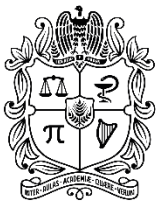


UNIVERSIDAD
NACIONAL
DE COLOMBIA

Método de Clasificación de Imágenes, Empleando Técnicas de Inteligencia Artificial, Integrado a una Plataforma IoT de Agricultura Inteligente

Juan Felipe Restrepo Arias

Universidad Nacional de Colombia – Sede Medellín
Facultad de Minas
Departamento de Ciencias de la Computación y de la Decisión
Medellín, Colombia
2023

Método de Clasificación de Imágenes, Empleando Técnicas de Inteligencia Artificial, Integrado a una Plataforma IoT de Agricultura Inteligente

Juan Felipe Restrepo Arias

Tesis presentada como requisito parcial para optar al título de:
Doctor en Ingeniería – Sistemas e Informática

Director:
John Willian Branch Bedoya, Ph.D
Codirector:
Gabriel Awad, Ph.D

Universidad Nacional de Colombia - Sede Medellín
Facultad de Minas
Departamento de Ciencias de la Computación y de la Decisión
Medellín, Colombia
2023

Dedicatoria

Muy especialmente a mi abuela, Fabiola Gómez, una mujer admirable.

A todas aquellas personas con las que me he cruzado y me han acompañado, en diferentes momentos de mi vida, han creído en mí, me han apoyado, me han animado, me han enseñado, me han soportado, gracias.

Declaración de obra original

Yo declaro lo siguiente:

He leído el Acuerdo 035 de 2003 del Consejo Académico de la Universidad Nacional. «Reglamento sobre propiedad intelectual» y la Normatividad Nacional relacionada al respeto de los derechos de autor. Esta disertación representa mi trabajo original, excepto donde he reconocido las ideas, las palabras, o materiales de otros autores.

Cuando se han presentado ideas o palabras de otros autores en esta disertación, he realizado su respectivo reconocimiento aplicando correctamente los esquemas de citas y referencias bibliográficas en el estilo requerido.

He obtenido el permiso del autor o editor para incluir cualquier material con derechos de autor (por ejemplo, tablas, figuras, instrumentos de encuesta o grandes porciones de texto).

Por último, he sometido esta disertación a la herramienta de integridad académica, definida por la universidad.

_____

Juan Felipe Restrepo Arias

Fecha 30/01/2023

Agradecimientos

Agradezco enormemente al profesor John Willian Branch, porque sin su valioso apoyo y guía, no habría emprendido está aventura, su valioso conocimiento es un regalo que comparte con todos sus alumnos, gracias además por ser un gran ser humano.

Agradezco al profesor Gabriel Awad porque ha contribuido en gran medida a mi formación como investigador, aprecio mucho su calidad humana y el conocimiento que me ha transmitido desde la maestría y ahora en el doctorado.

A quienes hicieron parte del proyecto titulado: *"Implementación de un modelo de smart farming, mediante la integración de tecnologías 4.0 e Inteligencia Artificial, para pequeños productores de hortalizas bajo invernadero"*. Financiado por la Universidad Nacional de Colombia sede Medellín, principalmente a Paulina Arregocés, Manuela Usme Martínez, Juan Pablo Yepes y Johan Sebastián Cano.

A mi alma mater Universidad Nacional de Colombia.

Resumen

La mayor parte de las variables que se miden en un cultivo agrícola solo pueden ser detectadas de manera visual, por ejemplo, el inventario de plantas y frutos, el desarrollo y las etapas fenológicas de un cultivo o la presencia de plagas y enfermedades. La llegada de las tecnologías de la Industria 4.0 a la agricultura, le ha dado la posibilidad a este dominio de resolver muchos de sus problemas. Una de las herramientas que actualmente vienen siendo implementadas son las plataformas basadas en el Internet de las Cosas (IoT por sus siglas en inglés), mejor conocidas como plataformas de Agricultura Inteligente. Sin embargo, muchas veces las explotaciones agrícolas cubren áreas muy grandes y/o remotas, en las cuales no es fácil acceder a recursos de energía o conectividad. Por lo tanto, implementar tecnologías como las ofrecidas por la Industria 4.0 algunas veces se convierte en un desafío. Este trabajo de investigación tiene como objetivo principal aportar en la solución de este tipo de problemas, al proponer un método de clasificación de imágenes integrado a una plataforma de agricultura inteligente, que busca reducir el costo computacional del proceso de clasificación de imágenes digitales, aplicado en un contexto rural con dispositivos que tienen limitaciones en su capacidad de cómputo local.

La primera parte de esta investigación se enfoca en la revisión de trabajos previos, con el fin de reconocer cuales son las estrategias arquitectónicas que otros investigadores han propuesto para resolver esta problemática, y en qué tipo de aplicaciones se han enfocado. Con base en esta revisión se seleccionó una arquitectura IoT de referencia, que posteriormente fue usada en la implementación de la solución. Esta arquitectura se basó en el uso de la tecnología de comunicación LoRa (*Long Range*), especialmente creada para trabajar en contextos con limitaciones de conectividad y energía. Luego se seleccionó el caso de aplicación de la clasificación de enfermedades en plantas, por ser uno de los que más impacto tiene en la economía y productividad de los agricultores, para lo cual se generó un conjunto de datos de imágenes digitales, basado en el conjunto de datos (*dataset)*

PlantVillage, uno de los más usados en investigaciones de este tipo. Posteriormente, con base en los resultados que otros investigadores han obtenido en el entrenamiento de algoritmos de Inteligencia Artificial con el conjunto de datos seleccionado, se hizo una preselección de métodos basados en redes neuronales convolucionales que combinan dos características: (1) un desempeño con exactitud en la clasificación por encima del 90 % y (2) un numero de parámetros de entrenamiento menor de cinco millones. El método seleccionado fue MobileNet, con los siguientes resultados de desempeño: exactitud (*Accuracy*) del 96,31 %, precisión (*Precision*) del 95,55 %, sensibilidad (*Recall)* del 95,93 %, *F1—score* del 95,72 %, con 3.762.056 de parámetros y un tamaño de 28,7 MB. Finalmente, el método seleccionado fue evaluado en tres escenarios de reducción de su arquitectura, para conocer su robustez al tener que adaptarse a condiciones con limitadas capacidades de cómputo. Para la evaluación se implementó una plataforma de agricultura inteligente en condiciones reales de trabajo, en dos unidades productivas de cultivo de tomate bajo invernadero, obteniendo métricas por encima del 90 % en todos los casos.

Palabras clave: Agricultura Inteligente, clasificación de imágenes, Inteligencia Artificial, Internet de las Cosas.

Image Classification Method, Using Artificial Intelligence Techniques, Integrated into a Smart Farming IoT Platform

Abstract

Most of the variables measured in an agricultural crop can only be detected visually, for example, the inventory of plants and fruits, the development and phenological stages of a crop, or the presence of pests and diseases. The arrival of industry 4.0 technologies in agriculture has allowed this domain to solve many of its problems. One of the tools currently being implemented is the platforms based on the Internet of Things (IoT), better known as smart agriculture platforms. However, farms often cover very large and/or remote areas where it is not easy to access energy resources or connectivity. Therefore, implementing technologies like those offered by Industry 4.0 sometimes becomes challenging. Therefore, the main objective of this research is to contribute to the solution of this type of problem by proposing an image classification method integrated into a smart agriculture platform. The proposed method seeks to reduce the computational cost of the digital image classification process applied in a rural context with devices that have limitations in their local computing capacity.

The very first part of this research focuses on reviewing previous works to recognize the architectural strategies that other researchers have proposed to solve this problem and what type of applications they have focused on. Based on this review, a reference IoT architecture was selected and later used in implementing the solution. This architecture was based on LoRa (Long Range) communication technology, specially created to work in contexts with connectivity and energy limitations. Then, the case of application of the disease classification in plants was selected, as it is one of those that have the greatest impact on the economy and productivity of farmers. Next, a data set of digital images was generated based on the dataset PlantVillage, one of the most used in research of this type. Subsequently, based on the results from previous research works whit plant village dataset, a pre-

selection of methods based on convolutional neural networks was made that combine two characteristics: (1) accurate performance in the classification above 90 % and (2) the number of training parameters less than five million.

The selected method was MobileNet, with the following performance results: 96,31 % accuracy, 95,55 % precision, 95,93 % recall, and 95,72 % F1-score, with 3,762,056 parameters and a size of 28.7 MB. Finally, the selected method was evaluated in three reduction scenarios of its architecture to know its robustness when adapting to conditions with limited computing capabilities. For the evaluation, a smart agriculture platform was implemented in real working conditions in two productive units of greenhouse tomato cultivation, obtaining metrics above 90 % in all cases.

Keywords: smart agriculture, image classification, artificial intelligence, internet of things.

Tabla de contenido

Lista de figuras

Lista de tablas

1 Introducción

En la literatura se pueden encontrar definiciones diferentes para referirse al concepto de Agricultura Inteligente, en inglés *Smart Farming*. Incluso, algunos investigadores se refieren a la Agricultura Inteligente y a la Agricultura de Precisión como dos conceptos similares, aunque los dos tienen orígenes y se basan en tecnologías diferentes [1], [2]. En este trabajo se hará referencia a la Agricultura Inteligente como la aplicación de las tecnologías de la Industria 4.0 a la agricultura [3]. Estas tecnologías son principalmente: El Internet de las Cosas (IoT por sus siglas en inglés), *Big Data*, Inteligencia Artificial, la computación en la Nube, los drones (*UAVs* por sus siglas en inglés), robots, impresión 3D entre otros [1].

La implementación de este tipo de tecnologías en ambientes rurales no es una tarea trivial. En algunos casos se requiere de conectividad a internet de alta velocidad para la realización de tareas que requieren gran capacidad de cómputo, que solo puede ser obtenida en servidores que están a cientos o miles de kilómetros de distancia. Con el objetivo de encontrar alternativas que permitan acceder cada vez más a los beneficios de este tipo de tecnologías, esta tesis doctoral propone un método para integrar un proceso de clasificación de imágenes digitales con técnicas de Inteligencia Artificial, a una plataforma IoT de Agricultura Inteligente (AI).

Este capítulo introductorio está organizado de la siguiente manera: en la primera parte se presenta una introducción a la problemática de integrar datos procedentes de imágenes digitales en las plataformas de AI. En segundo lugar, se presentan algunos trabajos previos que apuntan a solucionar esta problemática. En la tercera sección, se establece el problema de investigación. Luego, en cuarto lugar, se presenta la motivación para la búsqueda de soluciones al problema planteado. Finalmente, en las secciones quinta y sexta se establecen los objetivos y la hipótesis de investigación respectivamente.

1.1 El IoT y las imágenes digitales en las plataformas de AI

El uso de tecnologías de la industria 4.0 en agricultura se enmarca en un ciclo ciber-físico de captura de datos, transmisión, almacenamiento y procesamiento, para finalmente obtener información y conocimiento que permita tomar decisiones y actuar sobre el proceso (figura 1)

[4]. El Internet de las Cosas o IoT es la tecnología base o conjunto de tecnologías que permiten completar, en tiempo real o casi real, el ciclo ciber-físico de la AI, siendo esta su columna vertebral [5]. Por lo tanto, su influencia en la agricultura ha sido objeto de intenso estudio en los últimos cinco años [6]–[15].

Figura 1. Ciclo ciber-físico del *Smart Farming*

Fuente: elaboración propia.

Las plataformas IoT de agricultura inteligente generalmente capturan datos que son livianos, desde el punto de vista computacional, los cuales no requieren gran capacidad de cómputo para su almacenamiento y transmisión, ya que generalmente son datos numéricos, se pueden organizar de manera estructurada y se basan en sensores modernos y eficientes que se instalan fácilmente y no consumen grandes cantidades de energía [16].

Sin embargo, en la última década se han desarrollado aplicaciones en el campo de la visión artificial, con un potencial enorme para la captura de datos de variables agronómicas, que en su mayoría solamente se pueden detectar de manera visual, para poder dar diagnósticos acertados a problemas propios de las plantas en los cultivos [17]. Las imágenes digitales necesitan generalmente más capacidad de cómputo para su captura, almacenamiento y procesamiento, lo cual hace que se requieran recursos adicionales en las plataformas de AI para poder integrarlas.

Por otro lado, en los últimos años se han venido realizando esfuerzos tendientes a solucionar esta problemática. De acuerdo con los hallazgos de esta investigación, hay tres tipos de estrategias para clasificar imágenes digitales en plataformas IoT de AI:

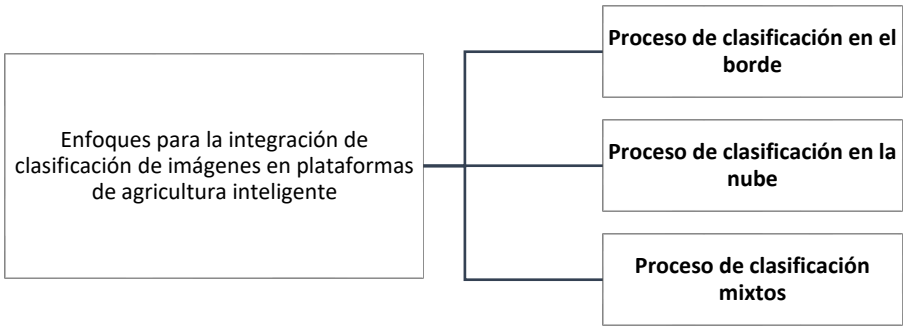
1. Proceso de clasificación de imágenes en el Borde.
2. Proceso de clasificación de imágenes en la Nube.
3. Proceso de clasificación de imágenes mixto.

La principal ventaja del primero es la obtención de datos en tiempo real, y su principal desventaja es el costo de implementación. Por otra parte, la principal ventaja del segundo es la capacidad de procesar imágenes digitales de alta resolución, y su principal desventaja es la necesidad de conectividad de gran ancho de banda. Finalmente, la estrategia mixta puede beneficiar significativamente la inversión en infraestructura, pero aumenta la complejidad de la solución.

1.2 Trabajos previos

Aunque en la literatura existen tres enfoques diferentes para integrar los métodos de clasificación de imágenes digitales a las plataformas de AI (figura 2), los trabajos previos se enfocan principalmente en desarrollar el proceso de clasificación en la Nube y en el Borde.

Figura 2. Enfoques para la integración de métodos de clasificación de imágenes a plataformas de agricultura inteligente



Fuente: elaboración propia.

1.2.1 Proceso de clasificación en la Nube.

En [18] los autores proponen el uso de la red neuronal NasNet, desarrollada por Google, y entrenada con el *dataset* PlantVillage, capaz de diagnosticar de manera efectiva enfermedades en las plantas, con una aplicación móvil integrada en teléfonos inteligentes. Los autores utilizaron aprendizaje por transferencia (*Transfer Learning*), para el entrenamiento de la red neuronal, todo el proceso de clasificación se realiza en la Nube en una instancia Amazon EC2, con la cual se comunica a través de un microservicio para diagnosticar enfermedades, usando las imágenes de las hojas de plantas capturadas a través de la aplicación móvil.

El modelo propuesto logró resultados prometedores calculados con la métrica que arroja la exactitud *(Accuracy)* de 99,24 %, 99,28 %, 99,31 % y 99,28 %. Aunque de acuerdo con los autores, el tiempo de respuesta en el microservicio API que sirve para el diagnóstico es muy alto y se vuelve más lento cuando el tamaño de la imagen es mayor. En [19] los autores, proponen un marco de agricultura inteligente basado en gemelos digitales, que utiliza LoRaWAN para la comunicación de la red de sensores en campos agrícolas, y procesamiento de imágenes digitales aéreas para la detección de enfermedades y deficiencias nutricionales. Los autores utilizaron el modelo CNN (*Convolutional neural network*) MobileNet y lograron un *Accuracy* de 95 %.

Por su parte, los autores en [20] presentan un enfoque que aprovecha diferentes tecnologías, como el IoT para detectar condiciones ambientales, y el aprendizaje profundo para la detección de enfermedades de las plantas. La idea central de los autores de este trabajo es poder tener datos para predecir la probabilidad de que aparezca una enfermedad. Su investigación se soportó en la ayuda de expertos para la clasificación de las enfermedades y el etiquetado. Los autores propusieron una CNN concatenada con una red neuronal LSTM (*Long Short-Term Memory*) para conseguir la predicción. La limitación más importante de este trabajo fue el entrenamiento con muy pocos datos de hojas de café, el proceso de clasificación se llevó a cabo en la Nube.

1.2.2 Proceso de clasificación en el Borde.

En [21] se propone un enfoque con proceso de clasificación en el Borde, utilizando los algoritmos ConvNet/CNN y MASK-RCNN. Los autores abordan un total de 10 tipos de enfermedades y 3 etapas de madurez en cultivos de tomate bajo invernadero. También combinan el proceso de clasificación con la captura de datos climáticos. La evaluación general del modelo se calculó utilizando *Precision, Recall* y *F1-score,* los cuales obtuvieron una precisión media y ponderada de 91 % y 93 % respectivamente.

En [22] los autores proponen un método integrado a una plataforma IoT para la detección de enfermedades en cultivos, con una arquitectura CNN nueva para la identificación y clasificación de veinte enfermedades en cuatro plantas diferentes. A partir de los resultados experimentales, se concluye que la arquitectura CNN propuesta por ellos realiza bien la clasificación de enfermedades de las hojas, dando una precisión alrededor del 96,88 %, superior en comparación con la precisión lograda por las arquitecturas existentes en el estado del arte. El modelo probado se implementó en un dispositivo Nvidia JetsonTX1 en el Borde, lo cual representa un costo adicional importante, que puede ser una desventaja para su implementación.

Finalmente, en [23] los autores proponen el uso de redes con tamaños reducidos, y probaron varios modelos del estado del arte: MobileNetV2, NasNetMobile, Xception y MobileNetV3. Esta comparación se realizó con el conjunto de datos de PlantVillage. Los cuatro modelos se implementaron en un microprocesador Raspberry Pi para demostrar su capacidad de ser incorporados en condiciones de trabajo real. Las principales contribuciones de este trabajo fueron: primero, el *transfer learning* realizado para los modelos seleccionados en una forma exhaustiva, en segundo lugar, la evaluación y el análisis con diferentes métricas de calidad para describir el comportamiento de los modelos cuantitativamente y cualitativamente en gran medida, en tercer lugar, la cuantificación de los modelos para su implementación en microprocesadores Raspberry Pi 4 y finalmente, el desarrollo de una interfaz gráfica de usuario (GUI por sus siglas en inglés) para el uso de los modelos entrenados en PC (*Personal Computer*), Raspberry Pi 4, o en dispositivos móviles.

1.3 Motivación

Las tecnologías de la Industria 4.0 y en especial la evolución del IoT son una oportunidad para solucionar diferentes problemas en el sector agrícola [36]. A diferencia de los sectores industriales, el enfoque heurístico todavía domina en la agricultura para la solución de problemas [3], y muchas de sus actividades de monitoreo requieren de una inspección visual para poder entregar diagnósticos acertados [17]. Estas inspecciones están a cargo de productores, técnicos e ingenieros que necesitan desplazarse a sitios apartados y muchas veces de difícil acceso, por lo que la asistencia técnica puede verse limitada.

Uno de los principales beneficios desde el punto de vista técnico que ofrece el IoT en la agricultura, es poder acceder a datos que se encuentran en sitios remotos en cuestión de segundos, así como a diferentes ubicaciones en muy corto tiempo, lo que hace posible una asistencia técnica más oportuna y eficiente, muchas veces necesaria en cuestiones que no dan espera como, por ejemplo, la identificación rápida de plagas o enfermedades [36]. Así mismo, la colección de grandes cantidades de datos, conocidos como *Big Data,* permiten aplicar herramientas del campo de la analítica para anticiparse a posibles problemas, lo que a su vez se traduce en ahorro de costos y tiempo, y puede hacer más competitivos tanto a pequeños como grandes productores agrícolas [37]–[39].

Por otro lado, existen retos aún no resueltos en la implementación de plataformas de IoT para la agricultura. En la última década la mayoría de las soluciones de IoT en entornos rurales, se han enfocado en la captura de datos principalmente climáticos y de condiciones del suelo, provenientes de sensores analógicos y digitales [8]. Si bien, esto es importante debido al avance que se ha dado a la agricultura, aún falta camino por recorrer en la integración de sensores visuales (cámaras), que permitan potencializar aún más el uso de las plataformas de IoT, como ya se viene dando en otros dominios [40]–[43]. Esto redundaría también en beneficios para la agroindustria en general, que vería como la administración de empresas del sector, tendría más información confiable para la toma de decisiones, no solamente en las actividades de campo, sino, por ejemplo, en el manejo adecuado de inventarios, gestión del personal, prevención de fraudes, minimización de los desperdicios y, sobre todo, minimización del impacto ambiental [39].

1.4 Problema de investigación

Una de las principales limitantes para la adopción del IoT en ambientes rurales es la falta de conectividad, debido a la escasez de redes celulares en áreas remotas [24]. Existen tecnologías alternativas a este tipo de redes, las cuales son conocidas como LPWAN (*Low Power Wide Area Network*) que tienen un bajo consumo de energía (las baterías de los dispositivos pueden durar hasta diez años) y un alcance entre los 10 y los 40 km en áreas abiertas [25]. Sin embargo, esta tecnología tiene un limitado ancho de banda, que puede variar entre 100 bps a 50 kbps, lo que restringe la cantidad de datos que se transmiten [25], [26]. Las tecnologías de comunicación más representativas de la familia LPWAN son: LoRa (*Long Range*), LoRaWAN (*Long Range Wide Area Network*) y Sigfox (*Sigfox S.A*.) [25]–[27].

Debido a esto, la mayoría de las plataformas IoT de AI, se limitan a la captura de datos muy livianos y relativamente fáciles de transmitir, como los provenientes de sensores analógicos o digitales que miden variables espacio-temporales (temperatura, velocidad del viento, humedad del suelo, humedad relativa, etc.), [28]. Esto limita la posibilidad de transferir datos más pesados, desde el punto de vista computacional, como imágenes digitales y videos [25].

Por otro lado, los desarrollos de soluciones basadas en visión artificial para agricultura vienen en aumento en los últimos años como, por ejemplo, detección de plagas y enfermedades [29], [30], identificación de malezas [31], conteo de frutos y de árboles [32]–[34], detección de estados fenológicos [35], entre otras aplicaciones, a través de imágenes digitales capturadas con diferentes dispositivos como UAVs (*Unmanned Aerial Vehicles*), teléfonos móviles o cámaras de dispositivos IoT.

Desafortunadamente, al tratar de integrar estas soluciones de visión artificial, a las plataformas IoT, se generan cuellos de botella, debido a las condiciones limitadas de las tecnologías de transmisión o hardware usadas en algunos entornos, lo cual genera errores en los datos transmitidos, alta latencia, altos consumos de energía y en algunos casos la interrupción de las transmisiones por completo [16]**.** En la literatura, se plantean dos desafíos que permanecen abiertos en la comunidad que investiga las posibles soluciones en dominios similares al rural [16]:

1. Minimizar la cantidad de cálculos locales en los dispositivos IoT, sin perder eficacia en los procesos.
2. Diseñar esquemas de transmisión más robustos para estos entornos, con limitantes en las comunicaciones.

Pregunta de investigación: Con base en lo anterior, esta tesis plantea una pregunta de investigación: ¿cuál es el método de clasificación de imágenes digitales y la arquitectura IoT que permiten una integración efectiva para su implementación en una plataforma de Agricultura Inteligente?

1.5 Hipótesis

La hipótesis que se plantea en este trabajo establece que: es posible implementar una plataforma de Agricultura Inteligente que integre de manera efectiva un método de clasificación de imágenes digitales y una arquitectura IoT.

1.6 Objetivos

1.6.1 Objetivo general.

Proponer un método de clasificación de imágenes digitales, empleando técnicas de Inteligencia Artificial, para integrarlo a una plataforma IoT de agricultura inteligente.

1.6.2 Objetivos específicos.

- Identificar una arquitectura de referencia para una plataforma IoT de agricultura inteligente, que permita integrar un método de clasificación de imágenes digitales empleando técnicas de Inteligencia Artificial.
- Seleccionar un método de clasificación de imágenes digitales empleando técnicas de Inteligencia Artificial, para integrarlo a una plataforma IoT de agricultura inteligente.
- Evaluar el desempeño del método de clasificación de imágenes digitales empleando técnicas de Inteligencia Artificial, para integrarlo a una plataforma IoT de AI.

1.7 Estructura de la tesis

Esta tesis está organizada de la siguiente manera: El primer capítulo presenta la introducción. En el capítulo 2, se presenta el marco metodológico seguido para la consecución de los objetivos. En el capítulo 3, se presenta la Revisión Sistemática de Literatura (RSL) llevada a cabo. Luego en el capítulo 4, se expone la selección de la arquitectura de referencia. Posteriormente en el capítulo 5, se presenta la selección del dataset y las modificaciones realizadas para la obtención de un nuevo conjunto de datos. En el capítulo 6, se presenta el método de clasificación de imágenes seleccionado. Luego en el capítulo 7, se presenta la implementación de la plataforma IoT de Agricultura Inteligente en condiciones reales de trabajo. Y finalmente en el capítulo 8, se presentan las conclusiones y el trabajo a seguir en futuras investigaciones.

2 Metodología

En el capítulo anterior se presentó la introducción a este trabajo, en la cual se explicaron algunos conceptos sobre Agricultura Inteligente y algunos trabajos previos. Se expuso además el problema de investigación, las motivaciones para resolverlo, la hipótesis que se plantea en esta tesis y los objetivos a cumplir en este trabajo.

En este capítulo se presentan los aspectos metodológicos empleados para dar cumplimiento a los objetivos de esta tesis doctoral. Esta investigación es aplicada de tipo experimental, con un enfoque cuantitativo. La metodología se dividió en cinco fases. Una de estas fases no estaba considerada en un principio: la creación de un nuevo conjunto de datos.

Este capítulo está organizado de la siguiente manera: Primero se describen los aspectos metodológicos de la Revisión Sistemática de Literatura (RSL) llevada a cabo para establecer el estado del arte. Segundo, se presentan los elementos metodológicos tenidos en cuenta para la selección de la arquitectura de referencia. En tercer lugar, se presenta la metodología seguida para la creación del nuevo conjunto de datos para el entrenamiento de los métodos de clasificación de imágenes. Luego, en la cuarta sección, se muestran los aspectos metodológicos seguidos para la elección del método de clasificación de imágenes y las métricas utilizadas. Finalmente se presenta en la quinta sección la metodología seguida para la implementación y evaluación del método seleccionado en una plataforma de Agricultura Inteligente y las métricas usadas en esta última fase.

2.1 Aspectos metodológicos de la Revisión Sistemática de Literatura

En la primera fase de esta tesis doctoral se desarrolló una Revisión Sistemática de la Literatura (RSL), con el fin de establecer el estado del arte que permitiera realizar una selección objetiva de la arquitectura de referencia (objetivo específico No. 1) y de un método de clasificación de imágenes (Objetivo específico No. 2). También fue muy importante la RSL para la selección de un caso de aplicación, con el cual probar el método de clasificación de imágenes.

2.1.1 Preguntas planteadas para la RSL.

Para guiar la búsqueda de las estrategias utilizadas en la integración de procesos de clasificación de imágenes digitales y plataformas de Agricultura Inteligente, se consideraron las siguientes preguntas:

- ¿En qué tipo de problemas se enfocan las plataformas IoT de agricultura inteligente, que incorporan métodos de clasificación de imágenes?
- ¿Cuáles son las principales estrategias para incorporar métodos de clasificación de imágenes, en plataformas IoT de agricultura inteligente?
- ¿Cuáles son las principales técnicas de adquisición, preprocesamiento, transmisión y clasificación de imágenes, utilizadas en las plataformas IoT de agricultura inteligente?

2.1.2 Criterios de selección.

Para garantizar la calidad de los artículos, solo se consideraron en el proceso de revisión aquellos que pasaron los siguientes criterios de selección:

• Artículos publicados en revistas arbitradas.

• Documentos publicados en inglés.

• Documentos que respondieron a alguna de las tres preguntas de investigación.

• Documentos publicados entre 2018 y 2022 (ambos años inclusive).

Después de buscar en las bases de datos y analizar los títulos y palabras clave, se eliminaron los artículos en los que el tema principal era irrelevante o estaba fuera del alcance de este estudio. Luego se aplicaron criterios de selección adicionales en la búsqueda, para poder obtener varias fuentes de alta calidad, que pudieran usarse para responder las preguntas de investigación:

2.1.3 Criterios de inclusión basados en resúmenes.

En esta fase se descartaron los artículos en los que, nuevamente, de acuerdo con el resumen apuntaban a resolver alguna de las tres preguntas de investigación. Los artículos que cumplieron con este criterio de inclusión se mantuvieron para su posterior procesamiento, es

decir, trabajos que integran clasificación de imágenes digitales en plataformas IoT para agricultura inteligente. Los artículos con poca información relevante en sus resúmenes se mantuvieron temporalmente en la lista y se procesaron en la siguiente etapa.

2.1.4 Criterios de inclusión basados en la revisión del texto completo.

En esta fase se eliminaron los artículos que no respondían a las preguntas de investigación, es decir, aunque en los resúmenes aparecían términos que apuntaban a la solución de la problemática planteada, en su contenido principal, como metodología, resultados y conclusiones, no se respondía a las preguntas de investigación.

2.1.5 Fuentes de información.

Se realizó una búsqueda en Internet utilizando algunos de los buscadores más importantes de artículos científicos e información académica. Los resultados obtenidos se recolectaron manualmente, con el fin de seleccionar las mejores fuentes de información que respondieran las preguntas de investigación antes mencionadas. Luego de un análisis exploratorio con palabras clave, se escogieron las bibliotecas digitales descritas en la tabla 1. Estas fuentes son reconocidas en el ámbito científico por su contenido científico-técnico y su relación con las ingenierías y tecnologías asociadas al objetivo de este trabajo.

Tabla 1. Fuentes de información usadas para la fase de búsqueda

Fuente	Tipo	URL
Scopus	Digital Library	https://www.scopus.com/ (Revisada 8 Julio 2022)
Web of Science	Digital Library	https://clarivate.com/webofsciencegroup/solutions/web-of-science (Revisada 8 Julio 2022)
Springer Link	Digital Library	https://link.springer.com/ (Revisada 8 Julio 2022)
IEEE Xplore	Digital Library	https://ieeexplore.ieee.org/Xplore/home.jsp (Revisada 8 Julio 2022)

Nota. Estas fuentes de información permiten el uso de algoritmos de búsqueda compuestos por operadores lógicos que son útiles en la extracción de la información deseada para realizar la revisión sistemática.

Fuente: elaboración propia.

2.1.6 Estrategia de búsqueda.

El siguiente paso fue definir términos clave de búsqueda y un procedimiento consistente para buscar documentación científica y técnica en las bibliotecas digitales elegidas. En primer lugar, se seleccionó un conjunto de palabras clave asociadas a las preguntas de investigación, con lo cual se conformaron tres grupos de palabras, que se muestran en la tabla 2. Cada grupo contenía expresiones consolidadas con sinónimos o términos con significados relacionados.

Tabla 2. Palabras clave usadas en los algoritmos de búsqueda

Grupo	Palabras clave
Grupo 1	Agriculture, farming, smart farming, smart agriculture, intelligent agriculture, digital farming, precision agriculture, agriculture smart system, farm management system.
Grupo 2	Architecture, IoT, platform, internet of things, wireless sensor network, cloud computing, edge computing
Grupo 3	Image, image classification, image recognition, object detection, computer vision

Nota. El grupo 1 incluyó términos asociados con la agricultura inteligente, mientras que el grupo 2 contenía un conjunto de términos generales relacionados con las plataformas IoT y las tecnologías asociadas. Finalmente, el grupo 3 incluyó términos asociados con imágenes digitales y visión artificial. Se utilizaron operadores lógicos soportados por la búsqueda avanzada de bibliotecas digitales para construir cadenas de búsqueda basadas en las tres preguntas de investigación, combinando términos de los grupos 1, 2 y en los algoritmos de búsqueda para todas las librerías digitales. La estructura general de los algoritmos de búsqueda que se aplicaron a las fuentes de información se muestra en el anexo A.

Fuente: elaboración propia.

2.1.7 Proceso de búsqueda y selección.

El proceso de búsqueda se llevó a cabo utilizando las palabras de los grupos de la tabla 2 para definir los algoritmos de búsqueda utilizados en las bibliotecas digitales. El proceso de búsqueda de artículos se limitó al título, resumen y palabras claves en las bases de datos. Después de eso, se revisó cada artículo que pasó los criterios de elegibilidad iniciales con una lectura detallada para responder a las preguntas de investigación. En el proceso, se detectaron

también artículos sobre plataformas que no usaban datos de imágenes digitales y, por lo tanto, fueron excluidos. Este proceso dejó setenta y nueve estudios, en su mayoría de los años 2020 y 2021 como se muestra en la figura 3.

Figura 3. Distribución de los artículos seleccionados por año

Fuente: elaboración propia.

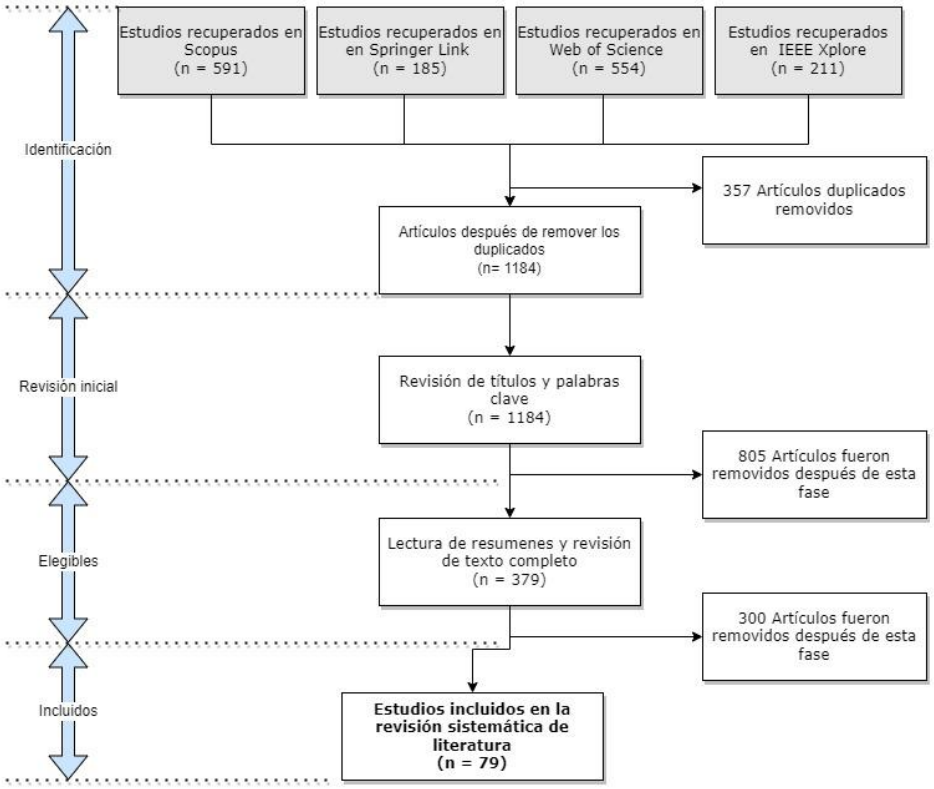
El protocolo empleado para la revisión sistemática se basó en *The PRISMA 2020 statement* [44], el cual ofrece ciertas ventajas. Primero, acelera la búsqueda de dominios que tienen literatura disponible limitada, lo que permite una rápida identificación de autores reconocidos y centros de investigación dentro de ese dominio específico. También permite identificar rápidamente tecnologías o grupos de tecnologías utilizadas para un análisis más detallado, y finamente permite a los lectores evaluar la idoneidad de los métodos y, por lo tanto, la confiabilidad de los hallazgos. Sin embargo, el protocolo utilizado puede tener debilidades en la evaluación de la calidad de los trabajos, ya que no se realiza un análisis riguroso de las pares evaluadores, solo se basa en la confiabilidad de las bases de datos científicas. Otra debilidad puede ser la estrategia de búsqueda, en la que las palabras clave pueden tener algún sesgo debido a los intereses particulares que se consideraron.

Inicialmente, se seleccionaron 1.541 estudios de las bases de datos electrónicas elegidas. Distribuidos de la siguiente manera: quinientos noventa y un *Scopus,* ciento ochenta y cinco *Springer Link,* quinientos cincuenta y cuatro *Web of Science*, y doscientos once *IEEE Xplore.*

En primer lugar, se excluyeron los duplicados, es decir, los estudios disponibles en varias bases de datos, eliminando trescientos cincuenta y siete ejemplares, y luego se extrajeron los artículos que no cumplían con las preguntas de investigación y el contexto de la búsqueda, que sumaron

ochocientos cinco artículos. Luego, a los trescientos setenta y nueve estudios restantes se les aplicó, primero, el criterio de lectura resumida y tamizaje de texto completo, lo que resultó en setenta y nueve artículos, a los que se les dio lectura completa (figura 4).

Figura 4. Resumen de la aplicación del protocolo de búsqueda



Fuente: elaboración propia.

2.2 Selección de la arquitectura de referencia

La selección de la arquitectura de referencia se realizó siguiendo la siguiente metodología:

1. Se identificaron las estrategias arquitectónicas usadas en los trabajos recuperados en la RSL.
2. Los estudios recuperados en la RSL se clasificaron también de acuerdo con sus estrategias arquitectónicas en tres tipos, que fueron usadas para poder integrar los métodos de clasificación a las plataformas IoT:

a. Proceso de clasificación en el Borde.
b. Proceso de clasificación en la Nube.
c. Proceso de clasificación combinada.

3. Se revisaron los tres tipos de estrategias y se revisaron las tecnologías usadas en cada una.
4. Con base en los análisis llevados a cabo en 2 y 3 se seleccionó la estrategia a seguir: procesamiento de clasificación en el Borde.
5. La selección de la arquitectura de referencia se realizó siguiendo la estrategia elegida y con base los criterios de diseño propuestos en [45], para la creación de plataformas IoT en Agricultura Inteligente:
   I. Aumento en la cantidad de datos, especialmente de todo tipo de equipos agrícolas.
   II. Interoperabilidad entre varios sistemas a nivel de plantación y en toda la red de la cadena de suministro que la rodea.
   III. Estandarización de datos
   IV. Ir más allá de la pequeña escala y enfocarse en la producción agrícola a nivel regional, y al mismo tiempo.
   V. cumplir con diferencias nacionales o regionales en las prácticas agrícolas

   Finalmente, no se tuvo en cuenta el quinto criterio, por la dificultad que involucraba tener en cuenta todas las prácticas agrícolas de un país o una región.

2.3 Selección del conjunto de datos y creación de un nuevo *dataset*

Para el entrenamiento del método elegido se seleccionó el conjunto de datos PlantaVillage [46]. Sin embargo, después de detectar problemas con el *dataset*, que ya habían sido reportados en la literatura, fue necesario crear un nuevo conjunto de datos. Los pasos seguidos para la obtención de este nuevo *dataset* fueron los siguientes:

2.3.1 Selección del dataset.

En el capítulo 3 se llevó a cabo un análisis de las aplicaciones de la clasificación de imágenes en las plataformas de Agricultura Inteligente. Con base en el análisis de las aplicaciones se eligió el caso de estudio: detección de enfermedades, debido tres criterios:

a. Mayor impacto en el dominio de la agricultura.
b. La flexibilidad de las técnicas usadas.
c. La variedad de fuentes de datos.

Se seleccionó el conjunto de datos PlantVillage por ser uno de los que más especies de cultivos y clases de enfermedades contiene y también es uno en los que muchos métodos de clasificación de imágenes se han utilizado.

Se realizó un análisis del poder predictivo de la silueta de las hojas, debido a que estas no son necesariamente influenciadas por las enfermedades, sino también por factores fenotípicos y genotípicos. Este análisis arrojó que existe un poder predictivo en la silueta.

2.3.2 Segmentación de imágenes.

Debido a que el fondo de las imágenes del *dataset* seleccionado puede generar un sesgo en la clasificación de acuerdo con [47], se realizó un proceso de segmentación tendiente a eliminar el color del fondo. El procedimiento para realizar la segmentación fue el siguiente:

1. Se revisaron los métodos empleados en trabajos anteriores, realizando una búsqueda en la literatura.
2. Se seleccionaron los métodos más usados y se aplicaron al *dataset,* luego se compararon los resultados y se eligió el método que mejores resultados arrojó

En la tabla 3, se hace un resumen de la literatura revisada y las técnicas de preprocesamiento y segmentación aplicadas en diferentes plataformas IoT de Agricultura Inteligente.

Tabla 3. Técnicas de preprocesamiento y segmentación usadas en algunas plataformas de IoT de agricultura inteligente reportadas en la literatura

Objetivo de la plataforma IoT	Técnicas de preprocesamiento y segmentación	Referencia
Detección de plagas en cultivo de repollo en invernadero.	• Ecualización basada en histogramas, para ajuste de brillo y contraste. • Conversión del espacio de color de RGB a escala de grises. • *Clustering* con k-means. •Bi- level *Thresholding* • *Erosion and dilation.*	[48]
Detección de enfermedades en hojas de cultivos de algodón.	• Filtro espacial de nitidez (filtro de mediana). • Conversión del espacio de color de RGB a YCbCr. • Bi-level thresholding, • Conversión del espacio de color binario a RGB *(Masked RGB).* • Conversión del espacio de color de RGB a escala de grises.	[49]
Detección de enfermedades en hojas de diferentes cultivos.	• Transformación de espacio de color RGB a HSV. • Enmascaramiento de pixeles verdes (*Masking green pixels).*	[50]
Detección de enfermedades en hojas de cultivos de arroz.	• Transformación de espacio de color RGB a HSV. • Ecualización basada en histogramas, para ajuste de brillo y contraste.	[51]
Detección de plagas en trampas de cultivos de manzano.	• Transformación de espacio de color de RGB a escala de grises. • Suavizado del marco con un filtro gaussiano. • Extracción de bordes a través de un operador *Canny.* • Dilatación y erosión de la imagen.	[52]
Detección de enfermedades en hojas de un cultivo de banano.	• Transformación de espacio de color de RGB a escala de grises. • Ecualización basada en histogramas, para ajuste de brillo y contraste. • *Clustering* con k-means.	[53]
Detección de plagas en árboles de ojo de dragón *(Dimocarpus longan).*	• Transformación de espacio de color de RGB a escala de grises. • Aplicación de filtros Sobel para detección de bordes verticales.	[54]
Detección de enfermedades en hojas de un cultivo de pepino.	• Transformación de espacio de color RGB a Lab. • *Clustering* con k-means.	[55]

Nota. Los hallazgos en la literatura sugieren que los métodos de preprocesamiento y segmentación de imágenes integrados en las plataformas de Agricultura Inteligente generalmente tienen un esquema similar: primero se realiza una transformación del espacio de color, para después realizar segmentación de las imágenes, y finalmente llevar a cabo un agrupamiento de clases, para lo cual se emplea comúnmente el método de *K – means.*

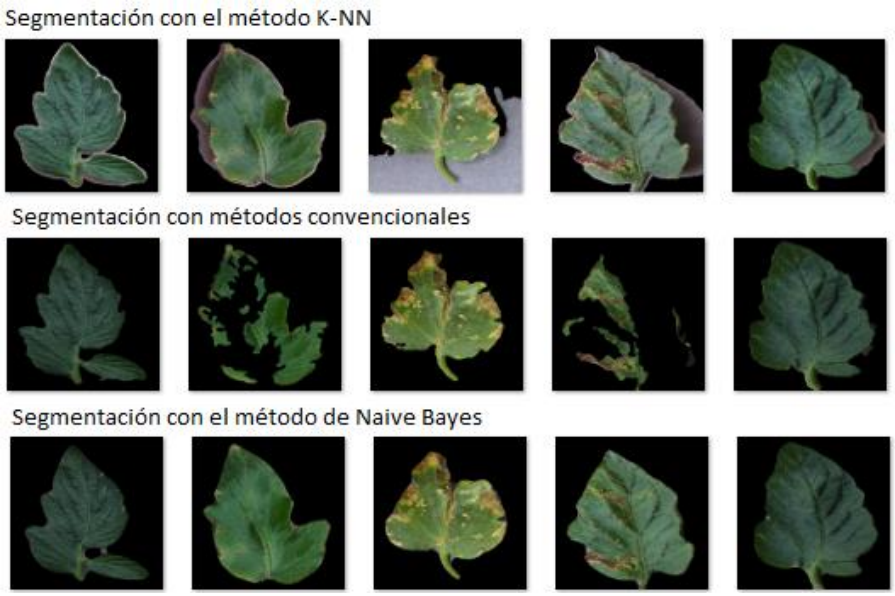
Fuente: elaboración propia.

Con base en estos hallazgos se evaluaron tres métodos diferentes:

1. Segmentación usando métodos convencionales de procesamiento digital de imágenes, (*Mean Adaptive Threshold* [56] y *Otsu Auto Threshold* [57]).
2. Segmentación con KNN (*K-Nearest Neighbors*) [58]
3. Clasificador de Bayes [58].

Con base en una inspección visual de los resultados, se eligió el método del clasificador de Bayes para realizar la fase de segmentación. Posteriormente se aplicaron técnicas de procesamiento morfológico de imágenes (*dilation & erosion*) y finalmente se eliminó el ruido residual con la aplicación de las técnicas de *Median blur* y *Fill holes* [56], como se observa en la figura 5.

Figura 5. Ejemplo de los resultados obtenidos en la elección del método de segmentación



Fuente: elaboración propia.

Para esta fase del preprocesamiento se utilizó el *framewok* PlantCV (*Plant Phenotyping Computer Visión*) que es un paquete de software de análisis de imágenes de código abierto destinado a la determinación del fenotipo de las plantas [59].

2.3.3 Clasificador de Bayes para la segmentación de imágenes.

Como se mencionó anteriormente, luego de la comparación de resultados de segmentación obtenidos con los tres métodos evaluados, se utilizó un clasificador de Bayes para la fase pre-procesamiento que consiste en la segmentación.

De manera formal el clasificador de Bayes actúa de la siguiente forma: la probabilidad de que un vector x (por ejemplo, los pixeles de una imagen RGB) provengan de la clase c_j (Por ejemplo, la clase enfermedad i) se denota por $p\,(c_i \mid x)$. Si el clasificador decide que x proviene de la clase c_j cuando en realidad proviene de c_i , incurre en una pérdida, denotada por L_{ij}. Como el vector x puede pertenecer a cualquiera de las N_c clases posibles, la pérdida promedio incurrida al asignar x a la clase c_j es como observamos en la ecuación 1:

$$r_j\,(x) = \sum_{k=1}^{N_c} L_{kj} p(c_k \mid x) \quad \textbf{(1)}$$

La cantidad $r_j(x)$ es llamada pérdida en la terminología de la teoría de decisiones. Por otro lado, el teorema de Bayes indica (ecuación 2) que:

$$p(c_k \mid x\,) = \frac{p(x \mid c_k) * p(c_k)}{p(x)} \quad \textbf{(2)}$$

Si reemplazamos la probabilidad condicional como indica Bayes, en la ecuación 3 tenemos:

$$r_j\,(x) = \frac{1}{p(x)} \sum_{k=1}^{N_c} [L_{kj} p(x \mid c_k) * p(c_k)] \quad \textbf{(3)}$$

Donde $p(x \mid c_k)$ es la función de densidad de probabilidad (PDF) de los datos tomados o muestras (por ejemplo, de los pixeles muestreados pertenecientes a cada clase) c_k, y $p(c_k)$ es la probabilidad de ocurrencia de la clase c_k (a $p(c_k)$ también se le llama la probabilidad a priori). Ahora, como $1/p(x)$ es positivo y común a todos los $r_j\,(x)$, $j = 1, 2, \ldots, N_c$, se puede eliminar de la ecuación 3 sin afectar el orden relativo de estas funciones desde el valor más pequeño hasta el más grande. La expresión para la pérdida promedio entonces se reduce a la ecuación 4:

$$r_j\,(x) = \sum_{k=1}^{N_c} [L_{kj} p(x \mid c_k) * p(c_k)] \quad \textbf{(4)}$$

Dado un vector desconocido, el clasificador tiene N_c clases posibles de las cuales elegir. Si el clasificador calcula $r_1(x), r_2(x), \dots r_{N_c}(x)$ para cada vector x y asigna el vector a la clase con la pérdida más pequeña, la pérdida promedio total con respeto a todas las decisiones será mínimo. El clasificador que minimiza el total de la pérdida promedio se llama el clasificador de Bayes. Este clasificador asigna un vector desconocido x a la clase c_i, si $r_i(x) < r_j(x)$ para $j = 1, 2, \dots, N_c; j \neq i$. En otras palabras, como vemos en la ecuación 5, se asigna x a la clase c_i, si:

$$\sum_{k=1}^{N_c}[L_{kj}p(x \mid c_k) * p(c_k)] < \sum_{q=1}^{N_c}[L_{qj}p(x \mid c_q) * p(c_q)] \textbf{ (5)}$$

para todo j; $j \neq i$. A la pérdida por una decisión correcta generalmente se le asigna un valor de 0, y a la pérdida por cualquier decisión incorrecta generalmente se le asigna un valor de 1. Entonces, la función de pérdida da como resultado la ecuación 6:

$$L_{ij} = 1 - \delta_{ij} \textbf{ (6)}$$

Donde $\delta_{ij} = 1$ si $i = j$, y $\delta_{ij} = 0$ si $i \neq j$. La ecuación 6 indica una pérdida de uno para decisiones incorrectas y una pérdida de cero para decisiones correctas, sustituyendo la ecuación 6 en la ecuación 4 obtenemos las ecuaciones 7 y 8:

$$r_j(x) = \sum_{k=1}^{N_c}[(1 - \delta_{kj}) * p(x \mid c_k) * p(c_k)] \textbf{ (7)}$$

$$r_j(x) = p(x) - p(x \mid c_j) * p(c_j) \textbf{ (8)}$$

El clasificador de Bayes luego asigna el vector x a la clase c_i si, para todo $j \neq i$,

$$p(x) - p(x \mid c_i) * p(c_i) < p(x) - p(x \mid c_j) * p(c_j) \textbf{ (9)}$$

O equivalentemente, si

$$p(x \mid c_i) * p(c_i) > p(x \mid c_j) * p(c_j) \;\; j = 1, 2, \dots, N_c; j \neq i \textbf{ (10)}$$

Luego, el clasificador de Bayes para la función de pérdida de 0 a 1, computa funciones de decisión de la forma:

$$d_j(x) = p(x \mid c_j) * p(c_j) \quad j = 1, 2, \ldots, N_c \textbf{ (11)}$$

Y asigna un vector x a la clase c_i si $d_i(x) > d_j(x)$ para todo $j \neq i$.

2.3.4 Recorte de las imágenes para evitar sesgo debido a la forma de las hojas.

Con el fin de evitar el sesgo debido a la forma de las hojas del *dataset* original, las imágenes segmentadas se subdividieron en nueve recuadros (*tiles* en inglés) de 85 x 85 pixeles cada una, y se seleccionaron solamente los recuadros de imágenes con los siguientes criterios para conformar el nuevo *dataset:*

1. Imágenes que tuvieran más del 80 % de pixeles con valor mayor de cero que corresponden al objeto de interés (píxel > 0 pertenece a las hojas, píxel = 0, pertenece al fondo), los recuadros con menos del 80 % de pixeles > 0 se eliminaron (figura 20).
2. El umbral del 80 % es arbitrario, y se tomó como valor de referencia porque se encontró en pruebas preliminares que arrojaba un número de recuadros igual o mayor de cuatro, lo cual permitió hacer una clasificación con la menor pérdida de información posible. Sin embargo, es importante aclarar que este umbral puede ser variable, dependiendo del tamaño y tipo de imágenes con las que se cuente.
3. Se eliminó la clase *Bacterial spot.* debido a que esta presenta diferentes etapas de la enfermedad en el *dataset* original, lo cual representa un reto adicional, pues las etapas tempranas presentan gran similitud con la clase sanas (*healthy),* y no permite su correcta clasificación. La detección de las enfermedades en diferentes etapas representa un reto adicional que no fue del alcance de este trabajo.
4. Finalmente se obtuvo un nuevo *dataset* conformado por 26148 imágenes y nueve clases (ocho enfermedades y la clase sana).

2.4 Selección del método de clasificación de imágenes

Para seleccionar el método de clasificación de imágenes se siguieron los siguientes pasos:

1. Se revisaron trabajos previos que aplicaron técnicas de *Machine Learning* al *dataset* PlantVillage para realizar clasificación de imágenes y se compararon los resultados obtenidos.

2. Se revisaron trabajos previos que aplicaron técnicas de Deep Learning al *dataset* PlantVillage para realizar clasificación de imágenes y se compararon los resultados obtenidos.
3. Luego de realizar las comparaciones de los pasos 1 y 2 se aplicaron dos criterios de selección:
    a. Tamaño: Métodos con menos de 5 millones de parámetros fueron elegidos. Este número de parámetros fue elegido como umbral, debido a que las redes del estado del arte diseñadas para trabajar con restricciones de capacidad de cómputo están en general por debajo de este valor [60] [61], [62], [63], [64], [65].
    b. Desempeño: Métodos que obtuvieron desempeños de exactitud (*Accuracy*) igual o superior al 90%. Esta métrica se eligió debido a que la métrica más usada en la literatura para medir el desempeño de los modelos es la exactitud y no hay mucha información sobre las demás métricas en la literatura revisada.
4. Luego, adicionalmente se eligieron métodos que están en el estado del arte, y fueron creados para trabajar en contextos de baja capacidad de cómputo y que no aparecieron en los trabajos revisados, para acabar de completar el conjunto de métodos pre-seleccionados.

2.4.1 Selección de los hiperparámetros para el entrenamiento.

Para la elección de los hiperparámetros del entrenamiento se realizó una comparación entre los métodos de optimización, basándose en una revisión de literatura. Los métodos evaluados fueron:

a. Búsqueda en cuadricula (Grid Search) [66]
b. Búsqueda en cuadricula aleatoria (Randomized Grid Search) [67]
c. Optimización Bayesiana (Bayesian Optimization) [68]
d. Algoritmos genéticos (Genetic Algorithms) [69]

Después de la comparación fue elegido el método de Optimización Bayesiana. Este método presenta ventajas respecto a los demás que son presentadas en la sección 6.4. El método de optimización bayesiana fue aplicado a los siete modelos elegidos para este estudio. El método

se aplicó con la librería SKOPT de *scikit-optimize.* Los fundamentos teóricos de la Optimización Bayesiana se presentan a continuación:

Dada una función "caja negra" (*Blackbox function*), la cual queremos aproximar, y un vector de datos $\boldsymbol{f}$. Los pasos del algoritmo son:

I. Se comienza con un pequeño número de puntos muestreados aleatoriamente escogidos (llamados *Bag1*).
II. Se usan estos *Bag1* para computar una función sustituta (*surrogate function*), que se aproxima a la función "caja negra".
III. Se itera en un bucle.
IV. Se adiciona un punto adicional al Bag1 usando una función de adquisición. (*Acquisition Function*).

i. Se re-evalua la función sustituta.
ii. Se continua el bucle a menos que ocurra alguna de las siguientes opciones:
    a) El máximo de la función sustituta no cambie.
    b) O se alcance la varianza por debajo de algún umbral.
    c) O $\boldsymbol{f}$ se agota.

- Se asume que el proceso aleatorio de la búsqueda es estacionario y sigue una distribución Gaussiana (ecuaciones 12 y 13).

$$\boldsymbol{f} \equiv \begin{pmatrix} f(x_1) \\ \vdots \\ f(x_n) \end{pmatrix} \sim N \quad \textbf{(12)} \qquad g(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{\frac{(x-\mu)^2}{2\sigma^2}} \quad \textbf{(13)}$$

- Estacionario quiere decir que la distribución de probabilidad conjunta no cambia en el tiempo.
- La media de $\boldsymbol{f}$ es **m**.
- La covarianza tendrá una forma gaussiana y es aproximada por una función RBF (*Radial Basis Function*) (ecuación 14).

$$K(x, x_i) = \sigma^2 e^{-\frac{1}{2l^2}\|(x-x_i\|^2} \textbf{ (14)}$$

- Función de predicción: formula bayesiana para la probabilidad condicional (ecuaciones 15 y 16).

$$p(f(x)|\boldsymbol{f}) = \frac{N(f(x),f(x_1),\ldots,f(x_n)|0,\tilde{C})}{N(f(x_1),\ldots f(x_n)|0,\tilde{C})} \textbf{ (15)}$$

$$p(f(x)|\boldsymbol{f}) = N(f(x)|\mu, \sigma^2) \textbf{ (16)}$$

Donde:

$\mu = \boldsymbol{k}^T C^{-1} \boldsymbol{f}$

$\sigma^2 = K(0) - \boldsymbol{k}^T C^{-1} \boldsymbol{k}$

$$C = \begin{pmatrix} K(0) & K(x_1 - x_2) & \ldots \\ K(x_2 - x_1) & \ddots & \vdots \\ \vdots & \ldots & K(0) \end{pmatrix}, \tilde{C} = \begin{pmatrix} K(0) & \boldsymbol{k}^T \\ \boldsymbol{k} & C \end{pmatrix}\mu$$

$$\boldsymbol{k} \equiv \begin{pmatrix} K(x - x_1) \\ \vdots \\ K(x - x_n) \end{pmatrix}$$

Con esta función de predicción podemos decir que, en promedio, nuestra función desconocida asumirá el valor $\boldsymbol{\mu}$ en $\boldsymbol{x}$ con una varianza $\boldsymbol{\sigma}$.

2.4.2 Entrenamiento de los métodos pre-seleccionados.

El entrenamiento de los métodos pre-seleccionados se realizó en la plataforma de *Google-Colab* con lenguaje *Python* y las principales librerías utilizadas fueron *Tensorflow, Numpy y Matplotlib*. Se utilizaron 30 épocas para cada entrenamiento y los hiperparámetros elegidos para cada modelo con la Optimización Bayesiana que se presentan el en capítulo 6 (tabla 12).

2.4.3 Métricas utilizadas para medir el desempeño.

Hay muchas formas de medir el desempeño de los modelos usados para clasificación en aprendizaje supervisado. Sin embargo, en este trabajo de investigación se usaron las métricas basadas en la matriz de confusión, porque en la revisión de literatura realizada, los hallazgos muestran que son las más usadas por su practicidad, fácil interpretación y confiabilidad para medir desempeño. Fueron llamadas por sus nombres en inglés para evitar confusiones en la traducción. Las explicaciones formales fueron extraídas de [70] y [71].

1. **Matriz de confusión.** Una matriz de confusión es una de las herramientas más usadas en Aprendizaje de Máquinas para la evaluación de modelos de clasificación. Es una matriz que compara el número de predicciones para cada clase que son correctas y las que son incorrectas. Las cuatro principales métricas que se extraen de la matriz de confusión son:

   Verdaderos positivos (*TP, True Positive*): el número de observaciones positivas que el modelo predijo correctamente como positivas.

   Falso positivo (*FP, False Positive*): el número de observaciones negativas que el modelo predijo incorrectamente como positivas.

   Negativo verdadero (*TN, True Negative*): el número de observaciones negativas que el modelo predijo correctamente como negativas.

   Falso negativo (*FN, False Negative*): el número de observaciones positivas que el modelo predijo incorrectamente como negativas.

2. ***Accuracy*.** La exactitud general de un modelo es simplemente el número de predicciones correctas dividido por el número total de predicciones. La exactitud de predicción de un modelo se mide con un valor porcentual entre 0 y 1; un valor de 1 indica un modelo con un desempeño perfecto. Se puede definir el *Accuracy* como la relación entre el número de predicciones correctas y el número total de predicciones (17).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad \textbf{(17)}$$

3. ***Precision.*** Mide qué tan bueno es el modelo para identificar correctamente la clase positiva. En otras palabras, de todas las predicciones para la clase positiva, ¿cuántas fueron realmente correctas? Usando solo esta métrica para optimizar un modelo estaríamos minimizando los falsos positivos (18).

$$Precision = \frac{TP}{TP+FP} \textbf{ (18)}$$

4. ***Recall.*** Mide qué tan bueno es el modelo para predecir correctamente todas las observaciones positivas en el conjunto de datos. Sin embargo, no incluye información sobre los falsos positivos (19).

$$Recall = \frac{TP}{TP+FN} \textbf{ (19)}$$

5. ***F1 score.*** Es la media armónica de *Precision y Recall. F1 score*, dará un número entre 0 y 1. Si el valor es 1, indica *Precision y Recall* perfectas. Si el valor 0, significa que la *Precision o el Recall* son 0 (20).

$$F1 = 2 * \frac{Precision*Recall}{Precision+Recall} \textbf{ (20)}$$

2.5 Implementación del método de clasificación de imágenes

La implementación del método estuvo en el marco del proyecto de investigación financiado por la Universidad Nacional de Colombia titulado: *"Implementación de un modelo de smart farming, mediante la integración de tecnologías 4.0 e Inteligencia Artificial, para pequeños productores de hortalizas bajo invernadero"*. Para la implementación del método seleccionado se siguieron los siguientes pasos:

1. Se seleccionaron dos predios rurales en que los agricultores se dedican a la producción de hortalizas bajo invernadero.
2. Con base en la arquitectura de referencia elegida y las tecnologías revisadas en el capítulo 4 se diseñó la arquitectura propia y su topología para distribuir los elementos de hardware en los dos predios.
3. Se instalaron los dispositivos en un transcurso de un mes aproximadamente.

4. Luego de la instalación se procedió a realizar pruebas de transmisión hasta verificar que todos los dispositivos transmitían sin restricciones.
5. Se realizó una evaluación del método propuesto, que consistió en modificar la arquitectura original reduciéndola en tamaño un 75, 50 y 25%.

2.5.1 Métricas utilizadas para la evaluación del método.

Se utilizaron dos métricas para medir el desempeño del método en condiciones de reducción de su arquitectura:

1. **Operaciones de punto flotante por segundo:** Para medir la capacidad de cómputo del hardware usado se utilizó la métrica de operaciones de punto flotante por segundo o FLOPs (por sus siglas en inglés), estandarizada por la IEEE [72]. Esta es una medida del rendimiento de una computadora, especialmente en cálculos científicos que necesitan un gran uso de operaciones. En el caso del dispositivo utilizado (Raspberry Pi4 4GB - 64bits) para la implementación del modelo propuesto, la capacidad es de 13,5 GFLOPs, medición llevada a cabo por el "*Weaver Computer Engineering Research Group*" de "*University* of *Maine*" [73].
2. **Tiempo de latencia**: La medición del tiempo de latencia consistió en medir el tiempo que tarda el modelo en realizar las predicciones. Para esta prueba se utilizaron 45 imágenes en todas las pruebas.

2.6 Resumen del capítulo

En este capítulo se presentaron los aspectos metodológicos utilizados en cada una de las cinco fases de esta investigación: (1) Revisión Sistemática de Literatura (RSL), (2) elección de una arquitectura de referencia, (3) creación de un nuevo conjunto de datos, (4) selección del método de clasificación de imágenes y (5) implementación del método elegido en una plataforma IoT de Agricultura Inteligente.

El objetivo de este capítulo fue dar un panorama general de los pasos seguidos en la consecución de los objetivos de esta tesis doctoral.

En el siguiente capítulo se muestra un estado del arte sobre las aplicaciones de los métodos de clasificación de imágenes en plataformas de Agricultura Inteligente. La cual fue realizada con

base en la RSL, y que buscó inicialmente elegir un caso de estudio para luego aplicarlo con el método de clasificación elegido y la arquitectura de referencia.

3 Clasificación de imágenes en Plataformas de Agricultura Inteligente

Antes de proponer un método de clasificación de imágenes en este capítulo se hace un análisis del estado del arte, sobre las plataformas de Agricultura Inteligente, sus aplicaciones, desafíos y cuáles los principales cultivos en los que vienen siendo usadas. En este capítulo se da respuesta a la siguiente pregunta de la RSL: ¿en qué tipo de problemas se enfocan las plataformas IoT de Agricultura Inteligente, que incorporan métodos de clasificación de imágenes?

Detectar cuales son los problemas que vienen siendo abordados en el dominio de la agricultura con este tipo de tecnologías es de gran importancia para determinar cuáles han sido las áreas de mayor impacto y donde todavía existen vacíos en el conocimiento. El análisis que a continuación se presenta también sirve de insumo para trabajos de investigación posteriores que vayan en la misma línea.

Este tercer capítulo está organizado en dos secciones principalmente: en la primera parte se presenta la arquitectura general que tienen las plataformas IoT en contextos agroindustriales, su estructura, las principales tecnologías usadas en las capas de captura de datos, comunicaciones, analítica y aplicaciones. En segundo lugar, se presentan las aplicaciones de estas plataformas, los problemas específicos del dominio de la agricultura que intentan resolver. Finalmente se hace un resumen del capítulo.

3.1 Plataformas de IoT de Agricultura Inteligente

Kevin Ashton propuso por primera vez el concepto de IoT en 1999, y se refirió al IoT como una serie de objetos conectados interoperables e identificables de forma única, inicialmente se basó en la identificación por radiofrecuencia (*RFID*, por sus siglas en inglés) [74]. Sin embargo, es solo hasta principios de la segunda década del siglo XXI, que comienza a consolidarse el uso del IoT como herramienta para aplicarse en agricultura, y se propone el término de Agricultura Inteligente en el proyecto *SmartAgriFood,* patrocinado por la Unión Europea [75], [76].

Aunque la evolución de los sistemas IoT sigue en constante desarrollo, las arquitecturas que se implementan en contextos agroindustriales constan generalmente de cuatro capas [15], como puede observarse en la figura 6.

Figura 6. Arquitectura IoT en contextos agroindustriales

Fuente: elaboración propia con base en [15]

3.1.1 Capa física o de captura de datos.

Esta capa también llamada capa de percepción es donde se ubican los diferentes dispositivos de captura de datos “las cosas”. En contextos agrícolas, en esta capa se ubican sensores para captura de variables climáticas, como temperatura, humedad relativa o velocidad del viento, también se consideran generalmente sensores de condiciones físicas o químicas del suelo [15], [77]–[80].

En esta capa también se ubican los sensores de imágenes (cámaras), que pueden ser de diferente tipo, por ejemplo, cámaras fijas de dispositivos IoT, cámaras de teléfonos móviles o incluso cámaras instaladas en drones [15], [79], [81]. Es en esta capa donde se usan las tecnologías para crear redes de “cosas”, tales como WSN (*Wireless Sensor Network*), RFID (*Radio Frequency Identification*), y recientemente, NFC *(Near Field Communications*), que se encargan de interconectar los diferentes dispositivos en red [82].

De igual manera en esta capa se encuentran los actuadores, que son los dispositivos encargados de activarse de acuerdo con algún protocolo, para realizar alguna actividad que

permita tener el control de alguna variable [15], por ejemplo, encender la bomba encargada de impulsar el agua a través de un sistema de riego, o un ventilador para bajar temperatura en un invernadero, o un robot encargado de eliminar malezas en un campo agrícola.

3.1.2 Capa de comunicaciones.

Las redes de dispositivos interconectados de la primera capa envían los datos a diferentes nodos que se encargan de guardarlos y retrasmitirlos a un servidor que puede estar ubicado en el la “niebla” (*Fog Computing*) o en la “Nube” (*Cloud Computing*), estos nodos son mejor conocidos como puertas de enlace o *gateways* [83]. Puede existir comunicación de doble vía entre los dispositivos y los nodos, con el fin de activar algún actuador [82]. Las tecnologías que aparecen en esta capa son protocolos de comunicación construidos sobre estándares inalámbricos, como el IEEE 802.15.4, mejor conocido como LR-WPAN (*Low-Rate Wireless Personal Area Network*), que facilitan la comunicación entre las redes y las puertas de enlace, y estas a su vez con los nodos finales que son los que se conectan a Internet. Dichos protocolos incluyen ZigBee, SigFox, LoRaWAN, ONE-NET, wirelessHART, ISA100.11, 6LowPan, Bluetooth, entre otros [82], [84].

3.1.3 Capa de servicio o middleware.

La capa de servicio, también conocida por su término en inglés como *middleware*, es considerada por algunos autores la capa principal del sistema IoT, pues es donde se centran las herramientas más potentes de almacenamiento y analítica de datos [85]. Está constituida principalmente por tecnología de software, que se encarga de la gestión de los datos transmitidos desde la capa de comunicaciones vía Internet, así como su almacenamiento, análisis, visualización y seguridad [15]. En esta capa se pueden encontrar soluciones de código abierto (*open source*) que en línea permiten la integración y visualización de los datos y la información generada a través de herramientas de analítica. El *middleware* ha ganado mucha atención en años recientes, debido a su papel principal en la simplificación del desarrollo de nuevos servicios [82], [85].

3.1.4 Capa de aplicaciones.

Finalmente, la capa de aplicaciones consume los servicios de la capa anterior en la arquitectura, y es la que permite al usuario gestionar el sistema completo [13], [15]. En esta capa residen todas las diferentes aplicaciones, como las relacionadas con el apoyo a la toma de decisiones (DSS – *Decision Support System*), la gestión de operaciones de las plantaciones (FMIS – *Farm Management Information System*), los cuadros de mando o la planificación de recursos empresariales (ERP - *Enterprise Resource Planning*) [84].

3.2 Aplicaciones de la clasificación de imágenes en plataformas de Agricultura Inteligente

De acuerdo con Terence y Purushothaman [8], las técnicas de Agricultura Inteligente basadas en IoT tienen tres aplicaciones principales: sistemas de monitoreo y control, sistemas de irrigación automática y sistemas para el monitoreo de plagas y enfermedades. La distribución porcentual de estas aplicaciones de acuerdo con [8] es la siguiente: sistemas de monitoreo y control (52 % de los estudios revisados), sistemas de irrigación automática (33 % de los estudios revisados), y los estudios restantes se clasifican en monitoreo de plagas y enfermedades (15 %). Los dos primeros tipos de plataformas se basan principalmente en datos de variables físicas o químicas como, por ejemplo, temperatura, humedad relativa, humedad del suelo y pH del suelo. Es decir, no usan imágenes, por lo cual, transmiten datos relativamente livianos. Por otro lado, según [8], las plataformas de *Smart Farming* que usan imágenes como fuente principal de datos, son las que se clasifican como sistemas de monitoreo de plagas y enfermedades. Las cuales, por lo general tienen más restricciones de transmisión por el tamaño de los archivos.

De acuerdo también con las conclusiones en [8], la mayoría de los experimentos con plataformas IoT en agricultura, se han llevado a cabo en ambientes controlados (*indoor*), es decir, en condiciones de laboratorio, dejando claro al final, que uno de los principales retos de futuros estudios, es llevar a cabo más experimentos en condiciones reales (*outdoor*), con el fin de observar mejor el comportamiento de las plataformas en condiciones de campo y en tiempo real.

Con el fin de agrupar las principales aplicaciones que se les da a las plataformas de Agricultura Inteligente que incluyen métodos de clasificación de imágenes, los trabajos revisados en la RSL se clasificaron en cinco grupos diferentes, como se muestra en la tabla 4.

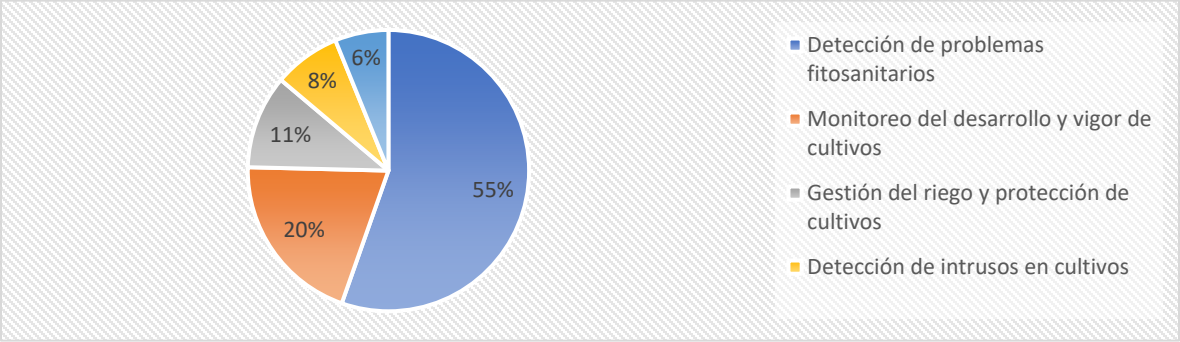
Tabla 4. Aplicaciones de la clasificación de imágenes en plataformas de Agricultura Inteligente

Aplicaciones	Referencias
Detección plagas y enfermedades	[18]–[23], [54], [86]–[114]
Monitoreo del desarrollo y vigor de cultivos	[115]–[127]
Gestión del riego y protección de cultivos	[128]–[134]
Detección de intrusos	[135]–[139]
Conteo de frutos y plantas	[140]–[143]

Fuente: elaboración propia.

En la figura 7, se puede ver que las tres aplicaciones principales ocupan el 86 %, la detección de plagas y enfermedades (55 %), el monitoreo del desarrollo y vigor de cultivos (20 %), y la gestión del riego y protección de cultivos (11 %), en segundo orden están: la detección de intrusos en cultivos (8 %) y el conteo de frutos y plantas (6 %), que ocupan el 14 % de las aplicaciones detectadas.

Figura 7. Distribución porcentual para las aplicaciones de clasificación de imágenes en las plataformas de Agricultura Inteligente



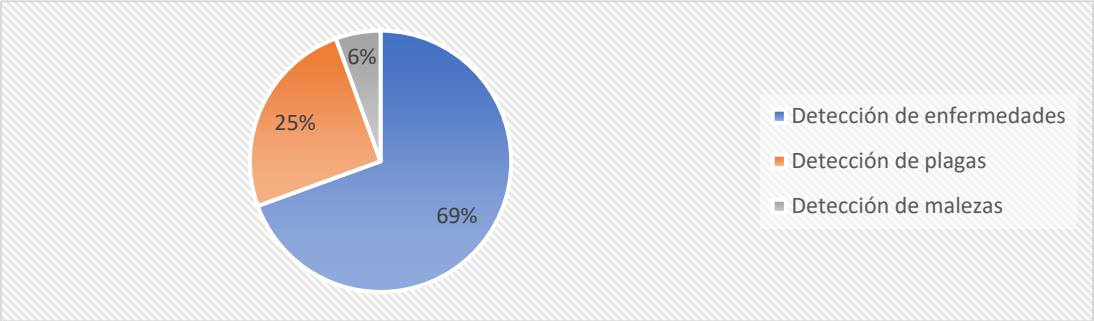
Fuente: elaboración propia.

3.2.1 Detección de problemas fitosanitarios.

La detección de problemas fitosanitarios es la aplicación que más se le da a la clasificación de imágenes en plataformas de Agricultura Inteligente, y se puede dividir en tres subgrupos de interés, como se muestra a continuación en la figura 8:

1. Detección de enfermedades.
2. Detección de plagas.
3. Detección de malezas.

Figura 8. Distribución porcentual de las aplicaciones de clasificación de imágenes en detección de problemas fitosanitarios



Fuente: elaboración propia.

Detección de enfermedades: Es la aplicación que más se reporta en la literatura consultada (69 %). La evidencia sugiere un fuerte interés por la detección de enfermedades en el cultivo de tomate [21]–[23], [88], [92], [95], [100]. Probablemente porque sigue siendo uno de los cultivos más extendidos a nivel mundial y porque representa una fuente importante de ingresos para muchos agricultores en el mundo. Algunos de estos trabajos se realizaron en cultivos bajo invernadero, en los que pueden existir algunas facilidades para la instalación de redes de sensores, por la protección que ofrecen estas estructuras [21], [88]. Además, el tomate es una hortaliza que se cultiva principalmente bajo invernadero.

El segundo cultivo en el que más se aplica la clasificación de imágenes para la detección de enfermedades es el arroz [61], [69], [70], probablemente porque también es uno de los alimentos básicos más importantes en el mundo. Le siguen en el trigo [89], [96], y el café [20], [105], por

otro lado, se detectaron aplicaciones en los que los modelos de clasificación se entrenaron utilizando conjuntos de datos (*datasets*), que incluían diferentes tipos de cultivos y de enfermedades [18], [91], [93], [101], y no se enfocaban en casos específicos. En general el *dataset* más usado, para el entrenamiento de estos modelos fue PlantVillage [46].

Finalmente, la evidencia mostró que las aplicaciones que involucran plataformas IoT con clasificación de imágenes en otros cultivos de importancia comercial no son muy abundantes, como por ejemplo en cultivos de cítricos [86], uva [102] y yuca (Cassava) [104].

Detección de plagas: La detección de plagas es uno de los usos que despierta más interés (25 % de los trabajos revisados en la detección de problemas fitosanitarios) [54], [107]–[114]. La mayoría de las implementaciones se hacen sobre trampas a las cuales se les adaptan cámaras, para luego clasificar los insectos atrapados. Existe un interés particular en los cultivos de manzana [107], [113], y cultivos bajo invernadero [108], [110]. También hay trabajos que proponen la detección de plagas de manera generalizada, sin referirse a algún tipo de cultivo en especial [54], [109], [112], [114], y en algunos trabajos se propone la detección por medio de drones, en combinación con otras técnicas de captura[54], [108]. Una de las principales dificultades en este caso es la correcta ubicación y el número de trampas necesarias en cultivos con grandes áreas, lo que puede hacer muy costosa su implementación si se hace utilizando cómputo en el Borde o se capturan imágenes de alta resolución, por lo cual uno de los principales retos es proponer sistemas de trampas de bajo costo [113], [114]. Por otro lado, la evidencia mostró en los trabajos revisados que la mayoría están a un nivel muy experimental, y hay retos importantes por resolver en su implementación a nivel de campo.

Detección de malezas: La evidencia recuperada en la RSL sugiere que la detección de malezas con visión artificial integrada a plataformas de Agricultura Inteligente aún está en sus inicios [19], [103], [106]. Existen aún varios retos por resolver en esta aplicación, como la gran variedad de malezas, la oclusión, la similitud con algunas especies de interés comercial, la etapa de desarrollo, su distribución en grandes áreas, entre otros [31], [144]–[146].

Una de las principales limitantes para la integración de estos métodos en plataformas IoT es probablemente el uso de imágenes aéreas en la mayoría de los trabajos citados, lo que

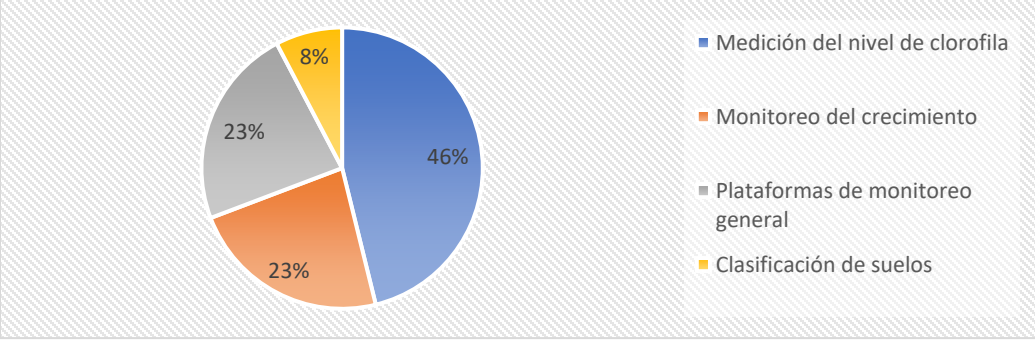
normalmente se traduce en imágenes de alta resolución. Se destaca el trabajo realizado en [103], en el cual que se propone un enfoque predictivo, pronosticando la aparición de malezas al combinar los datos de imágenes aéreas con variables climáticas y agronómicas capturadas con sensores. La solución de otros problemas combinados con la detección de malezas puede verse en la plataforma “AgriLora”, que se enfoca en la detección de enfermedades, problemas de nutrición y malezas al mismo tiempo [19].

3.2.2 Desarrollo y vigor de cultivos.

El seguimiento al desarrollo y el vigor de las plantas en los cultivos también es un uso que despierta un gran interés por parte de la comunidad académica, y se puede dividir en cuatro áreas de interés, como se muestra a continuación en la figura 9:

1. Medición del nivel de clorofila.
2. Monitoreo del crecimiento.
3. Plataformas para monitoreo general.
4. Clasificación de suelos.

Figura 9. Distribución porcentual de las aplicaciones de la clasificación de imágenes en el monitoreo del desarrollo y vigor de cultivos



Fuente: elaboración propia.

3.2.3 Medición del nivel de clorofila.

La detección de niveles de clorofila en las plantas se utiliza para medir indirectamente la presencia de nitrógeno en los tejidos, lo que representa una medida del vigor de las plantas

[115]–[117], [120], [121], [127]. La arquitectura IoT utilizada en la plataforma de Agricultura Inteligente llamada por sus autores como DIAS en [121], se enfoca en la detección de algunos índices espectrales como el NDVI (*Normalized Difference Vegetation Index*), el ExG (*Excess Greenness Index*), & NDI (*Normalized Difference Index*). Los autores en [88] proponen detectar a través de una imagen digital los tres principales contenidos de pigmentos fotosintéticos: clorofila, carotenoide y antocianina, con la ayuda de los sensores convencionales de las cámaras de celulares, lo que puede ser prometedor si se consideran los costos de esta solución.

3.2.4 Monitoreo del crecimiento.

En este tipo de monitoreos se intenta resolver el cálculo de tasas de crecimiento y estados de madurez de los cultivos, mediante el uso de cámaras estáticas [118], [119], [122] que hacen seguimiento al desarrollo vegetativo, la idea es poder solucionar la predicción del rendimiento, la medición de la biomasa producida, e incluso la calidad de frutos. La principal desventaja que presentan estos desarrollos es la implementación en campo, pues el crecimiento y desarrollo de plantas y frutos puede ser desigual en la mayoría de los cultivos, debido a variaciones genéticas, nutricionales, entre otras, por lo que sería necesario instalar muchos dispositivos estáticos en campos relativamente grandes para poder hacer un muestreo representativo del crecimiento. Como en otras propuestas la mayor parte de estos trabajos son a nivel experimental, y falta aún desarrollar trabajos para evaluar la implementación en campo.

3.2.5 Plataformas de monitoreo general.

En términos generales las arquitecturas IoT propuestas para este uso son más ambiciosas, pues se intenta resolver más de un problema a la vez, mediante diversas técnicas de clasificación de imágenes, diferentes tipos de sensores para capturar datos climáticos, diversas técnicas de pre-procesamiento, almacenamiento de datos y tecnologías de transmisión [123]–[125].

Una propuesta basada en tecnologías de bajo costo, se realiza con la arquitectura *FarmBeats* [123], en la que se propone la detección con cámaras elevadas en globos de helio, la transmisión de datos a través del espectro de televisión TVWS (*Television White Space*), que ha quedado libre en algunos países por el cambio de tecnología hacia la televisión digital, y el

uso de sensores de RF (*Radio Frequency*), que pueden aprovechar el uso de la tecnología WiFi incorporada en teléfonos celulares, para medir, por ejemplo, la humedad del suelo. Estas soluciones aún presentan ciertas limitantes, como por ejemplo el movimiento de los globos de helio en zonas de vientos fuertes, o la reglamentación inexistente para el uso del TVWS por parte de particulares en algunos países; sin embargo, son prometedores los resultados presentados por los autores, principalmente por los bajos costos de implementación.

3.2.6 Clasificación de suelos.

Este tipo de usos es muy escaso de acuerdo con la evidencia recopilada, solamente un trabajo fue detectado en la RSL. Sin embargo, puede ser un campo de investigación promisorio sobre todo en ambientes controlados, como los invernaderos [126]. Ya que la clasificación de suelos, de acuerdo con su estructura y coloración, permitiría ahorrar costos en algunos análisis que actualmente deben ser realizados en laboratorios. Por otro lado, este tipo de estudios pueden presentar ciertas limitantes que aún deben ser resueltas, como por ejemplo la instalación de cámaras en dispositivos móviles o robots que recorran diferentes sitios dentro de las áreas cultivadas.

3.2.7 Gestión de sistemas de riego y protección de cultivos.

El uso eficiente del agua incorporando el uso de imágenes para detectar presencia de estrés hídrico, y combinarlo con datos de sensores que detectan variables climáticas y del suelo para automatizar sistemas de riego, es uno de los usos que ha cobrado mayor interés [128]–[132]. Los cultivos que más reportan esta aplicación son: cítricos [130], arroz [131], lechuga [132], albahaca [133] y mostaza [129].

En este grupo se clasificaron también las aplicaciones que tienen como objetivo la protección de cultivos [133], [134], que se enfocan en la aplicación de pesticidas de manera eficiente. Los autores en [134], proponen un sistema de detección de tamaños de gotas para evaluar la calidad de la aspersión de pesticidas con drones, este puede ser un campo de investigación prometedor, en el que aún no hay muchos trabajos de acuerdo con la evidencia revisada.

3.2.8 Detección de intrusos en cultivos.

La detección de intrusos que pueden causar daños en cultivos es uno de los focos de interés en la implementación de plataformas de Agricultura Inteligente [135]–[139]. Existe un interés especial por detectar animales que causan daños, como es el caso de elefantes [137] o los ungulados (ciervos y jabalíes) [135], en países donde se expandido la frontera agrícola. También hay aplicaciones para detectar personas que invaden cultivos con fines delictivos como el robo de herramientas o productos [136], [139]. La mayor dificultad en este tipo de aplicaciones es la instalación de una red densa de cámaras, lo que aún representa un reto por resolver.

3.2.9 Conteo de frutos y plantas.

Este tipo de problemas ha despertado un gran interés por parte de los investigadores en el campo de la visión artificial, y existe literatura abundante enfocada en esta aplicación para diferentes cultivos en años recientes [32], [34], [147]–[163]. Sin embargo, la integración de estas soluciones a las plataformas de Agricultura Inteligente aún representa un campo muy amplio de investigación, que tiene retos importantes por resolver, como expresan varios autores en [140]–[143], como la oclusión, los frutos de colores similares a las hojas, la superposición de frutos, el traslapo de árboles de avanzada edad, entre otros.

Los cultivos de interés detectados son principalmente frutales como el mango [140] y manzana [142], pero también la detección de espigas de trigo con fines de fenotipado [143], y la detección de plántulas en algunas especies como curry, menta y cilantro [164][141] con el fin de medir porcentajes de germinación y necesidad de resiembra. La importancia de resolver este tipo de problema se centra en el pronóstico de producción, que sigue siendo un reto abierto para la agroindustria en general.

3.3 Resumen del capítulo

En este capítulo se presentó un estado del arte sobre las plataformas de Agricultura Inteligente, generalidades sobre su arquitectura IoT y principales componentes. Con base en la RSL desarrollada para este trabajo, también se pudieron determinar las principales aplicaciones que tienen las plataformas que realizan clasificación de imágenes, los cultivos en los que se vienen

usando estas aplicaciones, y algunos retos que siguen siendo objeto de investigación. El tema desarrollado en este capítulo fue fundamental para entender los usos de estas plataformas y las dificultades que enfrentan, así como, los sectores de la agricultura en los que se puede tener un mayor impacto con su aplicación. El uso que actualmente tiene un mayor interés es la detección de problemas fitosanitarios, y con base en esta evidencia, esta tesis doctoral abordó el caso de aplicación de la detección de enfermedades en cultivos para el método propuesto.

Con base en la RSL, en el siguiente capítulo se abordarán las diferentes estrategias arquitectónicas que están en el estado del arte para las plataformas IoT en Agricultura Inteligente, sus ventajas y desventajas, así como su uso dependiendo del caso de aplicación. Posteriormente se describen las arquitecturas de referencia preseleccionadas para realizar su comparación con base en los criterios definidos por los autores en [45], y finalmente se realiza una descripción de la arquitectura IoT de referencia seleccionada para la implementación.

4 Estrategias arquitectónicas

En el capítulo anterior se pudo determinar cuáles son las principales aplicaciones de las plataformas de Agricultura Inteligente que integran la clasificación de imágenes digitales en su arquitectura. Lo cual permitió conocer los subdominios de la agricultura donde actualmente se vienen desarrollando las plataformas y los problemas que se intentan resolver con esta tecnología. Como resultado del análisis en el capítulo anterior se determinó que el caso de uso en el cual se probaría el método propuesto en esta tesis doctoral sería el de la detección de enfermedades en cultivos.

En este capítulo se presenta el estado del arte sobre las estrategias arquitectónicas de referencia y tecnologías utilizadas para la integración entre la clasificación de imágenes digitales y las plataformas de Agricultura Inteligente con base en los hallazgos de la RSL. El análisis realizado con las evidencias encontradas permitió elegir arquitecturas de referencia candidatas, que fueron estudiadas posteriormente más a fondo, con el fin de determinar si cumplían con los criterios de selección elegidos para este trabajo y que fueron definidos en [165]. Para luego seleccionar la arquitectura de referencia.

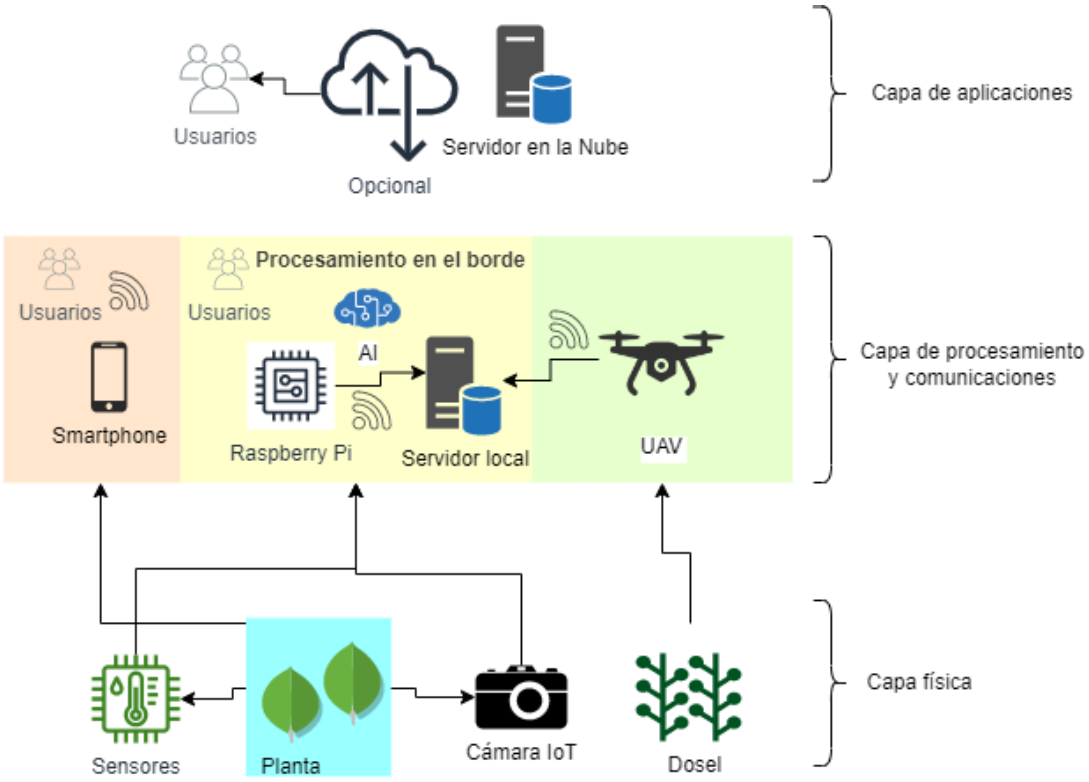
Este capítulo está organizado de la siguiente manera: primero se explica la estrategia de procesamiento en el Borde utilizada en las plataformas de Agricultura Inteligente revisadas. Luego se presenta la estrategia de procesamiento en la Nube. Posteriormente en la sección tres se presenta la estrategia de procesamiento mixta. Luego en la sección cuatro se describen de manera general las arquitecturas candidatas seleccionadas y finalmente se presenta una tabla comparativa con base en los criterios de selección con la cual se eligió la arquitectura de referencia utilizada luego para la implementación del método propuesto.

4.1 Proceso de clasificación en el Borde

La evidencia hallada en la RSL sugiere que, en la mayor parte de las aplicaciones detectadas la estrategia más usada es llevar a cabo todo el proceso de cómputo en el Borde (42 %), y así evitar los limitantes de las comunicaciones, este proceso es mejor conocido como *Edge Computing* [83].

El esquema general de las arquitecturas empleadas en esta estrategia se muestra a continuación en la figura 10.

Figura 10. Esquema arquitectónico general de plataformas que realizan el proceso de clasificación de imágenes en el Borde



Fuente: elaboración propia.

Las arquitecturas revisadas en esta estrategia son generalmente de tres capas: una primera capa donde se encuentran los dispositivos de captura o capa física, una segunda capa que agrupa las funcionalidades de comunicaciones y procesamiento y una tercera capa de aplicaciones. Estas arquitecturas basan su capacidad de cómputo en teléfonos móviles, microprocesadores Raspberry Pi y servidores locales, en ocasiones la carga de datos a servidores en la nube es opcional y se realiza para almacenar datos ya procesados y enviar alertas o informes a los usuarios.

La aplicación en la que más se detectó esta estrategia fue en la detección problemas sanitarios. Hay dos razones muy probables para que esto suceda son: primero la poca conectividad en

algunos entornos rurales, y segundo la gran variedad de casos (plagas o enfermedades) que pueden ser detectados de diversas maneras, y con imágenes que no exigen una alta resolución. Los dispositivos más usados para realizar el computo en el Borde son los microprocesadores Raspberry Pi [21], [23], [88], [92], debido principalmente a su bajo costo, la posibilidad de adicionar otros sensores y su capacidad de cómputo que puede ser variable dependiendo de la capacidad de la tarjeta microSD que se les instale. Un limitante del uso de estos dispositivos, es que las imágenes son generalmente capturadas en sitios fijos, lo que puede llevar a la instalación de varios nodos con estos microprocesadores en sitios homogéneamente distribuidos dentro de los cultivos.

Por otro lado, la evidencia muestra que el procesamiento en el Borde enfocado en la detección de enfermedades también se hace con teléfonos inteligentes (*smartphones*) [86], [90], [96], esto debido a que permite realizar toma de imágenes no destructivas directamente en la planta (y a su movilidad), lo que facilita las tomas de imágenes *off line* en sitios alejados y luego acceder a internet desde diferentes lugares o por diferentes alternativas, como redes WiFi o redes de servicio celular. La principal desventaja de esta técnica es que los datos en los teléfonos celulares son más susceptibles de pérdida, y las imágenes tomadas con diferentes cámaras y condiciones de iluminación y enfoque pueden variar enormemente, lo que causa alguna imprecisión al momento de realizar la clasificación. También existe una tercera alternativa usada en la detección de enfermedades, que es la instalación de servidores propios con mayor capacidad de cómputo, en sitios donde hay más facilidades de acceso a recursos e instalación [22], [94], [99].

En el subdominio de la detección de plagas por otro lado, hay una tendencia más fuerte a realizar el proceso de clasificación en la Nube [54], [108]–[111]. La evidencia muestra que no son muchas las aplicaciones en detección de plagas que usan cómputo en el Borde, muy pocos trabajos reportan esta estrategia [107], [112] en las que se implementan las llamadas trampas inteligentes, que realizan todo el computo *in situ*, apoyados en unidades de cómputo como las NCU (*Neural compute Stick*), que pueden tener incorporadas redes neuronales pre-entrenadas. Las marcas de NCU que reporta el estado del arte son: Movidius, Google Coral y Nvidia Jetson Nano [107]. Las principales técnicas de clasificación usadas en este dominio son en su mayoría

basadas en redes neuronales como MobileNets y sus diferentes versiones, VGG16, ResNet, AlexNet, técnicas de *Machine Learning* como Naive Bayes, SVM, KNN y Bosques Aleatorios.

En el dominio del monitoreo del desarrollo y vigor de cultivos, se hallaron una cantidad menor de trabajos basados en esta estrategia (38 %) [116], [117], [119], [120], [126]. En estos trabajos se identificó un interés importante por el desarrollo de aplicaciones móviles, [116], [117], [119], con fin de identificar el contenido de nitrógeno en las plantas, estas soluciones hacen principalmente uso de técnicas de procesamiento digital de imágenes a través de cambios y transformaciones del espacio de color, que normalmente no exigen mucha capacidad de cómputo.

En el caso del dominio de gestión del riego y protección de cultivos se descubrió una mayor proporción de trabajos que hacen uso de métodos basados en esta estrategia (71 %) [129], [131]–[134], en la que se prefiere usar como unidad de cómputo los microprocesadores Raspberry Pi, con captura de imágenes en dispositivos compatibles como las llamadas *Pi cameras.* En estos trabajos se identificó un enfoque por la clasificación de imágenes que detecta estados de estrés hídrico en las plantas o condiciones de humedad en el suelo, haciendo uso de métodos diversos de *Machine Learning*, principalmente Máquinas de Soporte Vectorial, Bosques Aleatorios y Redes Neuronales.

En las aplicaciones para la detección de intrusos en cultivos, la totalidad de los trabajos revisados emplean esta estrategia (100 %), [135]–[139]. Probablemente por la necesidad de identificar rápidamente si hay algún animal intruso en el cultivo y problemas de seguridad. También en este dominio el uso de esta estrategia se basa en el uso de microprocesadores Raspberry Pi y cámaras adaptables. La técnica de clasificación identificadas son *K Nearest Neighbour*, TINY-YOLOv3, *Single Shot detectors* (SSD) y Mobilenets.

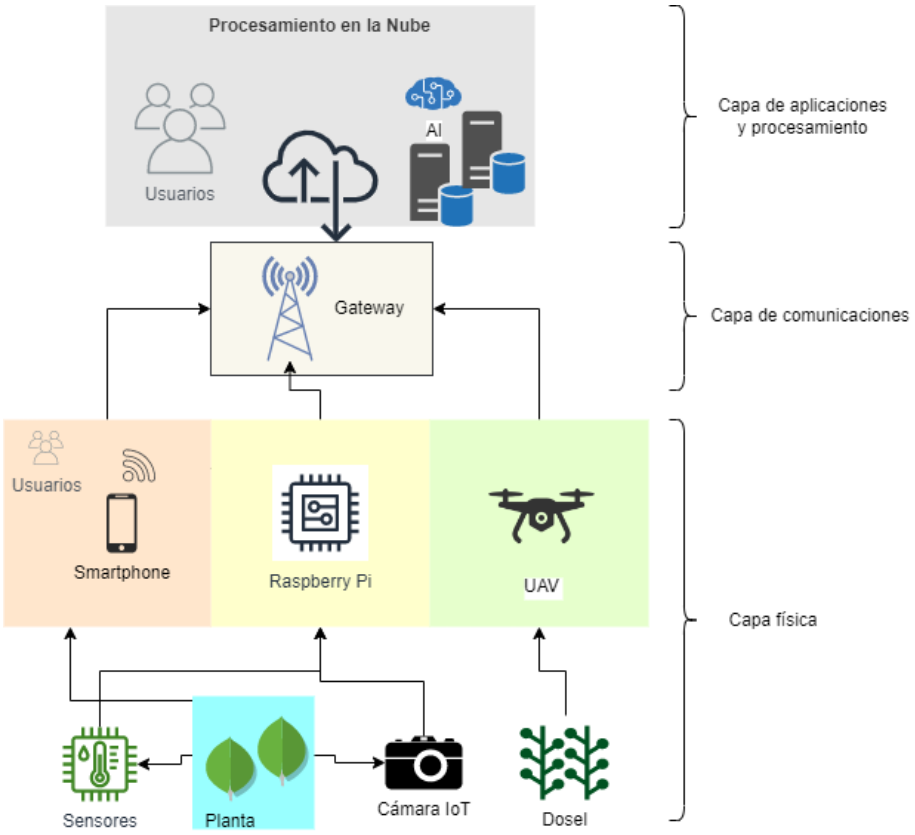
Finalmente, en la aplicación para el conteo de frutos y plantas también se identificó un mayor interés por esta estrategia en la mayoría de los trabajos revisados (75 %) [140]–[142]. De igual manera basan su estrategia en el uso de microprocesadores Raspberry Pi, y técnicas de clasificación, basadas en redes neuronales como TINY-YOLOv3, algunas técnicas de

procesamiento digital de imágenes clásicas, e incluso propuestas de redes neuronales del estado del arte modificadas.

4.2 Proceso de clasificación en la Nube

Otra de las estrategias es realizar lo contrario: todo el pre-procesamiento de las imágenes y su clasificación en la Nube *(Cloud Computing).* El modelo general de las arquitecturas con esta estrategia puede verse a continuación en la figura 11.

Figura 11. Esquema arquitectónico general de plataformas que realizan el proceso de clasificación de imágenes en la Nube



Fuente: elaboración propia.

Este tipo de arquitecturas también consta en la mayoría de los casos de tres capas: capa física, capa de comunicaciones, y capa de aplicaciones donde se realiza también todo el procesamiento de las imágenes capturadas y enviadas. Esta estrategia se usa principalmente cuando no se tienen limitantes de conectividad. De acuerdo con la evidencia recogida en la

RSL, esta es la segunda estrategia más usada en el dominio de la detección de problemas fitosanitarios (39 %). Los dispositivos más usados para la captura de las imágenes en este dominio son los teléfonos móviles [18], [97], [105], [166], en este caso el teléfono móvil solamente almacena las imágenes temporalmente. La principal ventaja de esta técnica es poder procesar imágenes de alta resolución, sin embargo, sigue existiendo el riesgo de pérdida de la información sobre todo para una captura *off line.* También es necesario contar con técnicas de transmisión sin restricciones importantes en el ancho de banda, y conectividad cerca de los cultivos.

En algunos casos se pueden capturar las imágenes con drones [89], [103], [167], lo cual es importante cuando se requiere detectar el comportamiento de enfermedades en el dosel, capturar imágenes hiperespectrales o multiespectrales, o la detección de malezas, aunque se hace necesario una buena conectividad también para la transmisión de imágenes de alta resolución. La capa de comunicaciones en esta estrategia se da principalmente por redes de telefonía celular 4G o 5G y Wifi [97], [105], aunque cada vez más se empieza a usar tecnología LoRa/LoRaWAN [19], [89], [102], [167] y Zigbee [103]. La captura con otro tipo de tecnología diferente a drones y teléfonos inteligentes se pudo evidenciar en las aplicaciones de monitoreo del desarrollo y vigor del cultivo, como las cámaras en globos de helio [123], y las cámaras Raspberry Pi v2 [120], [168].

De igual manera, como en la estrategia de cómputo en el Borde, las aplicaciones que se realizan en la detección de problemas fitosanitarios también hacen uso en gran medida de la estrategia de procesamiento en la Nube [54], [108]–[111], probablemente por los métodos de clasificación empleados en los que se requieren imágenes de alta resolución y gran capacidad de cómputo. Las técnicas de clasificación que se integran a estas plataformas no tienen casi ningún tipo de restricción en la capacidad de cómputo, por realizarse todo en servidores casi siempre de pago como AWS, Google, Azure entre otros. Los métodos de clasificación reportados en estos trabajos son en su mayoría basados en redes neuronales como NASNet, UNet, PSPNet, diferentes versiones de MobileNet, redes LSTM y algunas propuestas basadas en modificaciones de estas redes.

En la aplicación para monitoreo del desarrollo y vigor de cultivos esta es la estrategia más usada (54 %), [115], [118], [121], [122], [124], [125], [127], aunque también se apoya en tecnologías como los microprocesadores Raspberry Pi, para hacer un puente a los datos que posteriormente son enviados a un servidor en la Nube. Algunas plataformas de estas se apoyan en el proyecto FIWARE[121], [122], que es una plataforma *Open Source,* para la implementación de ambientes IoT, en la comunidad europea. Las fuentes de imágenes en este dominio son muy variadas, y se incluyen las de alta resolución, provenientes de satélite [118], o captura con drones [108]. Los métodos de clasificación se basan principalmente en redes neuronales como YOLOv3 y VGG16 [120], [127].

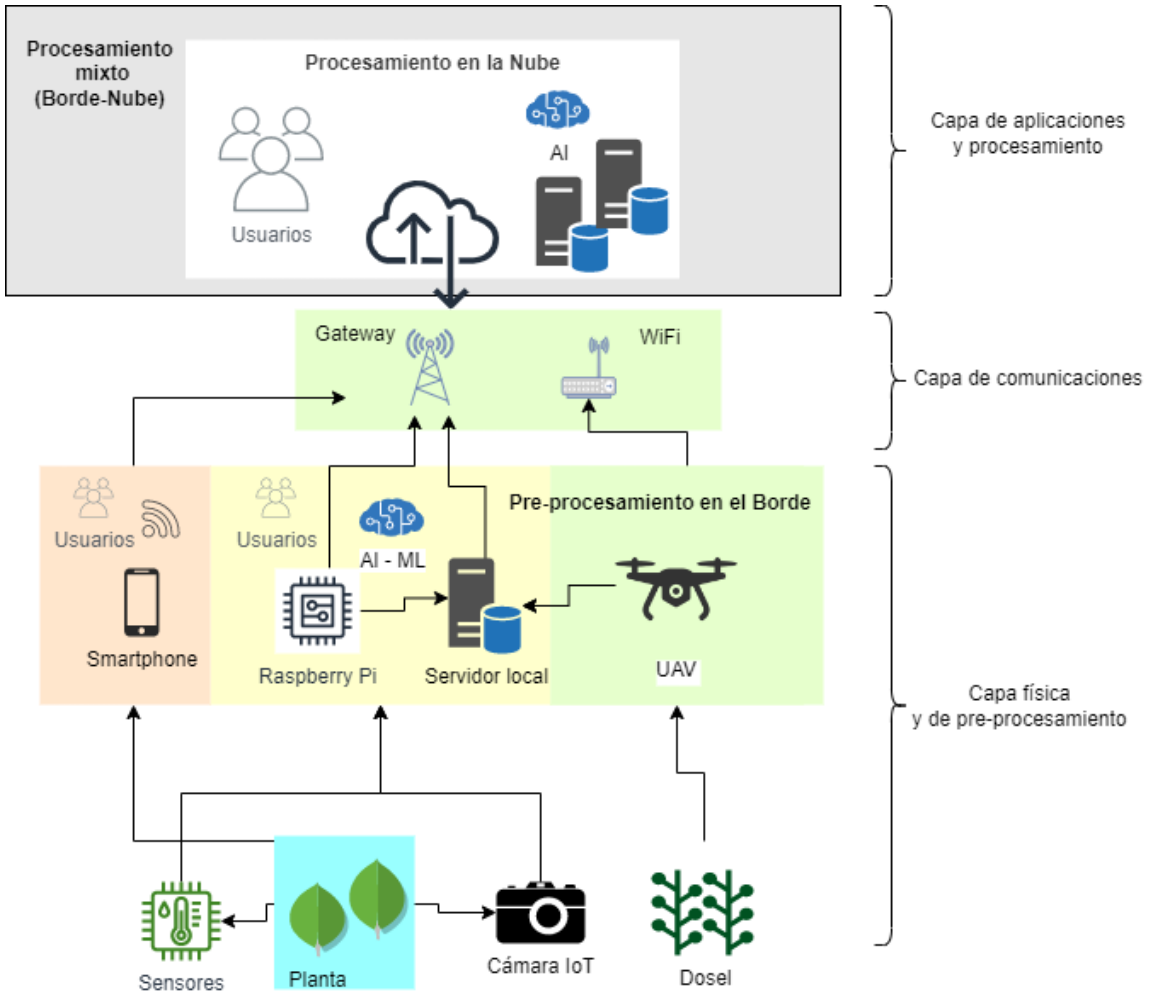
Cuando la clasificación de imágenes en las plataformas se aplica para la gestión del riego y protección de cultivos, esta es la segunda estrategia más usada, (29 %). Aquí se hace necesario el uso de procesamiento en la Nube por los objetivos que se proponen, como por ejemplo el establecimiento de redes de productores [128], o el uso de cámaras IP que no se apoyan en microprocesadores [130].

4.3 Proceso de clasificación mixto Borde-Nube

La tercera estrategia que se realiza en algunas plataformas es llevar a cabo la clasificación de imágenes de manera distribuida, usando capacidad de cómputo de diferentes partes de la plataforma IoT, en el Borde y en la Nube *(Edge Computing y Cloud Computing)* [83]. El esquema general en arquitecturas que emplean esta estrategia se muestra a continuación en la figura 12.

La evidencia encontrada en la RSL sugiere que esta es la estrategia menos empleada. El procedimiento general es que el pre-procesamiento de las imágenes se realiza en el Borde, aprovechando así la capacidad que pueden dar los dispositivos como microprocesadores o teléfonos móviles. Y la fase final de clasificación se realiza en la Nube, donde se dispone de mayor capacidad de cómputo para la aplicación de métodos, que en general se basan en redes neuronales artificiales.

Figura 12. Esquema arquitectónico general de plataformas que realizan el proceso de clasificación de imágenes mixto Borde-Nube



Fuente: elaboración propia.

Estas arquitecturas también tienen en la mayoría de los casos revisados, tres capas: capa física y de pre-procesamiento, capa de comunicaciones y capa de aplicaciones y procesamiento. Las aplicaciones en detección de problemas fitosanitarios también hacen un uso frecuente de esta estrategia [87], [98], [100], [104], [169]. Entre las técnicas de pre-procesamiento de imágenes que se realizan en el Borde fueron: el cambio de espacio de color y la disminución de ruido [98], [169]; la mejora del contraste, extracción de características [87], [100]. También se usa esta estrategia en aplicaciones de monitoreo del desarrollo y vigor de cultivos. Por ejemplo, la

plataforma llamada “*FarmBeats*” en la que los autores hacen uso combinado de la computación en la Nube y en el Borde [123].

4.4 Identificación de la arquitectura de referencia

La estrategia de realizar el proceso de clasificación en el Borde fue la elegida en este trabajo. Pues la hipótesis plantea que es posible implementar una plataforma de Agricultura inteligente que integre de manera efectiva un método de clasificación de imágenes digitales y una arquitectura IoT, en ambientes rurales con limitaciones de energía y comunicaciones. Debido a las limitaciones de energía y comunicaciones, para este trabajo resulta más confiable realizar todo el procesamiento en el Borde.

Con base en los hallazgos hasta aquí mostrados, dentro de los trabajos recuperados en la RSL, se seleccionaron inicialmente aquellos que mostraron arquitecturas de referencia con suficiente información para ser implementadas en contextos reales de trabajo. Luego, se hizo una comparación entre las arquitecturas preseleccionadas con base en los criterios de diseño propuestos por los autores en [45], quienes establecen que un marco de diseño para arquitecturas de referencia, en el dominio de la agricultura inteligente debe cumplir cinco principales desafíos, según [45]:

1. Aumento en la cantidad de datos, especialmente de todo tipo de equipos agrícolas.
2. interoperabilidad entre varios sistemas a nivel de plantación y en toda la red de la cadena de suministro que la rodea.
3. Estandarización de datos.
4. Ir más allá de la pequeña escala y enfocarse en la producción agrícola a nivel regional, y al mismo tiempo.
5. Cumplir con diferencias nacionales o regionales en las prácticas agrícolas.

Finalmente, la comparación entre arquitecturas se realizó con base en los primeros cuatro criterios mencionados, y no se tuvo en cuenta el quinto criterio, por la dificultad que involucra tener en cuenta todas las prácticas agrícolas de un país o una región (tabla 5).

Tabla 5. Arquitecturas de referencia comparadas para realizar la selección

Arquitectura / Referencia	Adaptable a Big Data	Interoperabilidad	Estandarización de datos	Escalabilidad
FARMIT [122]	x	x	✓	✓
DIAS [121]	✓	✓	✓	✓
Creative IoT Agriculture Platform [124]	x	x	✓	x
FarmBeats [123]	x	✓	✓	x
IoT Solution for Smart Agriculture [115]	x	x	✓	✓
PLATEM [128]	✓	x	✓	✓
AgriLoRa [19]	x	✓	✓	x
Remote Image Capture System[132]	x	✓	✓	x
LoraFarM [28]	✓	✓	✓	✓

Nota. Como puede observarse solamente dos arquitecturas: DIAS [121] y LoraFarM [28] cumplen con los primeros cuatro en criterios propuestos en [45]. Sin embargo, la arquitectura LoraFarM, cuenta con información más detallada sobre los dispositivos utilizados y presenta menos complejidad para su implementación. Estos dos últimos factores fueron decisivos para seleccionarla como la arquitectura de referencia que se usó en la implementación de la plataforma IoT.

Fuente: elaboración propia.

En la figura 13 a continuación se muestra la arquitectura de referencia seleccionada, la cual consta de tres capas: (1) capa de aplicaciones y servicios, (2) capa de *middleware* y (3) capa física y de comunicaciones.

Figura 13. Arquitectura de referencia LoRaFarm

Fuente: Elaboración propia con base en [28].

4.4.1 Capa de servicios y aplicaciones.

Esta capa cumple la función de interconectar a los usuarios con los datos de la plataforma, y en esta arquitectura no involucra procesos de analítica.

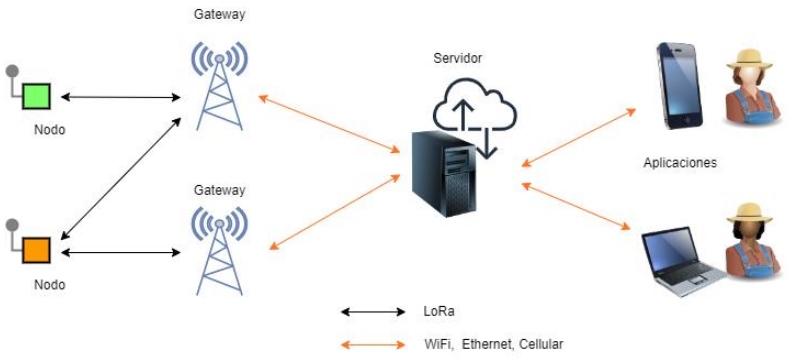
4.4.2 Capa de middleware.

En esta segunda capa, los autores en [28], proponen la integración de dos servidores: Servidor de Red (*Server Network*) y Servidor de Aplicaciones (*Application Server*), el primero recibe directamente los datos provenientes de la capa de nodos y comunicaciones, y los redirecciona a un servidor de aplicaciones que permite su posterior almacenamiento y visualización.

4.4.3 Capa de física y comunicaciones.

Esta capa integra los nodos y la tecnología de comunicaciones a través de una puerta de enlace (*gateway*), y se basa en el uso de la tecnología LoRa y LoRaWAN, además tiene una topología tipo estrella en la que los nodos están habilitados para comunicarse con varias puertas de enlace al mismo tiempo, como se describe a continuación en la figura 14.

Figura 14. Topología de implementación para la tecnología LoRa



Fuente: elaboración propia con base en [28].

Finalmente, Las ventajas de la arquitectura seleccionada además de las expuestas en la tabla comparativa, son las siguientes:

Fácil conectividad. Es una arquitectura que se basa en la tecnología LoRa, y el protocolo LoRaWAN que son de acceso libre y diseñados para trabajar en contextos de baja conectividad y escasos recursos de energía, como son los ambientes rurales.

Bajo costo. Los dispositivos IoT propuestos en la arquitectura son de bajo costo y fácil implementación en campo, así mismo LoRaWAN no requiere de la contratación de proveedores de servicios de comunicación, pues transmite en la banda ISM *(Industrial, Scientific and Medical*).

La principal desventaja de la arquitectura seleccionada es:

Capacidad de transmisión. Por estar basada en la Tecnología LoRa, la transmisión de datos se puede hacer a grandes distancias, pero con un limitado ancho de banda, que se disminuye en la medida que aumenta la distancia entre los nodos y la puerta de enlace.

4.5 Resumen del capítulo

En este capítulo se identificaron las tres principales estrategias arquitectónicas que emplean las plataformas de Agricultura Inteligente para integrar la clasificación de imágenes digitales: Procesamiento en el borde, procesamiento en la Nube y procesamiento mixto. También se identificaron las principales tecnologías usadas para implementar las estrategias. Con base en las arquitecturas analizadas, este capítulo permitió elegir la arquitectura de referencia: LoraFarm [28], dando cumplimiento así al primer objetivo específico de este trabajo.

En general, la evidencia encontrada en la literatura revisada sugiere que es necesario realizar más experimentos del comportamiento de estas arquitecturas en condiciones más parecidas a las encontradas en contextos rurales, como lo sugieren Terence y Purushothaman [8] en su revisión de 2020, con ancho de banda limitados, fuentes de energía limitadas, zonas remotas y condiciones de intemperie, entre otras posibles. Para poder tener una mejor comprensión de las mejoras que se deben realizar, para una correcta implementación en plataformas de Agricultura Inteligente.

En el próximo capítulo se propone un nuevo conjunto de datos para el entrenamiento del método propuesto. Como aporte adicional, durante el desarrollo de esta tesis se encontraron problemas

con el conjunto de datos original que debieron ser resueltos, y que dieron origen a un nuevo *dataset.*

5 Los datos

En el capítulo anterior se seleccionó la arquitectura de referencia, con base en un análisis de las diferentes estrategias arquitectónicas y tecnológicas detectadas en la RSL. Se describieron los diferentes elementos de cada capa, de acuerdo con las diferentes aplicaciones de las plataformas de Agricultura Inteligente.

En este capítulo se muestra un aporte adicional de esta tesis, que tiene que ver con los cambios realizados al *dataset* original seleccionado para el entrenamiento de los modelos. Esto debido a limitantes que presentó el *dataset* original y que no fueron detectados en trabajos previos. Como resultado de la realización de cambios en el *dataset,* está tesis propone un nuevo conjunto de datos.

Este capítulo está organizado de la siguiente forma: en la primera parte se presenta el caso de estudio seleccionado. En segundo lugar, se presenta el *dataset* seleccionado. Tercero, se presentan los problemas y los desafíos que presentó el *dataset.* Luego, en la cuarta sección se presentan los pasos para la creación del nuevo *dataset.* Posteriormente, se presenta el método usado para hacer el balanceo de datos en la quinta sección. Y finalmente en sexto lugar, se presenta una prueba que se realizó al *dataset* obtenido para constatar el poder predictivo de la textura.

5.1 Caso de estudio

En el capítulo 2 se eligió el caso de estudio de la clasificación de enfermedades en las plantas. De acuerdo con la evidencia obtenida en la RSL, la detección de enfermedades en las plantas es uno de los casos que más obtiene la atención de la comunidad investigativa. La detección de enfermedades en la agricultura es de vital importancia. La FAO estima que anualmente entre el 20 % y 40 % por ciento de la producción mundial de cultivos se pierde a causa de las plagas. Cada año, las enfermedades de las plantas le cuestan a la economía mundial alrededor de US $ 220 mil millones y los insectos invasores alrededor de US $ 70 mil millones [170].

La ciencia que se encarga del estudio de las enfermedades en las plantas es la fitopatología. Su estudio implica, conocer las causas, los patógenos responsables, su interacción con la

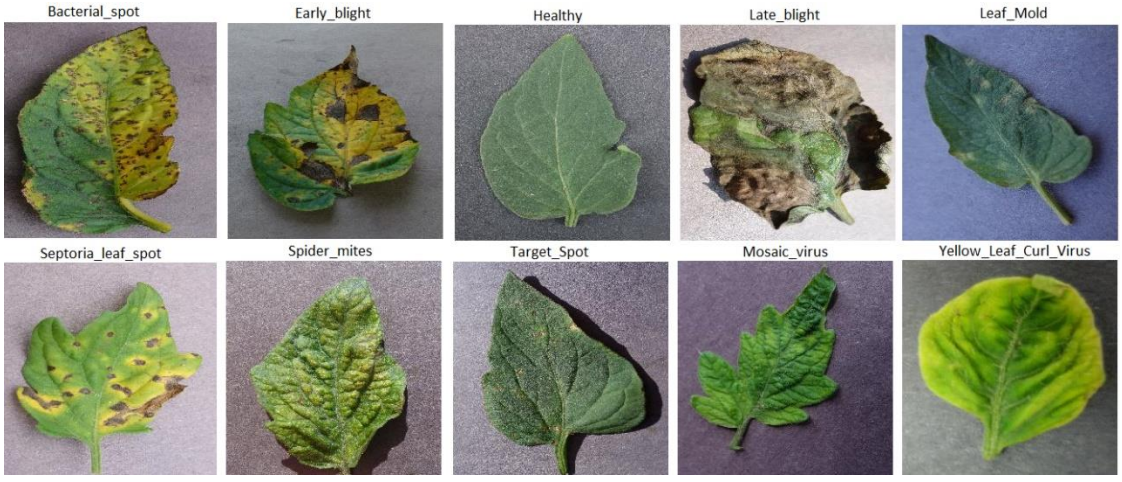
fisiología de las plantas y los métodos para controlarlos y reducir su impacto negativo [171]. Las enfermedades de las plantas son causadas por agentes infecciosos (factores bióticos como hongos, bacterias o virus) y agentes no infecciosos (factores abióticos como quemaduras solares, deficiencia de minerales, etc.) [164] Las enfermedades abióticas son menos peligrosas debido a su naturaleza de no transmisibilidad; por lo tanto, las enfermedades abióticas son en su mayoría evitables [171].

En este trabajo de investigación se utilizó el caso de la clasificación de enfermedades, debido a su alto impacto en agricultura. Solo se consideran enfermedades bióticas debido al impacto que tienen en la productividad de los cultivos y son las que más proliferan [170].

5.2 Conjunto de datos para entrenamiento (*dataset*)

Para el trabajo de investigación se seleccionó el conjunto de datos Plantvillage [46], porque contiene una gran variedad de imágenes correspondientes a diferentes tipos de enfermedades que se manifiestan principalmente en las hojas de las plantas. De este *dataset,* se seleccionó el cultivo de tomate por ser una de las hortalizas más importantes en la agricultura mundial, siendo el segundo cultivo más importante en el mundo después de la papa [172]. El tamaño de las imágenes es de 256 x 256 pixeles en RGB. Varios ejemplos de las diferentes clases de enfermedades del *dataset* pueden verse a continuación en la figura 15.

Figura 15. Ejemplos de cada clase en el *dataset* original PlantVillage



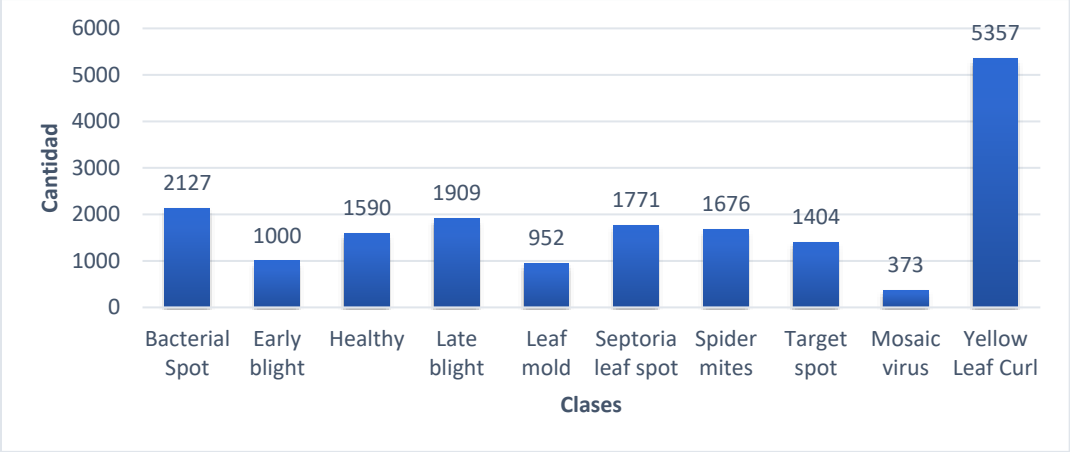
Fuente: [46]

5.3 Problemas y desafíos del *dataset* utilizado

PlantVillage contiene un conjunto de datos de imágenes de hojas de tomate, con 10 clases: clase sana y nueve tipos de enfermedades diferentes y en diferentes etapas de desarrollo. Sus imágenes fueron tomadas en ambientes controlados. Es uno de los *datasets* para enfermedades en cultivos más referenciados en la literatura. Sin embargo, de acuerdo con algunos autores presenta varios problemas que deben ser considerados.

1. Si bien el conjunto de datos conserva cierta homogeneidad en las condiciones de captura de la imagen como por ejemplo la distancia de la cámara a las hojas, algunas clases fueron recolectadas en lugares diferentes [46]. En el trabajo original de Mohanty y otros autores en el año 2016, no se indica si las variedades de tomate son la misma o distintas en cada clase, y por lo tanto existe incertidumbre al respecto.
2. Algunas características morfológicas de las hojas podrían obedecer a aspectos fenotípicos y genotípicos de la variedad, y no necesariamente sea una consecuencia de la enfermedad. Los autores en [47] detectaron, que la forma de las hojas de cada clase tiene un poder predictivo importante, para lo cual realizaron una clasificación de la silueta de las hojas, obteniendo un *Accuracy* de 52,3 %, con base en el modelo ResNet-50.
3. El conjunto de datos fue elaborado con imágenes capturadas en diferentes condiciones de iluminación y fondo para cada una de las clases, lo que trae como consecuencia un sesgo en la clasificación. Esto fue detectado por los mismos autores en [47], que realizaron una clasificación de los fondos de las imágenes, obteniendo un *Accuracy* de 62,3 %, por lo cual sugieren hacer una segmentación y eliminar el fondo antes de realizar algún procedimiento de clasificación.
4. El conjunto de datos difiere en la cantidad de imágenes para cada clase de manera significativa en algunas de ellas. Se puede observar un desbalanceo de las clases principalmente entre la *Mosaic virus* y *Yellow Leaf Curl,* como puede verse a continuación en la figura 16.

Figura 16. Distribución de la cantidad de imágenes en el dataset original PlantVillage



Fuente: elaboración propia.

5.3.1 Efecto de la forma de las hojas en la clasificación de enfermedades.

Como ya se mencionó anteriormente, las imágenes del *dataset* utilizado en esta investigación, pueden presentar un sesgo debido a la forma de las hojas. Aunque en la literatura se reporta este sesgo [47], en este trabajo se realizó una prueba para medir su magnitud.

1. Se realizó una segmentación binara para obtener la silueta de las hojas.
2. Se realizó una aumentación de datos para eliminar el efecto negativo por el desbalanceo.
3. Con las imágenes obtenidas se realizó un entrenamiento del modelo ResNet-50, que fue el utilizado por los autores en [47], para medir el sesgo debido a la forma de las hojas.
4. Los hiperparámetros utilizados para el entrenamiento del modelo fueron: Learning rate: 0.00234, optimizador: SGD (*Stochastic Gradient Descent*), epochs = 20, Loss=categorical crossentropy.

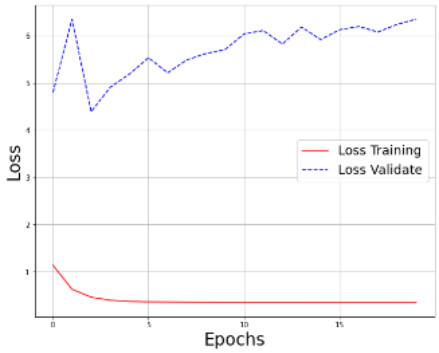
5.3.2 Resultados de la prueba de sesgo debido a la forma de las hojas.

En la figura 17 pueden observarse los resultados del entrenamiento y predicción con el conjunto de datos de prueba conformado solamente por las siluetas de las hojas, en los que se observa un *Accuracy* de 49**,**28 %, muy cercano al resultado reportado en [47] de 52,3 %. Incluso puede verse como para algunas clases tiene un *Recall* (proporción de instancias pertenecientes a la clase correctamente clasificadas*)* considerablemente alto, por ejemplo 91,33 % en la clase

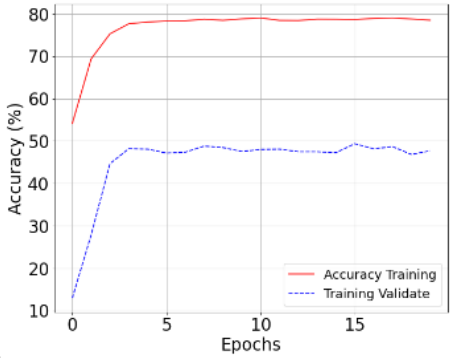
Early_Blight, 88 % para *Spider_Mites*, y 87,55 % en la clase *Late_Blight* por mencionar algunos. Esto confirma lo reportado por la literatura y ratifica la importancia de aislar la forma de las hojas en el proceso de clasificación de las enfermedades si se desconocen las variedades de las plantas.

Figura 17. Resultados del entrenamiento del modelo ResNet-50 con el set de datos de las siluetas de las hojas

a.



b.



c.

Class	precision	recall	f1-score	support
Bacterial_spot	0.4504	0.8683	0.5931	319
Early_blight	0.5502	0.9133	0.6867	150
Healthy	0.4959	0.7605	0.6003	238
Late_blight	0.5085	0.8755	0.6433	273
Leaf_Mold	0.6805	0.8042	0.7372	143
Septoria_spot	0.4564	0.4135	0.4339	266
Spider_mites	0.4536	0.8800	0.5986	250
Target_Spot	0.0000	0.0000	0.0000	211
Yellow_Leaf_Virus	0.4000	0.0250	0.0470	801
Mosaic_virus	0.6731	0.6250	0.6481	56
accuracy			0.4928	2707
macro avg	0.4669	0.6165	0.4988	2707
weighted avg	0.4334	0.4928	0.3898	2707

d.

Nota. a) *Train Loss*, b) *train Accuracy,* c) *Precision, Recall & F1-Score*, d) ejemplo de las imágenes segmentadas para las siluetas utilizadas en el entrenamiento.

Fuente: elaboración propia.

5.4 Creación de un nuevo *dataset*

Para crear un nuevo *datset* que superara las dificultades mostradas en la sección anterior se realizaron las operaciones de pre-procesamiento tendientes a superar los problemas planteados y crear un nuevo *dataset.*

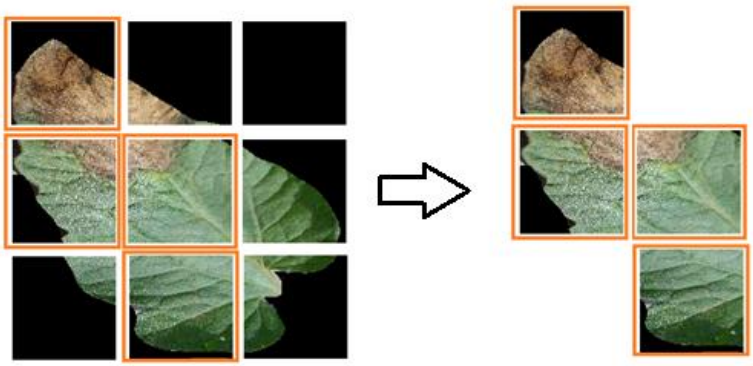
1. En primer lugar, se realizó una segmentación de las imágenes con un clasificador de Bayes como se explicó en las secciones 2.3.2 y 2.3.3, tendiente a separar el fondo del objeto (en este caso las hojas), con el fin de eliminar cualquier sesgo en la clasificación debido al color o la textura propias del material o la iluminación del fondo (figura 18).

Figura 18. Ejemplo de segmentación de las imágenes originales

Fuente: elaboración propia.

2. En segundo lugar, se realizó un recorte de las imágenes que busca independizar las características de textura y color de la enfermedad de la forma de las hojas, como se explicó en la sección 2.3.4 (figura 19).

Figura 19. Recorte de imágenes en recuadros de 85 x 85 pixeles y selección de recuadros



Fuente: elaboración propia.

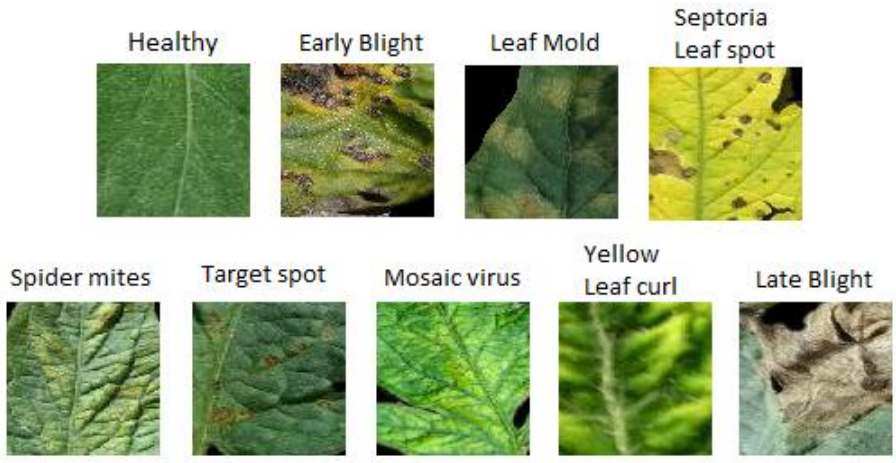
5.4.1 Estrategia basada en la textura para clasificar enfermedades en plantas.

Concentrarse en las características de textura para detectar enfermedades en las plantas es un enfoque que algunos autores ya han propuesto en diversos trabajos, y que sigue siendo un

campo abierto de investigación [173]–[175]. Esta idea, se propone en este trabajo para superar la incertidumbre que hay debido al origen de las imágenes en sus diferentes clases, ya que algunas de ellas fueron obtenidas en países y/o regiones distintas, y en el trabajo de Mohanty y otros autores [46], donde se propone el uso del set de datos PlantVillage, no se aclara si hay diferentes variedades de tomate en cada una de las clases.

Por lo tanto, luego de realizar el preprocesamiento a las imágenes y obtener el nuevo dataset, se realizó un balanceo de los datos, y se procedió a realizar un análisis previo del poder predictivo de las características de textura del nuevo *dataset*, como se muestra en las secciones siguientes. A continuación, en la figura 20 se presenta un ejemplo de las diferentes clases del nuevo *dataset.*

Figura 20. Ejemplo de recuadros del nuevo *dataset* modificado por cada clase



Fuente: elaboración propia.

5.5 Balanceo de datos.

Una vez creado el nuevo conjunto de datos, se pudo observar un desbalanceo de clases similar al desbalanceo de las imágenes originales, por lo que se realizó, un balanceo aplicando el método de aumentación de datos (*data augmentation*) para las primeras siete clases y una reducción de datos aleatoria para la clase *Yellow Leaf Curl*, obteniendo los resultados un total de imágenes de 23.215. Las técnicas de aumentación de datos aplicadas fueron: *Horizontal flip, Vertical flip y Random rotation* (-55°, 55°) (figura 21).

Figura 21. Ejemplo de aumentación de datos en el conjunto de datos de entrenamiento.

Fuente: elaboración propia.

Las cantidades obtenidas después de aplicar la aumentación de datos estuvieron entre 2531 (mínimo) y 2670 (máximo), como puede verse en la figura 22.

Figura 22. Cantidad de imágenes resultantes luego del balanceo de clases

Fuente: elaboración propia.

Luego el conjunto de datos fue dividido en tres grupos: entrenamiento, validación y prueba. Aunque no existe una regla estándar para esta subdivisión se realizó siguiendo como ejemplo trabajos similares recuperados en la RSL, que muestran divisiones generalmente entre 70 % - 80 % de datos para entrenamiento y 20 % - 30 % para los datos de validación y prueba (tabla 6).

Tabla 6. Cantidad de imágenes del nuevo dataset en los subconjuntos de entrenamiento, validación y prueba.

Clase	Entrenamiento	Validación	Prueba	Total
Early_blight	2031	543	207	2781
Late_blight	2016	517	291	2824
Leaf_Mold	2088	531	131	2750
Septoria_leaf_spot	2014	561	349	2924
Spider_mite	2015	533	276	2824
Target_Spot	2084	540	218	2842
Mosaic_virus	2042	502	49	2593
Yellow_Leaf_Curl	2167	503	1021	3691
Healthy	2055	559	305	2919
Total	**18512**	**4789**	**2847**	**26148**
Percentaje	71%	18%	11%	100%

Fuente: elaboración propia.

5.6 Poder predictivo de las características de textura

Luego del balanceo de datos, y con el fin de realizar una prueba con el nuevo *dataset* y obtener un grado de interpretabilidad para los métodos de clasificación usados en este trabajo, se buscó conocer el poder predictivo de las características de textura en el caso de las enfermedades que afectan las plantas. Se utilizó el método GLCM (*Gray Level Co-occurrence Matrix*). GLCM es un método estadístico para extraer las características de textura que considera la relación espacial de los pixeles. Mediante este método se construye una matriz de coocurrencia de nivel de gris (GLCM), también conocida como matriz de dependencia espacial de nivel de gris [182].

Las funciones GLCM caracterizan la textura de una imagen calculando con qué frecuencia se producen en una imagen pares de píxeles con valores específicos y en una relación espacial específica, creando una GLCM y luego extrayendo medidas estadísticas de esta matriz. Las funciones del método GLCM no proporcionan información sobre la forma de los objetos, es

decir, las relaciones espaciales de los pixeles. Las características de textura extraídas fueron: *Homogeneity, Contrast, Correlation & Dissimilarity* [176].

$$Homogeneity = \sum_i \sum_j \frac{C_{ij}}{1+|i-j|} \textbf{ (21)}$$

$$Contrast = \sum_i \sum_j (i-j)^2 C_{ij} \textbf{ (22)}$$

$$Correlation = \sum_i \sum_j \frac{(i-\mu_i)(j-\mu_j)C_{ij}}{\sigma_i \sigma_j} \textbf{ (23)}$$

$$Dissimilarity = \sum_i \sum_j C_{ij}|i-j| \textbf{ (24)}$$

Donde:

i = Número de fila.

j = Número de columna.

μ = Media de la GLCM (siendo esta un estimado de la intensidad de todos los pixeles que contribuyen a la GLCM.

σ = Varianza de las intensidades de los pixeles que contribuyen a la GLCM.

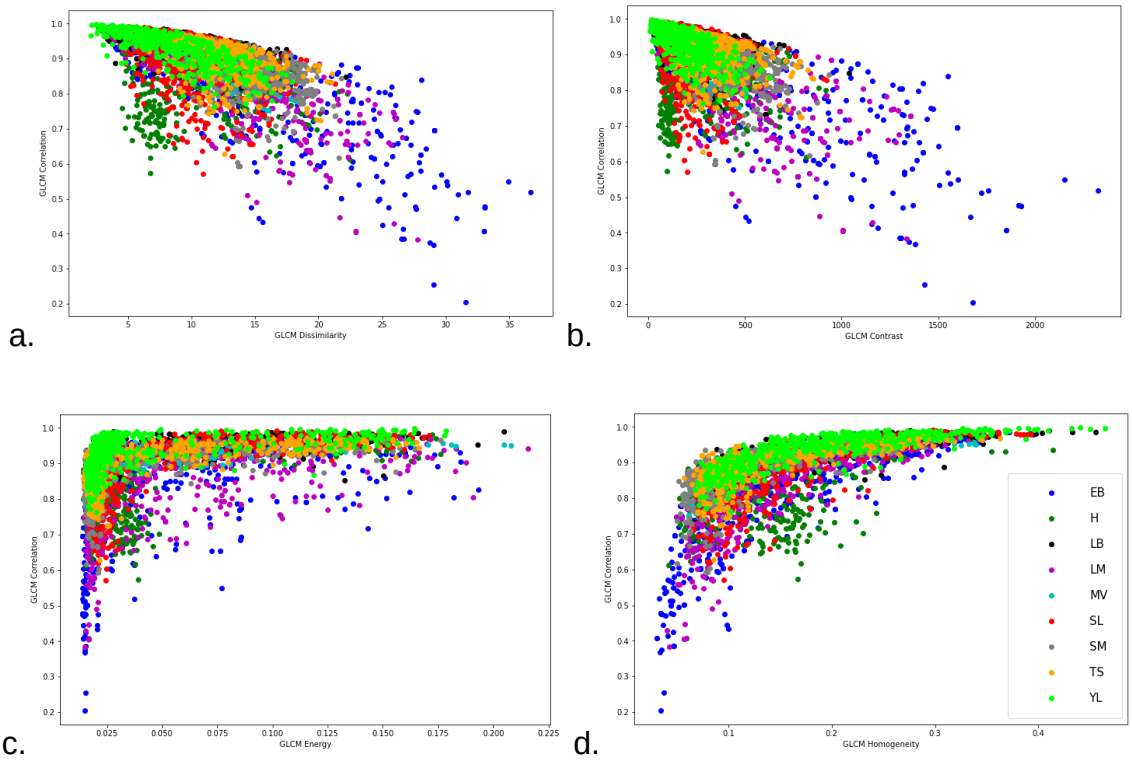
C = Elemento ij de la GLCM normalizada.

El contraste (*Contrast*) es una medida de las variaciones del nivel de gris entre los píxeles de referencia y sus vecinos. La correlación (*Correlation*) es indicativa de la dependencia lineal de niveles de gris en la GLCM. La homogeneidad (*Homogeneity*) suele estar inversamente relacionada con el contraste y es una indicación de la similitud de los elementos fuera de la diagonal en la GLCM. Y la disimilitud ($Dissimilarity$) es una medida de la distancia entre pares de píxeles en la región de interés [177].

El análisis se realizó con la librería Scikit-Image de Python, con las que se buscó la combinación de distancia y ángulo que mejor separara los grupos, para dos dimensiones, se realizaron combinaciones por pares de las características extraídas vs. la correlación. Los resultados de

este análisis se muestran en la figura 23 donde puede observarse que existen agrupamientos, debido a una alta correlación, entre los elementos de una misma clase, aunque también existe correlación entre las clases, lo que representa un reto de alta complejidad si se quieren separar las clases con base solo en estas combinaciones de características.

Figura 23. Análisis gráfico de las características de textura extraídas con GLMC



Nota. Correlation *vs.* a) Dissimilarity, b) Contrast, c) Energy, d) Homogeneity.

EB: Early Blight, **H:** Healthy, **LB:** Late Blight, **LM:** Leaf Mold, **MV:** Mosaic virus, **SL:** Septoria Leaf, **SM:** Spider mite, **TS:** Target spot, **YL:** Yellow leaf.

Fuente: elaboración propia.

5.7 Resumen del capítulo

En este capítulo se presentó como a partir de un conjunto de datos seleccionado (PlantVillage), se obtuvo un nuevo *dataset*. Los motivos para la obtención del *dataset* tienen que ver con el sesgo que presenta el conjunto de datos original, debido a dos factores: el color del fondo utilizado para la captura de las imágenes y la forma de las hojas que tienen poder predictivo.

También se realizó un balanceo de datos y una prueba inicial del *dataset* con el método GLCM para verificar el poder predictivo de las imágenes. La obtención de este nuevo *dataset*, no estaba prevista al inicio de este trabajo, por lo que representa un aporte adicional de esta tesis, pues el nuevo dataset podrá ser utilizado en trabajos futuros para la clasificación de enfermedades.

En el próximo capítulo se presenta el proceso de selección del método de clasificación de imágenes, para darle respuesta al objetivo específico número dos de esta tesis doctoral.

6 Método de clasificación de imágenes

En el capítulo anterior se mostró el proceso de creación de un nuevo conjunto de datos con base en el *dataset* PlantVillage. El *dataset* que resultó de este proceso fue utilizado en el entrenamiento de los modelos pre-seleccionados en este capítulo para el proceso de selección y con base en este so obtuvieron las métricas elegidas para su selección.

En este capítulo se muestra todo el proceso realizado para la selección del método de clasificación de imágenes. La característica principal, y que fue el punto de partida para la selección, fue el tamaño del modelo. Con este propósito se preseleccionaron métodos que en la comunidad del *Machine Learning,* que fueron creados para trabajar en ambientes muy limitados en capacidad de cómputo, como por ejemplo los dispositivos móviles.

Este capítulo está organizado de la siguiente manera: Primero se presentan algunos resultados de trabajos previos con el mismo *dataset* que utilizaron métodos de *Machine Learning* clásicos. En segundo lugar, se presentan trabajos previos que utilizaron métodos de *Deep Learning.* Tercero, se describen los métodos preseleccionados. Luego, en la cuarta sección se describe el método usado para la selección de los hiperparámetros. Quinto, se presentan los resultados del entrenamiento. Posteriormente en la sexta sección se muestran los resultados obtenidos con los datos de prueba. Y finalmente se presenta un análisis de los resultados y la elección del método.

6.1 Métodos de *Machine Learning* utilizados en trabajos previos

Los autores en [178] realizaron una comparación del desempeño entre métodos de *Machine Learning* clásicos y Redes Neuronales Convolucionales (CNN) con el conjunto de datos PlantVillage para tomate. Para la implementación de los algoritmos de *Machine Learning* clásicos los autores utilizaron diferentes métodos para extraer manualmente las características de cada enfermedad. En su estudio los autores extrajeron un total de cincuenta y dos características de textura usando los métodos *Local Binary Pattern* (LBP) y *Gray Local Co-occurrence Matrix* (GLCM). Además, extrajeron ciento cinco características de color usando los métodos de momento de color e histograma de color. Entre todos los métodos de extracción de características, el método COLOR+GLCM obtuvo el mejor resultado. Luego los mismos autores

probaron las redes neuronales convolucionales AlexNet, VGG16, ResNet34, EfficientNet-b0 y MobileNetV2, encontrando un mejor desempeño con los métodos basados en redes neuronales. El resumen de los resultados puede verse a continuación en la tabla 7.

Tabla 7. Resultados de desempeño comparados entre métodos clásicos de ML y algunas Redes Neuronales Convolucionales obtenidos

Métodos	***Accuracy***	***Precision***	***Recall***	***F1-score***
k-NN	82,1 %	82,1 %	82,5 %	82,0 %
SVM	91,0 %	90,9 %	91,0 %	91,0 %
Random forest	82,7 %	82,6 %	83,0 %	82,7 %
AlexNet	92,7 %	92,5 %	92,6 %	92,4 %
VGG16	98,9 %	98,6 %	98,5 %	98,5 %
ResNet34	99,7 %	99,6 %	99,7 %	99,7 %
EfficientNet-b0	98,9 %	98,9 %	98,9 %	98,8 %
MobileNetV2	91,2 %	91,3 %	91,3 %	91,2 %

Fuente: [178].

Por su parte, los autores en [179] propusieron un modelo de CNN simplificado que consta de 8 capas ocultas. Usando también el conjunto de datos PlantVillage para tomate, el modelo propuesto por los autores tuvo un mejor desempeño que los métodos clásicos de *Machine Learning* probados, logrando una precisión del 98,4 %. Los métodos entrenados por los autores fueron SVM (*Support Vector Machine*), árboles de decisión (*decision tree*), Regresión lineal (*lineal regression*), k-NN (*k-Nearest Neighbors*) y *Naïve Bayes*. Los métodos usados para extraer las características manualmente fueron: Los momentos de Hu (*Hu-moments),* método de Haralik (*Haralick features)*, *LBP*, y características de color *HSV.* El resumen de los resultados de este trabajo se puede ver a continuación en la tabla 8.

Tabla 8. Resultados de desempeño comparados entre métodos clásicos de ML

Características	**Método**	**Accuracy**	**Precision**	**Recall**	**F1-sc**
Haralick + Hu + HSV	SVM	72,0%	99,0%	26,2%	41,4%
	DecisionTree	73,4%	94,0%	26,6%	41,4%
	LR	89,7%	99,0%	49,3%	65,8%

	k-NN	94,1%	100,0%	62,9%	77,2%
	Naive Bayes	30,4%	1,0%	0,2%	0,3%
Haralick + Hu + LBP	SVM	72,8%	96,0%	26,4%	41,4%
	Decision Tree	75,1%	90,0%	27,4%	42,0%
	LR	89,7%	100,0%	49,3%	66,0%
	k-NN	94,9%	100,0%	66,2%	79,7%
	Naive Bayes	30,4%	1,0%	0,2%	0,3%
Haralick	SVM	37,7%	84,0%	12,2%	21,2%
	Decision Tree	25,5%	79,0%	9,8%	17,5%
	LR	35,3%	86,0%	12,0%	21,0%
	k-NN	39,4%	87,0%	12,8%	22,3%
	Naive Bayes	30,0%	100,0%	12,5%	22,2%
Hu	SVM	68,2%	89,0%	22,5%	35,9%
	Decision Tree	70,2%	43,0%	15,1%	22,4%
	LR	88,3%	98,0%	46,0%	62,6%
	k-NN	93,7%	100,0%	61,3%	76,0%
	Naive Bayes	39,4%	91,0%	13,2%	23,1%
	CNN	98,4%	100,0%	86,2%	92,6%

Fuente: [179]

Con base en los resultados de los trabajos en [178] y [179], parece existir un mejor desempeño de los métodos basados en redes neuronales convolucionales, al realizar clasificación con el conjunto de datos PantVillage de tomate. Por lo tanto, en esta investigación se decidió realizar la búsqueda de los métodos candidatos en trabajos realizados solamente con redes neuronales artificiales.

6.2 Métodos de *Deep Learning* utilizados en trabajos previos

En años recientes diversos trabajos han intentado realizar la clasificación de las clases del conjunto de datos PlantVillage con métodos modernos. Como puede observarse en la tabla 9, varios autores han probado el desempeño de diferentes arquitecturas de redes neuronales con este *dataset,* en algunos casos se utilizó completo y en otros solo una parte.

Tabla 9. Métodos basados en redes neuronales, empleados para la clasificación de enfermedades con el dataset Plantvillage.

Referencia /	Arquitectura	# de Imágenes	Clases	*Accuracy*	Parámetros
[180] **/ 2018**	SqueezeNet	1,400	6	86,92 %	Sin datos
[181] **/ 2019**	ResNet_101	6,888	7	98,80 %	Sin datos
[182] **/ 2020**	ToleD	17,500	10	91,20 %	208,802
	InceptionV3	17,500	10	63,40 %	41,247,146
	VGG 16	17,500	10	77,20 %	9,449,482
	MobilNet	17,500	10	63,75 %	20,507,146
[183] **/ 2020**	SimpleCNN model	22,930	10	91,82 %	3,715,754
[184] **/ 2020**	DensNet_Xception	13,112	18	97,10 %	29,200,000
	Xception	13,112	18	93,17 %	22,800,000
	ResNet 50	13,112	18	86,56 %	25,500,000
	MobilNet	13,112	18	80,11 %	4,200,000
	ShuffleNet	13,112	18	83,68 %	920,000
[185] **/ 2020**	ResNet 50	3,000	3	98 %	Sin datos
[186] **/ 2020**	LeafNet	54,306	38	79,61 %	324,000
	VGG-16	54,306	38	81,89 %	138,000,000
	Inception ResNet v2	54,306	38	90,91 %	54,300,000
	Reduced MobileNet	54,306	38	92,78 %	500,000
	Modified MobileNet	54,306	38	92,97 %	500,000
	ResNet-50	54,306	38	94,23 %	23,600,000
	AlexNet 60	54,306	38	95,78 %	60,000,000
	DenseNet-121	54,306	38	95,80 %	7,100,000
	MobileNet	54,306	38	96,32 %	3,200,000
	AgroAVNET	54,306	38	96,49 %	238,000,000
	Xception	54,306	38	97,98 %	22,800,000
[187] **/ 2021**	GoogLeNet	10,735	4	99,20 %	Sin datos
	VGG 16	10,735	4	98,00 %	Sin datos
[188] **/ 2021**	VGG 16	11,000	10	87,00 %	Sin datos
	VGG 19	11,000	10	87,00 %	Sin datos
	Inception v3	11,000	10	90,00 %	Sin datos
[189] **/ 2021**	VGG 16	609	5	89,50 %	Sin datos
	VGG 19	609	5	87,40 %	Sin datos
	Inception v3	609	5	89,00 %	Sin datos
	Squeezenet-SVM	609	5	87,00 %	Sin datos
	Squeezenet-SGD	609	5	86,50 %	Sin datos
	Squeezenet-RF	609	5	76,80 %	Sin datos
[190] **/ 2021**	MobileNet	87,900	38	98,89 %	1,900,000

Nota: resaltados en color gris los métodos preseleccionados.

Fuente: elaboración propia.

Criterios de pre-selección de métodos de *Deep Learning*. La pre-selección de los métodos candidatos se realizó con base en la anterior revisión del estado del arte. Se tuvieron en cuenta dos criterios de selección para este trabajo:

1. **Tamaño.** Arquitecturas por debajo de cinco millones de parámetros fueron tenidas en cuenta para la selección. Este número de parámetros fue elegido como umbral, debido a que las redes del estado del arte diseñadas para trabajar con restricciones de capacidad de cómputo están en general por debajo de este valor [60] [61], [62], [63], [64], [65].
2. **Desempeño.** Desempeños iguales o superiores al 90 % en la métrica *Accuracy.* Como pueden observarse resaltadas en la tabla 11 están las arquitecturas que cumplieron con las condiciones de análisis (Toled, SimpleCNN Model, Reduced MobileNet, Modified MobilNet, y MovileNet*)*. De las redes Toled y SimpleCNN no fue posible conocer sus arquitecturas de manera completa, de igual manera para las redes Reduced y Modified MobileNet*,* no fue posible conocer todos los detalles para que fueran reproducibles. Sin embargo, las tres versiones de las redes MobileNet fueron elegidas para realizar los entrenamientos del modelo propuesto.
3. **Otras arquitecturas del estado del arte.** Fueron seleccionados además modelos que tienen características similares: EfficientNet, NasNetMobile, SqueezeNet y ShuffleNet que han sido diseñadas para trabajar en dispositivos con limitaciones de capacidad de cómputo, pero que no fueron selecionados a partir de los estudios previos revisados en la tabla 7. Finalmente, las redes neuronales convolucionales preseleccionadas fueron:

    - MobileNet [60].
    - MobileNetV2 [191]
    - MobileNetV3 [61]
    - EfficientNetB0 [62]
    - NasNetMobile [63]
    - SqueezeNet [64]
    - ShuffleNet [65].

6.3 Métodos pre-seleccionados

A continuación, se hace un resumen de las arquitecturas de los métodos de *Deep Learning* seleccionados para la clasificación de imágenes digitales. En este resumen no se pretende hacer una explicación rigurosa de cada uno, para esto el lector puede consultar las referencias originales donde se pueden encontrar descripciones más detalladas. El objetivo es mostrar

principalmente la composición de sus arquitecturas y algunas diferencias significativas entre ellas.

6.3.1 MobileNet.

Utiliza el concepto de *Depth Separable Convolution* (DSC), que básicamente consiste en factorizar el *kernel* de convolución estándar en una convolución profunda (*depthwise convolution*) y una convolución de 1x1 llamada *pointwise convolution* [60]. Su arquitectura puede verse en la tabla 10. La primera capa de la red es la única con convolución estándar, y todas las capas son seguidas de *Batch Normalization* para regularización y la función de activación ReLU *(Rectified Linear Unit)*.

Tabla 10. Arquitectura del modelo MobileNet

Type / Stride	Filter Shape	Input Size
Conv / s2	3 × 3 × 3 × 32	224 × 224 × 3
Conv dw / s1	3 × 3 × 32 dw	112 × 112 × 32
Conv / s1	1 × 1 × 32 × 64	112 × 112 × 32
Conv dw / s2	3 × 3 × 64 dw	112 × 112 × 64
Conv / s1	1 × 1 × 64 × 128	56 × 56 × 64
Conv dw / s1	3 × 3 × 128 dw	56 × 56 × 128
Conv / s1	1 × 1 × 128 × 128	56 × 56 × 128
Conv dw / s2	3 × 3 × 128 dw	56 × 56 × 128
Conv / s1	1 × 1 × 128 × 256	28 × 28 × 128
Conv dw / s1	3 × 3 × 256 dw	28 × 28 × 256
Conv / s1	1 × 1 × 256 × 256	28 × 28 × 256
Conv dw / s2	3 × 3 × 256 dw	28 × 28 × 256
Conv / s1	1 × 1 × 256 × 512	14 × 14 × 256
5× Conv dw / s1 **Conv / s1**	3 × 3 × 512 dw 1 × 1 × 512 × 512	14 × 14 × 512 14 × 14 × 512
Conv dw / s2	3 × 3 × 512 dw	14 × 14 × 512
Conv / s1	1 × 1 × 512 × 1024	7 × 7 × 512
Conv dw / s2	3 × 3 × 1024 dw	7 × 7 × 1024
Conv / s1	1 × 1 × 1024 × 1024	7 × 7 × 1024
Avg Pool / s1	Pool 7 × 7	7 × 7 × 1024
FC / s1	1024 × 1000	1 × 1 × 1024
Softmax / s1	Classifier	1 × 1 × 1000

Fuente: [60].

6.3.2 MobileNetV2.

Esta red intenta mejorar la reducción de operaciones sin perder precisión en comparación con la primera versión de MobileNet, basándose en dos estrategias principales *Inverted Residuals* y *Linear Bottlenecks* [191]. *Inverted Residuals*, son bloques residuales como los propuestos en el modelo ResNet. La operación original de bloques residuales de ResNet, le permite a la red acceder a activaciones anteriores que no fueron modificadas en el bloque convolucional, y permite por lo tanto construir redes muy profundas. Además, siguen un patrón ancho - estrecho - ancho con respecto a la cantidad de canales. Por el contrario, en el modelo MovilNetV2 se propone una conexión residual invertida (*Inverted Residuals*), que en vez de seguir el patrón mencionado sigue el patrón estrecho – ancho – estrecho con respecto a la cantidad de canales [191]. Su arquitectura puede verse en la tabla 11.

Tabla 11. Arquitectura MobileNet V2 *

Input	Operator	t	C	N	S
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	K	-	

Nota.Cada línea describe una secuencia de 1 o más capas idénticas, repetidas **n** veces. Todas las capas en la misma secuencia tienen el mismo número **c** de canales de salida. La primera capa de cada secuencia tiene un *stride* **s** y todos los demás usan *stride* 1. Todas las convoluciones utilizan *kernels* de 3 × 3. El factor **t** (expansión) siempre se aplica al tamaño del *input size.*

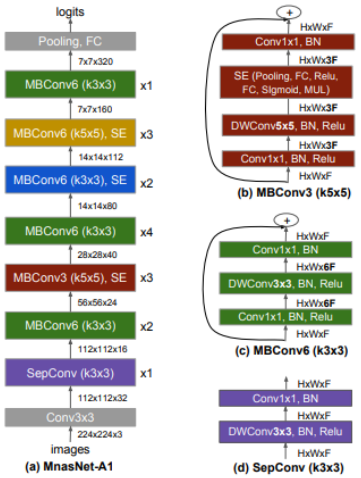
Fuente*:* [191].

Con los bloques residuales invertidos se reduce el número de parámetros, pero se comprimen las capas donde se vinculan las conexiones residuales. Esto perjudica el rendimiento de la red. Los autores de MobileNetV2 introdujeron la idea de un cuello de botella lineal (*Linear Bottlenecks)* [191], donde la última convolución de un bloque residual tiene una salida lineal antes de agregarse a las activaciones iniciales, y esto mitiga el efecto de pérdida de desempeño introducido por los bloques residuales invertidos (*Inverted Residuals)*.

6.3.3 NasNetMobile.

La estrategia utilizada por los autores de este método propone un enfoque de búsqueda de arquitectura neuronal automatizada para diseñar modelos de redes neuronales eficientes en recursos utilizando el aprendizaje por refuerzo [63]. Las ideas principales son incorporar información de latencia del mundo real y utilizar un novedoso espacio de búsqueda jerárquico factorizado para buscar modelos móviles con las mejores compensaciones entre precisión y latencia [63]. Su arquitectura se describe a continuación en la figura 24.

Figura 24. Arquitectura obtenida mediante el enfoque de NasNetMobile



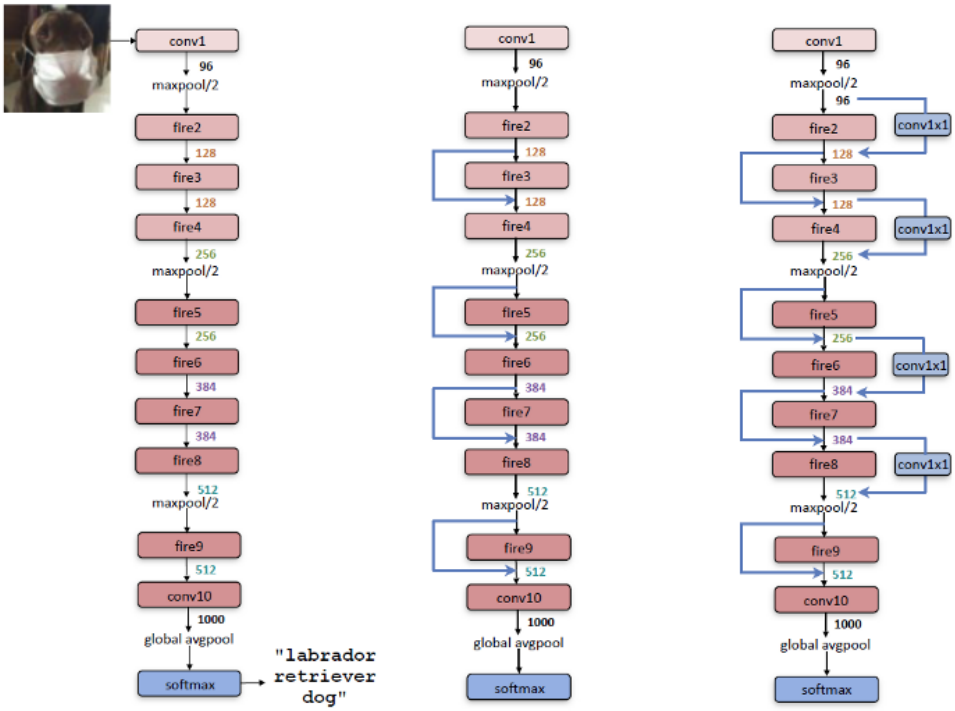
Nota. (a) es un modelo representativo obtenido por los autores en el artículo original, (b) - (d) son algunas estructuras de capas correspondientes. *MBConv* denota convolución de cuello de botella invertido móvil, *DWConv* denota *Depthwise convolution,* k3x3/k5x5 denota el tamaño de *kernel,* BN es *Batch Normalization,* HxWxF denota la forma de tensor (alto, ancho, profundidad) y × 1 / 2 / 3 / 4 denota el número de repeticiones de capas dentro del bloque.

Fuente: [63].

6.3.4 SquezzeNet.

Este modelo se basa en tres estrategias para construir una red con muy pocos parámetros sin perder precisión. Estrategia 1: Reemplaza los filtros 3×3 con filtros 1×1, ya que un filtro 1×1 tiene 9× menos parámetros que un filtro 3×3. Estrategia 2: Disminuir el número de canales de entrada a filtros 3×3, lo cual disminuye la cantidad de canales de entrada a filtros de 3×3 usando capas comprimidas. Estrategia 3. Reducir la muestra al final de la red para que las capas de convolución tengan mapas de activación grandes, la intuición detrás de esta última estrategia es que los mapas de activación grandes (debido a la reducción de muestreo retrasada) pueden conducir a una mayor precisión de clasificación. Con base en estas tres estrategias se conforma el *Fire Module, que se* compone de: una capa de convolución comprimida (que tiene solo filtros de 1×1), que alimenta una capa de expansión que tiene una combinación de filtros de convolución de 1×1 y 3×3 [64]. Su arquitectura se describe a continuación en la figura 25.

Figura 25. Modelo SqueezeNet

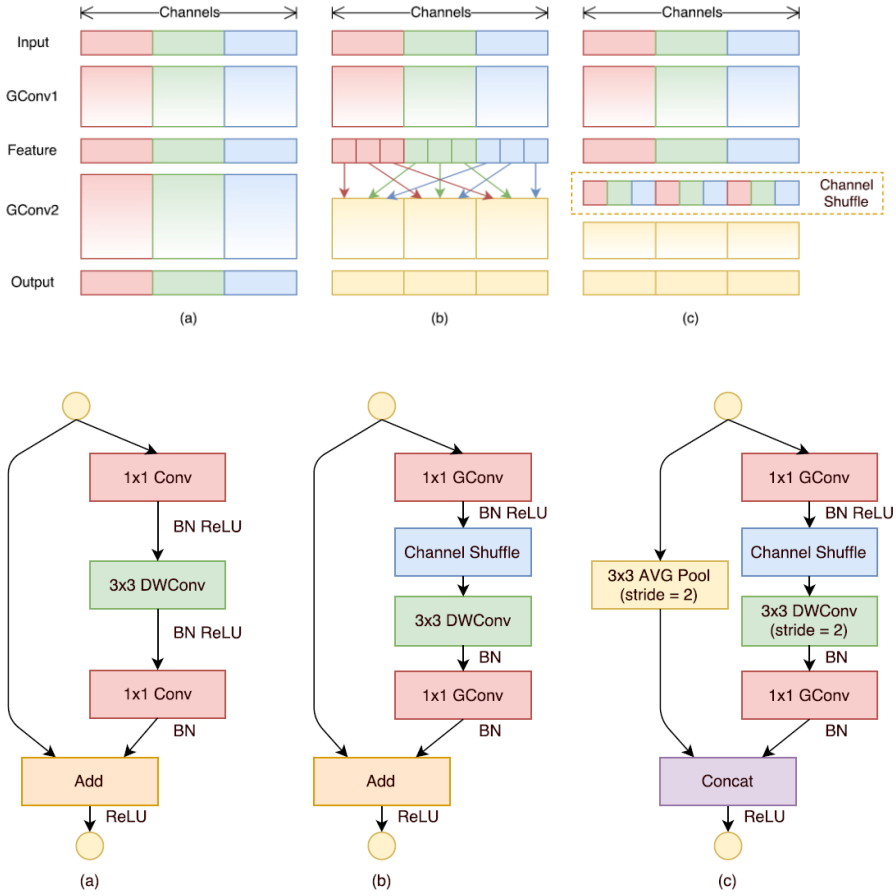


Fuente: [64].

6.3.5 ShuffleNet.

Este modelo se basa en la estrategia de dividir en grupos los tensores de entrada, utilizada por AlexNet para reducir la carga computacional y poder usar diferentes unidades de cómputo en paralelo, sin embargo, la desventaja que dejaba esta estrategia es que las características (*features)* no se podían combinar una vez se separaban y esto bajaba la precisión en algunos casos. La propuesta de los autores con el modelo ShuffleNet es crear bloque en los que se mezclan las características extraídas en convoluciones agrupadas anteriores para corregir esta pérdida de precisión [65]. Su arquitectura se describe a continuación en la figura 26.

Figura 26. Modelo ShuffleNet



Fuente: [65].

Los métodos MobileNetV3 [61] y EfficientNetB0 [62], fueron descartados después de hacer la optimización de hiperparámetros pues no convergieron en la prueba preliminar como se muestra a continuación. Por tal motivo no se describen en esta sección.

6.4 Selección de hiperparámetros para el entrenamiento

Para la optimización de hiperparámetros existen diversas técnicas; sin embargo, hay diferencias importantes entre ellas. Con el fin de seleccionar un método robusto, se realizó un análisis de cuatro técnicas:

1. Búsqueda en cuadricula (Grid Search) [66]
2. Búsqueda en cuadricula aleatoria (Randomized Grid Search) [67]
3. Optimización Bayesiana (Bayesian Optimization) [68]
4. Algoritmos genéticos (Genetic Algorithms) [69]

Las técnicas de Grid Search y Randomized Grid Search son sencillas de implementar, y probablemente son las más usadas por muchos investigadores e ingenieros para optimizar los hiperparametros [192]. Sin embargo, estas técnicas tienen un enfoque de "fuerza bruta", lo que significa que las opciones de hiperparámetros se eligen con base en la intuición o en la experiencia, y se ensayan esperando que dentro de las opciones elegidas se encuentre “la mejor”. Su ventaja es que son simples de implementar y rápidas. Estas técnicas también son conocidas como “no informadas”, debida a que la búsqueda se realiza sin tener en cuenta los resultados de las combinaciones de hiperparametros obtenidas anteriormente [193]. Los métodos “informados” siguen un proceso secuencial iterativo donde se busca encontrar las mejores combinaciones de hiperparametros, teniendo en cuenta los resultados de búsquedas anteriores, con base en algún algoritmo de optimización. Los métodos más usados de este tipo son la optimización bayesiana y los algoritmos genéticos. Una comparación de los diferentes métodos puede verse en [193].

Los métodos Grid Search y Randomized Grid Search se descartaron para este trabajo con el objetivo de basarse en un método más robusto. La selección del método se hizo entre la optimización bayesiana y la optimización con algoritmos genéticos. Para seleccionar uno de estos dos métodos se realizó una búsqueda en la literatura, en la que se exponen los pros y los

contras da cada uno de ellos [192]–[197]. Con base en las conclusiones de estos trabajos se puede resumir lo siguiente:

- La optimización con algoritmos genéticos no requiere ningún modelo probabilístico y trabaja directamente con la función objetivo. Por otro lado, la optimización bayesiana hace uso de una función sustituta, (*surrogate function*), que la hace más eficiente en el uso de recursos computacionales, pues esta función sustituta es una simplificación de la función original, normalmente desconocida en el caso de las redes neuronales.
- Lo anterior hace que se necesiten menos experimentos para lograr resultados aceptables. Y aunque en algunos estudios los resultados con algoritmos genéticos son mejores, la diferencia no es significativa.
- Los algoritmos genéticos tienen excelentes capacidades de paralelismo, por lo que el algoritmo genético tiene un muy buen desempeño cuando las soluciones se van almacenando en memoria, y esta se podrá ir mejorando con el tiempo. Por su parte, la optimización bayesiana no es capaz de explotar el paralelismo porque cada experimento depende del experimento anterior. Sin embargo, al tiempo, eso hace que la optimización con algoritmos genéticos sea un método más exigente en capacidad computacional.
- Existen más características que las diferencian, pero para efectos de este estudio el requerimiento de menos capacidad computacional por parte de la optimización bayesiana lo hacen más adecuado para la selección de hiperparametros en esta investigación, por lo tanto, este fue el método seleccionado.

El método de optimización bayesiana fue aplicado a los siete (7) modelos elegidos para este estudio, y los resultados pueden verse en la tabla 5.

6.4.1 Hiperparámetros obtenidos mediante la optimización bayesiana.

Los dos hiperparámetros elegidos para la optimización fueron la función de optimización y la tasa de aprendizaje (*learning rate*). Las funciones de optimización probadas fueron: *RmsProp, Adagrad, SGD, Adamax, Adam* y *Adadelta.* La tasa de aprendizaje se definió en un rango de $1e^{-6}$ hasta $1e^{-1}$ para la búsqueda.

Según la tabla 12, los resultados de las redes MobileNetV3 y EfficientNetB0 no convergieron, y su *Accuracy* estuvo por debajo del 33 % en el caso de MobileNetV3 y menos de 14 % para EfficientNetB0. Por lo anterior, fueron descartadas para la fase siguiente, que fue la del entrenamiento con los hiperparámetros encontrados. Para el proceso de optimización se emplearon cinco (5) épocas en cada iteración y se determinó un total de cien (100) iteraciones para cada optimización.

Tabla 12. Resultados de la optimización bayesiana para los modelos evaluados.

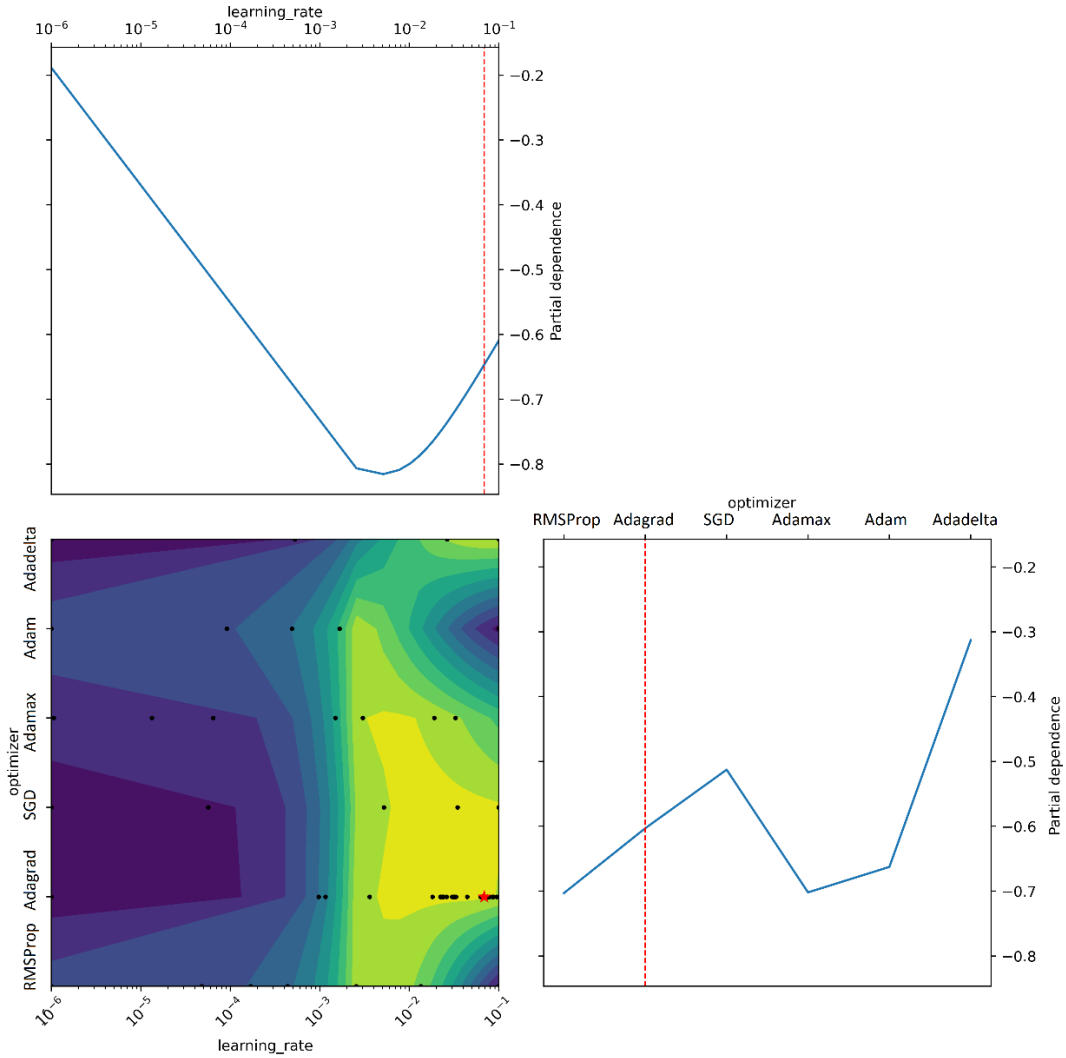
Arquitectura	Optimizador	Learning Rate	Accuracy
MobileNet	Adagrad	0,06851166	0,9197443
MobileNetV2	Adagrad	0,00537715	0,9088542
NasNetMobile	Adamax	0,00051483	0,8411458
ShuffleNet	Adagrad	0,03231302	0,8671875
SqueezeNet	Adagrad	0,02899546	0,8979640
MobileNetV3	SGD	0,03170745	0,3214962
EfficientNetB0	Adamax	1,06e-06	0,1387311

Fuente: elaboración propia.

6.4.2 Gráficos de dependencia parcial.

Estos gráficos constituyen un método de interpretabilidad en *Machine Learning* del tipo modelo-agnósticos [198]. En el gráfico, el eje vertical muestra la probabilidad promedio de las predicciones, y el eje horizontal muestra los valores de los hiperparámetros. La línea azul captura cómo cambia la probabilidad pronosticada promedio a medida que cambian los valores de los hiperparámetros. Se observa que la probabilidad más alta de las predicciones se da en la parte más baja de la curva. El cálculo de los gráficos de dependencia parcial tiene una interpretación causal; es decir, al cambiar uno de los hiperparámetros, se miden los cambios en las predicciones [198]. Al hacer esto se está analizando la relación causal entre el hiperparámetro y la predicción. En la figura 27 se muestra un ejemplo de los gráficos de dependencia parcial que arroja la optimización bayesiana en este estudio. La dependencia parcial muestra el efecto promedio en las predicciones a medida que cambia el valor de los hiperparámetros, en este caso el optimizador y la tasa de aprendizaje.

Figura 27. Ejemplo de los gráficos de dependencia parcial obtenidos para cada combinación de hiperparámetros con la optimización bayesiana aplicada a MobileNet.



Nota. Las líneas y el punto de color rojo muestran la combinación de hiperparámetros con mejor desempeño. Los puntos negros muestran las otras combinaciones probadas durante la iteración.

Fuente: elaboración propia.

6.5 Resultados del entrenamiento (*Train*)

Como resultado de las pruebas de optimización, se eligieron los modelos con mejor desempeño para ser entrenados y compararlos. Para el entrenamiento se utilizaron los datos para entrenamiento (71%) y validación (18%), 89% de los datos totales del *dataset.* El entrenamiento

se llevó a cabo en la plataforma de Google Colab con acceso a GPU. La GPU utilizada de marca NVIDIA tiene las siguientes características (figura 28):

Figura 28. Características de la GPU marca NVIDIA utilizada para el entrenamiento de los modelos en la plataforma Google – Colab

```
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 460.32.03    Driver Version: 460.32.03    CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla P100-PCIE...  Off  | 00000000:00:04.0 Off |                    0 |
| N/A   33C    P0    26W / 250W |      0MiB / 16280MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
```

Fuente: Elaboración propia.

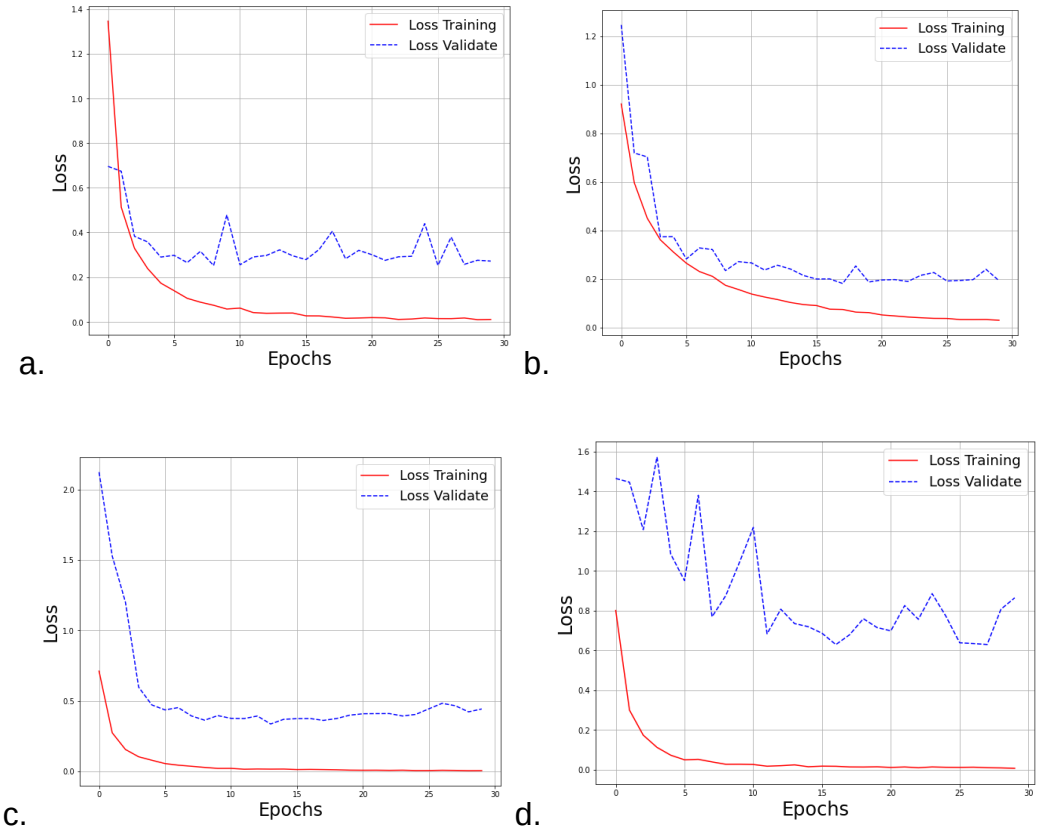
Transfer Learning. Para el entrenamiento de todos los métodos se aplicó aprendizaje por transferencia, más conocido por su nombre en inglés *"Transfer Learning"*, debido a que el número de imágenes por cada clase está entre 2531 y 2670, que no son significativos si se compara con la cantidad de imágenes de IMAGENET que han sido con las que se entrenado los modelos originales (quince millones de imágenes y mil clases). En cada caso se descongelaron alrededor del 70 % de los parámetros entrenables, dejando solo aproximadamente el 30 % de los parámetros al principio de cada red sin cambios. Esto, debido a que los parámetros de las primeras capas ya han aprendido elementos geométricos básicos como líneas o puntos, y se puede aprovechar esto para acelerar el entrenamiento y evitar el sobre-entrenamiento.

6.5.1 Error de predicción para el entrenamiento (Loss).

La figura 29 muestra la evolución de la pérdida durante el entrenamiento. La función de pérdida empleada fue *Categorical Croosentropy.* También llamada pérdida logarítmica o pérdida logística. Con esta función cada probabilidad de clase predicha se compara con la salida deseada 0 ó 1 de la clase real y se calcula una pérdida que penaliza la probabilidad en función de qué tan lejos está del valor esperado real, y la penalización del error es logarítmica. El valor es grande para diferencias grandes cercanas a 1 y es pequeño para diferencias pequeñas que

tienden a 0. La pérdida más alta con los datos de validación se obtuvo con el modelo NasNetMobile, que no bajó de 0,6 y el mejor desempeño lo obtuvo SqueezeNet con un valor cercano a 0,2.

Figura 29. Resultados del entrenamiento de los modelos evaluados (*Loss*)



Nota. *Loss*: a) MobileNet, b) SqueezeNet, c) MobileNetV2 y d) NasNet

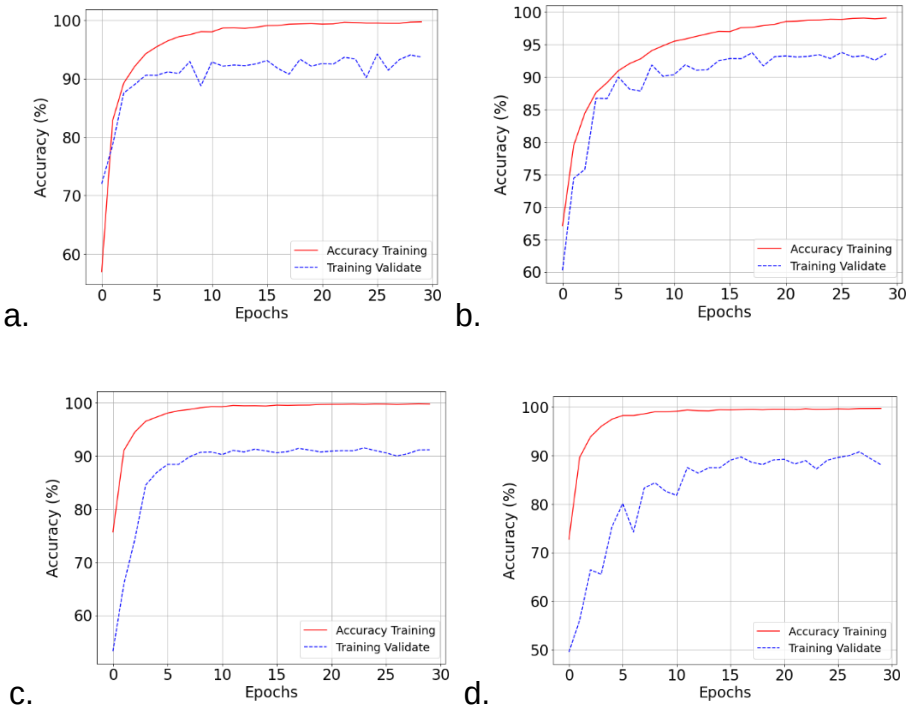
Fuente: elaboración propia.

6.5.2 Exactitud de la predicción obtenida en el entrenamiento (Accuracy).

Como se mencionó anteriormente, la exactitud general de un modelo es el número de predicciones correctas dividido por el número total de predicciones que da como resultado un valor porcentual. Esta medición se hace simultáneamente con los datos seleccionados para el entrenamiento y los datos para la validación. El objetivo principal es que los dos resultados no se alejen de manera significativa. En el caso del entrenamiento realizado con los modelos seleccionados (figura 30); la mejor exactitud con los datos de validación se obtuvo con el modelo

MobileNet, muy cercana al 95 % y la exactitud más baja se obtuvo nuevamente con el modelo NasNet, lo que corresponde con los resultados de pérdida obtenidos. También es importante resaltar que el comportamiento de ambos (pérdida y exactitud) muestra que no hubo sobre-entrenamiento, pues la brecha entre los resultados de entrenamiento y validación se mantiene constante y no es significativamente grande.

Figura 30 Resultados del entrenamiento de los modelos evaluados (Accuracy)



Nota. Accuracy: a) MobileNet, b) SqueezeNet, c) MobileNetV2 y d) NasNet

Fuente: elaboración propia.

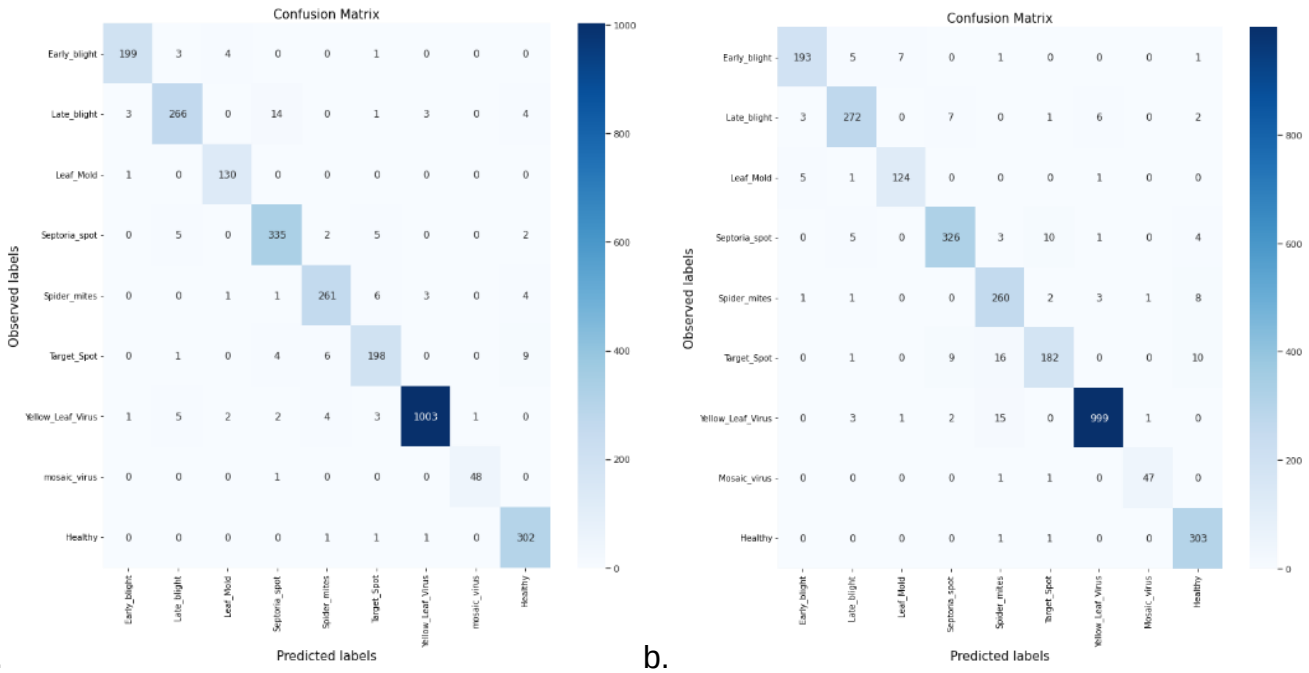
6.6 Resultados de la prueba (*Test*)

A continuación, se presentan los resultados de la prueba realizada con los datos de prueba, 11% de los datos totales del dataset. Igualmente, esta prueba se realizó en la plataforma de Google-Colab, en una GPU con las mismas especificaciones mostradas anteriormente en la figura 23. Es importante recordar que estos datos reservados para la prueba no han sido “vistos” por ninguno de los modelos evaluados, a diferencia de los datos de entrenamiento y validación.

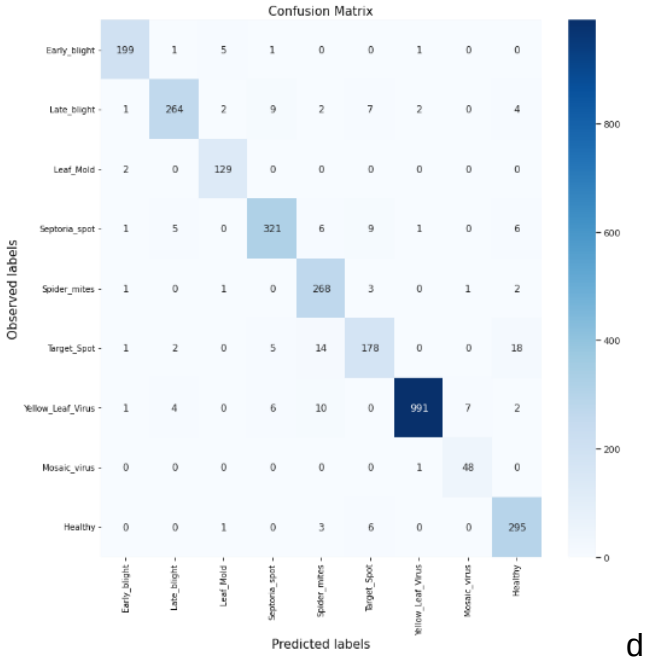
6.6.1 Matriz de confusión.

La matriz de confusión para cada uno de los cuatro métodos mejor calificados se muestra a continuación (figura 31). La matriz de MobileNet (a), muestra que la red presenta un mayor error al predecir la clase *Septoria_Spot* en la que catorce muestras fueron clasificadas como *Late_Blight,* en las demás no son significativos los errores. Por otro lado, parece existir una mayor probabilidad de error en general para los tres métodos restantes, al tratar de clasificar la clase *Spider_Mites,* y clasificarla como *Yellow_Leaf_Virus y Target_spot.* Lo que muestra que pueden tener características similares en algunos de los casos, aunque finalmente las métricas indican que los métodos son capaces de generalizar.

Figura 31. Resultados de la evaluación de los modelos con los datos de *test* “Matriz de confusión”



a. b.



c.

d.

Nota. a) MobileNet, b) SqueezeNet, c) MobileNetV2 y d) NasNet

Fuente: elaboración propia.

Las métricas individuales se muestran en el anexo D, donde se confirma una menor precisión en la clasificación de las clases, *Spider_Mites*, y *Target_Spot,* para los todos los métodos excepto para MobileNet. Esto puede indicar que para estas dos enfermedades podrían emplearse estrategias de clasificación por separado, si se quiere una mayor precisión, además de incluir más datos para el entrenamiento.

6.7 Análisis de resultados

Aunque en general las métricas obtenidas que se muestran en la tabla 13, obtenidas a partir de las matrices de confusión, son valiosas para el análisis de los resultados, el objetivo del análisis se centró en las métricas *Precision* y *Recall,* porque en el contexto del problema de investigación de este trabajo se consideran estas dos métricas como las más importantes, al tratarse de la detección de enfermedades en cultivos.

Tabla 13. Métricas obtenidas con las matrices de confusión para cada modelo

Modelo	Accuracy	Precision	Recall	F1-score	Parámetros	MB
MobileNet	96,31 %	95,55 %	95,93 %	95,72 %	3.762.056	28,7
SqueezeNet	95,05 %	93,98 %	93,95 %	93,91 %	120.760	1,2
NasNetMobile	95,01 %	92,73 %	94,22 %	93,29 %	5.495.132	64,7
MobileNetV2	94,59 %	92,35 %	94,20 %	93,17 %	3.741.448	28,8
ShuffleNet	91,50 %	89,36 %	90,80 %	89,93 %	969.256	8,2

Fuente: elaboración propia.

MobileNet: cuando el clasificador indica que los datos de la muestra representan una enfermedad específica, esto es correcto el 95,55 % de las veces en promedio (*Precision*). Además, este detecta el 95,93 % de las enfermedades presentes en promedio (*Recall*).

SqueezeNet: cuando el clasificador indica que los datos de la muestra representan una enfermedad específica, esto es correcto el 93,98 % de las veces en promedio (*Precision*). Además, este detecta el 93,95 % de las enfermedades presentes en promedio (*Recall*).

NasNetMobile: cuando el clasificador indica que los datos de la muestra representan una enfermedad específica, esto es correcto el 92,73 % de las veces en promedio (*Precision*). Además, este detecta el 94,22 % de las enfermedades presentes en promedio (*Recall*).

MobileNetV2: cuando el clasificador indica que los datos de la muestra representan una enfermedad específica, esto es correcto el 92,35 % de las veces en promedio (*Precision*). Además, este detecta el 94,20 % de las enfermedades presentes en promedio (*Recall*).

ShuffleNet: cuando el clasificador indica que los datos de la muestra representan una enfermedad específica, esto es correcto el 89,36 % de las veces en promedio (*Precision*). Además, este detecta el 90,80 % de las enfermedades presentes en promedio (*Recall*).

6.7.1 Métrica seleccionada para la comparación del desempeño.

Para el caso de las enfermedades en cultivos agrícolas, existen dos aspectos de vital importancia en la toma de decisiones para su control:

a) El costo que representa la pérdida de cultivos agrícolas; por lo tanto, debe actuarse de manera rápida para controlar el foco de la enfermedad y no tener que aplicar agentes químicos a áreas más grandes.
b) La detección correcta de la enfermedad es necesaria para no aplicar agentes de control equivocados, lo que puede provocar el aumento de la resistencia de la enfermedad a controles posteriores.

Por lo tanto, la métrica que brinda más confiabilidad para la toma de decisiones en este contexto es el *Recall.* Pues, aunque un modelo con un mejor *Accuracy* aciertan la mayoría de las veces que hacen una predicción, si su *Recall* es más bajo que otros, significa que dejan pasar por alto más enfermedades. Tratar de mejorar esta métrica es un objetivo importante, siempre y cuando se mantenga un costo computacional razonable para el hardware empleado en la solución.

6.7.2 Método seleccionado.

Con base en el análisis anterior, el método que presentó un mejor desempeño en todas las métricas fue MobileNet en su versión original. Y aunque es el modelo con el segundo mayor número de parámetros de todos los entrenados (3.762.056), fue el método elegido para integrarlo en la plataforma IoT de agricultura inteligente.

6.8 Resumen del capítulo

En este capítulo se detalló el proceso de selección del método de clasificación de imágenes, con base en modelos del estado del arte. Inicialmente se presentó la pre-selección de diferentes modelos que cumplieron con los criterios de selección. Luego se utilizó el método de optimización Bayesiana para elegir los hiperparámetros: *Leraning Rate* y Función de optimización. Posteriormente se evaluaron los resultados del entrenamiento, dando como resultado la elección de MobileNet como el método de clasificación de imágenes que mejor desempeño tuvo para el *dataset* propuesto en este trabajo. Con la selección del método se cumple el segundo objetivo específico de este trabajo.

En el siguiente capítulo se presenta la implementación del método de clasificación de imágenes elegido en una plataforma de Agricultura Inteligente en condiciones reales de trabajo, con base en la arquitectura de referencia seleccionada en el capítulo 3.

7 Implementación

En el capítulo anterior se seleccionó el método de clasificación de imágenes MobileNet. Una red neuronal convolucional creada para trabajar en ambientes con condiciones limitadas de capacidad de cómputo. Adicionalmente en el capítulo 2 se seleccionó la arquitectura de referencia LoRaFarm, creada para contextos rurales con escasas fuentes de energía y comunicaciones.

En este capítulo se presenta el proceso de implementación del método de clasificación de imágenes en una plataforma de Agricultura Inteligente creada con base en la arquitectura de referencia seleccionada. Adicionalmente se presenta una evaluación de desempeño de tres modelos modificados de MobileNet reducidos en su arquitectura 75, 50 y 25% con respecto a la arquitectura del modelo original. Esta parte de la investigación da respuesta al tercer y último objetivo específico de esta tesis doctoral.

La plataforma IoT de Agricultura Inteligente fue creada para pequeños productores de hortaliza bajo invernadero. La función de la plataforma es recolectar datos de diferentes variables a través de sensores, incluyendo cámaras para la clasificación de imágenes que permitiera detectar enfermedades en un cultivo de tomate.

Este capítulo está organizado de la siguiente manera: En la primera sección se describe el lugar elegido para la implementación. En la segunda sección se presenta la descripción general de la tecnología LoRa y el protocolo LoRaWAN. Luego, en la tercera sección se describe detalladamente la arquitectura de la plataforma implementada. Posteriormente, se describe el proceso de clasificación de imágenes dentro de la plataforma en la cuarta sección. Finalmente, en quinto lugar, se presenta la evaluación de las tres versiones modificadas del modelo original y su desempeño.

7.1 Lugar de implementación.

Las unidades productivas piloto están ubicadas en la vereda San José de la Montaña, corregimiento de San Cristóbal, municipio de Medellín, a una altura de 2200 msnm (figura 32).

Figura 32 .Ubicación del sitio seleccionado para la implementación

Nota. Panorámica general de la zona de estudio: 1: Invernadero No.1; 2: Invernadero No.2; N-A: Nodo tipo A; N-B: Nodo tipo B; M-E: Estación meteorológica. Coordenadas: 6°17’42.68’’ N 75°39’25.99’’ O

Fuente: elaboración propia.

Para el estudio se seleccionaron dos productores de hortalizas que tienen invernaderos, instalados desde el año 2020. Estos invernaderos son tipo semi-arco, cada uno de 300 m². El invernadero no.1 de 15m de ancho x 20m de longitud, y el invernadero no.2 de 10m de ancho x 30m de longitud, con sistema de riego por goteo, cortinas para ventilación de acción manual y 6m de altura máxima. Los cultivos principales son tomate, cebolla y cilantro, estos cultivos se alternan de acuerdo con la programación de los productores y la época del año (figura 33).

Figura 33. Imágenes de los invernaderos seleccionados

a.  b. 

Nota. a) Invernadero no.1, b) Invernadero no.2

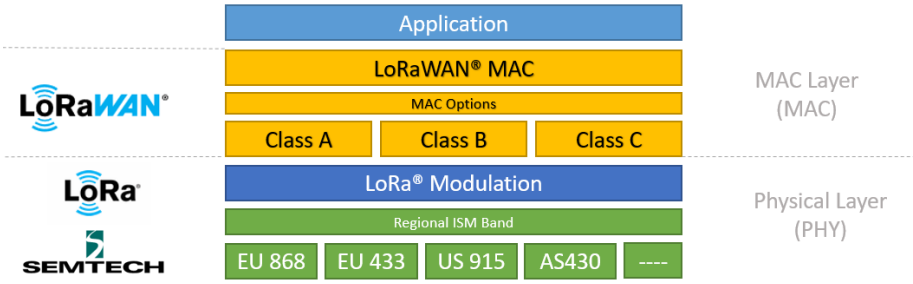
Fuente: elaboración propia.

7.2 Descripción general de la tecnología LoRa y LoRaWAN.

Existen distintas formas de conectividad, todas actúan en un rango del espectro electromagnético. Para el caso de comunicaciones a corta distancia se usan tecnologías como WiFi, Bluetooth o ZigBee [28], [199]. Por otro lado, para transmitir a largas distancias existen tecnologías como LoRa, GPRS, LTE o SigFox, que hacen uso del espectro de manera distinta [28], [199]. La tecnología LoRa (*Long Range*) proporciona una manera de utilizar el espectro inalámbrico sin licencia, en la banda ISM (*Industrial, scientific and medical purposes*), 869 MHz (Europa) y 915 MHz (Norte y Sur América), que permite ahorros importantes de energía y transmisiones a grandes distancias [200], [201].

LoRa es un tipo de modulación patentada basada en *Chirp Spread Spectrum* (CSS), por lo tanto, se refiere a la capa física (PHY) [28]. Por otro lado, LoRaWAN es el conjunto de mecanismos y protocolos que se usan para que varios dispositivos compartan el mismo canal de comunicación sin que se presenten pérdidas de información o interferencias; en otras palabras, proporciona lo que se conoce en informática y comunicaciones como MAC (*Medium Access Control*), [25], [28], [202]. La diferencia entre las dos tecnologías se presenta en la figura 34.

Figura 34. Diferencia entre LoRa y LoRaWAN desde la arquitectura



Fuente: [203].

La topología típica de LoRaWAN es tipo estrella. Los nodos finales pueden comunicarse con más de una puerta de enlace (*gateway*), y esta puede ser de doble vía ascendente (desde el nodo a la puerta de enlace) y descendente (desde la puerta de enlace hasta el nodo) (*downlink & uplink*) [203]. La comunicación con los servidores en la Nube puede realizarse con tecnología basada en conexión IP [203]. Los nodos se clasifican en tres categorías A, B y C, dependiendo del comportamiento del envío de paquetes *downlink.* La clase A es compatible con todos los dispositivos LoRaWAN, en este grupo un dispositivo puede recibir paquetes de enlace descendente solo después de enviar un paquete, lo que resulta en el modo de consumo de energía más bajo. Los dispositivos de clase B son adecuados para aplicaciones que requieren un tráfico de enlace descendente más intenso, ya que abren ventanas de comunicación a intervalos programados. Por último, los dispositivos de clase C siempre están escuchando el canal; por lo tanto, su consumo de energía es el más alto, pero pueden recibir un paquete de enlace descendente en cualquier momento, lo que lleva a la latencia de enlace descendente más baja [28], [203], [204]. La figura 36 muestra la arquitectura típica para la implementación de plataformas con LoRa y LoRaWAN.

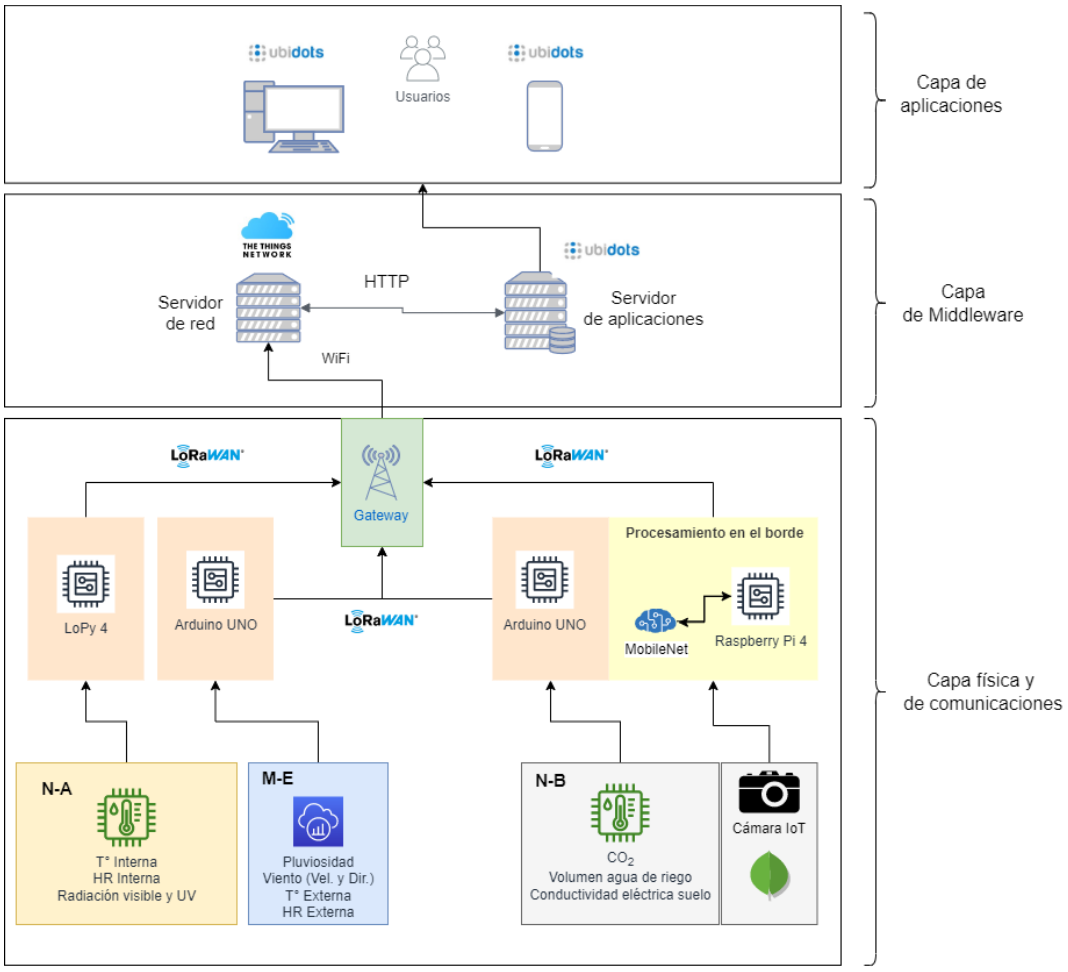
La modulación LoRa se caracteriza por un factor de propagación (SF, por sus siglas en inglés) que define la duración de la señal: cuanto mayor sea el SF, mayor será el tiempo, así como mayor será la distancia admitida por un enlace. La modulación LoRa tiene un total de seis factores de propagación (SF7 a SF12). Cuanto mayor sea el factor de propagación utilizado, más lejos podrá viajar la señal y el receptor de radio frecuencia (RF) aún la recibirá sin errores [203]. Con SF = 12 da como resultado la sensibilidad y el rango de transmisión más altos, pero

con la velocidad de datos más baja y el consumo de energía más alto. Una disminución de una unidad en SF duplica la tasa de datos transmitidos y reduce a la mitad la duración de la transmisión, también se reduce a la mitad el consumo de energía [203], [204].

7.3 Arquitectura de la Plataforma de Agricultura Inteligente implementada

La plataforma prototipo para la implementación del método de clasificación de imágenes seleccionado en el capítulo 5 se construyó con base en la arquitectura de referencia LoRaFARM seleccionada en el capítulo 3. La figura 35 a continuación muestra el esquema general de la arquitectura implementada.

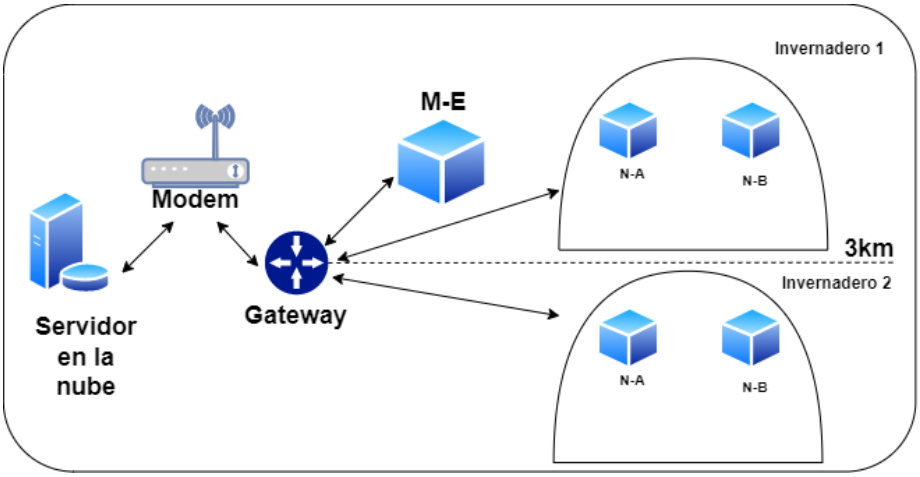
Figura 35. Arquitectura de la plataforma de Agricultura Inteligente implementada



Fuente: elaboración propia.

Por otro lado, como se mencionó anteriormente en la sección 7.1 la topología de la plataforma implementada incluyo dos unidades productivas. Estas unidades estaban ubicadas a 3 kilómetros de la puerta de enlace o *Gateway.* Luego esta puerta de enlace se conectó por medio de WiFi al servidor en la Nube, donde se almacenaron los datos. A continuación, en la figura 36 se muestra un esquema general de la topología que tuvo la plataforma.

Figura 36. Topología de la plataforma IoT implementada



Nota. M-E: Estación meteorológica, N-A: Nodo tipo A, N-B: Nodo tipo B

Fuente: elaboración propia.

7.3.1 Capa física y de comunicaciones.

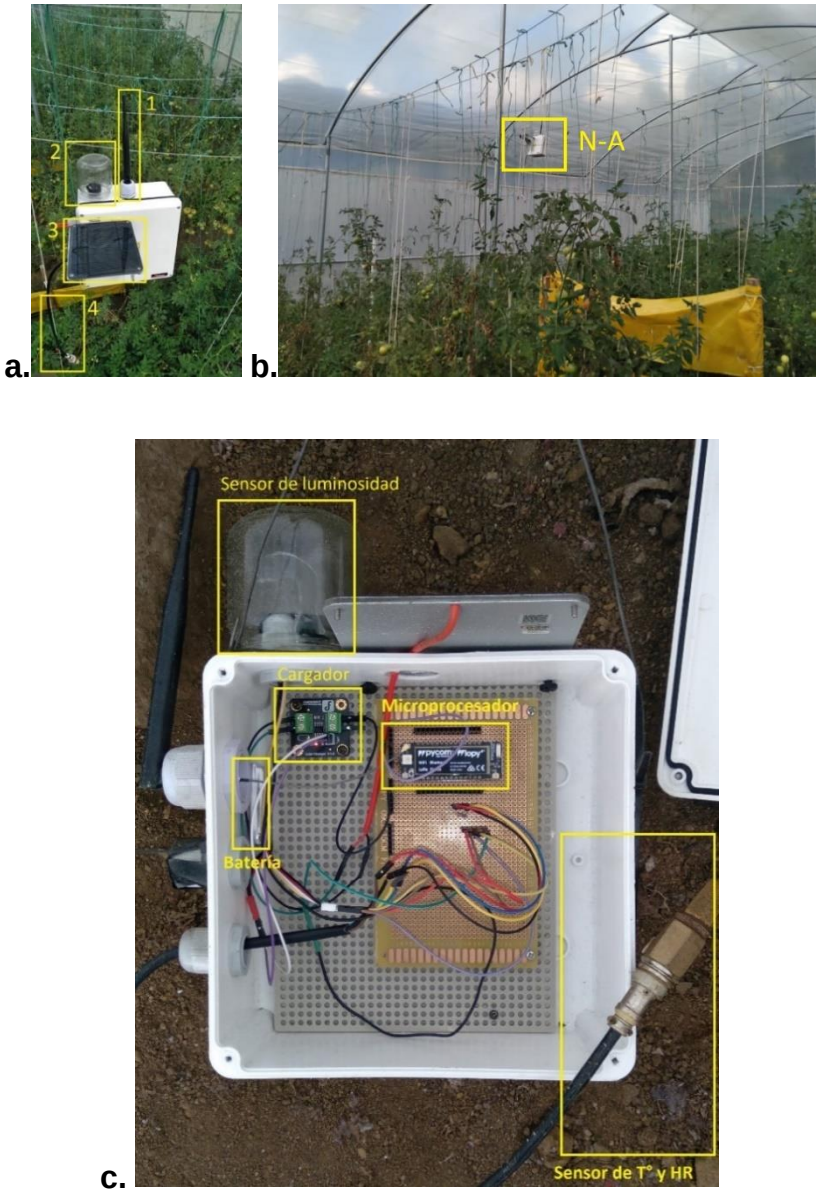
I. **Nodos tipo A (N-A).** Los nodos tipo A se encuentran compuestos por un sistema de unidad de procesamiento, diseñada para trabajar con LoRaWAN, un sistema de alimentación de energía, sensores y su caja de aislamiento o protección ambiental, como se observa en la figura 37. El nodo se une a la red LoRaWAN sin necesidad de un intermediario (directamente al *gateway*), para el envío de datos capturados. Cada N-A integra:

- **Unidad de procesamiento:** consta del microcontrolador y la antena para la comunicación inalámbrica. El microcontrolador es una placa de desarrollo LoPy4 [205], que puede implementarse con cualquiera de las tecnologías LoRa, Sigfox, WiFi o

Bluetooth. La integración de estas cuatro opciones en una sola tarjeta representa una gran ventaja, debido a que no requiere accesorios adicionales para la comunicación inalámbrica, como es el caso de las tarjetas Raspberry y Arduino. Otro criterio relevante al seleccionar la tarjeta LoPy4 es su ultra bajo consumo de energía (menor de 400 mA). Se utilizó la antena para LoRaWAN con conector IPEX a SMA de 915 MHz, según el estándar estadounidense. Con el uso de la tecnología inalámbrica LoRaWAN, la cual con su factor de propagación configurado en 7 (SF = 7), permite alcanzar una distancia de hasta 5 kilómetros (la más baja), con el menor consumo energético y la mayor velocidad que permite el protocolo.

- **Sistema de alimentación:** consta de una batería tipo LiPo (Litio-Polímero) con un voltaje de salida de 3,7 V y una corriente 780 mAh, un panel solar monocristalino de 2 W y un cargador solar Li-Po. Otra ventaja del microcontrolador elegido es que puede llevarse a un estado de “sueño profundo” o descanso, mientras no se requiere envío de datos, lo que reduce significativamente el consumo energético. Así que, para estimar el consumo total que requiere el sistema, basado en la ficha técnica del dispositivo, de los sensores y con un envío de datos cada veinte minutos (1.200 segundos), de los cuales toma solo diez segundos para el envío de datos, se estima un consumo inicial de 22,2 mAh, lo que supone una autonomía en la batería de 1,5 días aproximadamente.
- **Sensores**: el sistema de sensores tiene como objetivo recopilar variables climáticas dentro del invernadero. Se utilizó un sensor industrial SHT10 [206] para medir humedad relativa (%) y temperatura (°C). Para la medición de luz visible (lux), luz infrarroja y luz ultravioleta se utilizó un sensor de luz solar Si1145 [207]. Como criterios de selección básicos de los sensores se tuvo en cuenta el bajo consumo energético, el bajo costo y su adaptabilidad al entorno.
- **Protección ambiental:** para estos nodos se utilizaron cajas de protección Ip65 y prensas estopa que aíslan la parte interior, para garantizar una resistencia contra el agua, el polvo y otros factores ambientales que puedan ocasionar un daño en los equipos.

Figura 37. Características internas y externas de los nodos tipo A (N-A)



Nota. a) Componentes externos del N-A: (1) antena, (2) sensor de luz, (3) panel solar, (4) sensor SHT10. b) Ubicación del nodo a 3m de altura dentro del invernadero. c) Componentes internos del N-A

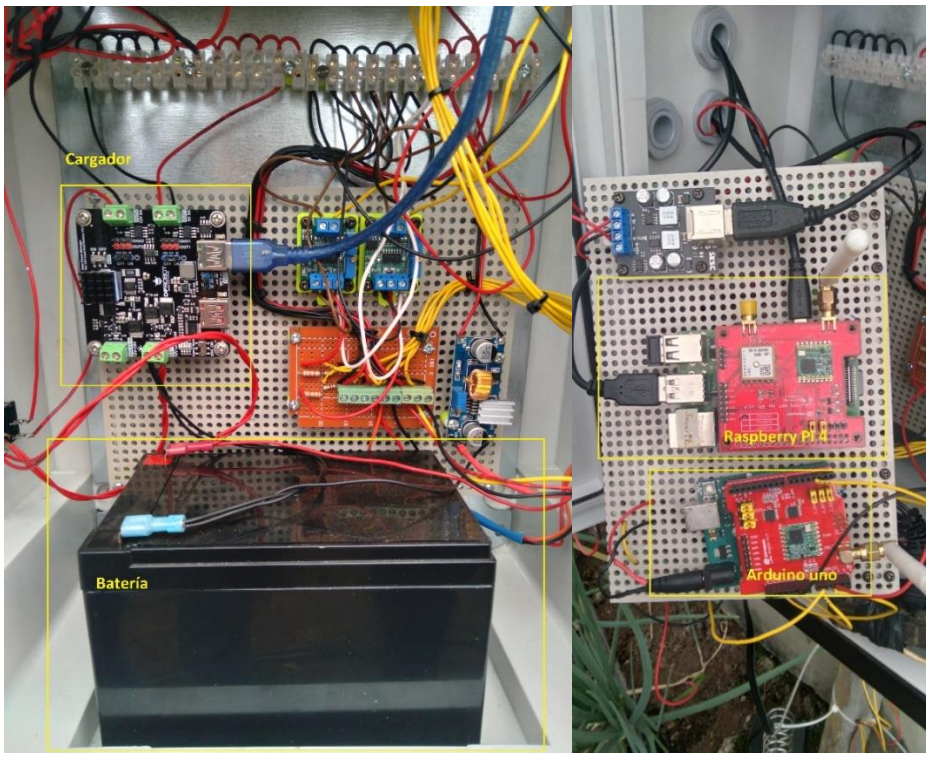
Fuente: elaboración propia.

II. **Nodos tipo B (N-B).** Los nodos tipo B se encuentran compuestos por un microprocesador Raspberry Pi-4, un microcontrolador Arduino UNO para el control de los sensores, con sus respectivos módulos de expansión LoRa, que les permite

conectarse a una red con LoRaWAN. Un sistema de alimentación de energía, sensores y su caja de aislamiento o protección ambiental, como se observa en la figura 38. Al igual que el N-A, el N-B se une a la red LoRaWAN sin necesidad de un intermediario, para el envío de datos capturados. El N-B también integra una unidad para la captura de imágenes con una cámara IoT. Cada N-B integra:

- **Unidades de procesamiento:** Unidad Raspberry Pi 4 con CPU ARM Cortex-A72, 1.5 GHz, 4GB de SDRAM con Micro-SD de 32 GB para el sistema operativo y almacenamiento de datos [208]. Esta unidad se utilizó para la implementación de la red neuronal propuesta y la captura de imágenes. Unidad Arduino UNO, con microcontrolador ATmega328P, 14 Pins I/O digitales, y 6 Pins I, para la conexión de sensores. Esta unidad de Arduino se utilizó para la conexión de los demás sensores.
- **Sistema de alimentación:** consta de una batería seca regulada por válvula tipo recargable sellada, con un voltaje de salida de 12 V y una corriente 3,6 A, un panel solar monocristalino de 10 W y un cargador solar especial para baterías ácido-plomo con salidas duales de alta potencia 5V 5A (OUT1) y 12V 8A (OUT2) y salidas USB duales 5V 2,5A (USB1/USB2).
- **Sensores**: el sistema de sensores del N-B tiene como objetivo recopilar variables de consumo de agua, contenido de CO_2 en el aire, y conductividad eléctrica del suelo. Para la medición del CO_2 se utilizó un sensor MG-811 [209]. Para medir la conductividad eléctrica del suelo se utilizó un sensor RK500-03EC [210] y para medir el consumo de agua un sensor FS400A [211].
- **Módulo para captura de imágenes**: el módulo para captura de imágenes se construyó con una impresora 3D, al cual se le adaptó un display LCD de 3,5” para la gestión del aplicativo de toma de imágenes y clasificación de enfermedades, con una cámara de 0,3 Mega pixeles y resolución de 640x480, con cable USB compatible con tarjetas Raspberry Pi y NVIDIA [212].
- **Protección ambiental:** para estos nodos también se utilizaron cajas de protección Ip65 y prensas estopa que aíslan la parte interior, para garantizar una resistencia contra el agua, el polvo y otros factores ambientales que puedan ocasionar un daño en los equipos.

Figura 38. Componentes internos del nodo tipo B (N-B)



Fuente: elaboración propia.

III. **Estación meteorológica (M-E).** Como parte de la plataforma se instaló una estación meteorológica con anemómetro, veleta de viento y pluviómetro, modelo SEN0186 [213], con las siguientes características (figura 39).

- Voltaje de operación: 5V.
- Rango de temperatura de operación: - 40 ºC ~ 80 ºC.
- Rango de humedad: 0 ~ 99 %.
- Peso: 4480g.
- Microcontrolador: Arduino UNO.

Figura 39. Estación meteorológica modelo SEN0186

Fuente: elaboración propia.

IV. **Puerta de enlace (*Gateway*).** Para la puerta de enlace se seleccionó modelo DLOS8N [214] de *DRAGUINO Technology Co* (figura 40). Es una puerta de enlace LoRaWAN para exteriores de código abierto, permite unir la red inalámbrica LoRa a una red IP a través de WiFi, Ethernet, celular 3G o 4G. Se ubicó a una distancia aproximada 3 km de distancia de los nodos, en un sitio con acceso a internet.

Su selección se hizo teniendo en cuenta que es un modelo compacto y fácil de instalar. DLOS8N es compatible con el reenviador de paquetes Semtech y la conexión de la estación LoRaWAN. Este *gateway* incluye un concentrador SX1302 LoRaWAN. Tiene bandas de frecuencia estándar preconfiguradas para usar en diferentes países. El usuario también puede personalizar las bandas de frecuencia para usar en su propia red LoRaWAN.

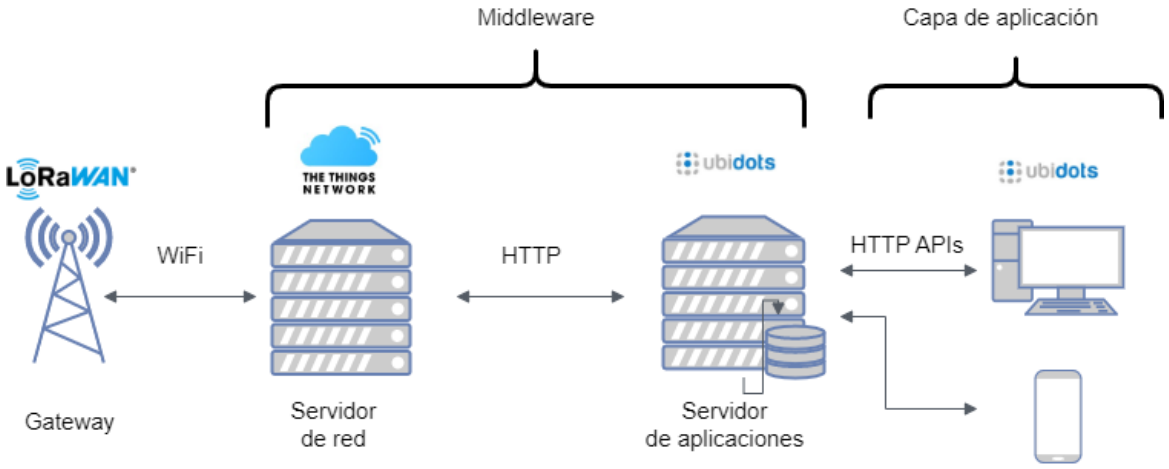
Figura 40. Puerta de enlace (gateway) Draguino Modelo DLOS8N

Fuente: [214].

7.3.2 Capa de middleware.

La capa de *middleware* tiene dos servidores. Se utilizó *The Things Network* (TTN) [215] como servidor de red (NS por sus siglas en inglés) y Ubidots [216] como servidor de aplicaciones (AS por sus siglas en inglés) de acuerdo con la arquitectura de referencia [28]. Una vez los datos llegan a la puerta de enlace, este usa una red Wifi para enviarlos a servidor y a la plataforma donde son decodificados con ayuda de código. La plataforma de acceso gratuito que se integra de forma transparente al proceso es TTN versión 3, cuya última versión de consola en el año 2022 es *The Thing Stack* (TTS), este servidor de IoT es de código abierto y permite programar la puerta de enlace y aplicaciones donde se pueden registrar diversos nodos. Básicamente se conectan los datos derivados de estos a internet a través del servidor disponible para uso libre "*eu1.cloud.thethings.network*". El *gateway*, como dispositivo multicanal, permite la transmisión de datos desde diversos nodos de manera simultánea; sin embargo, la cuenta gratuita en esta plataforma proporciona la integración de un dispositivo como puerta de enlace, una aplicación y diez dispositivos como máximo. Lo cual, para este proyecto, fue suficiente. Cada nodo puede transmitir hasta 128 bits y la trasmisión de los datos deben ser como mínimo cada tres minutos, se debe tener en cuenta que el tiempo de transmisión de datos debe ser optimizado con el fin de mantener un consumo bajo de energía (ver figura 41).

Figura 41. Conexión de la capa de middleware y la capa de aplicaciones



Fuente: elaboración propia.

The Things Network es una plataforma que no maneja almacenamiento de datos, solo permite visualizarlos en tiempo real si el usuario hace un seguimiento del estado de los dispositivos; por lo cual, se ofrece la opción de establecer integraciones con otras aplicaciones y bases de datos a través de protocolos como MQTT (*Message Queing Telemetry Transport*) o HTTP (*Hypertext Transfer Protocol*), con el fin de mantener un registro permanente de los datos recolectados y poder aplicar herramientas de ciencias de datos e Inteligencia Artificial para el tratamiento de estos en otra capa de la arquitectura.

7.3.3 Capa de aplicaciones.

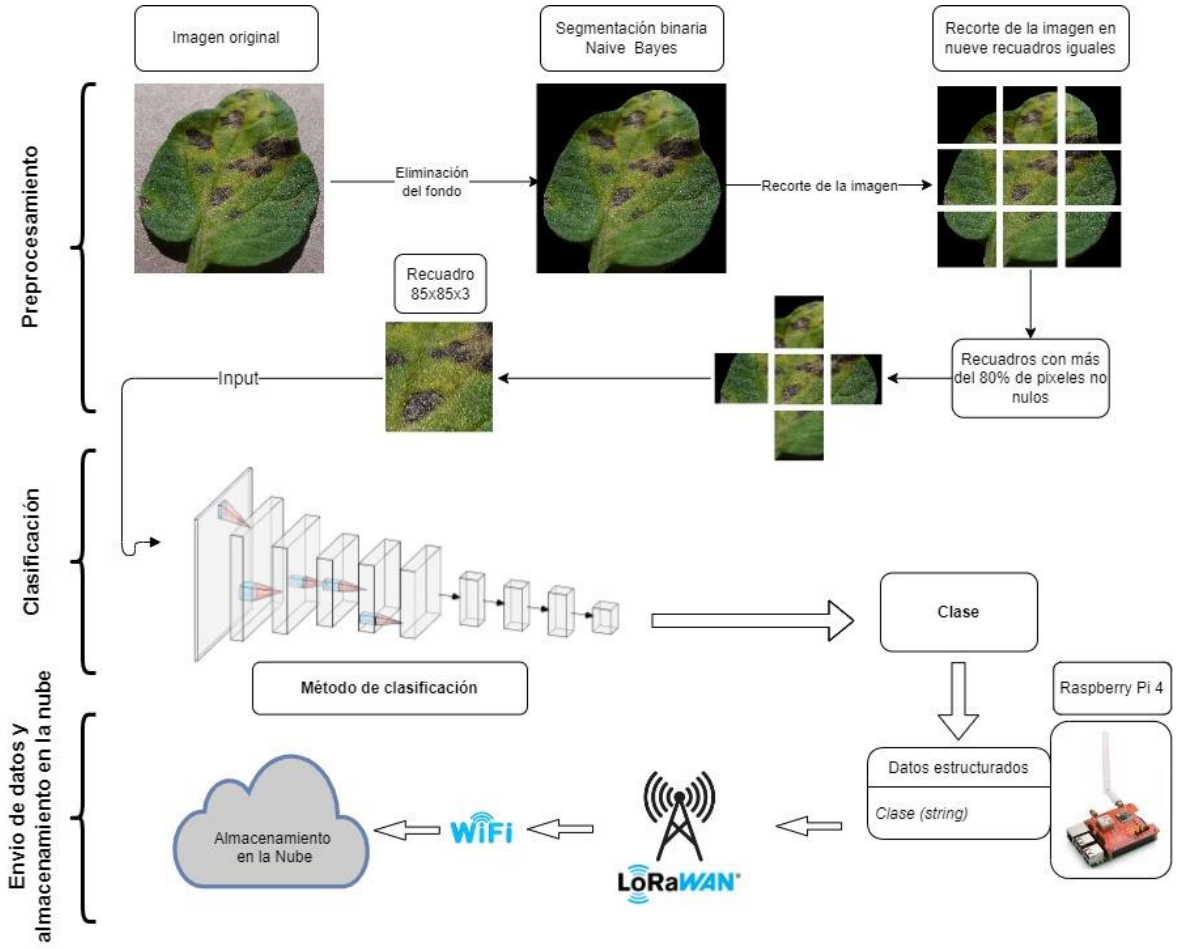
La plataforma de IoT Ubidots también aporta el componente de aplicaciones en esta investigación; este se integra con TTN a través del protocolo HTTP, la cual permite conectar los aplicativos IoT, almacenar la información de todas las variables de forma automática en la Nube, visualizar el último registro de forma más amigable para el usuario final y enviar alertas si hay algún comportamiento inusual. Ubidots también cuenta con su versión de instalación para Android, lo cual permite acceder a los datos desde cualquier lugar. Esta plataforma se utiliza de forma temporal, debido a que su versión gratuita tiene una limitación por un año en el almacenamiento de datos y limitaciones de visualización. El objetivo en un proyecto futuro es desarrollar una plataforma propia que sea lo más amigable posible para agricultores y asesores técnicos.

7.4 Descripción del proceso de clasificación de imágenes en la plataforma

La implementación del método de clasificación de imágenes, empleando técnicas de Inteligencia Artificial a una plataforma de agricultura inteligente, propuesto en este trabajo se implementó en los nodos tipo B (N-B) y tiene de tres etapas que se describen en la figura 42.

I. Pre-procesamiento de las imágenes.
II. Clasificación.
III. Envío de datos y almacenamiento en la Nube.

Figura 42. Esquema general del proceso de clasificación de imágenes en la plataforma IoT



Fuente: elaboración propia.

I. **Pre-procesamiento de las imágenes.** El preprocesamiento de las imágenes se llevó a cabo en tres subetapas: (1) la captura de las imágenes, que se realizó con un módulo independiente creado para tal fin como se muestra en la figura 43 y cuyos elementos están descritos en la sección anterior (Nodos tipo B - Módulo para captura de imágenes). (2) Segmentación, en la cual mediante un algoritmo de clasificación Bayesiana se retira el color del fondo. Y (3) el recorte de la imagen donde se seleccionan solamente los segmentos en los que la hoja ocupa un 80% de pixeles o más.

Figura 43. Módulo para la captura de imágenes

Fuente: elaboración propia.

II. **Clasificación.** En esta etapa los recortes con el 80% o más de pixeles pertenecientes a la hoja pasan por la red neuronal MobileNet y son clasificados de acuerdo con el entrenamiento de la red. La clase que es almacenada es la que presenta un mayor porcentaje de probabilidad según el algoritmo de clasificación, y por lo menos uno de los recortes que presente la enfermedad es suficiente para almacenar los datos como pertenecientes a una de las enfermedades. Esto debido a que algunos recortes pueden quedar con tejido sano y ser clasificados en la clase *Health.*

III. **Envío de datos y almacenamiento en la Nube.** Luego de ser clasificada la imagen, la aplicación creada para la integración envía el dato directamente desde el microprocesador Raspberry al Gateway y este a su vez la envía al servidor donde es

almacenado. Es importante aclarar que no se envían imágenes a través de la red y los recortes son eliminados cada cierto periodo de tiempo que puede ser programado de acuerdo con el volumen de imágenes que se puedan obtener. Esto es importante porque se debe cuidar que la memoria del microprocesador no se consuma con el almacenamiento de imágenes.

7.5 Evaluación del método seleccionado

Con el fin de evaluar la robustez del método que mostró un mejor desempeño (MobileNet), se entrenaron tres versiones reducidas de este modelo para medir su desempeño en condiciones limitadas de capacidad computacional:

- MobileNet_25: reducción de parámetros del 25 %.
- MobileNet_50: reducción de parámetros del 50 %.
- MobileNet_75: reducción de parámetros del 75 %.

7.5.1 Optimización de hiperparámetros para las modificaciones de MobileNet.

Previo al entrenamiento de estos tres modelos reducidos, se aplicó el método de optimización bayesiana para buscar las mejores tasas de aprendizaje y funciones de optimización posibles para cada uno de los modelos. Al igual que para los primeros siete métodos considerados, se realizó una exploración con los optimizadores: *RmsProp*, *Adagrad*, *SGD*, *Adamax*, *Adam* y *Adadelta.* La tasa de aprendizaje se definió en un rango de $1e^{-6}$ hasta $1e^{-1}$ para la búsqueda. Se realizaron cien iteraciones. Como puede observarse en la tabla 14, los resultados fueron los mismos para las tres redes, optimizador = Adagrad y tasa de aprendizaje = 0,1.

Tabla 14. Resultados de la optimización Bayesiana para los modelos reducidos de MobileNet

Arquitectura	Optimizador	Learning Rate	Accuracy
MobileNet_75	Adagrad	0,1	0,9089877
MobileNet_50	Adagrad	0,1	0.8896812
MobileNet_25	Adagrad	0,1	0.8578020

Fuente: elaboración propia.

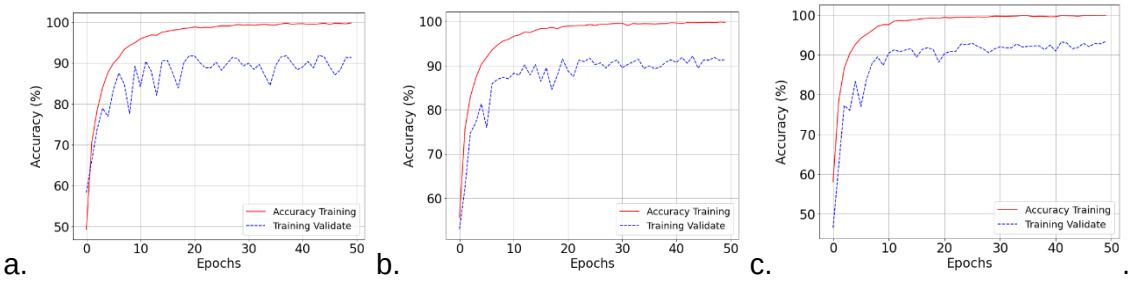
Finalmente, los modelos reducidos se entrenaron con los siguientes hiperparametros: *Optimizer = Adagrad; learning rate = 0,1; loss function = "categorical crossentropy"*, epochs = 50. Los resultados se presentan en la tabla 15 y en las figuras 44 y 45.

Tabla 15. Resultados del entrenamiento de los tres modelos reducidos basados en MobileNet

Modelo	Accuracy	Precision	Recall	F1-score	Parámetros	MB
MobileNet_75	94,87 %	93,15 %	94,33 %	93,69 %	824,201	6,6
MobileNet_50	94,24 %	91,89 %	93,39 %	92,51 %	1,637,257	12,8
MobileNet_25	93,33 %	91,40 %	92,52 %	91,87 %	2,452,873	19,1

Fuente: elaboración propia.

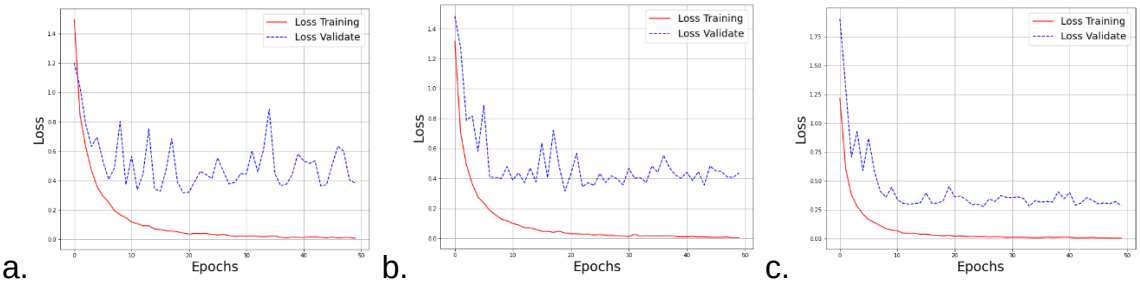
Figura 44. Exactitud (Accuracy) durante el entrenamiento de los modelos de MobileNet reducidos



Nota. a) MobileNet_25, b) MobileNet_50, c) MobileNet_75.

Fuente: elaboración propia.

Figura 45. Pérdida (Loss) durante el entrenamiento de los modelos de MobileNet reducidos



Nota. a) MobileNet_25, b) MobileNet_50, c) MobileNet_75.

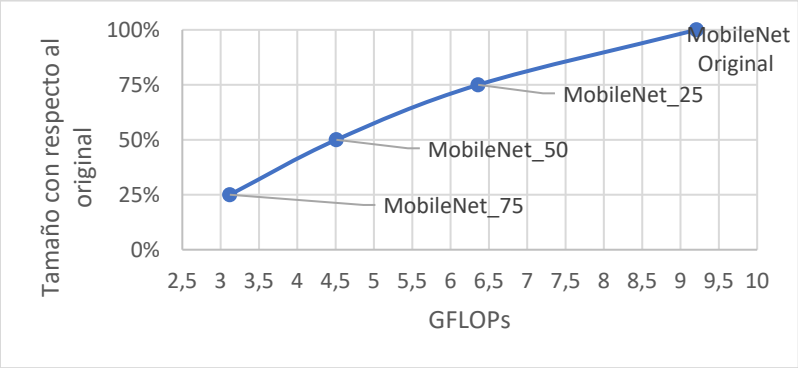
Fuente: elaboración propia.

Los resultados muestran que la versión reducida a un 75 % de MobileNet obtuvo un mejor desempeño que las demás versiones, e incluso obteniendo un *Recall* del 94,33 % siendo la segunda mejor después la versión original que obtuvo 95,93 %, pero con 3.762.056 de parámetros; mientras que la versión reducida lo hizo con 824.201 parámetros. Este resultado es prometedor desde el punto de vista del ahorro en capacidad computacional que puede obtenerse, sin una pérdida significativa en la métrica de *Recall.* Las matrices de confusión y los resultados detallados pueden verse en el anexo E.

7.5.2 Medición de operaciones de punto flotante por segundo (FLOPs).

Como puede observarse en la figura 46 y en la tabla 16, La versión original y las versiones reducidas del modelo propuesto desarrollan cantidades de FLOPs, que están dentro de la capacidad del dispositivo, y que a medida que se reduce el tamaño del modelo deja más capacidad disponible para otras operaciones.

Figura 46. Operaciones de punto flotante medidas en el modelo original y las versiones reducidas



Fuente: elaboración propia.

Tabla 16. Operaciones de punto flotante calculadas para las versiones reducidas del modelo propuesto

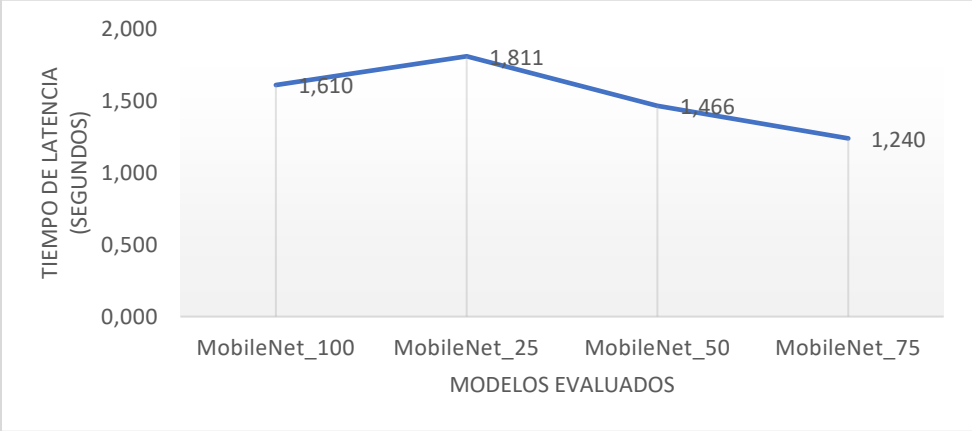
MobileNet	GFLOPs
Original	9,21
MobileNet_25	6,36
MobileNet_50	4,51
MobileNet_75	3,12

Fuente: elaboración propia.

7.5.3 Desempeño con base en el tiempo de procesamiento.

Finalmente se realizó una prueba de desempeño en el dispositivo Raspberry Pi4. La prueba consistió en medir el tiempo que le toma a cada modelo realizar la predicción y ver su rendimiento. Para medir el tiempo de predicción se tomaron cuarenta y cinco imágenes del conjunto de datos de prueba, cinco por cada clase, se midió el tiempo para cada imagen, luego se promediaron los tiempos de latencia para cada modelo y se compararon (figura 47).

Figura 47.Tiempos de latencia en el dispositivo Raspberry Pi4, para las versiones reducidas y original del modelo propuesto tomados en cuarenta y cinco imágenes del conjunto de prueba



Fuente: elaboración propia.

Como puede observarse en la figura anterior, hay un aumento del tiempo de latencia del 12,45 % al realizar una reducción del 25 % de los parámetros del modelo original, y luego se presenta una reducción gradual en los tiempos, primero del 8,97% al reducir el número de parámetros un 50 % y finalmente un 23,02 % de reducción en el tiempo al reducir un 75 % el número de parámetros.

Es necesario realizar más pruebas en trabajos futuros. No es claro por qué hay un aumento en el tiempo en el modelo reducido un 25%, y se necesitan otras pruebas para comprender este comportamiento. Igualmente, para corroborar las disminuciones de tiempo con los otros modelos es necesario realizar pruebas en trabajos futuros que permitan obtener más datos al respecto y ver su significancia desde el punto de vista estadístico.

7.6 Resumen del capítulo

En este capítulo se presentó la implementación del método de clasificación de imágenes en una plataforma de Agricultura Inteligente, utilizando la arquitectura de referencia seleccionada. Igualmente se realizó una evaluación del método en tres condiciones diferentes de reducción de la arquitectura, y los resultados mostraron que estas reducciones tienen desempeños muy cercanos, aunque por debajo comparados con el modelo original. Con este capítulo se da respuesta al tercer objetivo específico de esta tesis doctoral.

En el próximo capítulo se establecen las conclusiones de este trabajo y algunas pautas para la realización de trabajos futuros. Igualmente se exponen algunas limitaciones de este trabajo que deben ser tenidas en cuenta al momento de analizar los resultados.

8 Conclusiones y trabajo futuro

En el capítulo anterior se presentó la implementación del método de clasificación de imágenes en condiciones de trabajo real integrado a una plataforma IoT de Agricultura Inteligente. Lo cual constituye el alcance del tercer objetivo específico de esta tesis doctoral.

Este capítulo está dedicado a presentar algunas conclusiones sobre los objetivos alcanzados, la metodología empleada y otros aspectos relevantes. Además, se presentan líneas de trabajo futuro que se podrían mejorar los resultados que en este trabajo se presentaron. También se muestran algunos limitantes que tiene esta tesis y que deben ser tenidos en cuenta en el análisis de resultados.

Este capítulo está organizado de la siguiente manera: en la primera sección se presentan las conclusiones sobre las aplicaciones de la clasificación de imágenes en plataformas de Agricultura Inteligente. Segundo, se presentan las conclusiones sobre la arquitectura de referencia seleccionada. En tercer lugar, se presentan las conclusiones sobre las modificaciones realizadas al *dataset* original. Luego, se presentan las conclusiones sobre la selección de hiperparámetros por medio de la Optimización Bayesiana en la cuarta sección. Posteriormente, en la quinta sección se presentan las conclusiones sobre el método de clasificación de imágenes seleccionado. En sexto lugar, se presentan las conclusiones sobre la implementación del método de clasificación de imágenes y su evaluación. En séptimo lugar se presentan las posibilidades de desarrollar trabajos futuros en la misma línea de investigación. Luego se presentan las limitaciones de este trabajo en la sección ocho. Y finalmente en la sección nueve, se presentan las contribuciones de esta tesis doctoral.

8.1 Aplicaciones de la clasificación de imágenes en Agricultura Inteligente

- En este trabajo se realizó una revisión sistemática de literatura, con la cual se pudo detectar que el subdominio en el que más se viene aplicando la clasificación de imágenes utilizando plataformas de Agricultura Inteligente es en la detección de problemas fitosanitarios: enfermedades, plagas y malezas. Lo que se busca con esta aplicación es detectar de manera oportuna y con precisión problemas sanitarios en los cultivos, con el fin de tomar acciones rápidas.

- La segunda aplicación en orden de interés detectada es el de monitoreo del desarrollo y vigor de cultivos, el cual se basa en la detección indirecta del contenido de nitrógeno a través de la cuantificación del contenido de clorofila en las hojas. Esto representa una oportunidad para trabajos futuros que busquen la implementación de técnicas con Inteligencia Artificial en el seguimiento al desarrollo vegetativo y productivo de los cultivos.
- Las demás aplicaciones detectadas están aún en etapas muy tempranas de desarrollo, la gestión de sistemas de riego, la detección de intrusos, y el conteo de frutos y plantas; pues la mayoría de estos trabajos presentan resultados de etapas experimentales o de prototipos, lo que representa un campo de estudio amplio para la propuesta de soluciones de implementación.

8.2 Arquitectura de referencia seleccionada

- Las técnicas de Inteligencia Artificial empleadas en las plataformas IoT de Agricultura Inteligente, son muy diversas, en la fase de pre-procesamiento se emplean principalmente técnicas de procesamiento digital de imágenes clásicas, combinadas con algunas técnicas de *Machine Learning,* principalmente *K-means,* para la segmentación de imágenes. En la fase de extracción de características predominan los métodos de *Machine Learning* clásicos. Finalmente, en la fase de clasificación se vienen aplicando técnicas avanzadas como las redes neuronales, pero con limitaciones por capacidad de cómputo al momento de su implementación.
- Con base en los hallazgos de la RSL, se realizó una preselección de arquitecturas de referencia. Luego, con base en cuatro criterios de selección, se compararon las arquitecturas detectadas:
    1. Captura y almacenamiento de grandes volúmenes de datos.
    2. Interoperabilidad entre varios sistemas a nivel de plantación.
    3. Estandarización de formatos para el envío de datos.
    4. Escalabilidad a nivel regional. Como resultado de esta comparación.

Se seleccionó la arquitectura LoRaFARM [28], la cual basa su funcionamiento en las tecnologías LoRa y LoRaWAN, y que, además, cuenta con tres capas:

1. Capa de aplicaciones y servicios.

2. Capa de servidor en la Nube.
3. Capa de nodos y comunicaciones.

8.3 Modificaciones al conjunto de datos original

- Para este trabajo se seleccionó el conjunto de datos PlantVillage, uno de los más referenciados en la literatura; sin embargo, este presenta dos limitantes importantes que introducen un sesgo en la clasificación de las enfermedades: (1) el color del fondo, (2) la forma de las hojas.
- La estrategia propuesta en este trabajo para lograr una clasificación sin sesgo usando el *dataset* PlantVillage se compone de dos elementos: (1) se propone segmentar las imágenes para eliminar el fondo, con la técnica del clasificador de Bayesiano, y (2) se propone recortar las imágenes en segmentos de 85x85 pixeles para basar la clasificación en las características de textura que poseen poder predictivo y así evitar el sesgo debido a la forma de las hojas.
- La estrategia propuesta se justifica en que la forma de las hojas tiene un poder predictivo que puede sesgar el resultado de la clasificación. Si bien la forma de las hojas en las plantas puede estar influenciada por las enfermedades, también puede estar determinada por características fenotípicas y genotípicas que para el *dataset* de estudio son desconocidas.

8.4 Selección de hiperparámetros mediante optimización Bayesiana

- En este trabajo se propone la optimización de hiperparámetros para el entrenamiento de redes neuronales con la técnica de optimización bayesiana. Esta técnica se justifica en que la búsqueda de los hiperparámetros para la red se hace de manera “informada”; es decir, sigue un proceso secuencial iterativo donde se busca encontrar las mejores combinaciones de hiperparámetros, teniendo en cuenta los resultados de búsquedas anteriores, con base en algún algoritmo de optimización. Además, tiene la ventaja de ahorro en cálculos computacionales al realizar la búsqueda en una función sustituta (*surrogate function*) y no en un modelo “caja negra” como son los modelos de redes neuronales.

8.5 Método de clasificación seleccionado

- El método de clasificación de imágenes seleccionado fue MobileNet, porque obtuvo las mejores métricas en comparación con los demás métodos considerados en la evaluación: *Accuracy* de 96,31 % y *F1-score* de 95,72 %. Pero, además, con *Recall* de 95,93 %, el más alto de todos, que para este trabajo se consideró como la métrica de mayor importancia porque tener un *Recall* más alto significa que el método deja pasar por alto un porcentaje menor de individuos enfermos. Y en el caso de la detección de enfermedades es preferible tener algunos falsos positivos que posteriormente puedan ser verificados, que tener falsos negativos que pueden acelerar el crecimiento de una enfermedad sin ser detectados.

8.6 Implementación y evaluación del desempeño del método de clasificación seleccionado

- Para la evaluación del método propuesto se implementó una plataforma IoT de Agricultura Inteligente para la captura de datos en dos invernaderos de 300 m² cada uno, utilizados para la producción de tomate, los cuales están ubicados en el corregimiento de San Cristóbal de la ciudad de Medellín, a una altura sobre el nivel del mar de 2.200 m aproximadamente.
- La plataforma de Agricultura Inteligente incluyó dos tipos de nodos para la captura de datos y una estación meteorológica. Los nodos tipo A (N-A) se diseñaron para la captura de temperatura, humedad relativa y luminosidad; por otro lado, los nodos tipo B (N-B) se diseñaron para la captura de contenido de dióxido de carbono en el aire, la conductividad eléctrica del suelo, el volumen de agua aplicado en el sistema de riego y la captura de imágenes para la detección de enfermedades. En este último fue donde se integró el método de clasificación propuesto en este trabajo.
- El resultado final fue la integración de clasificación de imágenes para la detección de enfermedades con los demás tipos de datos en una plataforma IoT de Agricultura Inteligente. Esta integración se basó en la estrategia de realizar todos los cálculos de clasificación en el Borde, para lo cual se utilizó como procesador un dispositivo Raspberry Pi4. Esta estrategia fue seleccionada con base en los hallazgos de la RSL,

debido a que la transmisión de imágenes para realizar la clasificación en un servidor alojado en la Nube, no es siempre necesaria en este tipo de soluciones.

- Para evaluar la robustez del método y verificar si podía desempeñarse en dispositivos de menor capacidad de cómputo, se realizó una prueba experimental en la que se redujo el tamaño de la red neuronal original MobileNet en 25 %, 50 % y 75 %, obteniendo un *Accuracy* de 93,33 %, 94,24 % y 94,87 % respectivamente. También se obtuvo un *Recall* de 92,52 % 93,39 % 94,33 %, respectivamente. Estos resultados parecen ser contraintuitivos, pues a medida que se disminuyó el tamaño de la red, mejoraron las métricas. Las causas de este comportamiento no se investigaron en este trabajo y tendrán que ser el objetivo de trabajos futuros.
- Con el fin de evaluar la capacidad de cómputo que deben tener los dispositivos en los que se implemente el método seleccionado y sus versiones reducidas en futuros trabajos, se calcularon las operaciones de punto flotante por segundo o FLOPs por sus siglas en inglés, obteniendo como resultado 9,21 GFLOPs para la versión original, y 6,36, 4,51 y 3,12 GFLOPs para las versiones reducidas 25, 50 y 75 %, respectivamente. Por otro lado, la capacidad del procesador utilizado en la implementación de este trabajo, una Raspberry Pi4, tiene una capacidad de 13.5 GFLOPs; es decir, que la implementación de las versiones reducidas es posible inclusive en dispositivos de menor capacidad de cómputo.
- Con el fin de evaluar la latencia del método original y sus versiones reducidas, también se realizó una prueba experimental en la que se midió el tiempo de inferencia de cada uno en el dispositivo Raspberry Pi4. En promedio se obtuvieron tiempos de latencia de 1,61 segundos para la versión original, 1,81, 1,46 y 1,24 segundos para las versiones reducidas 25, 50 y 75 %, respectivamente. El aumento del tiempo en la versión reducida un 25 % parece ser contraintuitiva, porque en teoría al reducir la cantidad de parámetros debería reducir el tiempo de inferencia, como efectivamente sucedió con las versiones reducidas 50 % y 75 %. Para encontrar las causas de este comportamiento deberán realizarse pruebas más exhaustivas en trabajos futuros.

8.7 Trabajo futuro

- Para futuros trabajos deben realizarse pruebas con diferentes tamaños de imágenes que permitan mejorar la resolución para una mejor clasificación con base en la textura causada por las enfermedades en las hojas de las plantas.
- También deberán realizarse investigaciones tendientes a mejorar el desempeño con base en las métricas *Precision y Recall,* porque estas dan una mayor confiablidad en la identificación de posibles enfermedades oportunamente y de manera precisa.
- Es necesario realizar más investigaciones en la optimización de otros hiperparámetros como la cantidad de capas de la red neuronal, número de filtros, entre otros.
- Deben realizarse trabajos de investigación sobre diferentes estrategias de implementación de estos modelos en campo para conocer el desempeño en condiciones reales con limitaciones de energía y capacidad de cómputo diferentes.

8.8 Discusión

8.8.1 Objetivo específico No.1.

- Identificar una arquitectura de referencia para una plataforma IoT de agricultura inteligente, que permita integrar un método de clasificación de imágenes digitales empleando técnicas de Inteligencia Artificial.
    1. En el capítulo 4 se detalla el proceso mediante el cual se revisaron las estrategias desde el punto de vista arquitectónico que se detectaron en la literatura. En la RSL se pudo detectar que existen tres tipos de estrategias para integrar la clasificación de imágenes en las plataformas IoT de Agricultura Inteligente: (1) procesamiento de imágenes en el Borde, (2) procesamiento de imágenes en la Nube y (3) procesamiento mixto (en el Borde y en la Nube). La principal ventaja del primero es la obtención de los datos en tiempo real, y su principal desventaja es el costo de implementación. La principal ventaja del segundo es la capacidad de procesamiento de imágenes de alta resolución, y su principal desventaja es la necesidad de conectividad con grandes anchos de banda. Finalmente, la estrategia mixta puede tener grandes beneficios en inversión de infraestructura, pero su implementación puede ser más compleja.

2. Con base en lo anterior se eligió la estrategia de cómputo en el borde, y se seleccionaron las arquitecturas con este tipo de estrategia, con suficiente información que permitiera su implementación. Luego se compararon bajo los criterios de selección y se seleccionó la arquitectura llamada LoRaFarm, dando cumplimiento al objetivo específico No.1

8.8.2 Objetivo específico No.2.

- Seleccionar un método de clasificación de imágenes digitales empleando técnicas de Inteligencia Artificial, para integrarlo a una plataforma IoT de agricultura inteligente.
  1. En el capítulo 6 se detalla el proceso mediante el cual fue elegido el método de clasificación de imágenes. Su elección se basó en tres criterios de selección: (1) por su tamaño, modelos con menos de 5 millones de parámetros fueron preseleccionados; (2) por su desempeño, modelos que tuvieron un desempeño igual o mayor al 90% de *Accuracy* con el dataset PlantVillage en trabajos previos fueron tenidos en cuenta y (3) redes neuronales del estado del arte creadas para trabajar en contextos con limitaciones en capacidad de cómputo.
  2. Luego de esto fue elegida la red neuronal convolucional MobileNet, por presentar mejores medidas en su desempeño que las otras redes evaluadas, dando cumplimiento al objetivo específico No.2

8.8.3 Objetivo específico No.3.

- Evaluar el desempeño del método de clasificación de imágenes digitales empleando técnicas de Inteligencia Artificial, para integrarlo a una plataforma IoT de AI.
  1. El capítulo 7 presenta todo el proceso de implementación del método de clasificación de imágenes con la arquitectura de referencia LoRaFARM. En la cual se tuvieron en cuenta las tecnologías reportadas en la RSL en cada una de las estrategias arquitectónicas. La plataforma piloto fue instalada en dos unidades productivas, que cuentan con invernaderos que producen hortalizas, principalmente cultivo de tomate.
  2. Luego de la implementación se evaluó el desempeño del método integrado a la plataforma. La manera de evaluar este desempeño fue reduciendo la arquitectura original, para simular tres escenarios en los que por capacidades de computo debería

reducirse el tamaño de la red neuronal. Las métricas usadas fueron las operaciones de punto flotante por segundo y el tiempo de latencia, resultando en desempeños muy similares a la red original, aunque menores. Esto mostró que es posible su implementación, sin perdida significativa en sus métricas *Precision, Accuracy* y *Recall.* Con esto se dio cumplimiento al objetivo específico No.3

Con estos resultados se considera cumplido el objetivo general de esta tesis:

- Proponer un método de clasificación de imágenes digitales, empleando técnicas de Inteligencia Artificial, para integrarlo a una plataforma IoT de agricultura inteligente.

8.9 Limitaciones

- Este trabajo fue realizado con una parte del *dataset* PlantVillage, que corresponde a las imágenes en enfermedades que afectan cultivos de tomate. No fue del alcance de este estudio la investigación con las demás especies de plantas que hay en el *dataset*, como por ejemplo cultivos de papa, pimiento o manzana, que son muy importantes económicamente en diferentes lugares del mundo.
- Una de las principales limitaciones de este trabajo es que los resultados no se comparan con los obtenidos en trabajos anteriores. Sin embargo, en el capítulo 6 se hace referencia a investigaciones que han entrenado modelos con el conjunto de datos original.
- Una limitación importante que fue observada en la fase de evaluación se debe a que algunas enfermedades de los cultivos de tomate en la región de implementación son llamadas con nombres diferentes, por lo cual debería hacerse una actualización del dataset con los nombres de las clases que son comunes entre los agricultores de la zona para futuras implementaciones.

8.10 Contribuciones

Esta tesis doctoral hizo varias contribuciones al estudio de la integración de procesos de clasificación de imágenes en plataformas de Agricultura Inteligente.

- Se generó un conjunto de datos a partir del *dataset* PlantVillage, que es uno de los más referenciados en la literatura sobre clasificación de imágenes en contextos de agricultura.

Este nuevo conjunto soluciona dos de las problemáticas del *dataset* original reportadas en la literatura: a) un sesgo debido al color y la iluminación en el fondo y b) un sesgo debido a la forma de las hojas en las plantas. Este nuevo dataset está disponible a la comunidad que trabaja en temas similares en:
https://www.kaggle.com/datasets/jfrestrepoa/tomato-tiles

- Se identificó una arquitectura de referencia que permite la integración de procesos de clasificación de imágenes. Esta arquitectura emplea tecnologías que permiten su implementación en contextos con limitaciones de capacidad computacional y fuentes de energía.
- Se seleccionó un método de redes neuronales artificiales que combina eficiencia en uso de recursos computacionales y efectividad en la clasificación de imágenes en contextos de agricultura inteligente. Los resultados de esta selección fueron publicados en el *journal: Agriculture* de MDPI.
  https://www.mdpi.com/2077-0472/12/11/1964
- Se mostró la viabilidad de la implementación del método seleccionado en un contexto real de trabajo. Esto permitió conocer mejor las limitaciones y posibles caminos en la búsqueda de soluciones, que se adapten mejor a los contextos rurales, para estudios posteriores. La implementación se dio en el marco del proyecto HERMES No.202010030247. *Implementación de un Modelo de Smart Farming, Mediante la Integración de Tecnologías 4.0 e Inteligencia Artificial, para Pequeños Productores de Hortalizas Bajo Invernadero,* financiado por la Universidad Nacional de Colombia Sede Medellín, y desarrollado por el grupo de investigación GIDIA.

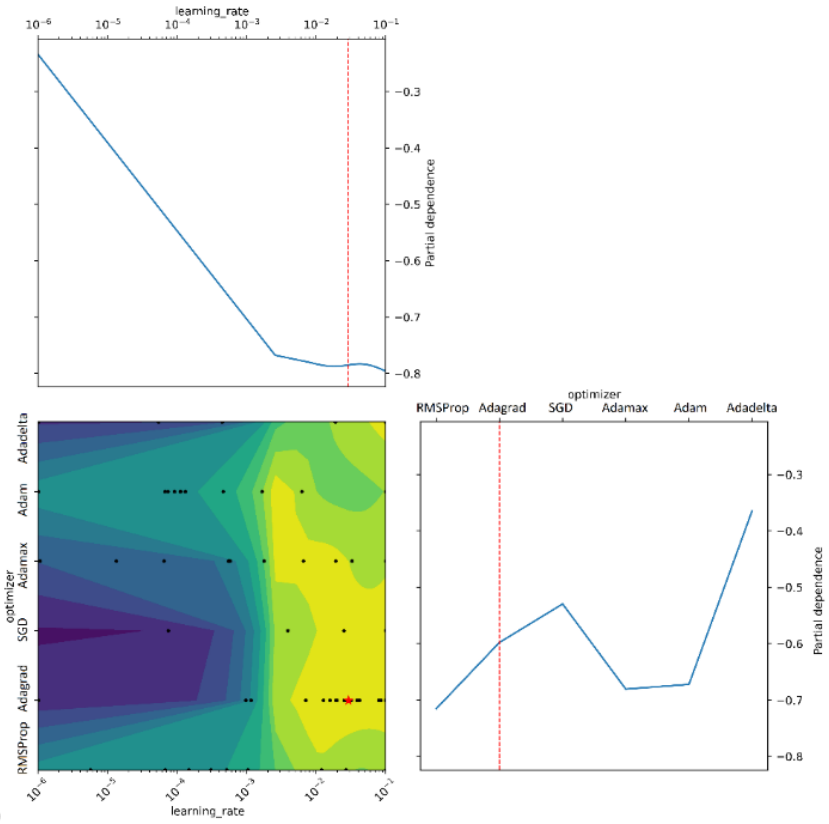
9 Anexos

Anexo A. Algoritmos de búsqueda en las bases de datos utilizadas para la revisión sistemática de literatura

Librería digital	Grupo	Algoritmo
Scopus, Web of Science, Springer Link	Grupos 1, 2 y 3	TITLE-ABS-KEY ((agriculture OR farming OR "smart farming" OR "smart agriculture" OR "intelligent agriculture" OR "digital farming" OR "precision agriculture" OR "agriculture smart system" OR "farm management system") AND (architecture OR iot OR platform OR "internet of things" OR "wireless sensor network" OR "cloud computing" OR "edge computing") AND (image OR "image classification" OR "image recognition" OR "object detection" OR "computer vision" OR "machine vision")) AND LIMIT-TO (PUBYEAR , 2022) OR LIMIT-TO (PUBYEAR , 2021) OR LIMIT-TO (PUBYEAR , 2020) OR LIMIT-TO (PUBYEAR , 2019) OR LIMIT-TO (PUBYEAR , 2018)) AND (LIMIT-TO (DOCTYPE , "ar") OR LIMIT-TO (DOCTYPE , "re")) AND (LIMIT-TO (LANGUAGE , "English")) AND (LIMIT-TO (SUBJAREA , "COMP") OR LIMIT-TO (SUBJAREA , "ENGI") OR LIMIT-TO (SUBJAREA , "AGRI")) AND (LIMIT-TO (SRCTYPE , "j"))
IEEE Xplore	Grupos 1, 2 y 3	(("All Metadata":agriculture OR "All Metadata":farming OR "All Metadata":“smart farming” OR "All Metadata":“smart agriculture” OR "All Metadata":“intelligent agriculture” OR "All Metadata":“digital farming” OR "All Metadata":“precision agriculture” OR "All Metadata":“agriculture smart system” OR "All Metadata":“farm management system”) AND ("All Metadata":architecture OR "All Metadata":IoT OR "All Metadata":platform OR "All Metadata":“internet of things” OR "All Metadata":“wireless sensor network” OR "All Metadata":“cloud computing” OR "All Metadata":“edge computing”) AND ("All Metadata":image OR "All Metadata":“image classification” OR "All Metadata":“image recognition” OR "All Metadata":“object detection” OR "All Metadata":“computer vision” OR "All Metadata":“machine vision”)) Filters Applied: Journals, Articles2018 – 2022

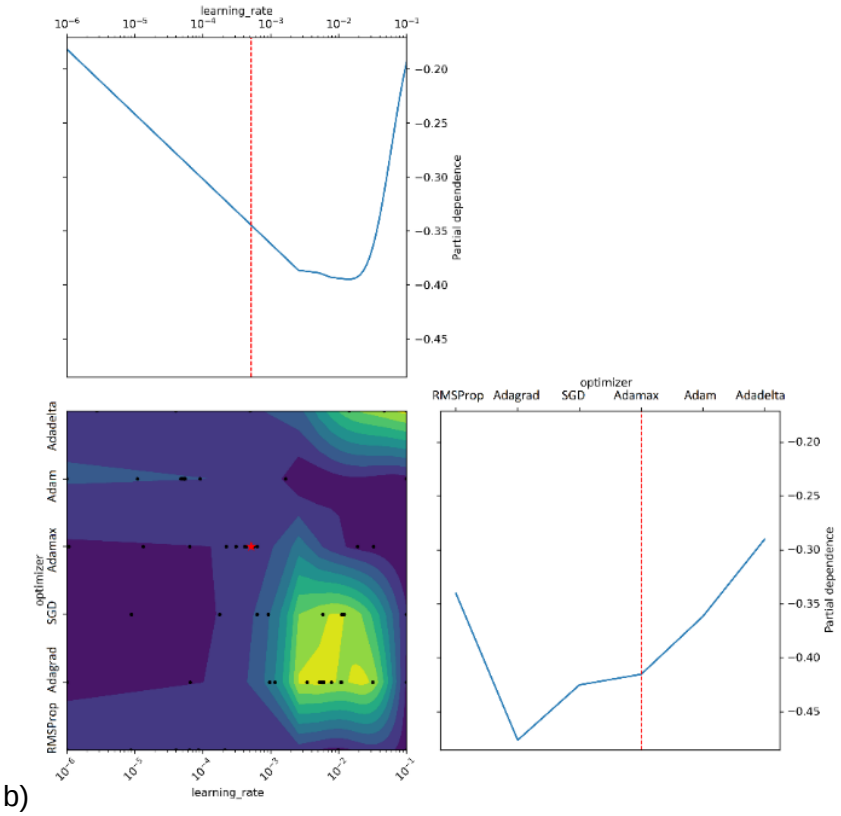
Fuente: elaboración propia.

Anexo B. Resultados de la optimización Bayesiana para los modelos: a) SqueezeNet, b) NasNetMobile, y c) MobileNetV2

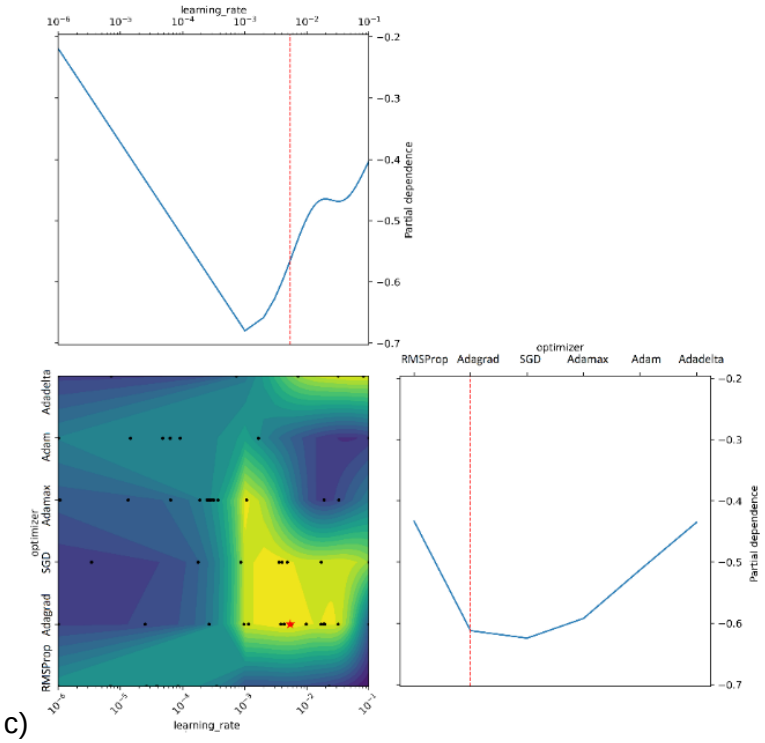


a)

b)

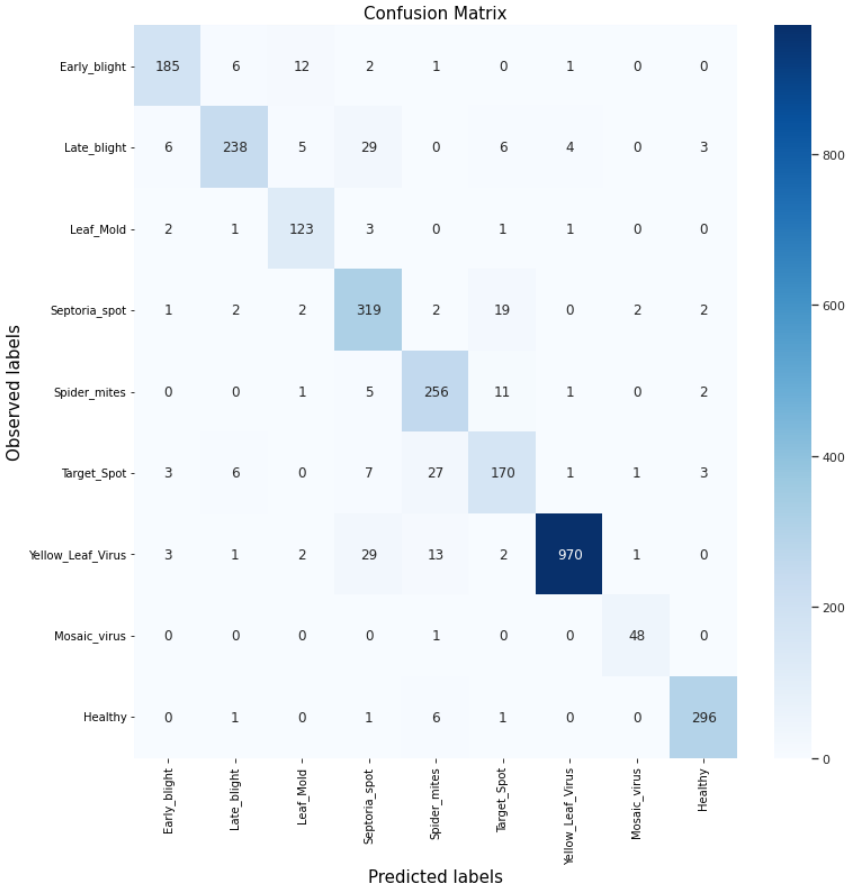


c)



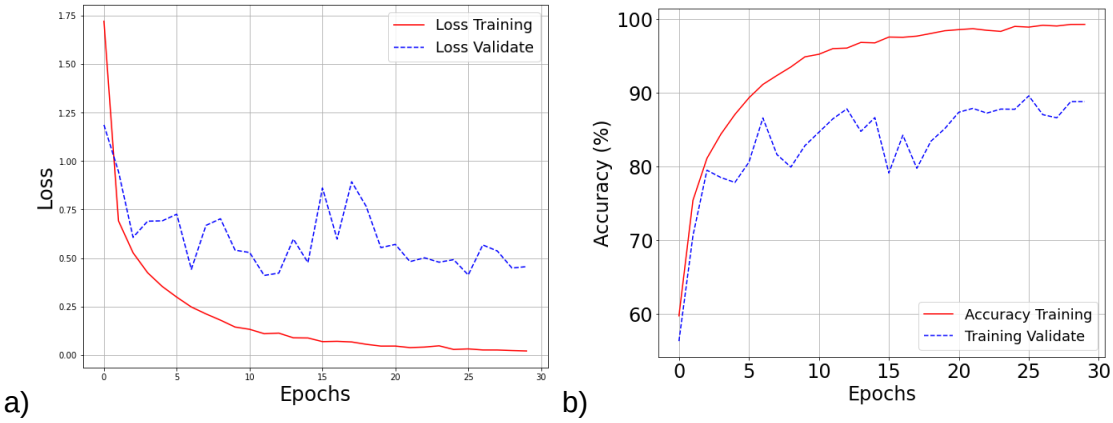
Fuente: elaboración propia.

Anexo C. Matriz de confusión del entrenamiento para el modelo ShuffleNet



Fuente: elaboración propia.

Anexo D. Resultados del entrenamiento para el modelo ShuffleNet. a) Pérdida, b) Exactitud



Fuente: elaboración propia.

Anexo E. Métricas obtenidas para los modelos de redes neuronales evaluados con la matriz de confusión. a) MobileNet, b) MobileNetV2, c) SqueezeNet, d) NasNetMobile, e) ShuffleNet

a)

MobileNet	precision	recall	f1-score	support
Early_blight	0.9755	0.9614	0.9684	207
Late_blight	0.9500	0.9141	0.9317	291
Leaf_Mold	0.9489	0.9924	0.9701	131
Septoria_spot	0.9384	0.9599	0.9490	349
Spider_mites	0.9526	0.9457	0.9491	276
Target_Spot	0.9209	0.9083	0.9145	218
Yellow_Leaf_Virus	0.9931	0.9824	0.9877	1021
mosaic_virus	0.9796	0.9796	0.9796	49
Healthy	0.9408	0.9902	0.9649	305
accuracy			0.9631	2847
macro avg	0.9555	0.9593	0.9572	2847
weighted avg	0.9634	0.9631	0.9631	2847

b)

MobileNetV2	precision	recall	f1-score	support
Early_blight	0.9660	0.9614	0.9637	207
Late_blight	0.9565	0.9072	0.9312	291
Leaf_Mold	0.9348	0.9847	0.9591	131
Septoria_spot	0.9386	0.9198	0.9291	349
Spider_mites	0.8845	0.9710	0.9257	276
Target_Spot	0.8768	0.8165	0.8456	218
Yellow_Leaf_Virus	0.9950	0.9706	0.9826	1021
Mosaic_virus	0.8571	0.9796	0.9143	49
Healthy	0.9021	0.9672	0.9335	305
accuracy			0.9459	2847
macro avg	0.9235	0.9420	0.9317	2847
weighted avg	0.9472	0.9459	0.9459	2847

c)

SqueezeNet	precision	recall	f1-score	support
Early_blight	0.9554	0.9324	0.9438	207
Late_blight	0.9444	0.9347	0.9396	291
Leaf_Mold	0.9394	0.9466	0.9430	131
Septoria_spot	0.9477	0.9341	0.9408	349
Spider_mites	0.8754	0.9420	0.9075	276
Target_Spot	0.9239	0.8349	0.8771	218
Yellow_Leaf_Virus	0.9891	0.9785	0.9838	1021
Mosaic_virus	0.9592	0.9592	0.9592	49
Healthy	0.9238	0.9934	0.9573	305
accuracy			0.9505	2847
macro avg	0.9398	0.9395	0.9391	2847
weighted avg	0.9512	0.9505	0.9504	2847

d)

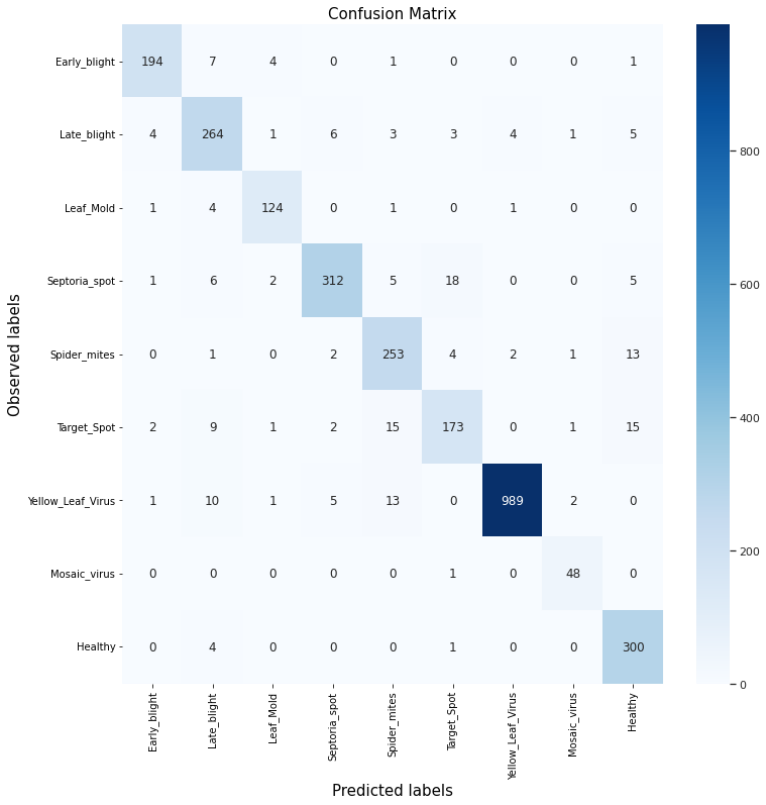
NasNetMobile	precision	recall	f1-score	support
Early_blight	0.9735	0.8889	0.9293	207
Late_blight	0.9418	0.8900	0.9152	291
Leaf_Mold	0.9478	0.9695	0.9585	131
Septoria_spot	0.8727	0.9628	0.9155	349
Spider_mites	0.9559	0.9420	0.9489	276
Target_Spot	0.9043	0.8670	0.8852	218
Yellow_Leaf_Virus	0.9941	0.9824	0.9882	1021
Mosaic_virus	0.8033	1.0000	0.8909	49
Healthy	0.9521	0.9770	0.9644	305
accuracy			0.9501	2847
macro avg	0.9273	0.9422	0.9329	2847
weighted avg	0.9519	0.9501	0.9503	2847

e)

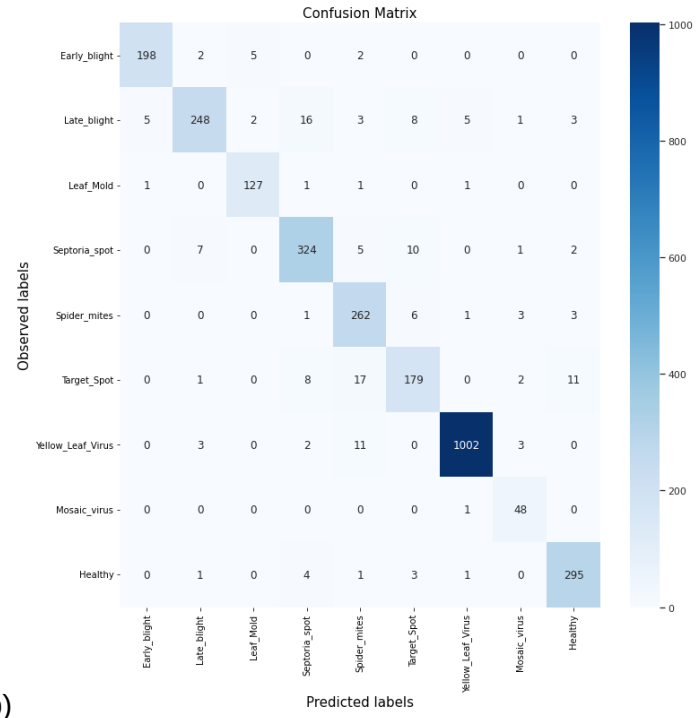
ShuffleNet	precision	recall	f1-score	support
Early_blight	0.9250	0.8937	0.9091	207
Late_blight	0.9333	0.8179	0.8718	291
Leaf_Mold	0.8483	0.9389	0.8913	131
Septoria_spot	0.8076	0.9140	0.8575	349
Spider_mites	0.8366	0.9275	0.8797	276
Target_Spot	0.8095	0.7798	0.7944	218
Yellow_Leaf_Virus	0.9918	0.9500	0.9705	1021
Mosaic_virus	0.9231	0.9796	0.9505	49
Healthy	0.9673	0.9705	0.9689	305
accuracy			0.9150	2847
macro avg	0.8936	0.9080	0.8993	2847
weighted avg	0.9190	0.9150	0.9156	2847

Fuente: elaboración propia.

Anexo F. Matrices de confusión obtenidas para los modelos reducidos basados en MobileNet. a) MobileNet_25, b) MobileNet_50, y c) MobileNet_75

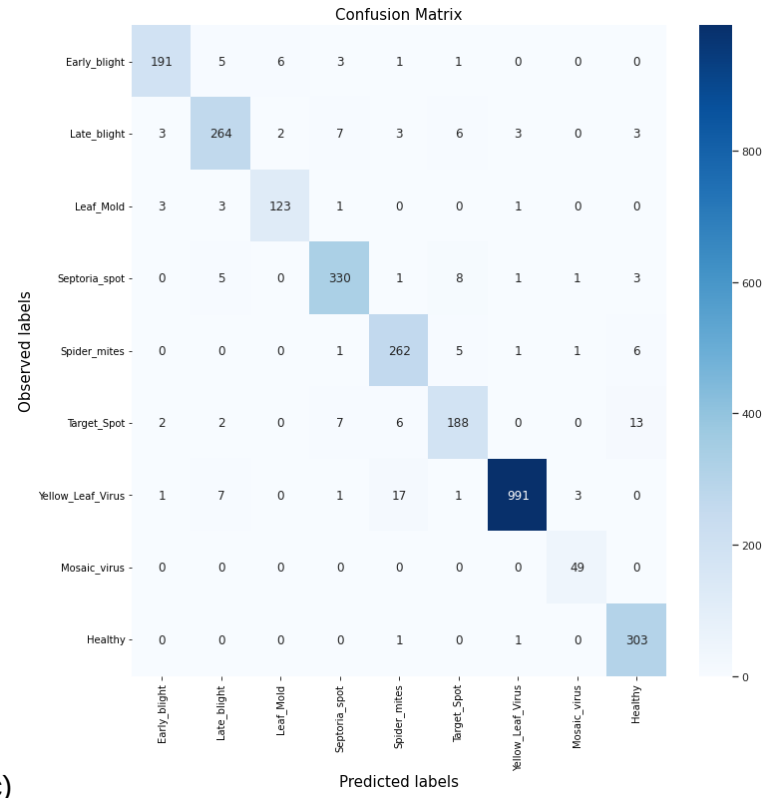


a)



b)

c)



Fuente: elaboración propia

Anexo G. Métricas obtenidas para los modelos reducidos basados en MobileNet. a) MobileNet_25, b) MobileNet_50, y c) MobileNet_75

a)

MobileNet_25	precision	recall	f1-score	support
Early_blight	0.9557	0.9372	0.9463	207
Late_blight	0.8656	0.9072	0.8859	291
Leaf_Mold	0.9323	0.9466	0.9394	131
Septoria_spot	0.9541	0.8940	0.9231	349
Spider_mites	0.8694	0.9167	0.8924	276
Target_Spot	0.8650	0.7936	0.8278	218
Yellow_Leaf_Virus	0.9930	0.9687	0.9807	1021
Mosaic_virus	0.9057	0.9796	0.9412	49
Healthy	0.8850	0.9836	0.9317	305
accuracy			0.9333	2847
macro avg	0.9140	0.9252	0.9187	2847
weighted avg	0.9348	0.9333	0.9333	2847

b)

MobileNet_50	precision	recall	f1-score	support
Early_blight	0.9706	0.9565	0.9635	207
Late_blight	0.9466	0.8522	0.8969	291
Leaf_Mold	0.9478	0.9695	0.9585	131
Septoria_spot	0.9101	0.9284	0.9191	349
Spider_mites	0.8675	0.9493	0.9066	276
Target_Spot	0.8689	0.8211	0.8443	218
Yellow_Leaf_Virus	0.9911	0.9814	0.9862	1021
Mosaic_virus	0.8276	0.9796	0.8972	49
Healthy	0.9395	0.9672	0.9532	305
accuracy			0.9424	2847
macro avg	0.9189	0.9339	0.9251	2847
weighted avg	0.9435	0.9424	0.9423	2847

c)

MobileNet_50	precision	recall	f1-score	support
Early_blight	0.9706	0.9565	0.9635	207
Late_blight	0.9466	0.8522	0.8969	291
Leaf_Mold	0.9478	0.9695	0.9585	131
Septoria_spot	0.9101	0.9284	0.9191	349
Spider_mites	0.8675	0.9493	0.9066	276
Target_Spot	0.8689	0.8211	0.8443	218
Yellow_Leaf_Virus	0.9911	0.9814	0.9862	1021
Mosaic_virus	0.8276	0.9796	0.8972	49
Healthy	0.9395	0.9672	0.9532	305
accuracy			0.9424	2847
macro avg	0.9189	0.9339	0.9251	2847
weighted avg	0.9435	0.9424	0.9423	2847

Fuente: elaboración propia.

Anexo H. Implementación en Tensorflow de arquitecturas para los modelos reducidos basados en MobileNet. a) MobileNet_25, b) MobileNet_50, y c) MobileNet_75)

a)

```
input = Input(shape=(width_shape, height_shape, depth))

x = Conv2D(filters=32, kernel_size=3, strides=2, padding='same')(input)
x = BatchNormalization()(x)
x = ReLU()(x)
x = mobilenet_block(x, filters = 64, strides = 1)
x = mobilenet_block(x, filters = 128, strides = 2)
x = mobilenet_block(x, filters = 128, strides = 1)
x = mobilenet_block(x, filters = 256, strides = 2)
x = mobilenet_block(x, filters = 256, strides = 1)
x = mobilenet_block(x, filters = 512, strides = 2)
for _ in range(6):
  x = mobilenet_block(x,filters = 512, strides = 1)
x = mobilenet_block(x, filters = 1024, strides = 2)
x = GlobalAveragePooling2D()(x)

output = Dense(units=num_classes, activation='softmax')(x)
```

b)
```
input = Input(shape=(width_shape, height_shape, depth))

x = Conv2D(filters=32, kernel_size=3, strides=2, padding='same')(input)
x = BatchNormalization()(x)
x = ReLU()(x)

x = mobilenet_block(x, filters = 64, strides = 1)
x = mobilenet_block(x, filters = 128, strides = 2)
x = mobilenet_block(x, filters = 128, strides = 1)
x = mobilenet_block(x, filters = 256, strides = 2)
x = mobilenet_block(x, filters = 256, strides = 1)
x = mobilenet_block(x, filters = 512, strides = 2)
for _ in range(3):
  x = mobilenet_block(x,filters = 512, strides = 1)
x = mobilenet_block(x, filters = 1024, strides = 2)
x = GlobalAveragePooling2D()(x)

output = Dense(units=num_classes, activation='softmax')(x)
```

c)
```
input = Input(shape=(width_shape, height_shape, depth))

x = Conv2D(filters=32, kernel_size=3, strides=2, padding='same')(input)
x = BatchNormalization()(x)
x = ReLU()(x)

x = mobilenet_block(x, filters = 64, strides = 1)
x = mobilenet_block(x, filters = 128, strides = 2)
x = mobilenet_block(x, filters = 128, strides = 1)
x = mobilenet_block(x, filters = 256, strides = 2)
x = mobilenet_block(x, filters = 256, strides = 1)
x = mobilenet_block(x, filters = 512, strides = 2)
x = mobilenet_block(x, filters = 512, strides = 1)
x = mobilenet_block(x, filters = 512, strides = 2)
x = GlobalAveragePooling2D()(x)

output = Dense(units=num_classes, activation='softmax')(x)
```

Fuente: elaboración propia.

10 Referencias

[1] L. Trivelli, A. Apicella, F. Chiarello, R. Rana, and A. Tarabella, “From precision agriculture to Industry 4.0 the agrifood sector,” *British Food Journal*, vol. 121, no. 8, pp. 1730–1743, 2019, doi: 10.1108/BFJ-11-2018-0747.

[2] F. Mazzetto, R. Gallo, and P. Sacco, “Reflections and methodological proposals to treat the concept of ‘information precision’ in smart agriculture practices,” *Sensors (Switzerland)*, vol. 20, no. 10, pp. 1–27, 2020, doi: 10.3390/s20102847.

[3] Y. Lu, “Industry 4.0: A survey on technologies, applications and open research issues,” *J Ind Inf Integr*, vol. 6, pp. 1–10, 2017, doi: 10.1016/j.jii.2017.04.005.

[4] 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. URL:http://dx.doi.org/10.1016/j.agsy.2017.01.023

[5] A. A. R. Madushanki, M. N. Halgamuge, W. A. H. S. Wirasagoda, and A. Syed, “Adoption of the Internet of Things (IoT) in agriculture and smart farming towards urban greening: A review,” *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 4, pp. 11–28, 2019.

[6] M. L. C. Delos Santos, L. L. Lacatan, and F. G. Balazon, “Cloud-based smart farming for crop production suitability using wireless sensor technology,” *Test Engineering and Management*, vol. 81, no. 11–12, pp. 5043–5052, 2019.

[7] S. P. Jaiswal, V. S. Bhadoria, A. Agrawal, and H. Ahuja, “Internet of things (Iot) for smart agriculture and farming in developing nations,” *International Journal of Scientific and Technology Research*, vol. 8, no. 12, pp. 1049–1056, 2019.

[8] S. Terence and G. Purushothaman, “Systematic review of Internet of Things in smart farming,” *Transactions on Emerging Telecommunications Technologies*, vol. 31, no. 6, 2020, doi: 10.1002/ett.3958.

[9] K. Sekaran, M. N. Meqdad, P. Kumar, S. Rajan, and S. Kadry, “Smart agriculture management system using internet of things,” *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 18, no. 3, pp. 1275–1284, 2020, doi: 10.12928/TELKOMNIKA.v18i3.14029.

[10] S. K. S. K. Verma, M. Rajesh, and R. Vincent, “Smart-farming using internet of things,” *J Comput Theor Nanosci*, vol. 17, no. 1, pp. 172–176, 2020, doi: 10.1166/jctn.2020.8646.

[11] K. Ravindranath, C. S. Bhargavi, K. Samaikya Reddy, and M. Sai Chandana, “Cloud of things for smart agriculture,” *International Journal of Innovative Technology and Exploring Engineering*, vol. 8, no. 6, pp. 30–33, 2019.

[12] P. Lashitha Vishnu Priya, N. Sai Harshith, and N. V. K. V. K. Ramesh, “Smart agriculture monitoring system using IoT,” *International Journal of Engineering and Technology(UAE)*, vol. 7, no. 4, pp. 308–311, 2018, doi: 10.17148/IJARCCE.2019.8419.

[13] E. Navarro, N. Costa, and A. Pereira, “A systematic review of iot solutions for smart farming,” *Sensors (Switzerland)*, vol. 20, no. 15, pp. 1–29, 2020, doi: 10.3390/s20154231.

[14] M. Ayaz, M. Ammad-Uddin, Z. Sharif, A. Mansour, and E.-H. M. Aggoune, “Internet-of-Things (IoT)-based smart agriculture: Toward making the fields talk,” *IEEE Access*, vol. 7, pp. 129551–129583, 2019, doi: 10.1109/ACCESS.2019.2932609.

[15] J. M. Talavera *et al.*, “Review of IoT applications in agro-industrial and environmental fields,” *Computers and Electronics in Agriculture*, vol. 142. 2017. doi: 10.1016/j.compag.2017.09.015.

[16] M. Jankowski, D. Gündüz, and K. Mikolajczyk, “Deep Joint Transmission-Recognition for Power-Constrained Iot Devices,” *ArXiv*, pp. 1–10, 2020.

[17] H. Tian, T. Wang, Y. Liu, X. Qiao, and Y. Li, “Computer vision technology in agricultural automation —A review,” *Information Processing in Agriculture*, vol. 7, no. 1, pp. 1–19, 2020, doi: 10.1016/j.inpa.2019.09.006. URL:https://doi.org/10.1016/j.inpa.2019.09.006

[18] A. O. Adedoja, P. A. Owolawi, T. Mapayi, and C. Tu, “Intelligent Mobile Plant Disease Diagnostic System Using NASNet-Mobile Deep Learning,” *IAENG Int J Comput Sci*, vol. 49, no. 1, pp. 216–231, 2022.

[19] P. Angin, M. H. Anisi, F. Göksel, C. Gürsoy, and A. Büyükgülcü, “Agrilora: A digital twin framework for smart agriculture,” *J Wirel Mob Netw Ubiquitous Comput Dependable Appl*, vol. 11, no. 4, pp. 77–96, 2020, doi: 10.22667/JOWUA.2020.12.31.077.

[20] G. Delnevo, R. Girau, C. Ceccarini, and C. Prandi, “A Deep Learning and Social IoT Approach for Plants Disease Prediction Toward a Sustainable Agriculture,” *IEEE Internet Things J*, vol. 9, no. 10, pp. 7243–7250, May 2022, doi: 10.1109/JIOT.2021.3097379.

[21] N. Farooqui, A. Mishra, and R. Mehra, “IOT based Automated Greenhouse Using Machine Learning Approach,” *International Journal of Intelligent Systems and Applications in Engineering*, vol. 10, no. 2, pp. 226–231, 2022, [Online]. Available: https://ijisae.org/index.php/IJISAE/article/view/1522. URL:https://ijisae.org/index.php/IJISAE/article/view/1522

[22] R. Gajjar, N. Gajjar, V. J. Thakor, N. P. Patel, and S. Ruparelia, “Real-time detection and identification of plant leaf diseases using convolutional neural networks on an embedded platform,” *Visual Computer*, 2021, doi: 10.1007/s00371-021-02164-9. URL:https://doi.org/10.1007/s00371-021-02164-9

[23] V. Gonzalez-Huitron, J. A. León-Borges, A. E. Rodriguez-Mata, L. E. Amabilis-Sosa, B. Ramírez-Pereda, and H. Rodriguez, “Disease detection in tomato leaves via CNN with lightweight architectures implemented in Raspberry Pi 4,” *Comput Electron Agric*, vol. 181, no. January, 2021, doi: 10.1016/j.compag.2020.105951.

[24] M. Ji, J. Yoon, J. Choo, M. Jang, and A. Smith, “LoRa-based Visual Monitoring Scheme for Agriculture IoT,” *SAS 2019 - 2019 IEEE Sensors Applications Symposium, Conference Proceedings*, 2019, doi: 10.1109/SAS.2019.8706100.

[25] N. Islam, B. Ray, and F. Pasandideh, “IoT Based Smart Farming: Are the LPWAN Technologies Suitable for Remote Communication?,” *Proceedings - 2020 IEEE International Conference on Smart Internet of Things, SmartIoT 2020*, no. October, pp. 270–276, 2020, doi: 10.1109/SmartIoT49966.2020.00048.

[26] K. Mekki, E. Bajic, F. Chaxel, and F. Meyer, “Overview of Cellular LPWAN Technologies for IoT Deployment: Sigfox, LoRaWAN, and NB-IoT,” *2018 IEEE International Conference on Pervasive*

Computing and Communications Workshops, PerCom Workshops 2018, no. January 2019, pp. 197–202, 2018, doi: 10.1109/PERCOMW.2018.8480255.

[27] A. Carlsson, I. Kuzminykh, R. Franksson, and A. Liljegren, “Measuring a LoRa Network: Performance, Possibilities and Limitations,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11118 LNCS, pp. 116–128, 2018, doi: 10.1007/978-3-030-01168-0_11.

[28] G. Codeluppi, A. Cilfone, L. Davoli, and G. Ferrari, “LoraFarM: A LoRaWAN-based smart farming modular IoT architecture,” *Sensors (Switzerland)*, vol. 20, no. 7, 2020, doi: 10.3390/s20072028.

[29] T. U. Pathan and S. Chakole, “Sensor based smart farming and plant diseases monitoring,” *Int J Eng Adv Technol*, vol. 8, no. 2, pp. 442–446, 2019.

[30] Y. Zhong, J. Gao, Q. Lei, and Y. Zhou, “A vision-based counting and recognition system for flying insects in intelligent agriculture,” *Sensors (Switzerland)*, vol. 18, no. 5, 2018, doi: 10.3390/s18051489.

[31] I. Sa *et al.*, “WeedNet: Dense Semantic Weed Classification Using Multispectral Images and MAV for Smart Farming,” *IEEE Robot Autom Lett*, vol. 3, no. 1, pp. 588–595, 2018, doi: 10.1109/LRA.2017.2774979.

[32] D. Koc-San, S. Selim, N. Aslan, and B. T. San, “Automatic citrus tree extraction from UAV images and digital surface models using circular Hough transform,” *Comput Electron Agric*, vol. 150, no. February, pp. 289–301, 2018, doi: 10.1016/j.compag.2018.05.001. URL:https://doi.org/10.1016/j.compag.2018.05.001

[33] R. G. R. G. De Luna, E. P. E. P. Dadios, A. A. A. A. Bandala, and R. R. P. R. R. P. Vicerra, “Tomato growth stage monitoring for smart farm using deep transfer learning with machine learning-based maturity grading,” *Agrivita*, vol. 42, no. 1, pp. 24–36, 2020, doi: 10.17503/agrivita.v42i1.2499.

[34] O. E. Apolo-Apolo, J. Martínez-Guanter, G. Egea, P. Raja, and M. Pérez-Ruiz, “Deep learning techniques for estimation of the yield and size of citrus fruits using a UAV,” *European Journal of Agronomy*, vol. 115, Apr. 2020, doi: 10.1016/j.eja.2020.126030.

[35] H. Yalcin, “Phenology recognition using deep learning: DeepPheno,” *26th IEEE Signal Processing and Communications Applications Conference, SIU 2018*, pp. 1–4, 2018, doi: 10.1109/SIU.2018.8404165.

[36] A. Khanna and S. Kaur, “Evolution of Internet of Things (IoT) and its significant impact in the field of Precision Agriculture,” *Comput Electron Agric*, vol. 157, no. December 2018, pp. 218–231, 2019, doi: 10.1016/j.compag.2018.12.039.

[37] X. Pham and M. Stack, “How data analytics is transforming agriculture,” *Bus Horiz*, vol. 61, no. 1, pp. 125–133, 2018, doi: 10.1016/j.bushor.2017.09.011.

[38] A. Kamilaris, A. Kartakoullis, and F. X. Prenafeta-Boldú, “A review on the practice of big data analysis in agriculture,” *Computers and Electronics in Agriculture*, vol. 143. Elsevier B.V., pp. 23–37, Dec. 2017. doi: 10.1016/j.compag.2017.09.037.

[39] O. Elijah, T. A. Rahman, I. Orikumhi, C. Y. Leow, and M. N. Hindia, “An Overview of Internet of Things (IoT) and Data Analytics in Agriculture: Benefits and Challenges,” *IEEE Internet Things J*, vol. 5, no. 5, pp. 3758–3773, 2018, doi: 10.1109/JIOT.2018.2844296.

[40] W. Feng, W. Ju, A. Li, W. Bao, and J. Zhang, “High-Efficiency Progressive Transmission and Automatic Recognition of Wildlife Monitoring Images with WISNs,” *IEEE Access*, vol. 7, pp. 161412–161423, 2019, doi: 10.1109/ACCESS.2019.2951596.

[41] S. W. Lee and H. Y. Kim, “An energy-efficient low-memory image compression system for multimedia IoT products,” *EURASIP J Image Video Process*, vol. 2018, no. 1, 2018, doi: 10.1186/s13640-018-0333-3.

[42] G. Campobello and A. Segreto, “A low complexity image compression algorithm for IoT multimedia applications,” *European Signal Processing Conference*, vol. 2019-Septe, pp. 1–5, 2019, doi: 10.23919/EUSIPCO.2019.8902678.

[43] Z. Liu *et al.*, “DeepN-JPEG: A deep neural network favorable JPEG-based image compression framework,” *Proc Des Autom Conf*, vol. Part F1377, no. 3, pp. 1–6, 2018, doi: 10.1145/3195970.3196022.

[44] M. J. Page *et al.*, “The PRISMA 2020 statement: An updated guideline for reporting systematic reviews,” *The BMJ*, vol. 372, 2021, doi: 10.1136/bmj.n71.

[45] S. Wolfert, D. Goense, and C. A. G. Sorensen, “A future internet collaboration platform for safe and healthy food from farm to fork,” *Annual SRII Global Conference, SRII*, no. April, pp. 266–273, 2014, doi: 10.1109/SRII.2014.47.

[46] S. P. Mohanty, D. P. Hughes, and M. Salathé, “Using deep learning for image-based plant disease detection,” *Front Plant Sci*, vol. 7, no. September, 2016, doi: 10.3389/fpls.2016.01419.

[47] P. Wspanialy and M. Moussa, “A detection and severity estimation system for generic diseases of tomato greenhouse plants,” *Comput Electron Agric*, vol. 178, no. August, p. 105701, 2020, doi: 10.1016/j.compag.2020.105701. URL:https://doi.org/10.1016/j.compag.2020.105701

[48] D. J. A. Rustia and T. te Lin, “An IoT-based wireless imaging and sensor node system for remote greenhouse pest monitoring,” *Chem Eng Trans*, vol. 58, no. January, pp. 601–606, 2017, doi: 10.3303/CET1758101.

[49] A. A. Sarangdhar, P. V. R. Pawar, and A. B. Blight, “Machine Learning Regression Technique for using IoT,” *International Conference on Electronics, Communication and Aerospace Technology ICECA 2017*, pp. 449–454, 2017.

[50] A. Thorat, S. Kumari, and N. D. Valakunde, “An IoT based smart solution for leaf disease detection,” *2017 International Conference on Big Data, IoT and Data Science, BID 2017*, vol. 2018-Janua, no. December, pp. 193–198, 2018, doi: 10.1109/BID.2017.8336597.

[51] N. Kitpo and M. Inoue, “Early rice disease detection and position mapping system using drone and IoT architecture,” *Proceedings - 12th SEATUC Symposium, SEATUC 2018*, no. September 2019, 2018, doi: 10.1109/SEATUC.2018.8788863.

[52] D. Brunelli, A. Albanese, D. Acunto, and M. Nardello, “E nergy N eutral M achine L earning B ased I o T D evice for P est D etection in P recision A griculture,” no. December 2019, pp. 2019–2022, 2019.

[53] R. D. Devi, S. A. Nandhini, R. Hemalatha, and S. Radha, “IoT enabled efficient detection and classification of plant diseases for agricultural applications,” *2019 International Conference on*

Wireless Communications, Signal Processing and Networking, WiSPNET 2019, pp. 447–451, 2019, doi: 10.1109/WiSPNET45539.2019.9032727.

[54] C. J. Chen, Y. Y. Huang, Y. S. Li, C. Y. Chang, and Y. M. Huang, "An AIoT Based Smart Agricultural System for Pests Detection," *IEEE Access*, vol. 8, pp. 180750–180761, 2020, doi: 10.1109/ACCESS.2020.3024891.

[55] S. Zhang, W. Huang, and H. Wang, "Crop disease monitoring and recognizing system by soft computing and image processing models," *Multimed Tools Appl*, vol. 79, no. 41–42, pp. 30905–30916, 2020, doi: 10.1007/s11042-020-09577-z.

[56] R. C. Gonzalez and R. E. Woods, *Digital Image Processing, 4e*, 4°. Pearson Education Limited, 2018.

[57] N. Otsu, "A Tlreshold Selection Method from Gray-Level Histograms," *IEEE Transaction on Systems, Man and Cybernetics*, vol. 20, no. 1, pp. 62–66, 1979.

[58] A. Géron, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow : concepts, tools, and techniques to build intelligent systems*, Second Edition. O'Reilly Media, Inc., 2019. Accessed: Oct. 02, 2022. [Online]. Available: http://oreilly.com/catalog/errata.csp?isbn=9781492032649. URL:http://oreilly.com/catalog/errata.csp?isbn=9781492032649

[59] M. Gehan and N. Fahlgren, "PlantCV." Missouri, United States., pp. 1–14, 2022.

[60] A. G. Howard *et al.*, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," Apr. 2017, [Online]. Available: http://arxiv.org/abs/1704.04861. URL:http://arxiv.org/abs/1704.04861

[61] A. Howard *et al.*, "Searching for mobileNetV3," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2019-Octob, pp. 1314–1324, 2019, doi: 10.1109/ICCV.2019.00140.

[62] M. Tan and Q. v. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," *36th International Conference on Machine Learning, ICML 2019*, vol. 2019-June, pp. 10691–10700, 2019.

[63] M. Tan *et al.*, "Mnasnet: Platform-aware neural architecture search for mobile," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2019-June, pp. 2815–2823, 2019, doi: 10.1109/CVPR.2019.00293.

[64] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size," pp. 1–13, 2016, [Online]. Available: http://arxiv.org/abs/1602.07360. URL:http://arxiv.org/abs/1602.07360

[65] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 6848–6856, 2018, doi: 10.1109/CVPR.2018.00716.

[66] Á. B. Jiménez, J. L. Lázaro, and J. R. Dorronsoro, "Finding optimal model parameters by deterministic and annealed focused grid search," *Neurocomputing*, vol. 72, no. 13–15, pp. 2824–2832, 2009, doi: 10.1016/j.neucom.2008.09.024.

[67] J. Bergstra, J. B. Ca, and Y. B. Ca, "Random Search for Hyper-Parameter Optimization Yoshua Bengio," 2012. [Online]. Available: http://scikit-learn.sourceforge.net. URL:http://scikit-learn.sourceforge.net.

[68] J. Wu, X. Y. Chen, H. Zhang, L. D. Xiong, H. Lei, and S. H. Deng, “Hyperparameter optimization for machine learning models based on Bayesian optimization,” *Journal of Electronic Science and Technology*, vol. 17, no. 1, pp. 26–40, Mar. 2019, doi: 10.11989/JEST.1674-862X.80904120.

[69] N. M. Aszemi and P. D. D. Dominic, “Hyperparameter Optimization in Convolutional Neural Network using Genetic Algorithms,” 2019. [Online]. Available: www.ijacsa.thesai.org. URL:www.ijacsa.thesai.org

[70] M. Sokolova and G. Lapalme, “A systematic analysis of performance measures for classification tasks,” *Inf Process Manag*, vol. 45, no. 4, pp. 427–437, Jul. 2009, doi: 10.1016/j.ipm.2009.03.002.

[71] A. Tharwat, “Classification assessment methods,” *Applied Computing and Informatics*, vol. 17, no. 1, pp. 168–192, 2018, doi: 10.1016/j.aci.2018.08.003.

[72] “IEEE Standard for Binary Floating-Point Arithmetic Standards Committee of the IEEE Computer Society IEEE Standards Board American National Standards Institute,” 1985.

[73] “The GFLOPS/W of the various machines in the VMW Research Group,” 2022. https://web.eece.maine.edu/~vweaver/group/green_machines.html (accessed Dec. 22, 2022). URL:https://web.eece.maine.edu/~vweaver/group/green_machines.html

[74] S. Li, L. Da Xu, and S. Zhao, “The internet of things: a survey,” *Information Systems Frontiers*, vol. 17, no. 2, pp. 243–259, 2015, doi: 10.1007/s10796-014-9492-7.

[75] A. Kaloxylos *et al.*, “Farm management systems and the Future Internet era,” *Comput Electron Agric*, vol. 89, pp. 130–144, 2012, doi: 10.1016/j.compag.2012.09.002.

[76] A. Kaloxylos *et al.*, “The Use of Future Internet Technologies in the Agriculture and Food Sectors: Integrating the Supply Chain,” *Procedia Technology*, vol. 8, no. Haicta, pp. 51–60, 2013, doi: 10.1016/j.protcy.2013.11.009.

[77] C. Li and B. Niu, “Design of smart agriculture based on big data and Internet of things,” *Int J Distrib Sens Netw*, vol. 16, no. 5, 2020, doi: 10.1177/1550147720917065.

[78] J. Muangprathub, N. Boonnam, S. Kajornkasirat, N. Lekbangpong, A. Wanichsombat, and P. Nillaor, “IoT and agriculture data analysis for smart farm,” *Comput Electron Agric*, vol. 156, pp. 467–474, 2019, doi: 10.1016/j.compag.2018.12.011.

[79] C. C. Baseca, S. Sendra, J. Lloret, and J. Tomas, “A smart decision system for digital farming,” *Agronomy*, vol. 9, no. 5, 2019, doi: 10.3390/agronomy9050216.

[80] M. Colezea, G. Musat, F. Pop, C. Negru, A. Dumitrascu, and M. Mocanu, “CLUeFARM: Integrated web-service platform for smart farms,” *Comput Electron Agric*, vol. 154, pp. 134–154, Nov. 2018, doi: 10.1016/j.compag.2018.08.015.

[81] D. Budaev *et al.*, “Conceptual design of smart farming solution for precise agriculture,” *International Journal of Design and Nature and Ecodynamics*, vol. 13, no. 3, pp. 307–314, 2018, doi: 10.2495/DNE-V13-N3-307-314.

[82] A. Tzounis, N. Katsoulas, T. Bartzanas, and C. Kittas, “Internet of Things in agriculture, recent advances and future challenges,” *Biosyst Eng*, vol. 164, pp. 31–48, 2017, doi: 10.1016/j.biosystemseng.2017.09.007.

[83] M. De Donno, K. Tange, and N. Dragoni, “Foundations and Evolution of Modern Computing Paradigms: Cloud, IoT, Edge, and Fog,” *IEEE Access*, vol. 7, pp. 150936–150948, 2019, doi: 10.1109/ACCESS.2019.2947652.

[84] IOF - European Union, “IOF-2020 (Internet of Food) REFERENCE ARCHITECTURE FOR INTEROPERABILITY , REPLICABILITY AND REUSE,” EU, 2020.

[85] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Cla, “Middleware for internet of things: A survey,” *IEEE Internet Things J*, vol. 3, no. 1, pp. 70–95, 2016, doi: 10.1109/JIOT.2015.2498900.

[86] U. Barman, R. D. Choudhury, D. Sahu, and G. G. Barman, “Comparison of convolution neural networks for smartphone image based real time classification of citrus leaf disease,” *Comput Electron Agric*, vol. 177, no. August, p. 105661, 2020, doi: 10.1016/j.compag.2020.105661. URL:https://doi.org/10.1016/j.compag.2020.105661

[87] S. S. Chouhan, U. P. Singh, and S. Jain, “Automated Plant Leaf Disease Detection and Classification Using Fuzzy Based Function Network,” *Wirel Pers Commun*, vol. 121, no. 3, pp. 1757–1779, 2021, doi: 10.1007/s11277-021-08734-3. URL:https://doi.org/10.1007/s11277-021-08734-3

[88] N. Fatima, S. A. Siddiqui, and A. Ahmad, “IoT-based Smart Greenhouse with Disease Prediction using Deep Learning,” *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 7, pp. 113–121, 2021, doi: 10.14569/IJACSA.2021.0120713.

[89] D. Gao, Q. Sun, B. Hu, and S. Zhang, “A framework for agricultural pest and disease monitoring based on internet-of-things and unmanned aerial vehicles,” *Sensors (Switzerland)*, vol. 20, no. 5, 2020, doi: 10.3390/s20051487.

[90] P. Joshi, D. Das, V. Udutalapally, M. K. Pradhan, and S. Misra, “RiceBioS: Identification of Biotic Stress in Rice Crops Using Edge-as-a-Service,” *IEEE Sens J*, vol. 22, no. 5, pp. 4616–4624, 2022, doi: 10.1109/JSEN.2022.3143950.

[91] S. S. Lee, Y. N. Jeong, S. R. Son, and B. K. Lee, “A self-predictable crop yield platform (SCYP) based on crop diseases using deep learning,” *Sustainability (Switzerland)*, vol. 11, no. 13, 2019, doi: 10.3390/su11133637.

[92] A. Mellit, M. Benghanem, O. Herrak, and A. Messalaoui, “Design of a novel remote monitoring system for smart greenhouses using the internet of things and deep convolutional neural networks,” *Energies (Basel)*, vol. 14, no. 16, 2021, doi: 10.3390/en14165045.

[93] M. Mishra, P. Choudhury, and B. Pati, “Modified ride-NN optimizer for the IoT based plant disease detection,” *J Ambient Intell Humaniz Comput*, vol. 12, no. 1, pp. 691–703, 2021, doi: 10.1007/s12652-020-02051-6. URL:https://doi.org/10.1007/s12652-020-02051-6

[94] G. Nagasubramanian, R. K. Sakthivel, R. Patan, M. Sankayya, M. Daneshmand, and A. H. Gandomi, “Ensemble Classification and IoT-Based Pattern Recognition for Crop Disease Monitoring System,” *IEEE Internet Things J*, vol. 8, no. 16, pp. 12847–12854, 2021, doi: 10.1109/JIOT.2021.3072908.

[95] S. Aasha Nandhini, R. Hemalatha, S. Radha, and K. Indumathi, “Web Enabled Plant Disease Detection System for Agricultural Applications Using WMSN,” *Wirel Pers Commun*, vol. 102, no. 2, pp. 725–740, 2018, doi: 10.1007/s11277-017-5092-4. URL:https://doi.org/10.1007/s11277-017-5092-4

[96] A. Picon, M. Seitz, A. Alvarez-Gila, P. Mohnke, A. Ortiz-Barredo, and J. Echazarra, "Crop conditional Convolutional Neural Networks for massive multi-crop plant disease classification over cell phone acquired images taken on real field conditions," *Comput Electron Agric*, vol. 167, no. September, p. 105093, 2019, doi: 10.1016/j.compag.2019.105093. URL:https://doi.org/10.1016/j.compag.2019.105093

[97] B. S. Rishiikeshwer, T. A. Shriram, J. S. Raju, M. Hari, B. Santhi, and G. R. Brindha, "Farmer-friendly mobile application for automated leaf disease detection of real-time augmented data set using convolution neural networks," *Journal of Computer Science*, vol. 16, no. 2, pp. 158–166, 2020, doi: 10.3844/JCSSP.2020.158.166.

[98] S. Ramesh and B. Rajaram, "Iot based crop disease identification system using optimization techniques," *ARPN Journal of Engineering and Applied Sciences*, vol. 13, no. 4, pp. 1392–1395, 2018.

[99] R. Rathinam, P. Kasinathan, U. Govindarajan, V. K. Ramachandaramurthy, U. Subramaniam, and S. Garrido, "Cybernetics approaches in intelligent systems for crops disease detection with the aid of IoT," *International Journal of Intelligent Systems*, vol. 36, no. 11, pp. 6550–6580, 2021, doi: 10.1002/int.22560.

[100] K. K. Sarma, K. K. Das, V. Mishra, S. Bhuiya, and D. Kaplun, "Learning Aided System for Agriculture Monitoring Designed Using Image Processing and IoT-CNN," *IEEE Access*, vol. 10, pp. 41525–41536, 2022, doi: 10.1109/ACCESS.2022.3167061.

[101] V. Udutalapally, S. P. Mohanty, V. Pallagani, and V. Khandelwal, "SCrop: A Novel Device for Sustainable Automatic Disease Prediction, Crop Selection, and Irrigation in Internet-of-Agro-Things for Smart Agriculture," *IEEE Sens J*, vol. 21, no. 16, pp. 17525–17538, 2021, doi: 10.1109/JSEN.2020.3032438.

[102] Z. Zinonos, S. Gkelios, A. F. Khalifeh, D. G. Hadjimitsis, Y. S. Boutalis, and S. A. Chatzichristofis, "Grape Leaf Diseases Identification System Using Convolutional Neural Networks and LoRa Technology," *IEEE Access*, vol. 10, pp. 122–133, 2022, doi: 10.1109/ACCESS.2021.3138050.

[103] I. Dasgupta, J. Saha, P. Venkatasubbu, and P. Ramasubramanian, "AI Crop Predictor and Weed Detector Using Wireless Technologies: A Smart Application for Farmers," *Arab J Sci Eng*, vol. 45, no. 12, pp. 11115–11127, 2020, doi: 10.1007/s13369-020-04928-2. URL:https://doi.org/10.1007/s13369-020-04928-2

[104] A. Coletta, N. Bartolini, G. Maselli, A. Kehs, P. McCloskey, and D. P. Hughes, "Optimal Deployment in Crowdsensing for Plant Disease Diagnosis in Developing Countries," *IEEE Internet Things J*, vol. 9, no. 9, pp. 6359–6373, 2022, doi: 10.1109/JIOT.2020.3002332.

[105] 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," *Information Processing in Agriculture*, vol. 9, no. 1, pp. 38–47, 2022, doi: 10.1016/j.inpa.2021.01.004.

[106] R. Kamath, M. Balachandra, and S. Prabhu, "Raspberry Pi as Visual Sensor Nodes in Precision Agriculture: A Study," *IEEE Access*, vol. 7, pp. 45110–45122, 2019, doi: 10.1109/ACCESS.2019.2908846.

[107] A. Albanese, M. Nardello, and D. Brunelli, "Automated Pest Detection with DNN on the Edge for Precision Agriculture," *IEEE J Emerg Sel Top Circuits Syst*, vol. 11, no. 3, pp. 458–467, 2021, doi: 10.1109/JETCAS.2021.3101740.

[108] V. Attada and S. Katta, “A methodology for automatic detection and classification of pests using optimized SVM in greenhouse crops,” *Int J Eng Adv Technol*, vol. 8, no. 6, pp. 1485–1491, 2019, doi: 10.35940/ijeat.F8133.088619.

[109] M. E. Karar, F. Alsunaydi, S. Albusaymi, and S. Alotaibi, “A new mobile application of agricultural pests recognition using deep learning in cloud computing system,” *Alexandria Engineering Journal*, vol. 60, no. 5, pp. 4423–4432, 2021, doi: 10.1016/j.aej.2021.03.009. URL:https://doi.org/10.1016/j.aej.2021.03.009

[110] M. A. Lopez, J. M. Lombardo, M. López, D. Álvarez, S. Velasco, and S. Terrón, “Traceable Ecosystem and Strategic Framework for the Creation of an Integrated Pest Management System for Intensive Farming,” *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 6, no. 3, p. 47, 2020, doi: 10.9781/ijimai.2020.08.004.

[111] B. Ramalingam *et al.*, “Remote insects trap monitoring system using deep learning framework and iot,” *Sensors (Switzerland)*, vol. 20, no. 18, pp. 1–17, 2020, doi: 10.3390/s20185280.

[112] I. Salehin *et al.*, “IFSG: Intelligence agriculture crop-pest detection system using IoT automation system,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 24, no. 2, pp. 1091–1099, 2021, doi: 10.11591/ijeecs.v24.i2.pp1091-1099.

[113] M. J. Schrader, P. Smytheman, E. H. Beers, and L. R. Khot, “An Open-Source Low-Cost Imaging System Plug-In for Pheromone Traps Aiding Remote Insect Pest Population Monitoring in Fruit Crops,” *Machines*, vol. 10, no. 1, 2022, doi: 10.3390/machines10010052.

[114] P. Thakare and V. Ravi Sankar, “Advanced pest detection strategy using hybrid optimization tuned deep convolutional neural network,” *Journal of Engineering, Design and Technology*, 2022, doi: 10.1108/JEDT-09-2021-0488.

[115] G. Gagliardi *et al.*, “An internet of things solution for smart agriculture,” *Agronomy*, vol. 11, no. 11, Nov. 2021, doi: 10.3390/agronomy11112140.

[116] L. Paleari *et al.*, “Estimating plant nitrogen content in tomato using a smartphone,” *Field Crops Res*, vol. 284, Aug. 2022, doi: 10.1016/j.fcr.2022.108564.

[117] K. R. Prilianti, S. Anam, T. H. P. Brotosudarmo, and A. Suryanto, “Real-time assessment of plant photosynthetic pigment contents with an artificial intelligence approach in a mobile application,” *Journal of Agricultural Engineering*, vol. 51, no. 4, pp. 220–228, 2020, doi: 10.4081/jae.2020.1082.

[118] J. Santhosh, P. Balamurugan, G. Arulkumaran, M. Baskar, and R. Velumani, “Image Driven Multi Feature Plant Management with FDE Based Smart Agriculture with Improved Security in Wireless Sensor Networks,” *Wirel Pers Commun*, no. 0123456789, 2021, doi: 10.1007/s11277-021-08710-x. URL:https://doi.org/10.1007/s11277-021-08710-x

[119] L. K. S. Tolentino *et al.*, “Yield evaluation of brassica rapa, Lactuca Sativa, and brassica Integrifolia using image processing in an IoT-based Aquaponics with temperature-controlled greenhouse,” *Agrivita*, vol. 42, no. 3, pp. 393–410, 2020, doi: 10.17503/agrivita.v42i3.2600.

[120] S. Park and J. Kim, “Design and implementation of a hydroponic strawberry monitoring and harvesting timing information supporting system based on nano ai-cloud and iot-edge,” *Electronics (Switzerland)*, vol. 10, no. 12, 2021, doi: 10.3390/electronics10121400.

[121] A. Triantafyllou, P. Sarigiannidis, and S. Bibi, “Precision agriculture: A remote sensing monitoring system architecture,” *Information (Switzerland)*, vol. 10, no. 11, Nov. 2019, doi: 10.3390/info10110348.

[122] Á. L. Perales Gómez, P. E. López-de-Teruel, A. Ruiz, G. García-Mateos, G. Bernabé García, and F. J. García Clemente, “FARMIT: continuous assessment of crop quality using machine learning and deep learning techniques for IoT-based smart farming,” *Cluster Comput*, vol. 25, no. 3, pp. 2163–2178, 2022, doi: 10.1007/s10586-021-03489-9.

[123] R. Chandra *et al.*, “Democratizing Data-Driven Agriculture Using Affordable Hardware,” *IEEE Micro*, vol. 42, no. 1, pp. 69–77, 2022, doi: 10.1109/MM.2021.3134743.

[124] T. C. Hsu, H. Yang, Y. C. Chung, and C. H. Hsu, “A Creative IoT agriculture platform for cloud fog computing,” *Sustainable Computing: Informatics and Systems*, vol. 28, p. 100285, 2020, doi: 10.1016/j.suscom.2018.10.006. URL:https://doi.org/10.1016/j.suscom.2018.10.006

[125] J. D. Wahl and J. X. Zhang, “Development and power characterization of an IoT network for agricultural imaging applications,” *Journal of Advances in Information Technology*, vol. 12, no. 3, pp. 214–219, 2021, doi: 10.12720/jait.12.3.214-219.

[126] A. Oliveira *et al.*, “Iot sensing platform as a driver for digital farming in rural africa,” *Sensors (Switzerland)*, vol. 20, no. 12, pp. 1–25, Jun. 2020, doi: 10.3390/s20123511.

[127] P. K. Sethy, S. K. Behera, N. Kannan, S. Narayanan, and C. Pandey, “Smart paddy field monitoring system using deep learning and IoT,” *Concurr Eng Res Appl*, vol. 29, no. 1, pp. 16–24, Mar. 2021, doi: 10.1177/1063293X21988944.

[128] C. C. Baseca, S. Sendra, J. Lloret, and J. Tomas, “A smart decision system for digital farming,” *Agronomy*, vol. 9, no. 5, May 2019, doi: 10.3390/agronomy9050216.

[129] E. A. Abioye *et al.*, “IoT-based monitoring and data-driven modelling of drip irrigation system for mustard leaf cultivation experiment,” *Information Processing in Agriculture*, vol. 8, no. 2, pp. 270–283, Jun. 2021, doi: 10.1016/j.inpa.2020.05.004.

[130] S. AlZu’bi, B. Hawashin, M. Mujahed, Y. Jararweh, and B. B. Gupta, “An efficient employment of internet of multimedia things in smart and future agriculture,” *Multimed Tools Appl*, vol. 78, no. 20, pp. 29581–29605, Oct. 2019, doi: 10.1007/s11042-019-7367-0.

[131] S. R. Barkunan, V. Bhanumathi, and J. Sethuram, “Smart sensor for automatic drip irrigation system for paddy cultivation,” *Computers and Electrical Engineering*, vol. 73, pp. 180–193, Jan. 2019, doi: 10.1016/j.compeleceng.2018.11.013.

[132] A. Mateo-Aroca, G. García-Mateos, A. Ruiz-Canales, J. M. Molina-García-Pardo, and J. M. Molina-Martínez, “Remote image capture system to improve aerial supervision for precision irrigation in agriculture,” *Water (Switzerland)*, vol. 11, no. 2, Feb. 2019, doi: 10.3390/w11020255.

[133] P. Ramya, T. R. Annapoorneswari, G. V. Bindu, N. V. Meghana, and U. Roshni, “Solar powered automated irrigation system with plant health indication using iot,” *International Journal of Recent Technology and Engineering*, vol. 8, no. 2 Special Issue 11, pp. 60–64, Sep. 2019, doi: 10.35940/ijrte.B1011.0982S1119.

[134] L. Wang *et al.*, “A Smart Droplet Detection Approach with Vision Sensing Technique for Agricultural Aviation Application,” *IEEE Sens J*, vol. 21, no. 16, pp. 17508–17516, Aug. 2021, doi: 10.1109/JSEN.2021.3056957.

[135] D. Adami, M. O. Ojo, and S. Giordano, “Design, Development and Evaluation of an Intelligent Animal Repelling System for Crop Protection Based on Embedded Edge-AI,” *IEEE Access*, vol. 9, pp. 132125–132139, 2021, doi: 10.1109/ACCESS.2021.3114503.

[136] S. Angadi and R. Katagall, “Agrivigilance: A Security System For Intrusion Detection In Agriculture Using Raspberry Pi And Opencv,” *INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH*, vol. 8, no. 11, 2019, [Online]. Available: www.ijstr.org. URL:www.ijstr.org

[137] V. Nithin, S. Mishra, P. Devarubiny, and S. Muthulakshmi, “Iot enabled farming assist and security using machine learning,” *ARPN Journal of Engineering and Applied Sciences*, vol. 14, no. 9, pp. 1809–1819, 2019.

[138] A. V. Prabu, G. S. Kumar, S. Rajasoundaran, P. P. Malla, S. Routray, and A. Mukherjee, “Internet of things-based deeply proficient monitoring and protection system for crop field,” *Expert Syst*, vol. 39, no. 5, Jun. 2022, doi: 10.1111/exsy.12876.

[139] M. R. Ramli, P. T. Daely, D.-S. Kim, and J. M. Lee, “IoT-based adaptive network mechanism for reliable smart farm system,” *Comput Electron Agric*, vol. 170, 2020, doi: 10.1016/j.compag.2020.105287.

[140] S. K. Behera, P. K. Sethy, S. K. Sahoo, S. Panigrahi, and S. C. Rajpoot, “On-tree fruit monitoring system using IoT and image analysis,” *Concurr Eng Res Appl*, vol. 29, no. 1, pp. 6–15, Mar. 2021, doi: 10.1177/1063293X20988395.

[141] S. Kamal *et al.*, “IOT Automation with Segmentation Techniques for Detection of Plant Seedlings in Agriculture,” *Wirel Commun Mob Comput*, vol. 2022, 2022, doi: 10.1155/2022/6466555.

[142] V. Mazzia, A. Khaliq, F. Salvetti, and M. Chiaberge, “Real-time apple detection system using embedded systems with hardware accelerators: An edge AI application,” *IEEE Access*, vol. 8, pp. 9102–9114, 2020, doi: 10.1109/ACCESS.2020.2964608.

[143] T. Misra *et al.*, “Web-SpikeSegNet: Deep Learning Framework for Recognition and Counting of Spikes from Visual Images of Wheat Plants,” *IEEE Access*, vol. 9, pp. 76235–76247, 2021, doi: 10.1109/ACCESS.2021.3080836.

[144] L. Parra, J. Marin, S. Yousfi, G. Rincón, P. V. Mauri, and J. Lloret, “Edge detection for weed recognition in lawns,” *Comput Electron Agric*, vol. 176, no. August, 2020, doi: 10.1016/j.compag.2020.105684.

[145] S. K. Valicharla, “Weed Recognition in Agriculture : A Mask R-CNN Approach Weed Recognition in Agriculture : A Mask R-CNN Approach,” 2021.

[146] A. Wang, W. Zhang, and X. Wei, “A review on weed detection using ground-based machine vision and image processing techniques,” *Comput Electron Agric*, vol. 158, pp. 226–240, 2019, doi: 10.1016/j.compag.2019.02.005.

[147] S. Bargoti and J. Underwood, “Deep fruit detection in orchards,” *Proc IEEE Int Conf Robot Autom*, pp. 3626–3633, 2017, doi: 10.1109/ICRA.2017.7989417.

[148] A. Koirala, K. B. Walsh, Z. Wang, and C. McCarthy, “Deep learning for real-time fruit detection and orchard fruit load estimation: benchmarking of ‘MangoYOLO,’” *Precis Agric*, no. 0123456789, 2019, doi: 10.1007/s11119-019-09642-0. URL:https://doi.org/10.1007/s11119-019-09642-0

[149] E. Bellocchio, G. Costante, S. Cascianelli, M. L. Fravolini, and P. Valigi, “Combining Domain Adaptation and Spatial Consistency for Unseen Fruits Counting: A Quasi-Unsupervised Approach,” *IEEE Robot Autom Lett*, vol. 5, no. 2, pp. 1079–1086, 2020, doi: 10.1109/LRA.2020.2966398.

[150] I. Sa, Z. Ge, F. Dayoub, B. Upcroft, T. Perez, and C. McCool, “Deepfruits: A fruit detection system using deep neural networks,” *Sensors (Switzerland)*, vol. 16, no. 8, 2016, doi: 10.3390/s16081222.

[151] S. Bargoti and J. P. Underwood, “Image Segmentation for Fruit Detection and Yield Estimation in Apple Orchards,” *J Field Robot*, vol. 34, no. 6, pp. 1039–1060, 2017, doi: 10.1002/rob.21699.

[152] U. O. Dorj, M. Lee, and S. seok Yun, “An yield estimation in citrus orchards via fruit detection and counting using image processing,” *Comput Electron Agric*, vol. 140, pp. 103–112, Aug. 2017, doi: 10.1016/j.compag.2017.05.019.

[153] J. Lu, W. S. Lee, H. Gan, and X. Hu, “Immature citrus fruit detection based on local binary pattern feature and hierarchical contour analysis,” *Biosyst Eng*, vol. 171, pp. 78–90, 2018, doi: 10.1016/j.biosystemseng.2018.04.009. URL:https://doi.org/10.1016/j.biosystemseng.2018.04.009

[154] P. Sindhu and G. Indirani, “IOT with cloud based smart farming for citrus fruit disease classification using optimized convolutional neural networks,” *International Journal on Emerging Technologies*, vol. 11, no. 2, pp. 52–56, 2020.

[155] P. Marques *et al.*, “UAV-Based Automatic Detection and Monitoring of Chestnut Trees,” *Remote Sens (Basel)*, vol. 11, no. 7, p. 855, 2019, doi: 10.3390/rs11070855.

[156] A. Gupta, J. Byrne, D. Moloney, S. Watson, and H. Yin, “Automatic Tree Annotation in LiDAR Data,” no. Gistam, pp. 36–41, 2018, doi: 10.5220/0006668000360041.

[157] W. S. Qureshi, A. Payne, K. B. Walsh, R. Linker, O. Cohen, and M. N. Dailey, “Machine vision for counting fruit on mango tree canopies,” *Precis Agric*, vol. 18, no. 2, pp. 224–244, 2017, doi: 10.1007/s11119-016-9458-5.

[158] M. Zortea, M. M. G. Macedo, A. B. Mattos, B. C. Ruga, and B. H. Gemignani, “Automatic Citrus Tree Detection from UAV Images based on Convolutional Neural Networks,” *Drones*, vol. 2, no. 4, p. 39, 2018.

[159] A. Khan *et al.*, “Remote Sensing: An Automated Methodology for Olive Tree Detection and Counting in Satellite Images,” *IEEE Access*, vol. 6, pp. 77816–77828, 2018, doi: 10.1109/ACCESS.2018.2884199.

[160] M. Balsi, S. Esposito, P. Fallavollita, and C. Nardinocchi, “Single-tree detection in high-density LiDAR data from UAV-based survey,” *Eur J Remote Sens*, vol. 51, no. 1, pp. 679–692, 2018, doi: 10.1080/22797254.2018.1474722.

[161] W. Li, H. Fu, L. Yu, and A. Cracknell, “Deep learning based oil palm tree detection and counting for high-resolution remote sensing images,” *Remote Sens (Basel)*, vol. 9, no. 1, 2017, doi: 10.3390/rs9010022.

[162] N. A. Mubin, E. Nadarajoo, H. Z. M. Shafri, and A. Hamedianfar, “Young and mature oil palm tree detection and counting using convolutional neural network deep learning method,” *Int J Remote Sens*, vol. 40, no. 19, pp. 7500–7515, 2019, doi: 10.1080/01431161.2019.1569282. URL:https://doi.org/10.1080/01431161.2019.1569282

[163] H. Jiang, S. Chen, D. Li, C. Wang, and J. Yang, “Papaya Tree Detection with UAV Images Using a GPU-Accelerated Scale-Space Filtering Method,” *Remote Sens (Basel)*, vol. 9, no. 7, p. 721, 2017, doi: 10.3390/rs9070721.

[164] A. Kelman, “Plant Disease,” 2017. https://www.britannica.com/science/plant-disease (accessed Jun. 26, 2022). URL:https://www.britannica.com/science/plant-disease

[165] A. Kamilaris, F. Gao, and F. X. Prenafeta-bold, “Agri-IoT : A Semantic Framework for Internet of Things-enabled Smart Farming Applications,” no. October 2017, 2016, doi: 10.1109/WF-IoT.2016.7845467.

[166] G. Delnevo, R. Girau, C. Ceccarini, and C. Prandi, “A Deep Learning and Social IoT Approach for Plants Disease Prediction Toward a Sustainable Agriculture,” *IEEE Internet Things J*, vol. 9, no. 10, pp. 7243–7250, 2022, doi: 10.1109/JIOT.2021.3097379.

[167] A. Triantafyllou, P. Sarigiannidis, and S. Bibi, “Precision agriculture: A remote sensing monitoring system architecture,” *Information (Switzerland)*, vol. 10, no. 11, 2019, doi: 10.3390/info10110348.

[168] T. C. Hsu, H. Yang, Y. C. Chung, and C. H. Hsu, “A Creative IoT agriculture platform for cloud fog computing,” *Sustainable Computing: Informatics and Systems*, vol. 28, p. 100285, 2020, doi: 10.1016/j.suscom.2018.10.006.

[169] S. Aasha Nandhini, R. Hemalatha, S. Radha, and K. Indumathi, “Web Enabled Plant Disease Detection System for Agricultural Applications Using WMSN,” *Wirel Pers Commun*, vol. 102, no. 2, pp. 725–740, 2018, doi: 10.1007/s11277-017-5092-4. URL:https://doi.org/10.1007/s11277-017-5092-4

[170] FAO, “New standards to curb the global spread of plant pests and diseases,” 2019. https://www.fao.org/news/story/en/item/1187738/icode/ (accessed Jun. 26, 2022). URL:https://www.fao.org/news/story/en/item/1187738/icode/

[171] V. K. Vishnoi, K. Kumar, and B. Kumar, *Plant disease detection using computational intelligence and image processing*, vol. 128, no. 1. Springer Berlin Heidelberg, 2021. doi: 10.1007/s41348-020-00368-0.

[172] S. Panno *et al.*, “A review of the most common and economically important diseases that undermine the cultivation of tomato crop in the mediterranean basin,” *Agronomy*, vol. 11, no. 11, pp. 1–45, 2021, doi: 10.3390/agronomy11112188.

[173] C. S. Hlaing and S. M. Maung Zaw, “Tomato Plant Diseases Classification Using Statistical Texture Feature and Color Feature,” in *Proceedings - 17th IEEE/ACIS International Conference on Computer and Information Science, ICIS 2018*, Sep. 2018, pp. 439–444. doi: 10.1109/ICIS.2018.8466483.

[174] N. Ahmad, H. M. S. Asif, G. Saleem, M. U. Younus, S. Anwar, and M. R. Anjum, “Leaf Image-Based Plant Disease Identification Using Color and Texture Features,” *Wirel Pers Commun*, vol. 121, no. 2, pp. 1139–1168, Nov. 2021, doi: 10.1007/s11277-021-09054-2.

[175] Anjna, M. Sood, and P. K. Singh, “Hybrid System for Detection and Classification of Plant Disease Using Qualitative Texture Features Analysis,” in *Procedia Computer Science*, 2020, vol. 167, pp. 1056–1065. doi: 10.1016/j.procs.2020.03.404.

[176] J. L. Raheja, S. Kumar, and A. Chaudhary, “Fabric defect detection based on GLCM and Gabor filter: A comparison,” *Optik (Stuttg)*, vol. 124, no. 23, pp. 6469–6474, 2013, doi: 10.1016/j.ijleo.2013.05.004. URL:http://dx.doi.org/10.1016/j.ijleo.2013.05.004

[177] X. Liu and C. Aldrich, “Deep Learning Approaches to Image Texture Analysis in Material Processing,” *Metals (Basel)*, vol. 12, no. 2, 2022, doi: 10.3390/met12020355.

[178] L. Tan, J. Lu, and H. Jiang, “Tomato Leaf Diseases Classification Based on Leaf Images: A Comparison between Classical Machine Learning and Deep Learning Methods,” *AgriEngineering*, vol. 3, no. 3, pp. 542–558, Sep. 2021, doi: 10.3390/agriengineering3030035.

[179] M. Agarwal, S. K. Gupta, and K. K. Biswas, “Development of Efficient CNN model for Tomato crop disease identification,” *Sustainable Computing: Informatics and Systems*, vol. 28, Dec. 2020, doi: 10.1016/j.suscom.2020.100407.

[180] A. Hidayatuloh, M. Nursalman, and E. Nugraha, *Identification of Tomato Plant Diseases by Leaf Image Using Squeezenet Model; Identification of Tomato Plant Diseases by Leaf Image Using Squeezenet Model*. 2018.

[181] M. Kaur and R. Bhatia, "Development Of An Improved Tomato Leaf Disease Detection And Classification Method," 2019 IEEE Conference on Information and Communication Technology, Allahabad, India, 2019, pp. 1-5, doi: 10.1109/CICT48419.2019.9066230.

[182] M. Agarwal, A. Singh, S. Arjaria, A. Sinha, and S. Gupta, “ToLeD: Tomato Leaf Disease Detection using Convolution Neural Network,” in *Procedia Computer Science*, 2020, vol. 167, pp. 293–301. doi: 10.1016/j.procs.2020.03.225.

[183] M. M. Gunarathna and R. M. K. T. Rathnayaka, “Experimental determination of CNN hyper-parameters for tomato disease detection using leaf images,” in *ICAC 2020 - 2nd International Conference on Advancements in Computing, Proceedings*, Dec. 2020, pp. 464–469. doi: 10.1109/ICAC51239.2020.9357284.

[184] H. Hong, J. Lin, and F. Huang, “Tomato Disease Detection and Classification by Deep Learning,” in *Proceedings - 2020 International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering, ICBAIE 2020*, Jun. 2020, pp. 25–29. doi: 10.1109/ICBAIE49996.2020.00012.

[185] D. Jiang, F. Li, Y. Yang and S. Yu, "A Tomato Leaf Diseases Classification Method Based on Deep Learning," 2020 Chinese Control And Decision Conference (CCDC), Hefei, China, 2020, pp. 1446-1450, doi: 10.1109/CCDC49329.2020.9164457.

[186] M. H. Saleem, J. Potgieter, and K. M. Arif, “Plant disease classification: A comparative evaluation of convolutional neural networks and deep learning optimizers,” *Plants*, vol. 9, no. 10, pp. 1–17, Oct. 2020, doi: 10.3390/plants9101319.

[187] H. Kibriya, R. Rafique, W. Ahmad, and S. M. Adnan, “Tomato Leaf Disease Detection Using Convolution Neural Network,” in *Proceedings of 18th International Bhurban Conference on Applied Sciences and Technologies, IBCAST 2021*, Jan. 2021, pp. 346–351. doi: 10.1109/IBCAST51254.2021.9393311.

[188] H. I. Peyal, S. M. Shahriar, A. Sultana, I. Jahan, and M. H. Mondol, “Detection of tomato leaf diseases using transfer learning architectures: A comparative analysis,” Jul. 2021. doi: 10.1109/ACMI53878.2021.9528199.

[189] R. Sujatha, J. M. Chatterjee, N. Z. Jhanjhi, and S. N. Brohi, “Performance of deep learning vs machine learning in plant leaf disease detection,” *Microprocess Microsyst*, vol. 80, Feb. 2021, doi: 10.1016/j.micpro.2020.103615.

[190] H. Younis, M. Z. Khan, M. U. G. Khan, and H. Mukhtar, “Robust Optimization of MobileNet for Plant Disease Classification with Fine Tuned Parameters,” in *2021 International Conference on Artificial Intelligence, ICAI 2021*, Apr. 2021, pp. 146–151. doi: 10.1109/ICAI52203.2021.9445261.

[191] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, “MobileNetV2: Inverted Residuals and Linear Bottlenecks,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, 2018, doi: 10.1109/CVPR.2018.00474.

[192] P. Liashchynskyi and P. Liashchynskyi, “Grid Search, Random Search, Genetic Algorithm: A Big Comparison for NAS,” no. 2017, pp. 1–11, 2019, [Online]. Available: http://arxiv.org/abs/1912.06059. URL:http://arxiv.org/abs/1912.06059

[193] H. Alibrahim and S. A. Ludwig, “Hyperparameter Optimization: Comparing Genetic Algorithm against Grid Search and Bayesian Optimization,” in *2021 IEEE Congress on Evolutionary Computation, CEC 2021 - Proceedings*, 2021, pp. 1551–1559. doi: 10.1109/CEC45853.2021.9504761.

[194] J. González, “Still optimizing in the dark ? Bayesian optimization for model configuration and experimental design,” 2016.- University of Sheffield, Sheffield, UK February 18, 2016. Groningen, The Netherlands. URL: http://javiergonzalezh.github.io/presentations/talk_groningen2016.pdf

[195] E. Bochinski, T. Senst, and T. Sikora, “Hyper-parameter optimization for convolutional neural network committees based on evolutionary algorithms,” *Proceedings - International Conference on Image Processing, ICIP*, vol. 2017-Septe, no. September, pp. 3924–3928, 2018, doi: 10.1109/ICIP.2017.8297018.

[196] L. Tani, D. Rand, C. Veelken, and M. Kadastik, “Evolutionary algorithms for hyperparameter optimization in machine learning for application in high energy physics,” *European Physical Journal C*, vol. 81, no. 2, pp. 1–9, 2021, doi: 10.1140/epjc/s10052-021-08950-y. URL:https://doi.org/10.1140/epjc/s10052-021-08950-y

[197] J. Moosbauer, J. Herbinger, G. Casalicchio, M. Lindauer, and B. Bischl, “Explaining Hyperparameter Optimization via Partial Dependence Plots,” *Adv Neural Inf Process Syst*, vol. 3, no. NeurIPS, pp. 2280–2291, 2021.

[198] C. Molnar, “Interpretable Machine Learning A Guide for Making Black Box Models Explainable,” 2019. [Online]. Available: http://leanpub.com/interpretable-machine-learning. URL:http://leanpub.com/interpretable-machine-learning

[199] S. Sadowski and P. Spachos, “Wireless technologies for smart agricultural monitoring using internet of things devices with energy harvesting capabilities,” *Comput Electron Agric*, vol. 172, no. September 2019, p. 105338, 2020, doi: 10.1016/j.compag.2020.105338. URL:https://doi.org/10.1016/j.compag.2020.105338

[200] P. Savvidis and G. A. Papakostas, “Remote Crop Sensing with IoT and AI on the Edge,” in *2021 IEEE World AI IoT Congress, AIIoT 2021*, May 2021, pp. 48–54. doi: 10.1109/AIIoT52608.2021.9454237.

[201] C.-C. Wei, P.-Y. Su, and S.-T. Chen, “Comparison of the LoRa Image Transmission Efficiency Based on Different Encoding Methods,” *International Journal of Information and Electronics Engineering*, vol. 10, no. 1, pp. 1–4, Mar. 2020, doi: 10.18178/IJIEE.2020.10.1.712. URL:http://www.ijiee.org/index.php?m=content&c=index&a=show&catid=89&id=823

[202] A. H. Jebril, A. Sali, A. Ismail, and M. F. A. Rasid, “Overcoming limitations of LoRa physical layer in image transmission,” *Sensors (Switzerland)*, vol. 18, no. 10, Oct. 2018, doi: 10.3390/s18103257.

[203] Semtech Corporation, “LoRa and LoRaWAN: A Technical Overview,” 2019. Accessed: Oct. 14, 2022. [Online]. Available: https://lora-developers.semtech.com/documentation/tech-papers-and-guides/lora-and-lorawan/. URL:https://lora-developers.semtech.com/documentation/tech-papers-and-guides/lora-and-lorawan/

[204] F. Adelantado, X. Vilajosana, P. Tuset-Peiro, B. Martinez, J. Melia, and T. Watteyne, “Understanding the limits of LoRaWAN,” Jul. 2016, doi: 10.1109/MCOM.2017.1600613. URL:http://arxiv.org/abs/1607.08011

[205] Pycom, “Lopy4-Datasheet, version 1.1,” 2020. https://docs.pycom.io/gitbook/assets/specsheets/Pycom_002_Specsheets_LoPy4_v2.pdf (accessed Oct. 15, 2022). URL:https://docs.pycom.io/gitbook/assets/specsheets/Pycom_002_Specsheets_LoPy4_v2.pdf

[206] sensirion company, “Datasheet_SHT1x_V5_C1,” 2011. https://www.mouser.it/datasheet/2/682/Sensirion_Humidity_Sensors_SHT1x_Datasheet-1539071.pdf (accessed Oct. 15, 2022). URL:https://www.mouser.it/datasheet/2/682/Sensirion_Humidity_Sensors_SHT1x_Datasheet-1539071.pdf

[207] Silicon Labs, “Si1145/46/47 P ROXIMITY/ UV/AMBIENT LIGHT SENSOR IC,” 2013. https://cdn-shop.adafruit.com/datasheets/Si1145-46-47.pdf (accessed Oct. 15, 2022). URL:https://cdn-shop.adafruit.com/datasheets/Si1145-46-47.pdf

[208] “Raspberry Pi 4,” 2020. https://www.raspberrypi.com/products/raspberry-pi-4-model-b/ (accessed Jul. 05, 2022). URL:https://www.raspberrypi.com/products/raspberry-pi-4-model-b/

[209] “MG-811 ETC sensors,” 2020. https://datasheetspdf.com/datasheet/MG811.html (accessed Jun. 04, 2022). URL:https://datasheetspdf.com/datasheet/MG811.html

[210] “Rika sensors - RK500-03 EC / Salinity Sensor,” 2021. https://www.rikasensor.com/rk500-03-ec-salinity-sensor.html (accessed Jul. 07, 2022). URL:https://www.rikasensor.com/rk500-03-ec-salinity-sensor.html

[211] “THAOYU electronics - G1" Water Flow Sensor,” 2021. https://www.hotmcu.com/g1-water-flow-sensor-p-312.html (accessed Jun. 01, 2022). URL:https://www.hotmcu.com/g1-water-flow-sensor-p-312.html

[212] “DFROBOT sensors - camera DSJ Raspberry NVIDIA.” https://www.dfrobot.com/product-2089.html (accessed Dec. 04, 2022). URL:https://www.dfrobot.com/product-2089.html

[213] “DFROBOT sensors - Weather Station Kit with Anemometer/Wind Vane/Rain Bucket,” 2021. https://www.dfrobot.com/product-1308.html (accessed Dec. 03, 2022). URL:https://www.dfrobot.com/product-1308.html

[214] L. Dragino Technology Co., “Datasheet_DLOS8N_LoRaWAN_Gateway,” 2020. https://www.dragino.com/products/lora-lorawan-gateway/item/225-dlos8n.html (accessed Jul. 02, 2022). URL:https://www.dragino.com/products/lora-lorawan-gateway/item/225-dlos8n.html

[215] “The Things Network -TTN.” https://www.thethingsnetwork.org/ (accessed Feb. 01, 2022). URL:https://www.thethingsnetwork.org/

[216] “Ubidots - Industrial IoT.” https://ubidots.com/ (accessed Apr. 01, 2022). URL:https://ubidots.com/