuracy: 0.9981 -
val_loss: 0.0141 - val_accuracy: 0.9975
Epoch 32/50
50/50 [==============================] - 24s 491ms/step - loss: 0.0072 - accuracy: 0.9981 -
val_loss: 0.0088 - val_accuracy: 0.9975
Epoch 33/50
50/50 [==============================] - 24s 490ms/step - loss: 0.0092 - accuracy: 0.9975 -
val_loss: 0.0095 - val_accuracy: 0.9975
Epoch 34/50
50/50 [==============================] - 25s 508ms/step - loss: 0.0043 - accuracy: 0.9987 -
val_loss: 0.0106 - val_accuracy: 0.9950
Epoch 35/50
50/50 [==============================] - 25s 492ms/step - loss: 0.0099 - accuracy: 0.9981 -
val_loss: 0.0227 - val_accuracy: 0.9925
Epoch 36/50
50/50 [==============================] - 25s 492ms/step - loss: 0.0067 - accuracy: 0.9981 -
val_loss: 0.0074 - val_accuracy: 0.9950
Epoch 37/50
50/50 [==============================] - 25s 511ms/step - loss: 0.0063 - accuracy: 0.9987 -
val_loss: 0.0073 - val_accuracy: 0.9975
Epoch 38/50
```

```
50/50 [==============================] - 24s 489ms/step - loss: 0.0039 - accuracy: 1.0000 -
val_loss: 0.0083 - val_accuracy: 0.9975
Epoch 39/50
50/50 [==============================] - 24s 488ms/step - loss: 0.0072 - accuracy: 0.9975 -
val_loss: 0.0160 - val_accuracy: 0.9925
Epoch 40/50
50/50 [==============================] - 25s 492ms/step - loss: 0.0151 - accuracy: 0.9950 -
val_loss: 0.0344 - val_accuracy: 0.9875
Epoch 41/50
50/50 [==============================] - 25s 509ms/step - loss: 0.0064 - accuracy: 0.9981 -
val_loss: 0.0163 - val_accuracy: 0.9975
Epoch 42/50
50/50 [==============================] - 24s 490ms/step - loss: 0.0076 - accuracy: 0.9975 -
val_loss: 0.0114 - val_accuracy: 0.9975
Epoch 43/50
50/50 [==============================] - 24s 491ms/step - loss: 0.0065 - accuracy: 0.9981 -
val_loss: 0.0059 - val_accuracy: 0.9975
Epoch 44/50
50/50 [==============================] - 25s 491ms/step - loss: 0.0115 - accuracy: 0.9962 -
val_loss: 0.0085 - val_accuracy: 0.9975
Epoch 45/50
50/50 [==============================] - 25s 505ms/step - loss: 0.0036 - accuracy: 0.9994 -
val_loss: 0.0110 - val_accuracy: 0.9950
Epoch 46/50
50/50 [==============================] - 24s 490ms/step - loss: 0.0041 - accuracy: 0.9994 -
val_loss: 0.0051 - val_accuracy: 0.9975
Epoch 47/50
50/50 [==============================] - 24s 491ms/step - loss: 0.0056 - accuracy: 0.9987 -
val_loss: 0.0093 - val_accuracy: 0.9950
Epoch 48/50
50/50 [==============================] - 25s 508ms/step - loss: 0.0050 - accuracy: 0.9994 -
val_loss: 0.0046 - val_accuracy: 0.9975
Epoch 49/50
50/50 [==============================] - 24s 490ms/step - loss: 0.0031 - accuracy: 0.9994 -
val_loss: 0.0095 - val_accuracy: 0.9975
Epoch 50/50
50/50 [==============================] - 25s 494ms/step - loss: 0.0099 - accuracy: 0.9975 -
val_loss: 0.0183 - val_accuracy: 0.9925
```

3.2.6.4. Resultados del sistema clasificador

La gráfica de precisión es una herramienta visual que se utiliza en TensorBoard para analizar la evolución del rendimiento de los modelos de redes neuronales convolucionales durante el entrenamiento permitiendo comprender lo que ocurre en cada época de entrenamiento y validar la efectividad del modelo [42].

En el presente trabajo, se ha incluido una gráfica de precisión (ilustración 33) que muestra los resultados de las pruebas de validación y entrenamiento. Esta gráfica permite una fácil visualización de la precisión del modelo en cada etapa del proceso de entrenamiento y validación. La inclusión de esta gráfica es importante para la interpretación y evaluación de los resultados obtenidos, ya que permite identificar cualquier posible sobreajuste del modelo y realizar los ajustes necesarios en consecuencia. La implementación del código para generar la gráfica de precisión se verá detallada en la ilustración 32.

```
#Graficas de precisión
acc = historial.history['accuracy']
val_acc = historial.history['val_accuracy']

loss = historial.history['loss']
val_loss = historial.history['val_loss']

rango_epocas = range(50)

plt.figure(figsize=(8,8))
plt.subplot(1,2,1)
plt.plot(rango_epocas, acc, label='Precisión Entrenamiento')
plt.plot(rango_epocas, val_acc, label='Precisión Pruebas')
plt.legend(loc='lower right')
plt.title('Precisión de entrenamiento y pruebas')

plt.subplot(1,2,2)
plt.plot(rango_epocas, loss, label='Pérdida de entrenamiento')
plt.plot(rango_epocas, val_loss, label='Pérdida de pruebas')
plt.legend(loc='upper right')
plt.title('Pérdida de entrenamiento y pruebas')
plt.show()
```

Ilustración 32. Línea de código para mostrar la gráfica de precisión del sistema ya entrenado

Al analizar el gráfico de precisión (ilustración 33) del algoritmo implementado, se puede notar que la línea de precisión de entrenamiento y la línea de precisión de pruebas están cercanas entre sí. Esta cercanía sugiere que el modelo está generalizando bien a nuevos datos y no sufre de sobreajuste.

La unión de ambas líneas indica que el modelo ha aprendido patrones generales en los datos de entrenamiento que se aplican a nuevos datos. Como resultado, el modelo tiene una alta precisión tanto en los datos de entrenamiento como en los datos de prueba, lo que

sugiere que es capaz de hacer predicciones precisas para datos nuevos y no vistos durante el entrenamiento.

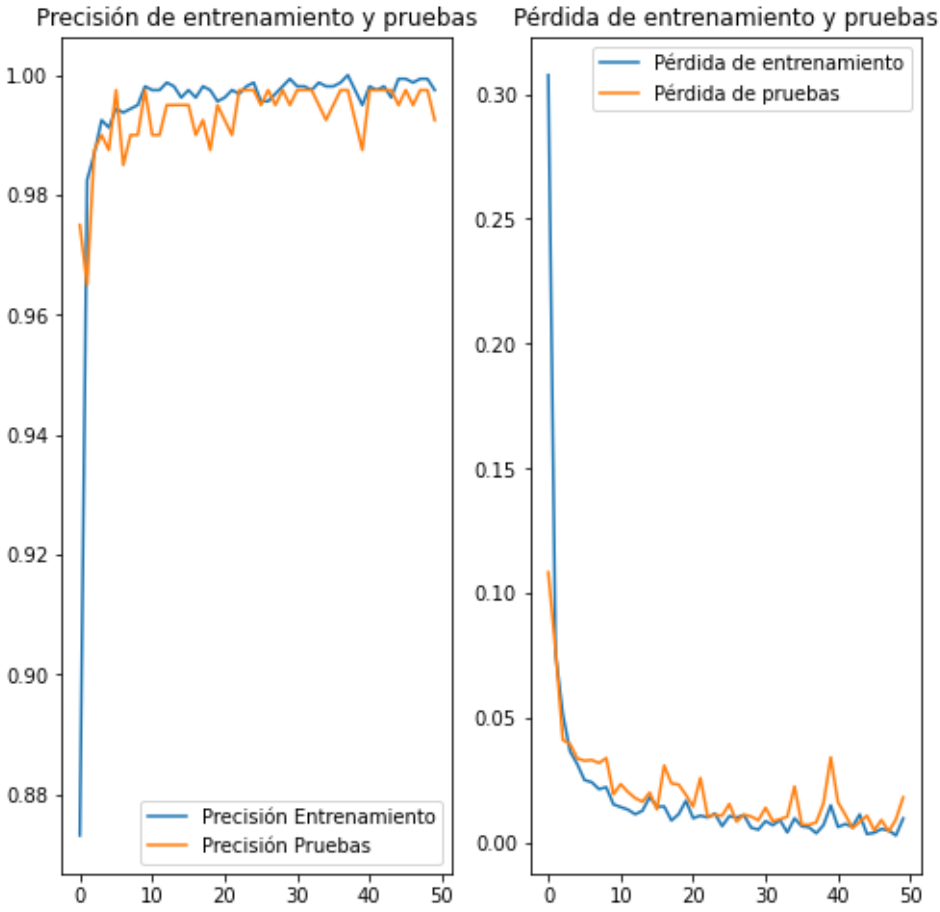


Ilustración 33. Gráfica de precisión del modelo MobileNetV2

Sin embargo, es importante realizar pruebas adicionales para confirmar el rendimiento del modelo. La validación cruzada y otras pruebas pueden ser útiles para verificar que el modelo funcione correctamente en todos los casos y evitar posibles problemas de sobreajuste.

Al observar el gráfico de precisión del algoritmo en la ilustración 34, se puede notar una separación significativa entre la línea de precisión de entrenamiento y la línea de precisión de pruebas.

Esta disparidad sugiere la presencia de un fenómeno de sobreajuste en el modelo, caracterizado por una alta capacidad de adaptación a los datos de entrenamiento específicos, pero una menor habilidad para generalizar y aplicar los patrones aprendidos a nuevas instancias. Este exceso de ajuste se traduce en una sobre calibración del modelo a los datos de entrenamiento, lo que conduce a la memorización de detalles y ruido

inherente a dichos datos en lugar de capturar las características subyacentes más relevantes y aplicables a nuevos conjuntos de datos.

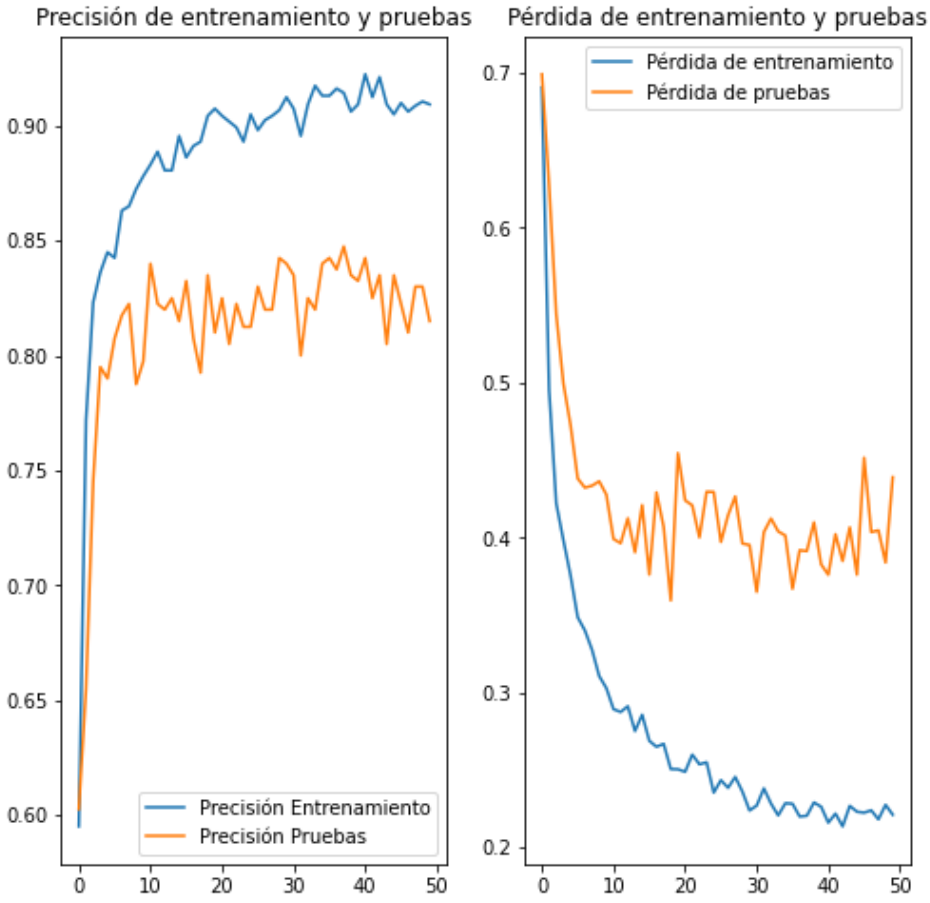


Ilustración 34. Gráfica de precisión del modelo MobileNetV1

Este tipo de sobreajuste puede conducir a una alta precisión en los datos de entrenamiento, pero una baja precisión en los datos de prueba, lo que significa que el modelo no puede generalizar a nuevos datos de manera efectiva. Para resolver este problema, existen técnicas como la regularización, la selección cuidadosa de características, la reducción de la complejidad del modelo, el aumento de datos y la validación cruzada, que pueden ayudar a mejorar la capacidad del modelo para generalizar a nuevos datos y reducir el sobreajuste.

CAPÍTULO 4: DISCUSIÓN DE LOS RESULTADOS

En el capítulo tres, se presentaron dos algoritmos de clasificación de imágenes basados en redes neuronales convolucionales: MobileNetV2 y MobileNetV1. Ambos modelos fueron entrenados utilizando un conjunto de datos de imágenes de granos de café, y se evaluaron sus resultados utilizando métricas como la precisión y la pérdida.

Para comprender mejor los resultados obtenidos por cada modelo, se procedió a elaborar tablas que contienen información detallada sobre el proceso de entrenamiento y los resultados de la clasificación. La información presentada en estas tablas incluye el número de épocas de entrenamiento, la precisión de entrenamiento y validación, así como la pérdida de entrenamiento y validación.

Además, para evaluar los modelos de manera completa, se realizaron pruebas adicionales utilizando diferentes conjuntos de imágenes y casos de clasificación. Estas pruebas permitieron obtener conclusiones más sólidas sobre la capacidad de los modelos para generalizar y clasificar correctamente imágenes de granos de café en diferentes situaciones.

En general, la elaboración de estas tablas y la evaluación de los modelos en diferentes casos permitió una mejor comprensión de los algoritmos implementados y sus resultados. Esto es importante para determinar la efectividad de cada modelo en la clasificación de imágenes de granos de café, y para obtener una comprensión más profunda de los procesos y desafíos involucrados en la implementación de algoritmos de clasificación de imágenes en general.

4.1. Tablas de características de los algoritmos de clasificación

En el presente estudio, se aplicó una metodología basada en redes neuronales convolucionales para la selección de imágenes de granos de café mediante el modelo MobileNetV2. MobileNetV2 es un modelo de red neuronal convolucional diseñado específicamente para dispositivos móviles con recursos limitados. La arquitectura de MobileNetV2 se basa en el uso de bloques de construcción llamados "módulos de

expansión" que permiten el aumento de la complejidad y profundidad de la red mientras se mantiene la eficiencia computacional.

El algoritmo implementado consistió en un proceso de entrenamiento supervisado mediante la técnica de aprendizaje profundo. Se utilizó un conjunto de imágenes de granos de café previamente etiquetadas para el entrenamiento del modelo. El objetivo del entrenamiento fue lograr que el modelo pudiera clasificar de manera efectiva y precisa las imágenes de granos de café en dos categorías: "aceptable" y "rechazable".

Una vez entrenado, el modelo MobileNetV2 fue utilizado para clasificar un conjunto de imágenes de granos de café. El proceso de clasificación se llevó a cabo mediante la aplicación de una técnica de detección de objetos que permitió identificar la presencia de granos de café en la imagen y evaluar su calidad de acuerdo con los criterios de clasificación previamente definidos.

Entre las principales ventajas del modelo MobileNetV2 se encuentra su eficiencia computacional, lo que lo hace adecuado para aplicaciones en dispositivos móviles con recursos limitados. Además, su arquitectura modular permite la adaptación a diferentes tareas de clasificación de imágenes.

En cuanto a las desventajas, MobileNetV2 puede presentar problemas de precisión en la clasificación de imágenes muy complejas o con características atípicas que no se encuentren representadas en el conjunto de entrenamiento.

4.1.2. Características del modelo MobileNetV2

En la tabla 6 presentada a continuación, se pueden observar los porcentajes de precisión obtenidos durante el proceso de entrenamiento y validación de los modelos. Como se mencionó en capítulos previos, se evaluó el rendimiento de diferentes algoritmos para la selección de imágenes de granos de café, entre los cuales se encontraba el modelo MobileNetV2.

Los resultados indican que el sistema empleado con el modelo MobileNetV2 tuvo un desempeño superior al sistema donde se utilizó el modelo MobileNetV1. En el proceso de entrenamiento, el modelo logró una eficiencia 99.18%, y un promedio de error y pérdidas del 1,97%.

Tabla 6. Características de entrenamiento del modelo con MobileNetV2

Épocas	Pérdida de precisión (%)	Precisión (%)	Pérdida de validación (%)	Validación (%)
1	30,76	87,31	10,84	97,5

2	7,61	92,25	7,77	96,5
3	5,18	92,69	4,13	92,75
4	3,67	99,25	3,97	99
5	3,16	99,12	3,39	98,75
6	2,52	99,44	3,3	99,75
7	2,44	99,37	3,33	98,5
8	2,16	99,44	3,21	99
9	2,24	99,5	3,4	99
10	1,55	99,81	1,95	99,75
11	1,44	99,75	2,36	99
12	1,34	99,75	2,04	99
13	1,15	99,87	1,5	99,5
14	1,29	99,81	1,65	99,5
15	1,85	99,62	2,02	99,5
16	1,46	99,75	1,35	99,5
17	1,47	99,62	3,11	99
18	0,91	99,81	2,39	99,25
19	1,15	99,75	2,34	98,75
20	1,69	99,56	1,93	99,5
21	1	99,62	1,46	99,25
22	1,1	99,75	2,61	99
23	1,03	99,69	1,07	99,75
24	1,18	99,81	1,13	99,75
25	0,68	99,87	1,1	99,75
26	1,08	99,56	1,57	99,5
27	1,02	99,56	0,86	99,75
28	1,12	99,69	1,14	99,5
29	0,62	99,81	1,07	99,75
30	0,54	99,84	0,91	99,5
31	0,88	99,81	1,41	99,75
32	0,72	99,81	0,88	99,75
33	0,92	99,75	0,95	99,75
34	0,43	99,87	1,06	99,5
35	0,99	99,81	2,27	99,25
36	0,67	99,81	0,74	99,5
37	0,63	99,87	0,73	99,75
38	0,39	100	0,83	99,75
39	0,72	99,75	1,6	99,25
40	1,51	99,5	3,44	98,75
41	0,64	99,81	1,63	99,75
42	0,75	99,75	1,14	99,75
43	0,65	99,81	0,59	99,75
44	1,15	99,62	0,85	99,75

45	0,36	99,94	1,1	99,5
46	0,41	99,94	0,51	99,75
47	0,56	99,87	0,93	99,5
48	0,5	99,94	0,64	99,75
49	0,31	99,75	0,95	99,75
50	0,99	99.99	1,83	99,25
Promedio	**1,97%**	**99,18%**	**2,07%**	**99,20%**

4.1.3. Características del modelo MobileNetV1

En comparación con MobileNetV2, MobileNetV1 enfrentó dificultades para solucionar los sesgos y las pérdidas de datos, lo que afectó su proceso de entrenamiento. La tabla de resultados muestra una pérdida del 27.25% en la precisión del entrenamiento y una pérdida del 42.21% en la validación. Además, su precisión fue del 88,72% y una validación del 81.25%. Los datos antes mencionados se pueden observar en la tabla 7.

La precisión de los datos es un aspecto crucial en el entrenamiento de modelos de clasificación de imágenes, y es esencial seleccionar el modelo adecuado para el diseño del algoritmo. Si bien MobileNetV1 es eficiente en la detección de imágenes con características similares a las del conjunto de datos de entrenamiento, el modelo puede experimentar dificultades para reconocer imágenes con características distintas. En resumen, estos resultados muestran que MobileNetV1 tiene limitaciones significativas en el reconocimiento de imágenes en comparación con su versión mejorada, MobileNetV2.

Tabla 7. Tabla de las características de entrenamiento del modelo con MobileNetV1

Épocas	Perdida de precisión (%)	Precisión (%)	Perdida de validación (%)	Validación (%)
1	69,07	59,5	69,93	60,25
2	49,54	77,19	62,77	64,5
3	42,21	82,31	54,43	75,5
4	39,83	83,63	49,95	79,5
5	37,54	85,5	47,25	79
6	34,85	84,25	43,81	80,75
7	34	86,31	43,22	81,75
8	32,75	86,5	43,35	82,25
9	31,05	87,25	43,63	78,75
10	30,27	87,81	42,77	79,75
11	28,73	88,31	39,9	84
12	29,1	88,88	39,63	82,25
13	27,5	88,06	41,24	82
14	28,55	88,06	39,05	82,5

15	26,87	89,56	42,09	81,5
16	26,48	88,63	37,61	83,25
17	26,67	89,13	42,92	80,75
18	25,04	89,31	40,69	79,25
19	25,03	90,44	35,93	83,5
20	24,87	90,75	45,45	81
21	25,98	90,44	42,4	82,5
22	25,35	90,19	42,08	80,5
23	25,48	89,94	40,01	82,25
24	23,52	89,31	42,96	81,25
25	24,33	90,5	42,94	82,25
26	25,56	89,81	39,71	83
27	23,83	90,25	41,44	82
28	24,54	90,44	42,65	82
29	23,59	90,69	39,61	84,25
30	22,37	91,25	39,52	84
31	22,69	90,75	36,49	83,5
32	23,8	89,56	40,35	80
33	22,81	90,94	41,23	82,5
34	22,05	91,75	40,4	82
35	22,84	91,31	40,13	84
36	22,81	91,31	36,69	84,25
37	21,98	91,62	39,18	83,75
38	22,03	91,44	39,12	84,75
39	22,88	90,62	40,98	83,5
40	22,61	90,94	38,26	83,25
41	21,58	92,25	37,6	84,25
42	22,17	91,25	40,21	82,5
43	21,34	92,12	38,48	83,5
44	22,67	90,94	40,66	80,5
45	22,29	90,5	37,6	83,5
46	22,23	91	45,16	82,25
47	22,38	90,62	40,35	81
48	21,79	90,87	40,47	83
49	22,73	91,06	38,39	83
50	22,1	90,94	40	81,5
Promedio	**27,25 %**	**88,72 %**	**42,21 %**	**81,25 %**

4.2. Algoritmo de clasificación de acuerdo con su aplicación

El algoritmo para la clasificación de granos de café por forma y color ha sido desarrollado y teóricamente probado, y ahora se encuentra en la fase de implementación práctica. Para

llevarlo a cabo, se ha diseñado un sistema tipo manipulador robótico que permitirá aplicar el algoritmo en un entorno real y obtener resultados precisos y confiables.

Este sistema de robótica se compone de diversos elementos, como cámaras de muy buena resolución, brazos robóticos y software especializado. El objetivo es lograr una clasificación precisa y eficiente de los granos de café en función de sus características específicas, lo que permitirá mejorar la calidad y consistencia del producto final.

Un ejemplo de cómo este sistema podría ser utilizado es en las plantaciones de café, donde los granos pueden ser clasificados automáticamente según su forma y color antes de ser enviados a la siguiente etapa del proceso de producción. También podría ser utilizado por los distribuidores y tostadores de café, quienes podrían utilizar el sistema para garantizar la calidad y consistencia de los granos que utilizan en su producción. Para la fase de integración del sistema robótico con los algoritmos de clasificación de imágenes, se lo realizó en el Laboratorio de Robótica Industrial junto con el equipo de trabajo de la UTPL. En las siguientes subsecciones se documenta parte de dicha integración de los componentes del sistema final. Para mayor detalle del componente correspondiente al brazo robot industrial se sugiere revisar la referencia [6].

4.2.1. Arquitectura del sistema clasificador de café

La arquitectura del sistema clasificador de imágenes (ilustración 34) consta de varios pasos que se describen a continuación. El primer paso es la adquisición de la imagen, que se realiza utilizando un dispositivo de captura de imágenes. La imagen adquirida se procesa mediante un preprocesamiento de datos, que consiste en la corrección de la iluminación, la eliminación de ruido y la normalización de la escala.

La segmentación de la imagen por otro lado implica la separación de los objetos de interés de la imagen de fondo. En este caso, el objeto de interés son los granos de café. Una vez segmentados, se procede a la extracción de características mediante un algoritmo de extracción de características. Las características extraídas pueden incluir el tamaño, la forma y la textura de los granos de café.

A continuación, se entrena una red neuronal convolucional (CNN) utilizando las características extraídas. La CNN se entrena para clasificar los granos de café en dos categorías: defectuosos y no defectuosos. Una vez entrenada, la CNN se utiliza para clasificar los granos de café en la imagen segmentada.

La siguiente etapa es la conversión de coordenadas, que implica la transformación de las coordenadas de los granos de café clasificados por la CNN a coordenadas que puedan ser

procesadas por el brazo robótico o dispositivo electrónico inteligente. Esta conversión permite la localización precisa de los granos de café defectuosos.

En la etapa final, el brazo robótico o dispositivo electrónico inteligente procesa los granos de café defectuosos, extrayéndolos de la imagen segmentada y separándolos de los granos de café no defectuosos. El resultado final es la extracción de los granos de café defectuosos del resto de la muestra.

En resumen, la arquitectura del sistema clasificador de imágenes es una combinación de técnicas de procesamiento de imágenes, aprendizaje automático y robótica que permite la identificación y extracción precisa de granos de café defectuosos de una muestra. Para entender mejor este concepto, en la ilustración 35, se muestra la arquitectura del sistema de reconocimiento de granos de café.

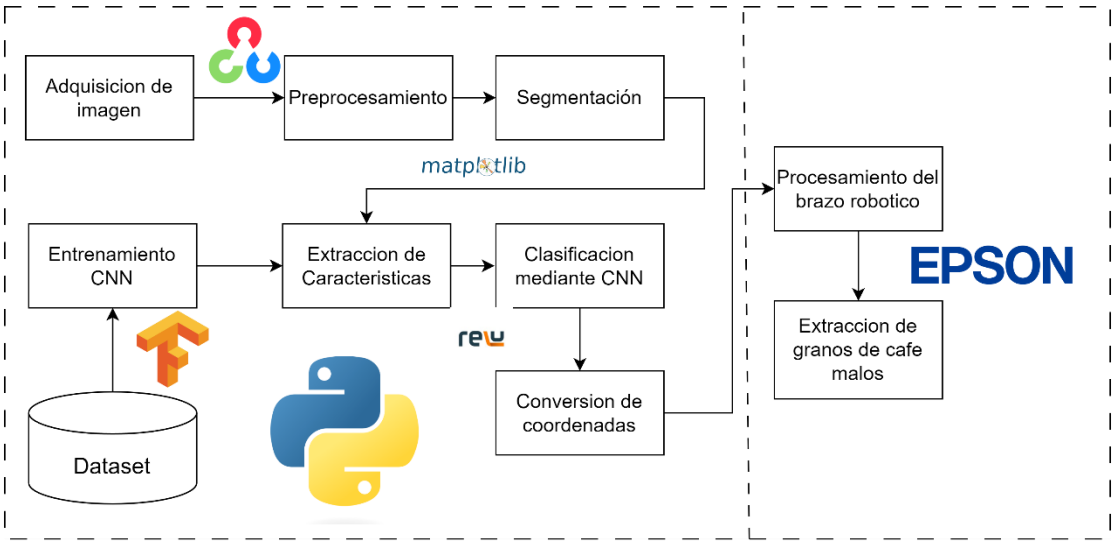


Ilustración 35. Arquitectura de sistema clasificador de imágenes

4.2.2. Aplicación del sistema clasificador de imágenes

En la provincia de Loja se comprobó el funcionamiento de un brazo robótico Epson LS10 – B602S en conjunto con un procesamiento de imagen aplicando el software de Python [6]. En dicho estudio se tuvo como objetivo encontrar la manera de automatizar los procesos de clasificación, aumentando niveles de productividad y disminuyendo el tiempo de trabajo para así garantizar un producto final de alta calidad.

Existen varios datos interesantes de [6], pero uno de los más relevantes es la manera en como se realizo las pruebas de funcionamiento. Porque para ello se tomaron en cuenta varios bloques de granos de café, en donde se colocaron diferentes cantidades de granos, algunos en buen estado y otros granos en mal estado, para que así el brazo robótico

mediante una programación lograra captar los granos de café en mal estado y apartarlos de los que estaban buenos.

En la ilustración 36 se puede observar el brazo robótico ensamblado y listo para el proceso de selección de granos de café. En la misma imagen se puede notar paneles de luz de 24W que están alrededor los cuales garantizan que la imagen este lo más clara posible y sin sombras que dañen o confundan el funcionamiento del sistema.

Ilustración 36. Arquitectura hardware y software [6]

Así mismo, en la ilustración 37 se ve un bloque de granos de café para pruebas, en donde [6] decidieron colocar 28 granos de café (25 granos con defectos y 3 granos en buen estado) para hacer que el sistema detectara los granos con defectos (ver ilustración 38). Cabe destacar que los granos tienen una separación de no más de 5 milímetros.

Ilustración 37. Bloque de granos de café[6]

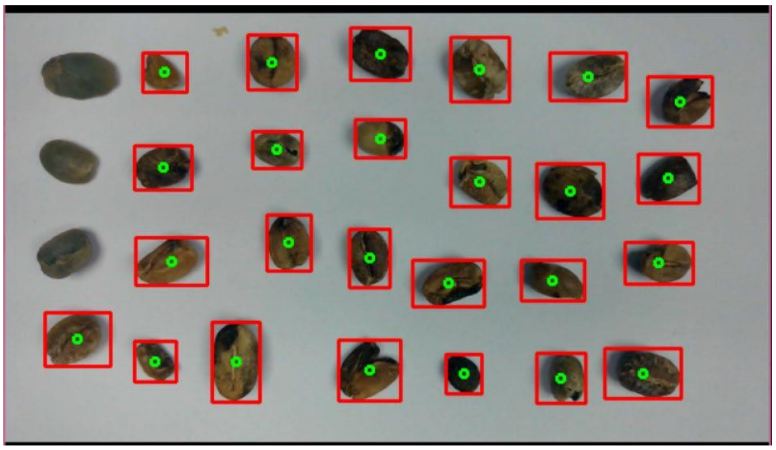


Ilustración 38. Bloque de granos de café con sistema detectando los granos de café defectuosos [38]

En la ilustración 39, se puede observar el sistema detectando los granos de café defectuosos en un bloque de 36 granos de café en donde se podrá notar que elige los 18 granos con defectos a la perfección sin que este tome en cuenta los otros 18 granos de café en buen estado.

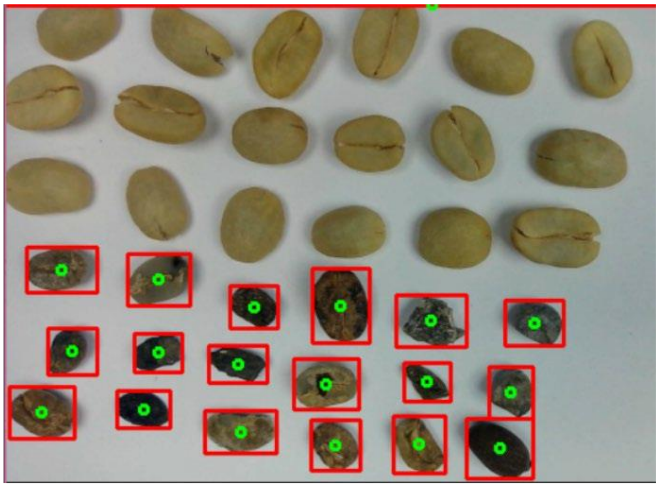


Ilustración 39. Granos de café con imperfecciones detectados por el sistema seleccionador [6]

La implementación del sistema se llevó a cabo en múltiples entornos de trabajo, utilizando conjuntos de imágenes que contenían bloques de muestras de café con variaciones en la cantidad de granos presentes. Se establecieron escenarios con diferentes proporciones de granos de café defectuosos y en buen estado, incluyendo casos con una mayor presencia de granos defectuosos, así como aquellos con una distribución equilibrada entre granos defectuosos y en buen estado.

Para realizar un análisis más detallado sobre estas variaciones, se proporciona en la tabla 8 un resumen exhaustivo que recopila información relevante. La tabla incluye métricas específicas como la proporción de granos defectuosos y en buen estado en cada conjunto

de imágenes, así como estadísticas relacionadas con la distribución espacial y características físicas de los granos presentes.

Tabla 8. Resultados de bloques de pruebas [38]

Resultados de pruebas			
Granos de café separados a una distancia ≥ 5mm entre granos			
Bloques de prueba	**Numero de granos por identificar**	**Numero de granos identificados**	**% de efectividad**
Bloque 1	25	25	100%
Bloque 2	33	33	100%
Bloque 3	18	18	100%
Bloque 4	28	28	100%
Bloque 5	19	19	100%
Bloque 6	5	5	100%
Granos de café separados a una distancia de entre 3 a 4 mm entre granos			
Bloques de prueba	**Numero de granos por identificar**	**Numero de granos identificados**	**% de efectividad**
Bloque 1	25	25	100%
Bloque 2	33	33	100%
Bloque 3	18	18	100%
Bloque 4	28	28	100%
Bloque 5	19	19	100%
Bloque 6	5	5	100%
Granos de café separados a una distancia ≤ 2mm entre granos			
Bloques de prueba	**Numero de granos por identificar**	**Numero de granos identificados**	**% de efectividad**
Bloque 1	25	12	48%
Bloque 2	33	18	55%
Bloque 3	18	8	44%
Bloque 4	28	21	75%
Bloque 5	19	9	47%
Bloque 6	5	5	100%

En la tabla 8 se puede observar que la efectividad del sistema implementado en un entorno real en donde la mayoría de las pruebas cuenta con el 100%. Cuando la imagen captada por el sistema tiene granos de café separados a una distancia mayor a 3mm, el sistema tiende a funcionar de una manera idónea. Pero cuando la imagen captada por el sistema tiene granos de café separados a una distancia menor de 2mm, este tiene un resultado variado de efectividad, que por una parte depende del número de granos de café en la imagen y así mismo la separación de estos. Para revisar mejor esta información ver el anexo 3.

CONCLUSIONES

Se diseñó e implementó un algoritmo de procesamiento de imágenes basado en Redes Neuronales Convolucionales (CNN) para la clasificación de granos de café por forma y por color, la salida del sistema corresponde a la agrupación de granos en buen estado y de granos en mal estado. El tipo de CNN utilizada es la MobileNet, la cual se basa en una arquitectura optimizada en las convoluciones para generar redes neuronales profundas con menor número de parámetros y en consecuencia con bajo requerimiento computacional y con baja latencia.

Para la implementación del algoritmo de clasificación se utilizó un marco de trabajo compuesto por las siguientes herramientas computacionales: Google Colab como infraestructura en la nube para procesamiento en GPU; Python como el lenguaje central de desarrollo acompañado de sus librerías Numpy y Matplotlib; TensorFlow para la parametrización de las redes CNN a diseñar y evaluar; y TensorBoard como herramienta de visualización del comportamiento de los algoritmos de aprendizaje automático.

Los resultados de la comparación de las arquitecturas revelaron una notable diferencia en el rendimiento entre el modelo MobileNetV2 y el modelo MobileNetV1 en términos de precisión y validación. El modelo MobileNetV2 logró una precisión impresionante del 99.18% y una validación del 99.20%, en contraste con el modelo MobileNetV1, que alcanzó una precisión del 88.72% y una validación del 81.25%. Estos resultados demuestran claramente que el modelo basado en MobileNetV2 supera significativamente al MobileNetV1 en la tarea de clasificación de granos de café.

La alta precisión obtenida por el modelo MobileNetV2 es fundamental en el contexto de la clasificación de granos de café, ya que garantiza una identificación precisa de los diferentes tipos de granos. Esto es de suma importancia para la industria cafetera, ya que la calidad y clasificación precisa de los granos son elementos críticos en el proceso de producción. Al lograr una precisión del 99.18%, el modelo MobileNetV2 cumple con los estándares más exigentes de precisión en esta tarea.

RECOMENDACIONES

Basado en los resultados obtenidos y en la experiencia del proyecto de clasificación de granos de café, se pueden hacer las siguientes recomendaciones:

Considerar el uso de una base de datos más grande, aunque se utilizó una base de datos de granos de café de tamaño considerable, se debe considerar la inclusión de más variedades de granos de café en la base de datos. Esto ayudaría a mejorar la precisión del modelo y hacerlo más robusto.

Optimización de la red neuronal, aunque el modelo utilizado para la clasificación de granos de café fue altamente preciso, se puede optimizar aún más para mejorar el tiempo de procesamiento y reducir el uso de recursos computacionales. Se podría considerar el uso de técnicas de compresión de modelo o el diseño de una red neuronal personalizada para el proyecto.

Desarrollo de un prototipo de sistema automatizado, se puede explorar la posibilidad de desarrollar un prototipo de sistema automatizado para la clasificación de granos de café en un entorno industrial. Esto podría involucrar la integración de sensores, mecanismos de recolección y transporte, y una interfaz de usuario para monitorear el proceso.

Investigación adicional, para mejorar aún más la eficiencia y la precisión del modelo, se podría realizar una investigación adicional sobre la selección de características, la normalización de datos, y la selección de algoritmos de aprendizaje automático para la clasificación de granos de café.

BIBLIOGRAFÍA

[1] N. Caporaso, M. B. Whitworth, and I. D. Fisk, "Prediction of coffee aroma from single roasted coffee beans by hyperspectral imaging," *Food Chem.*, vol. 371, no. September 2021, p. 131159, 2022, doi: 10.1016/j.foodchem.2021.131159.

[2] J. H. Contenido and F. Prieto, "Clasificación de Granos de Café usando FPGA," *Ing. Y Compet.*, vol. 7, no. 2, pp. 35–42, 2011, doi: 10.25100/iyc.v7i2.2516.

[3] M. Zhu *et al.*, "Fast determination of lipid and protein content in green coffee beans from different origins using NIR spectroscopy and chemometrics," *J. Food Compos. Anal.*, vol. 102, no. June, p. 104055, 2021, doi: 10.1016/j.jfca.2021.104055.

[4] C. D. E. Machala and C. Art, "COMERCIALIZACIÓN Y CONSUMO DE CAFÉ (COFFEA ARÁBICA) EN LA".

[5] H. C. Bazame, J. P. Molin, D. Althoff, and M. Martello, "Detection, classification, and mapping of coffee fruits during harvest with computer vision," *Comput. Electron. Agric.*, vol. 183, no. February, 2021, doi: 10.1016/j.compag.2021.106066.

[6] F. D. E. I. Y. Arquitecura, *UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA La Universidad Católica de Loja Sistema de clasificación automática de café basado en un brazo robótico industrial y máquina de visión Director :* 2022.

[7] P. C. Camporredondo, "" M ETODO DE SELECCIÓN DE GRANOS CON IoT y VISIÓN ARTIFICIAL PARA MEJORAR EL RENDIMIENTO DEL CAFÉ PERGAMINO EN LA ASOCIACIÓN," 2019.

[8] F. Ferreira, J. Tadeu, F. Rosas, R. N. Martins, L. D. A. Viana, and J. D. P. Gonçalves, "Evaluación de la calidad de los granos de café mediante visión artificial y algoritmos de aprendizaje automático," 2020.

[9] S. Wolfert, L. Ge, C. Verdouw, and M. J. Bogaardt, "Big Data in Smart Farming – A review," *Agric. Syst.*, vol. 153, pp. 69–80, 2017, doi: 10.1016/j.agsy.2017.01.023.

[10] J. G. M. Esgario, P. B. C. de Castro, L. M. Tassis, and R. A. Krohling, "An app to assist farmers in the identification of diseases and pests of coffee leaves using deep learning," *Inf. Process. Agric.*, no. xxxx, pp. 1–10, 2021, doi: 10.1016/j.inpa.2021.01.004.

[11] A. Dubois, F. Teytaud, and S. Verel, "Short term soil moisture forecasts for potato crop farming: A machine learning approach," *Comput. Electron. Agric.*, vol. 180, no. November 2020, 2021, doi: 10.1016/j.compag.2020.105902.

[12] R. M. Mohana, C. K. K. Reddy, P. R. Anisha, and B. V. R. Murthy, "Random forest algorithms for the classification of tree-based ensemble," *Mater. Today Proc.*, no. xxxx, 2021, doi: 10.1016/j.matpr.2021.01.788.

[13] R. An, Y. Xu, and X. Liu, “A rough margin-based multi-task ν-twin support vector machine for pattern classification,” *Appl. Soft Comput.*, vol. 112, p. 107769, 2021, doi: 10.1016/j.asoc.2021.107769.

[14] C. Quintero, F. Merchán, A. Cornejo, and J. S. Galán, “Uso de Redes Neuronales Convolucionales para el Reconocimiento Automático de Imágenes de Macroinvertebrados para el Biomonitoreo Participativo,” *KnE Eng.*, vol. 3, no. 1, p. 585, 2018, doi: 10.18502/keg.v3i1.1462.

[15] R. Janandi, “para la clasificación de la calidad de los granos de café en un sistema de información móvil,” pp. 13–14, 2020.

[16] M. Vázquez-Marrufo, E. Sarrias-Arrabal, M. García-Torres, R. Martín-Clemente, and G. Izquierdo, “A systematic review of the application of machine-learning algorithms in multiple sclerosis,” *Neurologia*, no. xxxx, 2021, doi: 10.1016/j.nrl.2020.10.017.

[17] P. Oliveri, C. Malegori, M. Casale, E. Tartacca, and G. Salvatori, “An innovative multivariate strategy for HSI-NIR images to automatically detect defects in green coffee,” *Talanta*, vol. 199, no. October 2018, pp. 270–276, 2019, doi: 10.1016/j.talanta.2019.02.049.

[18] D. I. Patrício and R. Rieder, “Computer vision and artificial intelligence in precision agriculture for grain crops: A systematic review,” *Comput. Electron. Agric.*, vol. 153, pp. 69–81, 2018, doi: 10.1016/j.compag.2018.08.001.

[19] P. Oliveri, C. Malegori, M. Casale, E. Tartacca, and G. Salvatori, “An innovative multivariate strategy for HSI-NIR images to automatically detect defects in green coffee,” *Talanta*, vol. 199, pp. 270–276, 2019, doi: 10.1016/j.talanta.2019.02.049.

[20] B. Fu *et al.*, “Synergy of multi-temporal polarimetric SAR and optical image satellite for mapping of marsh vegetation using object-based random forest algorithm,” *Ecol. Indic.*, vol. 131, p. 108173, 2021, doi: 10.1016/j.ecolind.2021.108173.

[21] R. Janandi and T. W. Cenggoro, “An Implementation of Convolutional Neural Network for Coffee Beans Quality Classification in a Mobile Information System,” *Proc. 2020 Int. Conf. Inf. Manag. Technol. ICIMTech 2020*, no. August, pp. 218–222, 2020, doi: 10.1109/ICIMTech50083.2020.9211257.

[22] Y. Jamtsho, P. Riyamongkol, and R. Waranusast, “Real-time license plate detection for non-helmeted motorcyclist using YOLO,” *ICT Express*, vol. 7, no. 1, pp. 104–109, 2021, doi: 10.1016/j.icte.2020.07.008.

[23] J. Arcilla, “Crecimiento y desarrollo de la planta de café,” *Sist. Prod. café en Colomb.*, pp. 22–60, 2007, [Online]. Available: http://www.cenicafe.org/es/documents/LibroSistemasProduccionCapitulo2.pdf

[24] A. Jim and P. Massa-s, “Producción de café y variables climáticas: El caso de Espíndola, Ecuador,” *Economía*, vol. 0, no. 40, pp. 117–137, 2016.

[25] S. Venegas Sánchez, D. Orellana Bueno, and P. Pérez Jara, “La realidad

Ecuatoriana en la producción de café," *Recimundo*, vol. 2, no. 2, pp. 72–91, 2018, doi: 10.26820/recimundo/2.(2).2018.72-91.

[26] L. Rouhiainen, "Inteligencia artificial 101 cosas que debes saber," *Alienta Editor.*, p. 352, 2018, [Online]. Available: https://planetadelibrosar0.cdnstatics.com/libros_contenido_extra/40/39307_Inteligencia_artificial.pdf

[27] M. L. López Franco, S. G. Lovato Torres, and G. Abad Peña, "El impacto de la cuarta revolución industrial en las relaciones sociales y productivas de la industria del plástico IMPLASTIC S. A. en Guayaquil-Ecuador: retos y perspectivas," *Univ. y Soc.*, vol. 10, no. 5, pp. 153–160, 2018.

[28] J. Eguia and R. C. Espinosa, *Experiencias de gamificación en aulas*. 2017. [Online]. Available: http://repositorio.minedu.gob.pe/handle/MINEDU/5932

[29] S. Neethirajan, "The role of sensors, big data and machine learning in modern animal farming," *Sens. Bio-Sensing Res.*, vol. 29, no. June, p. 100367, 2020, doi: 10.1016/j.sbsr.2020.100367.

[30] G. J. Gonzalez Osorio, "Reconocimiento de objetos utilizando Open CV y Python en una Raspberry Pi 2 en una tlapalería," pp. 1–120, 2017, [Online]. Available: http://ri.uaemex.mx/handle/20.500.11799/68150

[31] W. C. Parke, *Neural Networks and Brains*. 2020. doi: 10.1007/978-3-030-44146-3_14.

[32] S. Herculano-Houzel, "The human brain in numbers: A linearly scaled-up primate brain," *Front. Hum. Neurosci.*, vol. 3, no. NOV, pp. 1–11, 2009, doi: 10.3389/neuro.09.031.2009.

[33] "Vista de Diseño de una Arquitectura de Red Neuronal Convolucional para la clasificación de objetos _ Ciencia Nicolaita".

[34] M. Arias, J. Sierra, Z. Sandoval, and F. Prieto, "Procesamiento de imágenes para la clasificación de café verde," 2016, [Online]. Available: http://www.redalyc.org/pdf/4962/496250975010.pdf

[35] O. Reyes, M. Mejia, and J. S. Useche-Castelblanco, "Técnicas de inteligencia artificial utilizadas en el procesamiento de imágenes y su aplicación en el análisis de pavimentos / Article in press Artificial Intelligence Techniques Used in the Processing of Images and Its Application in Pavement Analysis," *Rev. EIA*, vol. 16, no. 31, pp. 189–207, 2019.

[36] C. W. Rosas-Echevarría, H. Solís-Bonifacio, and A. F. Cerna-Cueva, "Efficient and low-cost system for the selection of coffee beans: An application of artificial vision," *Sci. Agropecu.*, vol. 10, no. 3, pp. 347–351, 2019, doi: 10.17268/sci.agropecu.2019.03.04.

[37] Mark Sandler, A. Howard, M. Zhu, A. Zhmoginov, and Liang-Chieh Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks Mark," *etc-Convolutional Neural Networks with Swift Tensorflow*, pp. 99–107, 2019.

[38] A. Khandakar *et al.*, “A machine learning model for early detection of diabetic foot using thermogram images,” *Comput. Biol. Med.*, vol. 137, no. August, p. 104838, 2021, doi: 10.1016/j.compbiomed.2021.104838.

[39] D. R. Seninde and E. Chambers, “Coffee flavor: A review,” *Beverages*, vol. 6, no. 3, pp. 1–25, 2020, doi: 10.3390/beverages6030044.

[40] Y. Xu and R. Goodacre, “On Splitting Training and Validation Set: A Comparative Study of Cross-Validation, Bootstrap and Systematic Sampling for Estimating the Generalization Performance of Supervised Learning,” *J. Anal. Test.*, vol. 2, no. 3, pp. 249–262, 2018, doi: 10.1007/s41664-018-0068-2.

[41] A. Bakhtiarnia, Q. Zhang, and A. Iosifidis, “Efficient High-Resolution Deep Learning: A Survey,” no. 957337, pp. 1–19, 2022, [Online]. Available: http://arxiv.org/abs/2207.13050

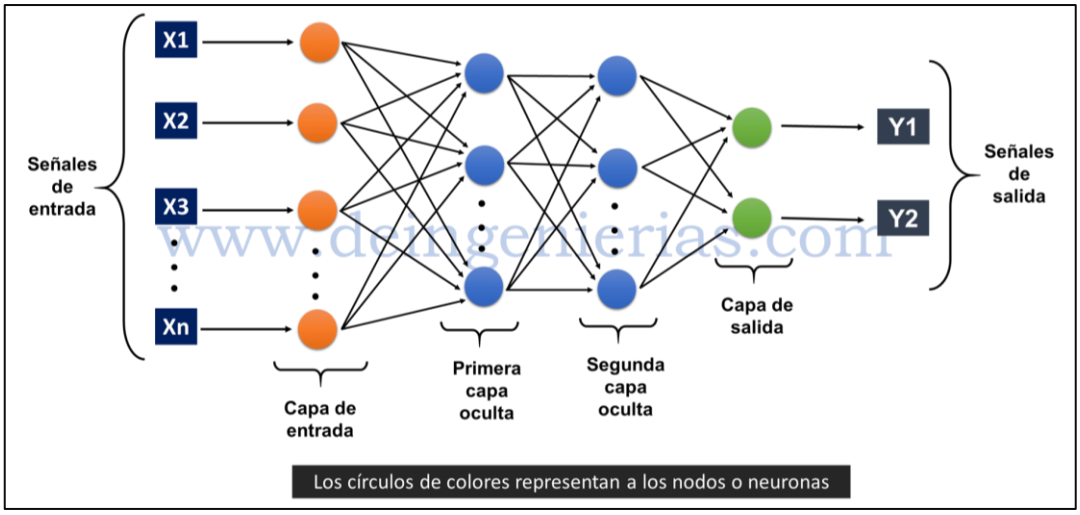
[42] F. Luus, N. Khan, and I. Akhalwaya, “Active Learning with TensorBoard Projector,” pp. 1–7, 2019, [Online]. Available: http://arxiv.org/abs/1901.00675

ANEXOS

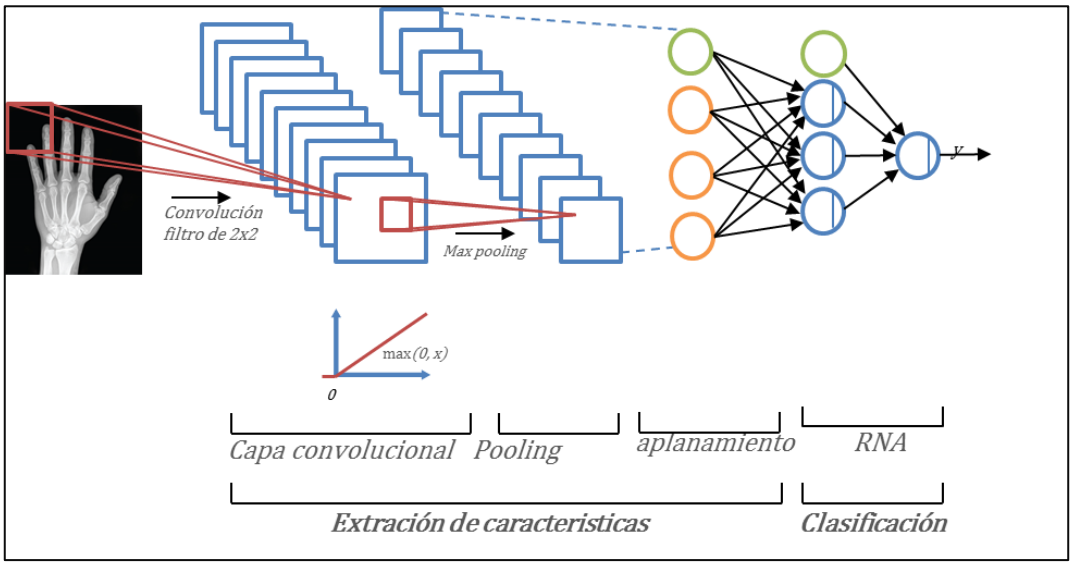
Anexo 1. Arquitecturas de las Redes y del Modelo de Clasificación

A continuación, se pueden observar ejemplos de proyectos, donde se aplicaron diferentes arquitecturas de redes neuronales artificiales:

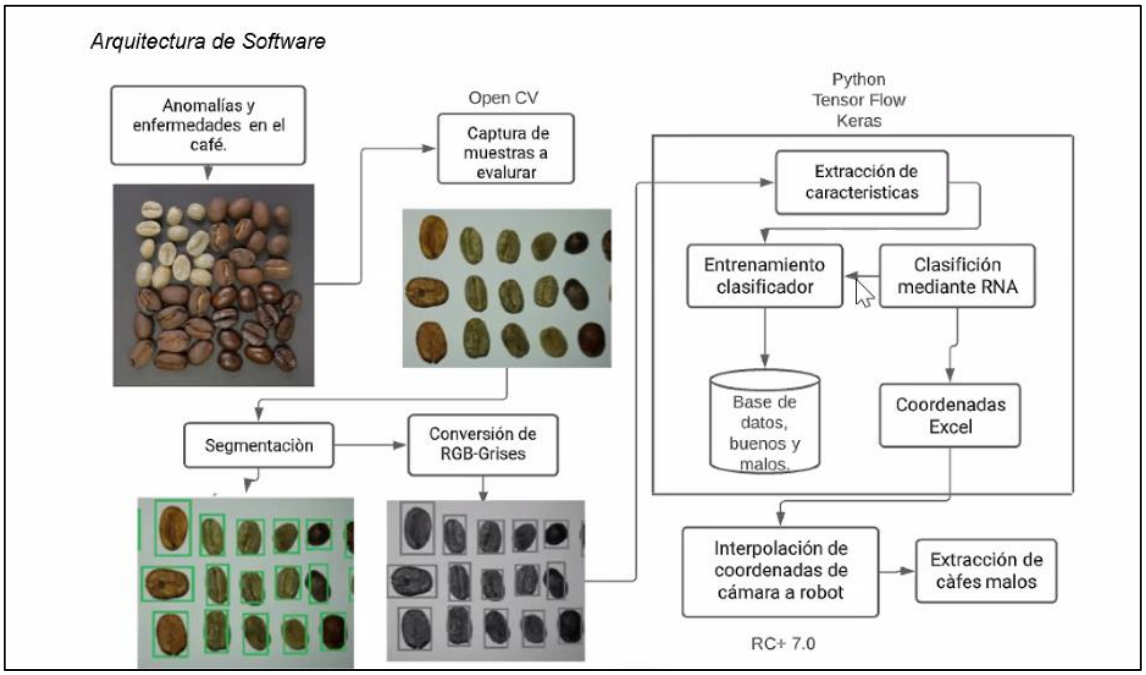
Anexo 1.1. Gráfico arquitectónico de un perceptrón multicapa con dos capas ocultas.



Anexo 1.2. Gráfico arquitectónico de una red neuronal convolucional.



Anexo 1.3. Arquitectura de modelo para clasificación de granos de café.



Anexo 2. Código del proyecto

En el siguiente apartado se ve el algoritmo del proyecto, se encuentra en formato py.

```
#Crear las carpetas para subir las imagenes
!mkdir Cbueno
!mkdir Cmalo

# Commented out IPython magic to ensure Python compatibility.
# %cd Cbueno
!unzip Cbueno.zip
# %cd ..

# %cd Cmalo
!unzip Cmalo.zip
# %cd ..

#Borrar los archivo ZIP
!rm -rf /content/Cbueno/Cbueno.zip
!rm -rf /content/Cmalo/Cmalo.zip
```

```
#Mostrar cuantas imagenes tengo de cada categoria
!ls /content/Cbueno/Cbueno | wc -l #2149

!ls /content/Cmalo/Cmalo | wc -l #1284

#Mostrar algunas imagenes con pyplot
import os
import matplotlib.pyplot as plt
import matplotlib.image as mpimg

plt.figure(figsize=(15,15))

carpeta = '/content/Cbueno/Cbueno'
imagenes = os.listdir(carpeta)

for i, nombreimg in enumerate(imagenes[:25]):
  plt.subplot(5,5,i+1)
  imagen = mpimg.imread(carpeta + '/' + nombreimg)
  plt.imshow(imagen)

#Crear carpetas para hacer el set de datos

!mkdir dataset
!mkdir dataset/Cbueno
!mkdir dataset/Cmalo

#Copiar imagenes que subimos a carpetas del dataset
#Limitar para que todos tengan la misma cantidad de imagenes
#maximo 1000 (el num. menor de imagenes que subi)

import shutil
carpeta_fuente = '/content/Cbueno/Cbueno'
```

```
carpeta_destino = '/content/dataset/Cbueno'

imagenes = os.listdir(carpeta_fuente)

for i, nombreimg in enumerate(imagenes):
  if i < 1000:
    #Copia de la carpeta fuente a la destino
    shutil.copy(carpeta_fuente + '/' + nombreimg, carpeta_destino + '/' + nombreimg)

#Copiar imagenes que subimos a carpetas del dataset
#Limitar para que todos tengan la misma cantidad de imagenes
#maximo 419 (el num. menor de imagenes que subi)

import shutil
carpeta_fuente = '/content/Cmalo/Cmalo'
carpeta_destino = '/content/dataset/Cmalo'

imagenes = os.listdir(carpeta_fuente)

for i, nombreimg in enumerate(imagenes):
  if i < 1000:
    #Copia de la carpeta fuente a la destino
    shutil.copy(carpeta_fuente + '/' + nombreimg, carpeta_destino + '/' + nombreimg)

#Mostrar cuantas imagenes tengo de cada categoria en el dataset
!ls /content/dataset/Cbueno | wc -l
!ls /content/dataset/Cmalo | wc -l

#Aumento de datos con ImageDataGenerator
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import numpy as np

#Crear el dataset generador
```

```
datagen = ImageDataGenerator(
    rescale=1. / 255,
    rotation_range = 30,
    width_shift_range = 0.25,
    height_shift_range = 0.25,
    shear_range = 15,
    zoom_range = [0.5, 1.5],
    validation_split=0.2 #20% para pruebas
)

#Generadores para sets de entrenamiento y pruebas
data_gen_entrenamiento = datagen.flow_from_directory('/content/dataset',
target_size=(224,224),
                                    batch_size=32, shuffle=True, subset='training')
data_gen_pruebas = datagen.flow_from_directory('/content/dataset',
target_size=(224,224),
                                    batch_size=32, shuffle=True, subset='validation')

#Imprimir 10 imagenes del generador de entrenamiento
for imagen, etiqueta in data_gen_entrenamiento:
  for i in range(10):
    plt.subplot(2,5,i+1)
    plt.xticks([])
    plt.yticks([])
    plt.imshow(imagen[i])
  break
plt.show()

import tensorflow as tf
import tensorflow_hub as hub

url = "https://tfhub.dev/google/tf2-preview/mobilenet_v2/feature_vector/4"
mobilenetv2 = hub.KerasLayer(url, input_shape=(224,224,3))
```

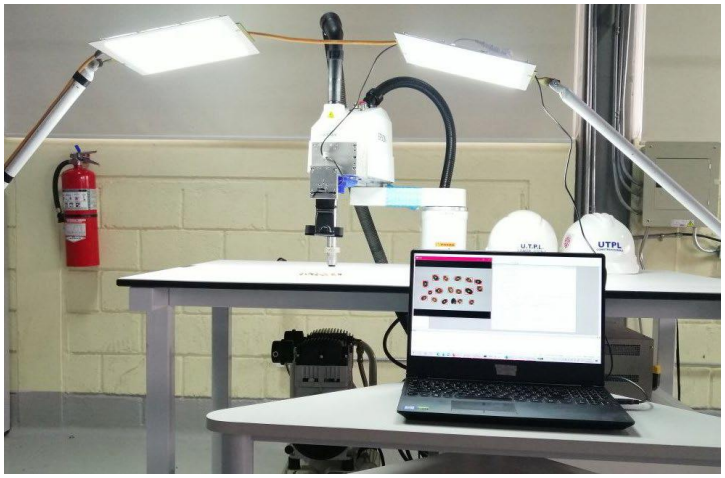
```
#Congelar el modelo descargado
mobilenetv2.trainable = False

modelo = tf.keras.Sequential([
    mobilenetv2,
    tf.keras.layers.Dense(2, activation='softmax')
])

modelo.summary()

#Compilar el modelo
modelo.compile(
    optimizer='adam',
    loss='categorical_crossentropy',
    metrics=['accuracy']
)

#Entrenar el modelo
EPOCAS = 50

historial = modelo.fit(
    data_gen_entrenamiento, epochs=EPOCAS, batch_size=32,
    validation_data=data_gen_pruebas
)

#Graficas de precisión
acc = historial.history['accuracy']
val_acc = historial.history['val_accuracy']

loss = historial.history['loss']
val_loss = historial.history['val_loss']
```

```
rango_epocas = range(50)

plt.figure(figsize=(8,8))
plt.subplot(1,2,1)
plt.plot(rango_epocas, acc, label='Precisión Entrenamiento')
plt.plot(rango_epocas, val_acc, label='Precisión Pruebas')
plt.legend(loc='lower right')
plt.title('Precisión de entrenamiento y pruebas')

plt.subplot(1,2,2)
plt.plot(rango_epocas, loss, label='Pérdida de entrenamiento')
plt.plot(rango_epocas, val_loss, label='Pérdida de pruebas')
plt.legend(loc='upper right')
plt.title('Pérdida de entrenamiento y pruebas')
plt.show()

#Categorizar una imagen de internet
from PIL import Image
import requests
from io import BytesIO
import cv2

def categorizar(url):
  respuesta = requests.get(url)
  img = Image.open(BytesIO(respuesta.content))
  img = np.array(img).astype(float)/255

  img = cv2.resize(img, (224,224))
  prediccion = modelo.predict(img.reshape(-1, 224, 224, 2))
  return np.argmax(prediccion[0], axis=-1)

#0 = grano malo, 1 = grano bueno
url = '(insertar direccion de la imagen)'
```

```
prediccion = categorizar (url)
print(prediccion)
```

Anexo 3. Resultados e Implementación de Epson RC

Anexo 3.1. Implementación de proyecto y respuesta de la cámara.



Anexo 3.2. Imagen de respuesta de algoritmo con granos de café

En estos anexos se pueden observar diferentes pruebas hechas por el sistema en donde el punto crucial es la manera en como el sistema detecta granos de café en mal estado en diferentes casos de distancias de separación entre granos de café.