

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE ELECTRÓNICA E INFORMÁTICA
CONCEPCIÓN - CHILE



**GreenGuardian: Modelo CNN Aplicado a la Detección
de Enfermedades de Plantas.**

Duván Alexander Apiolaza Caroca

**MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO EN INFORMÁTICA**

Profesor Guía: Rhoddy Angel Viveros Muñoz

Diciembre - 2024

DEDICATORIA

Dedico este trabajo con todo mi cariño a mi familia, por su amor incondicional y por ser mi mayor fuente de inspiración. A mi madre, por su apoyo constante y su confianza en mí.

Este logro es también de ustedes.

AGRADECIMIENTOS

Agradezco profundamente a Rhoddy por su valiosa guía y apoyo durante la realización de este trabajo. También, a mi familia por su constante motivación y a mis amigos por su ánimo y consejos. Finalmente, a la universidad por facilitar los recursos necesarios para completar este proyecto.

Gracias a todos por su contribución.

RESUMEN

El presente trabajo aborda el desarrollo de un sistema automatizado basado en redes neuronales convolucionales (CNN) para la detección temprana de enfermedades en cultivos hidropónicos. El modelo fue entrenado utilizando el dataset **PlantVillage (Hughes & Salathé, 2015)**, reestructurado en dos clases: **sanas** y **enfermas**, permitiendo optimizar su capacidad de clasificación. Se implementó un pipeline de preprocesamiento que incluye técnicas como normalización, aumento de datos y balanceo mediante pesos de clase, mejorando la robustez del modelo.

El entorno de desarrollo fue **Paperspace Gradient**, donde se logró un rendimiento del modelo con una precisión del **99%** en el conjunto de validación. La solución se integra en el ecosistema **GreenGuardian** a través de cámaras de monitoreo y una aplicación móvil, generando alertas automáticas en tiempo real. Los resultados demuestran la eficacia del modelo para optimizar la gestión de cultivos hidropónicos, reduciendo pérdidas económicas y promoviendo prácticas agrícolas sostenibles.

Palabras Clave— *Red neuronal convolucional; Cultivos hidropónicos; Detección de enfermedades; Inteligencia Artificial.*

ABSTRACT

This study presents the development of an automated system based on Convolutional Neural Networks (CNN) for the early detection of diseases in hydroponic crops. The model was trained using the PlantVillage dataset, which was restructured into two classes: healthy and diseased, optimizing its classification performance. A preprocessing pipeline was implemented, including normalization, data augmentation, and class weight balancing, improving the model's robustness.

The development environment was Paperspace Gradient, where the model achieved a validation accuracy of 99%. The solution is integrated into the GreenGuardian ecosystem through monitoring cameras and a mobile application, providing real-time alerts. The results demonstrate the effectiveness of the model in optimizing hydroponic crop management, reducing economic losses, and promoting sustainable agricultural practices.

Keywords— *Convolutional Neural Networks, Disease Detection, Hydroponic Crops, Artificial Intelligence*

GLOSARIO

AI: Artificial Intelligence (Inteligencia Artificial). Rama de la informática que simula procesos de inteligencia humana mediante algoritmos y modelos matemáticos.

CNN: Convolutional Neural Network (Red Neuronal Convolucional). Modelo de red neuronal diseñado para el procesamiento de imágenes y visión por computadora.

FAO: Food and Agriculture Organization (Organización de las Naciones Unidas para la Alimentación y la Agricultura). Organización internacional que lidera esfuerzos para erradicar el hambre y mejorar la seguridad alimentaria.

ML: Machine Learning (Aprendizaje Automático). Subcampo de la inteligencia artificial que permite a los sistemas aprender y mejorar a partir de datos sin programación explícita.

PlantVillage: Base de datos abierta que contiene imágenes de cultivos clasificadas en categorías, utilizada para entrenar modelos de aprendizaje profundo.

RGB: Red Green Blue (Rojo, Verde, Azul). Modelo de representación de imágenes basado en la combinación de estos tres colores primarios.

TDS: Total Dissolved Solids (Sólidos Totales Disueltos). Medida que cuantifica la concentración total de sustancias disueltas en una solución, comúnmente en agua.

UTFSM: Universidad Técnica Federico Santa María. Institución educativa chilena de alto prestigio en ciencias e ingeniería.

XLA: Accelerated Linear Algebra (Aceleración de Álgebra Lineal). Framework de optimización de modelos de aprendizaje automático, diseñado para TensorFlow.

PIB (Producto Interno Bruto): Indicador económico que mide el valor total de los bienes y servicios producidos en un país durante un período de tiempo determinado, generalmente un año. Es utilizado para evaluar el tamaño y el desempeño de la economía.

INDICE DE CONTENIDOS

INTRODUCCIÓN.....	11
CAPÍTULO 1: DEFINICIÓN DEL PROBLEMA	12
1.1. Problemática Específica	12
1.2. Justificación.....	13
1.3. Objetivo General	13
1.4. Objetivos Específicos	14
CAPÍTULO 2: MARCO CONCEPTUAL.....	15
2.1. Redes Neuronales Convolucionales (CNNs)	15
2.2. Componentes Principales.....	15
2.2.1. Capas Convolucionales:.....	15
2.2.2. Capas de Pooling.....	15
2.2.3. Capas Densas:	15
2.2.4. Regularización	16
2.3. Funcionamiento General	16
2.4. Dataset PlantVillage.....	17
2.4.1. Relevancia	17
2.4.1. Limitaciones	17
2.5. Reestructuración del Dataset PlantVillage.....	18
2.6. Procesamiento de Datos	18
2.6.1. Class Weights.....	18
2.6.2. Normalización	18
2.6.3. Aumento de Datos	18
2.7. Pipeline de preprocesamiento	18

2.8. Integración Tecnológica	19
2.8.1. Paperspace Gradient.....	19
2.8.2. TensorFlow/Keras.....	20
2.8.3. Cámaras	20
2.9. Arquitectura del modelo CNN en GreenGuardian	20
CAPÍTULO 3: PROPUESTA DE SOLUCION.....	22
3.1. Entorno de Desarrollo.....	22
3.1.1. Plataforma: Paperspace Gradient.....	22
3.1.2. Hardware Utilizado:	22
3.2. Dataset.....	23
3.2.1 Reducción a Dos Clases:	23
3.2.2. Cantidad de Datos:	23
3.3. Preprocesamiento de Datos	24
3.3.1. Redimensionamiento.....	24
3.3.2. Normalización	24
3.3.3. Balanceo de Clases:.....	24
3.3.4. Aumento de Datos:	24
3.4. Estructura del Modelo	24
3.4.1. Capas de Entrada:	25
3.4.2. Capas Convolucionales:.....	25
3.4.3. Capas Ocultas:	25
3.4.4. Capa de Salida:	25
3.6. Implementación del Pipeline.....	26
CAPÍTULO 4: VALIDACIÓN DE LA SOLUCIÓN	27

4.1. Resultados del Entrenamiento	27
4.1.1. Proceso de Entrenamiento	27
4.1.2. Curvas de Entrenamiento.....	27
4.2. Evaluación del Modelo.....	29
4.2.1. Matriz de Confusión.....	29
4.2.2. Métricas de Clasificación.....	29
4.3. Validación de Predicciones.....	30
4.3.1. Predicciones Correctas	30
4.3.2. Predicciones Incorrectas	31
4.4. Discusión de Resultados.....	31
CAPÍTULO 5: CONCLUSIONES	32
5.1 Resultados Principales	32
5.1.1. Reestructuración del Dataset PlantVillage:	32
5.1.2. Optimización del Modelo CNN:	32
5.1.3. Pipeline de Preprocesamiento de Imágenes:	32
5.1.4. Integración Propuesta en GreenGuardian:	32
5.2 Beneficios Proyectados de la Solución	33
5.3 Limitaciones del Trabajo	33
5.4 Proyecciones a Futuro.....	33
REFERENCIAS BIBLIOGRÁFICAS	34
ANEXOS	35
Anexo A: Configuración del Entorno de Ejecución	35
Anexo B: Pipeline de Preprocesamiento	36
Anexo C: Arquitectura del Modelo CNN en GreenGuardian	39

INDICE DE FIGURAS

<i>Figura 1. Arquitectura de la Red Convolucional (CNN)</i>	16
<i>Figura 2. Pipeline de Preprocesamiento de Imágenes</i>	19
<i>Figura 3. Arquitectura de GreenGuardian con Integración del Modelo CNN</i>	20
<i>Figura 4. Especificaciones de GPU Utilizada</i>	23
<i>Figura 5. Arquitectura CNN</i>	25
<i>Figura 6. Precisión por Época</i>	28
<i>Figura 7. Pérdida por Época</i>	28
<i>Figura 8. Matriz de Confusión</i>	29
<i>Figura 9. Predicciones Correctas</i>	30
<i>Figura 10. Predicciones Incorrectas</i>	31
<i>Figura 11. Código del Preprocesamiento</i>	37
<i>Figura 12. Imágenes Preprocesadas</i>	38

INDICE DE TABLAS

<i>Tabla 1. Comparativa Pérdidas Económicas por Enfermedades en Cultivos Tradicionales vs Hidropónicos</i>	11
<i>Tabla 2. Alcances y Limitaciones del Modelo</i>	14
<i>Tabla 3. Hiperparámetros</i>	26
<i>Tabla 4. Informe de Clasificación</i>	30
<i>Tabla 5. Librerías Instaladas</i>	35

INTRODUCCIÓN

Las enfermedades y plagas representan un desafío crítico para la agricultura global, con pérdidas estimadas entre el 20% y el 40% de la producción total (FAO, n.d.). En sistemas hidropónicos, esta problemática se agrava debido a la naturaleza recirculante de sus sistemas, que facilita la propagación de patógenos como *Pythium spp.* y *Fusarium spp.* (Universidad de Puerto Rico, n.d.).

En Chile, la hidroponía ha surgido como una alternativa sostenible frente a desafíos como el cambio climático y la escasez hídrica. Sin embargo, enfrenta riesgos significativos debido a enfermedades radicales que comprometen la productividad. La agricultura chilena contribuye aproximadamente al 3% del Producto Interno Bruto (PIB), subrayando la importancia de abordar estas problemáticas para garantizar la sostenibilidad del sector (Banco Mundial, 2023).

El proyecto GreenGuardian integra un modelo de red neuronal convolucional (CNN) diseñado para identificar enfermedades en cultivos hidropónicos a partir de imágenes capturadas en tiempo real. Este enfoque permite detectar anomalías en etapas tempranas, mejorando el monitoreo y reduciendo las pérdidas económicas. La gravedad de la problemática queda reflejada en la **Tabla 1**, que compara las pérdidas entre cultivos tradicionales e hidropónicos.

Tabla 1. Comparativa Pérdidas Económicas por Enfermedades en Cultivos Tradicionales vs Hidropónicos

TIPO DE CULTIVO	INCIDENCIA DE ENFERMEDADES (%)	IMPACTO EN LA PRODUCCIÓN (%)
Cultivos Tradicionales	20-30	10-15
Cultivos Hidropónicos	30-40	20-30

Fuente: Elaboración propia basada en datos de la [FAO \(n.d.\)](#) y el [Banco Mundial \(2023\)](#).

CAPÍTULO 1: DEFINICIÓN DEL PROBLEMA

La agricultura hidropónica, al eliminar el uso de suelo y minimizar el consumo de agua, se posiciona como una solución clave en la búsqueda de la sostenibilidad agrícola. En Chile, este sistema ha ganado relevancia como respuesta a la escasez hídrica y al cambio climático, contribuyendo a la diversificación de cultivos y al aumento de la eficiencia en el uso de recursos ([Banco Mundial, 2023](#)). Sin embargo, los sistemas hidropónicos enfrentan desafíos críticos debido a su vulnerabilidad frente a enfermedades, que pueden propagarse rápidamente a través de las soluciones recirculantes ([Diario del Agro, 2023](#)).

A pesar de sus beneficios, la hidroponía enfrenta retos similares a los cultivos tradicionales en términos de enfermedades, pero con características específicas que amplifican el impacto de estos problemas, como, por ejemplo:

- **Demanda tiempo:** El monitoreo manual no es viable a gran escala.
- **Es subjetivo:** La experiencia del observador influye en la precisión.
- **Carece de escalabilidad:** No se adapta bien a sistemas grandes o automatizados.

El uso de tecnologías de inteligencia artificial, como las redes neuronales convolucionales (CNNs), permite superar estas limitaciones al ofrecer un análisis sistemático, rápido y preciso.

1.1. Problemática Específica

La ausencia de herramientas automatizadas para la detección de enfermedades en cultivos hidropónicos genera los siguientes problemas:

- 1.1.1. Detección Tardía:** Las enfermedades suelen identificarse en etapas avanzadas, cuando ya han reducido significativamente la productividad del cultivo. En casos severos, pueden causar pérdidas de hasta el 40% de la producción global ([FAO, n.d.](#)).
- 1.1.2. Subjetividad en la Identificación:** La detección depende exclusivamente de la experiencia del agricultor, lo que aumenta el riesgo de diagnósticos incorrectos o inconsistentes. Esto es especialmente crítico en sistemas como la hidroponía, donde los signos tempranos de enfermedad son sutiles ([Universidad de Puerto Rico, n.d.](#)).
- 1.1.3. Impacto Económico:** Las pérdidas económicas asociadas a la propagación de enfermedades en cultivos hidropónicos pueden generar una reducción en la rentabilidad del sector agrícola, afectando un área que constituye una parte significativa del Producto Interno Bruto (PIB) de Chile. ([Banco Mundial \(2023\)](#)).

1.2. Justificación

El desarrollo de un sistema automatizado basado en inteligencia artificial, como el modelo CNN propuesto, responde a la necesidad de optimizar el manejo de cultivos hidropónicos. Estas herramientas permiten detectar enfermedades en etapas iniciales, reduciendo las pérdidas económicas y mejorando la sostenibilidad agrícola. De acuerdo con el [Banco Mundial \(2023\)](#), la integración de tecnologías avanzadas puede aumentar en un 20% la rentabilidad de los sistemas agrícolas sostenibles, incentivando su adopción a nivel nacional. Además de ofrecer:

1.2.1. Monitoreo Continuo:

- Las cámaras del Plan Premium capturan imágenes regularmente, proporcionando datos en tiempo real.
- Esto asegura que los problemas se detecten en las etapas iniciales, permitiendo intervenciones oportunas.

1.2.2. Análisis Automatizado:

- El modelo CNN elimina la subjetividad al clasificar de manera precisa las plantas como sanas o enfermas.
- Esto facilita el acceso a soluciones avanzadas, independientemente de la experiencia del agricultor.

1.2.3. Escalabilidad y Eficiencia:

- Al integrarse en el ecosistema GreenGuardian, el sistema se adapta fácilmente a cultivos de diferentes tamaños.
- Reduce la carga operativa, liberando recursos humanos para otras actividades.

1.2.4. Impacto en la Sostenibilidad:

- Promueve prácticas agrícolas más sostenibles al minimizar las pérdidas por enfermedades.
- Mejora la rentabilidad de los agricultores, fomentando la adopción de tecnologías innovadoras.

1.3. Objetivo General

Desarrollar e integrar un modelo de red neuronal convolucional (CNN) dentro del Plan Premium de GreenGuardian, capaz de detectar enfermedades en cultivos hidropónicos mediante imágenes capturadas por cámaras, proporcionando alertas en tiempo real.

1.4. Objetivos Específicos

1. Entrenar un modelo CNN utilizando el dataset **reestructurado** de PlantVillage, clasificando hojas en dos categorías: **sanas** y **enfermas**.
2. Implementar un pipeline de preprocesamiento que incluya normalización, aumento de datos y técnicas de **balanceo de clases**.
3. Evaluar el modelo en un entorno robusto utilizando **Paperspace Gradient**, midiendo su desempeño con métricas como precisión, recall y F1-score.
4. Integrar el modelo CNN con las cámaras del Plan Premium y el sistema de notificaciones en la aplicación móvil de GreenGuardian.
5. Validar el sistema en un entorno simulado para garantizar su funcionalidad antes de la implementación en campo.

Tabla 2. Alcances y Limitaciones del Modelo

Aspecto	Alcances	Limitaciones	Beneficios
Clasificación	Detectará enfermedades en plantas con imágenes de 128x128 píxeles.	Requiere imágenes de buena calidad para garantizar precisión.	Incrementa la precisión y reduce el tiempo de diagnóstico.
Integración	Enviará alertas automáticas a través de la app móvil.	Depende de una conectividad constante para el procesamiento.	Permite una respuesta rápida para mitigar problemas.
Escalabilidad	Permite añadir nuevas clases de enfermedades en el futuro.	Inicialmente limitado a enfermedades incluidas en <i>PlantVillage</i> .	Mejora la adaptabilidad para diferentes tipos de cultivos.

Fuente: Elaboración propia basada en datos de *PlantVillage* (Hughes & Salathé, 2015).

CAPÍTULO 2: MARCO CONCEPTUAL

2.1. Redes Neuronales Convolucionales (CNNs)

Las redes neuronales convolucionales, también llamadas CNN, son un tipo especializado de red neuronal diseñado para procesar datos estructurados espacialmente, como imágenes. Estas redes son especialmente efectivas para tareas de clasificación, detección de objetos y segmentación, debido a su capacidad para extraer automáticamente características relevantes de las imágenes.

2.2. Componentes Principales

Las CNN están compuestas por diversos elementos clave que trabajan en conjunto para procesar imágenes y extraer características relevantes. Cada componente tiene un rol específico, desde la extracción de patrones básicos hasta la combinación de información para tomar decisiones de clasificación.

2.2.1. Capas Convolucionales:

Las capas convolucionales son el núcleo de las CNN. Cada capa utiliza filtros (kernels), que son matrices pequeñas, para recorrer la imagen de entrada y detectar patrones como bordes, texturas o formas específicas. Este proceso, conocido como convolución, genera un mapa de características que representa las propiedades detectadas en la imagen. A medida que avanzan las capas, los patrones detectados se vuelven más complejos, pasando de detalles básicos como líneas a elementos más específicos como formas u objetos completos.

2.2.2. Capas de Pooling:

Las capas de pooling, también conocidas como capas de reducción, disminuyen la dimensionalidad de los mapas de características generados por las capas convolucionales. Esto reduce la carga computacional y mejora la eficiencia del modelo. La técnica más común es el *max-pooling*, donde se selecciona el valor máximo dentro de una región definida (por ejemplo, 2x2). Alternativamente, el *average pooling* toma el promedio de los valores en esa región. Estas capas también contribuyen a hacer el modelo más robusto frente a pequeñas variaciones en la posición de las características.

2.2.3. Capas Densas:

Las capas densas, también llamadas capas completamente conectadas (*fully connected layers*), toman las características procesadas por las capas convolucionales y de pooling y las combinan para tomar decisiones de clasificación. Cada neurona de una capa densa está conectada a todas las neuronas de la capa anterior. La última capa densa, en problemas

de clasificación, utiliza una función de activación *softmax* para asignar probabilidades a las posibles clases. Este diseño permite que las capas densas interpreten los patrones detectados en las etapas previas.

2.2.4. Regularización:

La regularización es una técnica utilizada para prevenir el sobreajuste, asegurando que el modelo generalice correctamente en datos nuevos. Una de las técnicas más comunes es el *dropout*, que desactiva de manera aleatoria un porcentaje de neuronas durante el entrenamiento. Esto fuerza al modelo a distribuir el aprendizaje entre todas las neuronas, evitando que dependa demasiado de algunas en particular. Por ejemplo, un *dropout* del 50% indica que la mitad de las neuronas se desactivan en cada paso, fomentando la robustez del modelo.

2.3. Funcionamiento General

El flujo de datos en una CNN sigue un proceso jerárquico que combina la extracción de características y la clasificación. Este proceso, representado en la **Figura 1**, detalla las etapas principales de la arquitectura utilizada en este proyecto para clasificar imágenes de plantas como “Sanas” o “Enfermas”. La red opera en tres fases principales:

- **Entrada de la imagen:** La imagen de entrada es una matriz de píxeles normalizada de tamaño 128x128x3.
- **Extracción de características:** Las capas convolucionales identifican patrones relevantes, como bordes, texturas y otras características visuales, mientras que las capas de pooling reducen la dimensionalidad para optimizar el procesamiento.
- **Clasificación:** Las capas densas integran las características extraídas y asignan probabilidades a las clases posibles, utilizando una capa de salida con activación softmax.

La siguiente figura ilustra cómo los datos fluyen a través de las diferentes capas de la CNN utilizada en este proyecto:

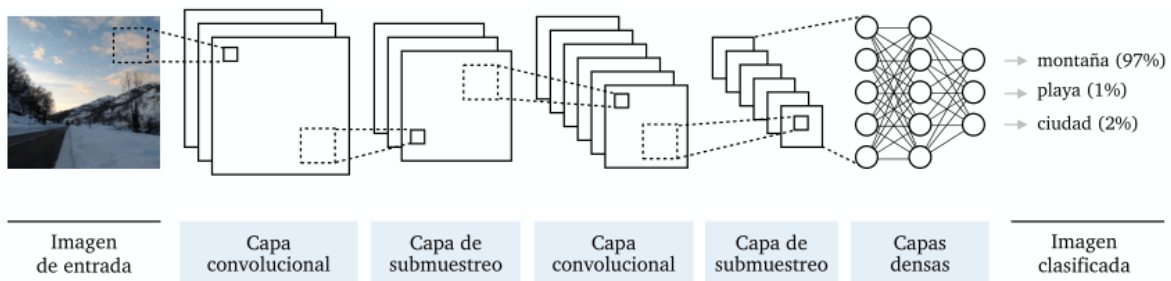


Figura 1. Arquitectura de la Red Convolucional (CNN)

Fuente: González-Muñoz (2023).

El diagrama refleja cómo cada capa realiza su función específica dentro de la arquitectura de la CNN:

- **Entrada:** Recibe una imagen.
- **Capas convolucionales:** Aplican filtros para extraer características relevantes de la imagen, generando mapas de características.
- **Capas de pooling:** Reducen la dimensionalidad de los mapas de características, destacando las partes más importantes de la imagen.
- **Capa Densa:** Integra todas las características extraídas en un vector, que sirve para la clasificación final.
- **Capa de salida:** Genera las probabilidades de que la imagen pertenezca a cada clase (montaña, playa o ciudad), entregando como resultado la clase con mayor probabilidad.

2.4. Dataset PlantVillage

El dataset PlantVillage contiene más de 50,000 imágenes de plantas, clasificadas en diferentes estados de salud. Este dataset es crucial para el entrenamiento del modelo CNN debido a su diversidad y calidad, pero presenta limitaciones al no incluir imágenes tomadas en entornos reales. Esto se aborda mediante pruebas con imágenes adicionales obtenidas en sistemas hidropónicos simulados.

2.4.1. Relevancia

- **Diversidad:** Contiene imágenes de alta calidad y diversas condiciones, lo que permite entrenar modelos robustos.
- **Etiquetado:** Cada imagen está clasificada con precisión, facilitando el entrenamiento supervisado.
- **Compatibilidad:** El dataset es ideal para problemas de clasificación binaria (sana o enferma) o multiclase.

2.4.1. Limitaciones

- Aunque es exhaustivo, algunas categorías tienen menos imágenes, lo que puede causar desbalances en los datos.
- Aunque el dataset PlantVillage es exhaustivo en diversidad y calidad, presenta limitaciones al no incluir imágenes obtenidas en entornos hidropónicos reales, lo que puede afectar la capacidad del modelo para generalizar a escenarios prácticos. Este desafío será abordado mediante pruebas en entornos controlados con imágenes externas.

2.5. Reestructuración del Dataset PlantVillage

Para este proyecto, se seleccionaron únicamente imágenes clasificadas en dos categorías: **sanas** y **enfermas**. Esta reestructuración responde a la necesidad de reducir la complejidad del problema, enfocando el modelo en la detección binaria de anomalías

Total de imágenes: 55,448.

Distribución:

- **Enfermas:** 40,364 imágenes.
- **Sanas:** 15,084 imágenes.

2.6. Procesamiento de Datos

2.6.1. Class Weights

Debido al desbalance observado en las clases (enfermas: 40,364; sanas: 15,084), se implementó un mecanismo de pesos de clase (Class Weights) para ajustar la influencia de cada categoría durante el entrenamiento. Esta técnica permite mejorar el rendimiento del modelo en datasets desequilibrados al penalizar más los errores en la clase menos representada.

2.6.2. Normalización

Todas las imágenes se redimensionan a 128x128 píxeles y se normalizan dividiendo los valores de los píxeles entre 255, asegurando que estén en el rango [0, 1]. Esto facilita el entrenamiento y mejora la estabilidad del modelo.

2.6.3. Aumento de Datos

El aumento de datos incluye técnicas como:

- **Rotación:** Variaciones en el ángulo de la imagen.
- **Volteo horizontal:** Simulación de diferentes perspectivas.
- **Ajustes de brillo:** Modificaciones en la iluminación.

Estas técnicas aumentan la diversidad del dataset y previenen el sobreajuste.

2.7. Pipeline de preprocesamiento

El pipeline de preprocesamiento de imágenes es un paso esencial en la preparación del dataset para el modelo CNN. Este proceso garantiza que las imágenes estén en un formato uniforme y optimizado para el entrenamiento. La **Figura 2** detalla las etapas principales del pipeline implementado en este proyecto:

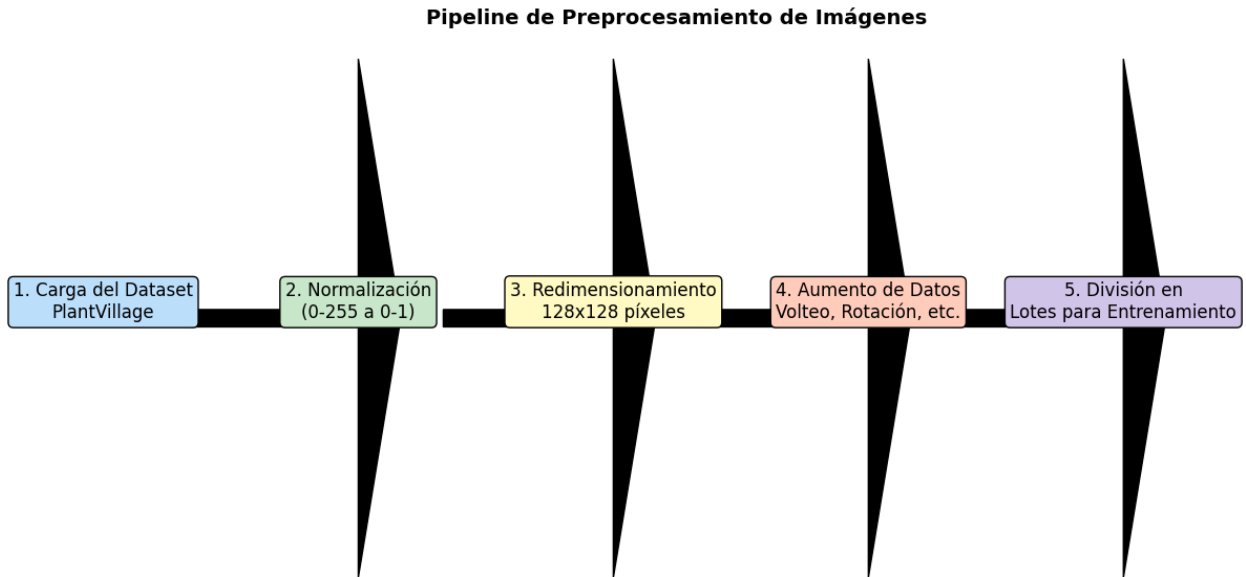


Figura 2. Pipeline de Preprocesamiento de Imágenes

Fuente: Elaboración Propia

Cada paso realiza una función específica dentro del pipeline:

1. **Carga del dataset:** Las imágenes se obtienen del dataset PlantVillage.
2. **Normalización:** Los valores de los píxeles, originalmente en el rango 0-255, se normalizan a un rango de 0 a 1.
3. **Redimensionamiento:** Todas las imágenes se ajustan a un tamaño estándar de 128x128 píxeles.
4. **Aumento de datos:** Se aplican técnicas como volteo y rotación para mejorar la generalización del modelo.
5. **División en Lotes y Aplicación de Class Weights:** Se aplican pesos de clase para abordar el desbalance entre las categorías (40,364 imágenes enfermas vs 15,084 sanas), mejorando la capacidad del modelo para generalizar.

Para más detalles técnicos sobre el flujo de implementación del pipeline de preprocesamiento, consultar el [**Anexo B**](#).

2.8. Integración Tecnológica

2.8.1. Paperspace Gradient

El modelo se entrenó utilizando **Paperspace Gradient**, una plataforma diseñada para tareas intensivas en recursos computacionales. Esta elección permitió aprovechar una **GPU NVIDIA Quadro P4000** con soporte de alto rendimiento y un entorno optimizado para el entrenamiento de modelos de aprendizaje profundo.

2.8.2. TensorFlow/Keras

El modelo CNN fue implementado utilizando TensorFlow y su API de alto nivel, Keras. Estas herramientas ofrecen flexibilidad y potencia para construir y entrenar redes neuronales.

2.8.3. Cámaras

El modelo está diseñado para integrarse con cámaras de video incluidas en el Plan Premium. Estas cámaras capturan imágenes a intervalos regulares o bajo demanda, enviándolas al backend para su análisis.

2.9. Arquitectura del modelo CNN en GreenGuardian

La integración del modelo CNN se realiza a través del Microservicio de Predicción, que incluye módulos clave como el preprocesamiento de imágenes, el modelo de predicción, el monitoreo y registro de eventos, así como un sistema de retroalimentación para mejorar continuamente el modelo. Este microservicio interactúa con la infraestructura general del sistema, como se muestra en la **Figura 3**, proporcionando un flujo eficiente desde la captura de imágenes hasta la notificación al usuario final.

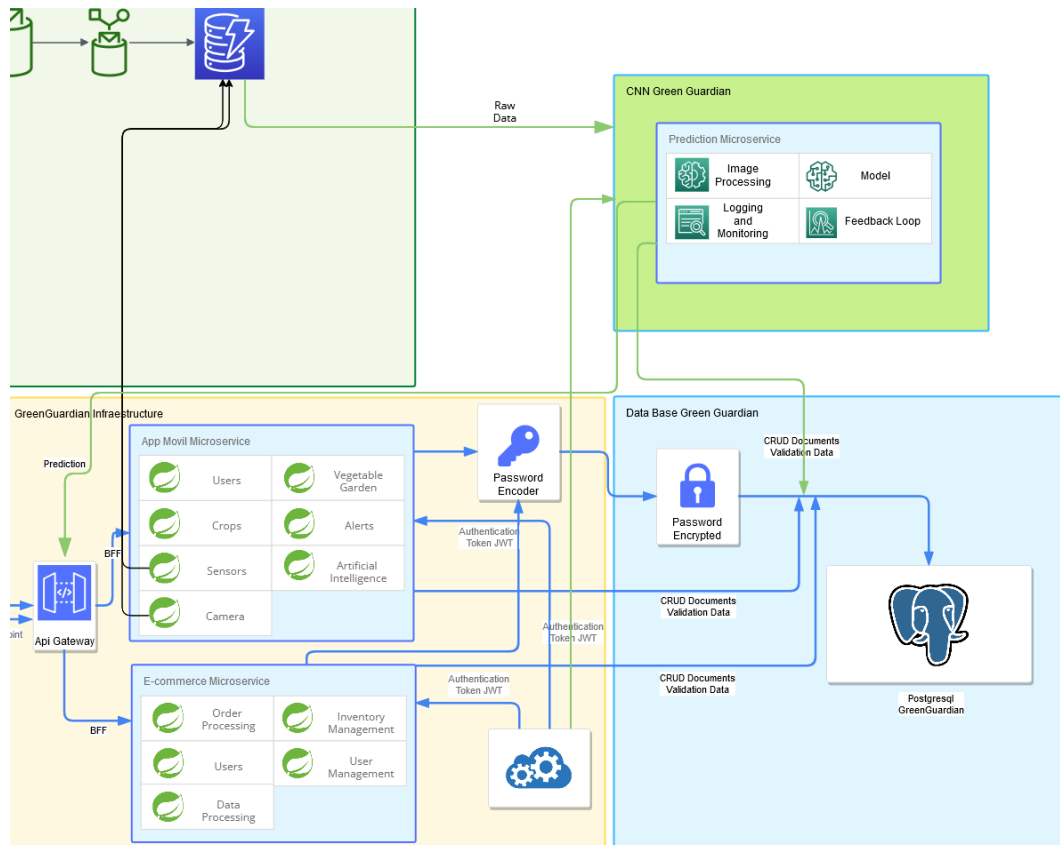


Figura 3. Arquitectura de GreenGuardian con Integración del Modelo CNN

Fuente: Elaboración Propia

La **figura 3** muestra la arquitectura de GreenGuardian, destacando la integración del modelo CNN a través del Microservicio de Predicción. La cámara transmite imágenes al sistema de almacenamiento temporal, donde son preprocesadas y clasificadas mediante el modelo CNN. Los resultados de la predicción se registran en la base de datos y, en caso de detectar una planta enferma, se genera una alerta que se envía al usuario a través de la aplicación móvil. Este flujo permite una respuesta rápida y eficaz ante posibles problemas en los cultivos hidropónicos.

La arquitectura completa del sistema, incluyendo la integración del modelo CNN, se describe con mayor detalle en el **Anexo C**.

El marco conceptual presentado establece las bases para el desarrollo e implementación del modelo CNN en el sistema GreenGuardian. Desde los fundamentos teóricos hasta las herramientas tecnológicas empleadas, cada componente contribuye a la creación de una solución innovadora que aborda los desafíos de la agricultura hidropónica.

CAPÍTULO 3: PROPUESTA DE SOLUCION

La detección temprana de enfermedades en cultivos hidropónicos es fundamental para garantizar la sostenibilidad y rentabilidad de esta práctica agrícola. Las enfermedades pueden causar pérdidas significativas en la producción y comprometer la calidad de los cultivos. Este capítulo presenta una solución innovadora basada en inteligencia artificial para abordar este desafío.

Se ha desarrollado y entrenado un modelo CNN para clasificar imágenes de plantas como sanas o enfermas. La solución propuesta se integra en el sistema GreenGuardian, permitiendo el monitoreo continuo de los cultivos y la generación de alertas en tiempo real. A diferencia de las soluciones tradicionales, este enfoque ofrece una detección temprana y precisa de enfermedades, lo que permite tomar medidas correctivas de manera oportuna.

Se espera que el modelo de CNN desarrollado sea capaz de clasificar imágenes de plantas con una precisión superior al 95%, lo que permitirá reducir significativamente las pérdidas en la producción hidropónica. Los siguientes apartados detallarán la arquitectura del modelo, el proceso de entrenamiento y los resultados obtenidos.

3.1. Entorno de Desarrollo

Se seleccionó un entorno robusto y eficiente para entrenar modelos de aprendizaje profundo debido a los altos requerimientos computacionales.

3.1.1. Plataforma: Paperspace Gradient

- **Justificación:** Proporciona acceso a GPU de alto rendimiento y un entorno de ejecución optimizado para Machine Learning.

3.1.2. Hardware Utilizado:

- GPU: **NVIDIA Quadro P4000** (versión utilizada en la máquina virtual).
- RAM: 16GB.
- CPU: 4 vCPUs.

NVIDIA-SMI 525.116.04 Driver Version: 525.116.04 CUDA Version: 12.0									
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	Quadro P4000	Off		00000000:00:05.0	Off			N/A	
46%	39C	P0	25W / 105W	2MiB / 8192MiB		0%	Default	N/A	
Processes:									
GPU	GI	CI	PID	Type	Process name	GPU	Memory		
ID	ID	ID				Usage			
No running processes found									

Figura 4. Especificaciones de GPU Utilizada

Fuente: Elaboración Propia

El modelo fue desarrollado y entrenado en un entorno robusto utilizando TensorFlow y GPU, detallado en el [Anexo A](#)

3.2. Dataset

El dataset original de **PlantVillage** contiene imágenes clasificadas en múltiples categorías según el tipo de planta y la enfermedad específica. Para este proyecto, **GreenGuardian**, se realizó una **reestructuración del dataset** con el objetivo de ajustarlo a los requerimientos del modelo. Las modificaciones aplicadas fueron:

3.2.1 Reducción a Dos Clases:

Se seleccionaron únicamente dos categorías generales:

1. **Enfermas:** Imágenes de hojas con cualquier tipo de enfermedad.
2. **Sanas:** Imágenes de hojas sin signos visibles de enfermedad.

Justificación: Simplificar la clasificación para enfocarse únicamente en determinar si una planta requiere intervención.

3.2.2. Cantidad de Datos:

- **Total de imágenes después de la reestructuración: 55,448.**

- **Distribución de Clases:**
 - **Enfermas:** 40,364 imágenes.
 - **Sanas:** 15,084 imágenes.

3.3. Preprocesamiento de Datos

Para optimizar el entrenamiento del modelo, se realizaron las siguientes técnicas sobre el dataset reestructurado:

3.3.1. Redimensionamiento

Todas las imágenes se ajustaron a 128x128 píxeles.

3.3.2. Normalización

Los valores de los píxeles se escalaron al rango [0, 1].

3.3.3. Balanceo de Clases:

- Debido a la diferencia significativa en el número de imágenes entre clases, se aplicó la técnica de *Class Weights* durante el entrenamiento.

3.3.4. Aumento de Datos:

- Rotaciones aleatorias (hasta 20°).
- Volteos horizontales.
- Variaciones de zoom, brillo y contraste aleatorios.

3.4. Estructura del Modelo

La arquitectura propuesta fue diseñada teniendo en cuenta el balance entre la complejidad del modelo y la capacidad de generalización en el problema de clasificación binaria (sanas y enfermas). Las capas convolucionales progresivamente más profundas (32, 64 y 128 filtros) permiten extraer patrones desde características simples (bordes, texturas) hasta más complejas (formas específicas). Este diseño es adecuado para el tamaño del dataset PlantVillage, que proporciona suficientes ejemplos para entrenar un modelo con esta capacidad sin riesgo elevado de sobreajuste. Además, la inclusión de *MaxPooling2D* y *Dropout* permite reducir la dimensionalidad y mitigar el sobreajuste, asegurando un modelo eficiente y robusto. La última capa *Dense* con activación *Softmax* permite una clasificación binaria precisa, necesaria para este problema específico.

La arquitectura incluye las siguientes capas:

3.4.1. Capas de Entrada:

- Aceptan imágenes en formato (128, 128, 3).

3.4.2. Capas Convolucionales:

- **Capa 1:** Conv2D con 32 filtros (3x3) y activación **ReLU**.
- **Capa 2:** Conv2D con 64 filtros (3x3) y activación **ReLU**.
- **Capa 3:** Conv2D con 128 filtros (3x3) y activación **ReLU**.

Cada capa convolucional está seguida por una **MaxPooling2D** (2x2) para reducir la dimensionalidad.

3.4.3. Capas Ocultas:

- **Flatten:** Aplanamiento de la salida de las capas convolucionales.
- **Dropout (50%):** Regularización para evitar el sobreajuste.
- **Dense:** Una capa densa con 128 unidades y activación **ReLU**.

3.4.4. Capa de Salida:

- **Dense:** Capa final con **2 unidades** y activación **Softmax** para la clasificación en 2 clases.

Arquitectura de Red Neuronal Convolucional (CNN)

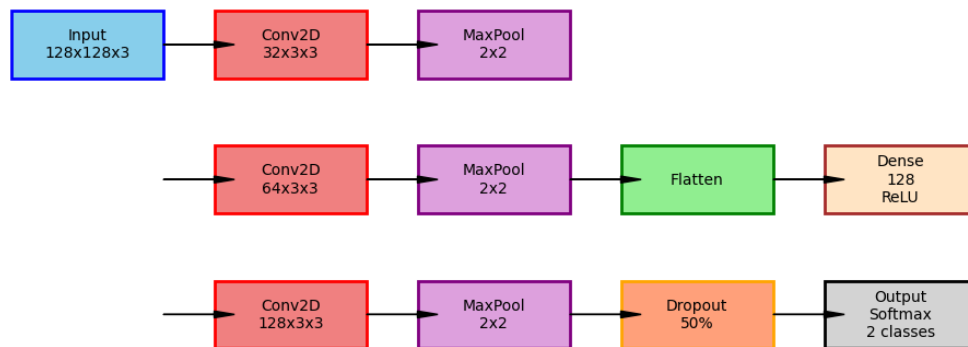


Figura 5. Arquitectura CNN

Fuente: Elaboración Propia

3.5. Hiperparámetros

Se seleccionaron y ajustaron los siguientes hiperparámetros para optimizar el rendimiento del modelo:

Tabla 3. Hiperparámetros

PARÁMETRO	VALOR
TASA DE APRENDIZAJE	0.001 (ajuste con ReduceLROnPlateau)
BATCH SIZE	32
ÉPOCAS MÁXIMAS	20 (con EarlyStopping)
FUNCIÓN DE COSTO	sparse_categorical_crossentropy
OPTIMIZADOR	Adam

Fuente: Elaboración Propia

Callbacks Implementados:

- **EarlyStopping:** Detiene el entrenamiento si no mejora el *val_loss* después de 5 épocas.
- **ReduceLROnPlateau:** Reduce la tasa de aprendizaje automáticamente si el *val_loss* no mejora tras 3 épocas consecutivas.

3.6. Implementación del Pipeline

El desarrollo de la solución se implementó como un **pipeline** modular, permitiendo claridad y replicabilidad. Las etapas del pipeline son:

1. **Cargar y Preprocesar los Datos:**
 - Balanceo con Class Weights.
 - Aumento de datos y normalización.
2. **Definir la Arquitectura del Modelo:**
 - Creación de la CNN con TensorFlow/Keras.
3. **Entrenamiento del Modelo:**
 - Uso de fit() con los datos de entrenamiento y validación.
 - Aplicación de **Class Weights** en el proceso de entrenamiento.
4. **Evaluación:**
 - Se incluye validación cruzada y evaluación final.

En este capítulo se planteó la solución propuesta utilizando un modelo CNN para clasificar hojas enfermas y sanas. El uso de un entorno de ejecución (Paperspace), técnicas de balanceo de clases (Class Weights), preprocesamiento de datos y la implementación de una arquitectura optimizada permiten validar esta solución en el siguiente capítulo.

CAPÍTULO 4: VALIDACIÓN DE LA SOLUCIÓN

En este capítulo se presentan los resultados obtenidos durante el proceso de entrenamiento y validación del modelo CNN implementado. Se analizan las métricas de desempeño y se discuten los resultados obtenidos en el entorno de ejecución robusto proporcionado por **Paperspace**, utilizando el dataset reestructurado de **PlantVillage**.

4.1. Resultados del Entrenamiento

4.1.1. Proceso de Entrenamiento

El modelo fue entrenado utilizando una arquitectura CNN, con un ajuste de hiperparámetros enfocado en optimizar la tasa de aprendizaje y la configuración de class weights para abordar el desbalance del dataset. La configuración final del entrenamiento fue:

- **Tamaño de Imagen:** 128x128 píxeles.
- **Batch Size:** 32.
- **Optimización:** Adam con tasa de aprendizaje inicial de 0.0005.
- **Class Weights:**
 - Clase *enfermas*: 0.68.
 - Clase *sanas*: 1.83.
- **Número de Épocas:** 15.

4.1.2. Curvas de Entrenamiento

Las siguientes figuras muestran la evolución de la precisión (*accuracy*) y la pérdida (*loss*) tanto en el conjunto de entrenamiento como en el de validación a lo largo de las 15 épocas.

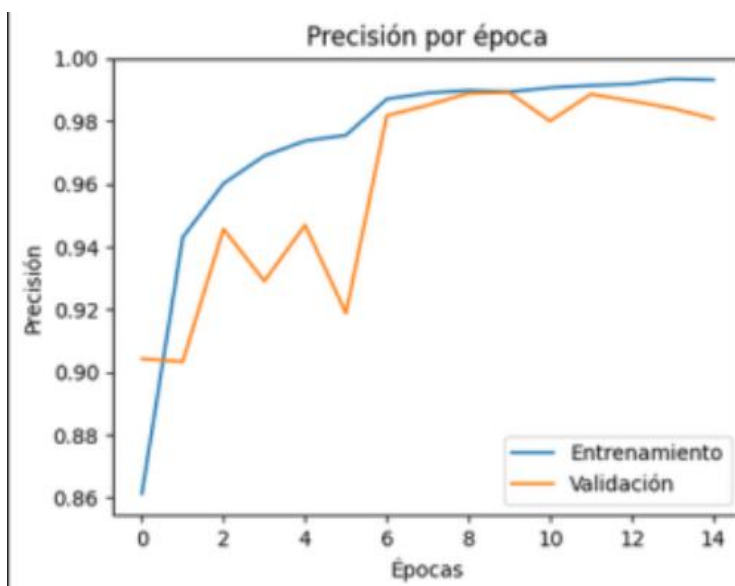


Figura 6. Precisión por Época

Fuente: Elaboración Propia

La precisión se estabiliza cerca del 99% en el entrenamiento y alrededor del 98% en la validación.

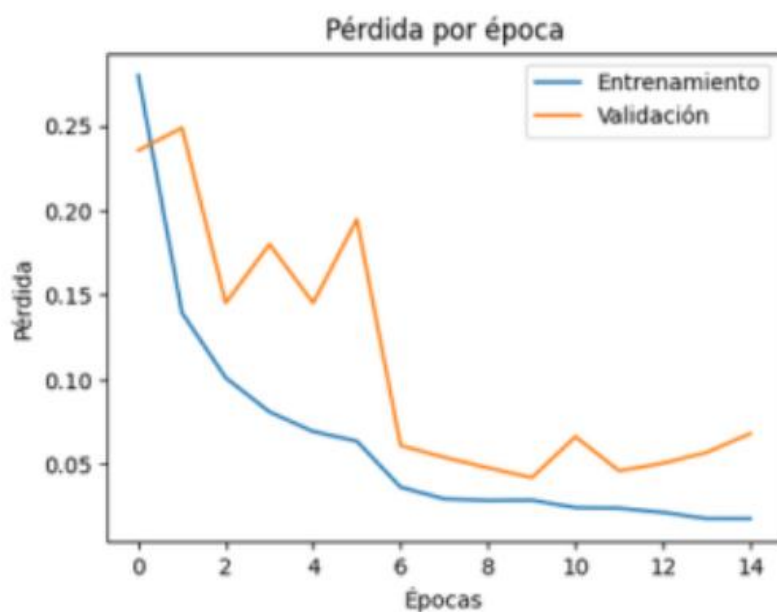


Figura 7. Pérdida por Época

Fuente: Elaboración Propia

Se observa una disminución progresiva de la pérdida en ambas fases, con ligeras fluctuaciones en la validación.

4.2. Evaluación del Modelo

4.2.1. Matriz de Confusión

La siguiente matriz de confusión ilustra el desempeño del modelo en el conjunto de validación. Se puede observar una clasificación efectiva de las clases "enfermas" y "sanas", con una precisión y *recall* superiores al 98%.

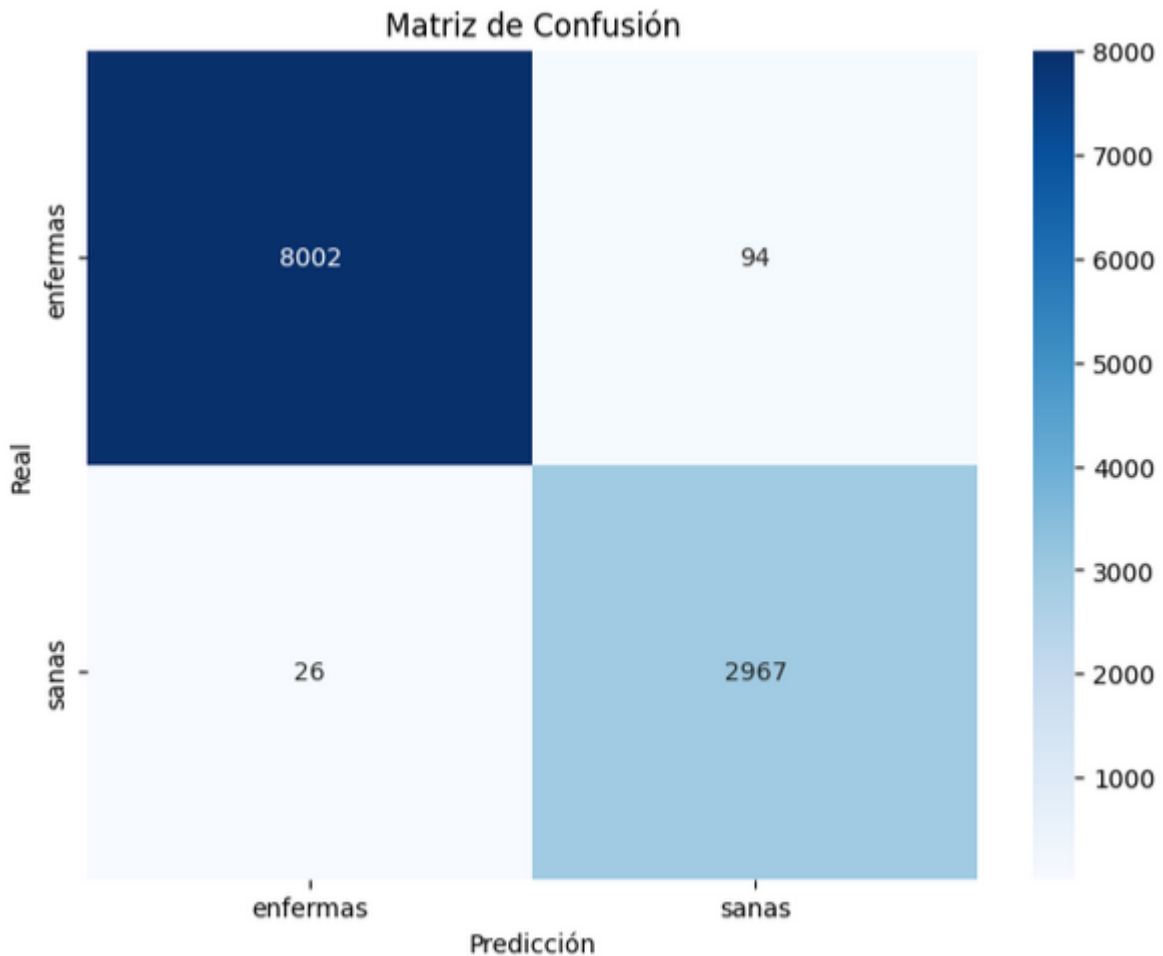


Figura 8. Matriz de Confusión

Fuente: Elaboración Propia

4.2.2. Métricas de Clasificación

El informe de clasificación resume las métricas más relevantes:

Tabla 4. Informe de Clasificación

MÉTRICA	ENFERMAS	SANAS	PROMEDIO PONDERADO
PRECISIÓN	1.00	0.97	0.99
RECALL	0.99	0.99	0.99
F1-SCORE	0.99	0.98	0.99
ACCURACY			0.99

Fuente: Elaboración Propia

Estas métricas indican un desempeño robusto del modelo en la tarea de clasificación binaria.

4.3. Validación de Predicciones

Se realizó una validación visual de las predicciones utilizando un subconjunto del conjunto de validación. A continuación, se muestran ejemplos representativos:

4.3.1. Predicciones Correctas

El modelo clasificó correctamente las siguientes imágenes:

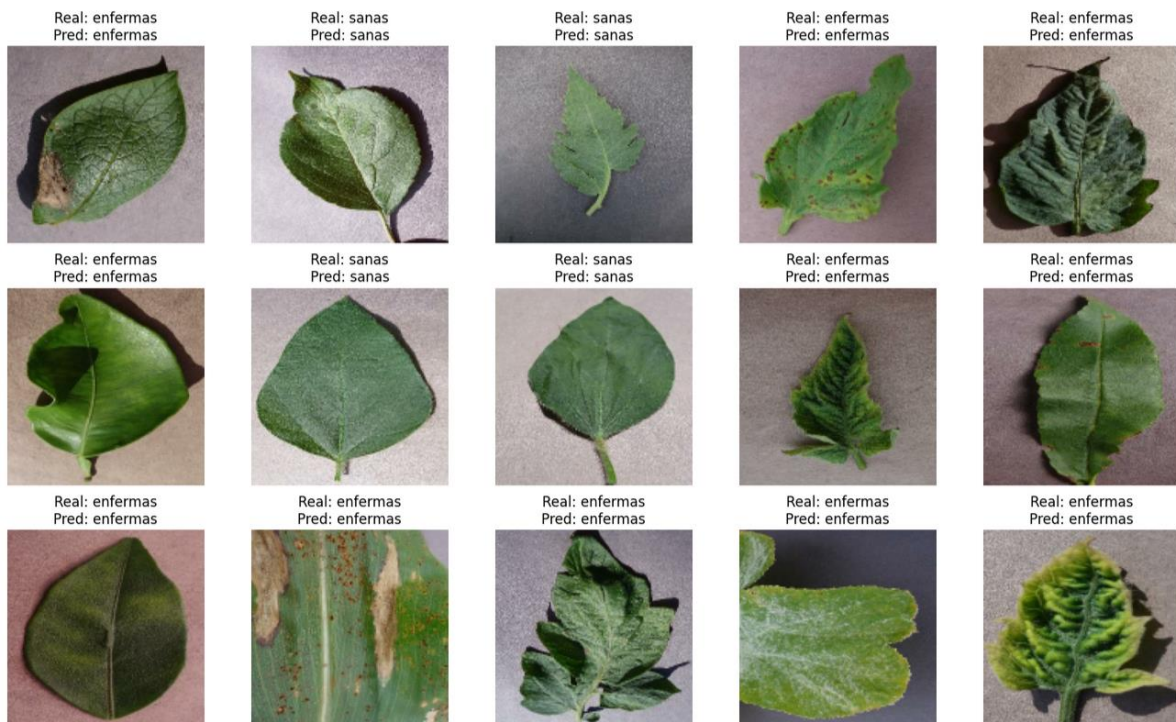


Figura 9. Predicciones Correctas

Fuente: Elaboración Propia

4.3.2. Predicciones Incorrectas

A pesar del alto desempeño general, el modelo presenta algunos errores. En las siguientes imágenes se observa la clasificación errónea:

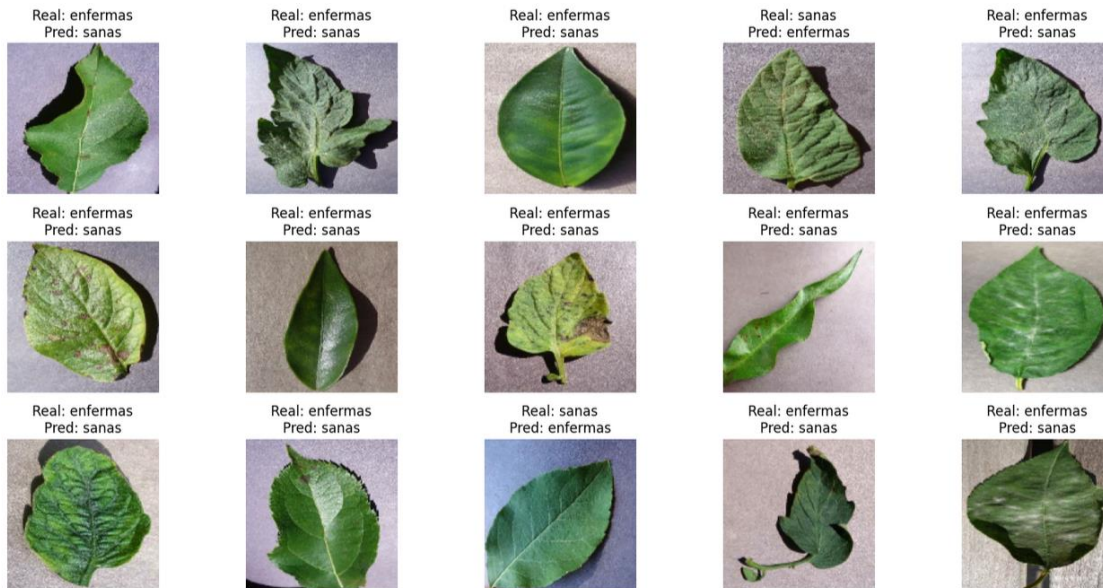


Figura 10. Predicciones Incorrectas

Fuente: Elaboración Propia

4.4. Discusión de Resultados

El análisis de los resultados muestra que el modelo CNN ha alcanzado un desempeño sobresaliente con una precisión del 99%, superando ampliamente las expectativas iniciales. La inclusión de *class weights* fue clave para manejar el desbalance del dataset, logrando mejoras significativas en la clasificación de la clase minoritaria (sanas).

No obstante, las fluctuaciones en la pérdida de validación sugieren posibles indicios de sobreajuste en las etapas finales del entrenamiento. Este desafío podría abordarse implementando técnicas complementarias como la validación cruzada o ajustes más detallados de los hiperparámetros.

El flujo descrito en el **Anexo C** corresponde a una planificación futura para la implementación del modelo CNN en la arquitectura de GreenGuardian. Actualmente, el modelo ha sido validado en un entorno simulado, y se proyecta su integración operativa como siguiente etapa del desarrollo.

El modelo final propuesto satisface los objetivos planteados al inicio del proyecto, demostrando una alta precisión en la clasificación de hojas en dos categorías: sanas y enfermas. La solución es sólida y confirma la efectividad de la arquitectura CNN para resolver problemas de clasificación en contextos agrícolas.

CAPÍTULO 5: CONCLUSIONES

El presente trabajo abordó el problema de detección temprana de enfermedades en cultivos hidropónicos mediante el uso de un modelo CNN integrado dentro del ecosistema GreenGuardian. A través de un enfoque sistemático, se logró diseñar, implementar y validar una solución que optimiza el manejo de los cultivos y contribuye a la sostenibilidad agrícola.

5.1 Resultados Principales

5.1.1. Reestructuración del Dataset PlantVillage:

El dataset original fue reconfigurado a dos clases (sanas y enfermas), garantizando la representatividad de las imágenes relevantes para los cultivos hidropónicos. El uso de técnicas de aumento de datos y **class weights** permitió superar el desbalance inicial entre ambas clases (40,364 enfermas vs. 15,084 sanas).

5.1.2. Optimización del Modelo CNN:

El modelo final, entrenado en un entorno robusto proporcionado por **Paperspace**, alcanzó un desempeño sobresaliente:

- **Precisión en validación:** 98.92%.
- **Matriz de Confusión:** Alta capacidad de detección tanto para plantas enfermas (recall de 0.99) como sanas (precision de 0.97), minimizando los falsos positivos y negativos.

5.1.3. Pipeline de Preprocesamiento de Imágenes:

Se implementó un pipeline efectivo que incluyó:

- Normalización y redimensionamiento.
- Técnicas avanzadas de aumento de datos (volteo, rotación, zoom).
- Uso de **pesos de clase** para equilibrar el impacto del desbalance en el entrenamiento.

5.1.4. Integración Propuesta en GreenGuardian:

El sistema se diseñó para proporcionar:

- **Monitoreo automatizado** mediante cámaras conectadas.
- **Alertas en tiempo real** a través de la aplicación móvil.
- Una solución **escalable y adaptable** para distintos tamaños de cultivo.

El modelo ha demostrado su eficacia en un entorno simulado, pero su implementación operativa requerirá la integración en el sistema completo descrito en el **Anexo C**, así como pruebas en condiciones reales

5.2 Beneficios Proyectados de la Solución

El sistema automatizado propuesto se proyecta como una herramienta con el potencial de generar los siguientes beneficios en la agricultura hidropónica:

- **Reducción de pérdidas económicas:** Se espera que, al detectar enfermedades en etapas tempranas, permita intervenciones oportunas y evite pérdidas significativas de producción, estimadas hasta en un 40%.
- **Optimización del tiempo:** El monitoreo continuo podría sustituir la subjetividad del diagnóstico manual, agilizando el proceso de toma de decisiones.
- **Escalabilidad y adaptabilidad:** La solución está diseñada para ser implementada en diversos escenarios, desde pequeños agricultores hasta explotaciones comerciales, lo que incentivaría el uso de tecnologías avanzadas en el sector agrícola.

5.3 Limitaciones del Trabajo

A pesar de los logros alcanzados, se identifican ciertas limitaciones:

- **Dependencia de la calidad de las imágenes:** El modelo requiere imágenes claras y bien iluminadas para garantizar una clasificación precisa.
- **Dataset limitado:** Aunque el modelo fue entrenado con éxito en dos clases, la detección de enfermedades específicas requerirá datasets más amplios y detallados en el futuro.

5.4 Proyecciones a Futuro

Como proyección, el sistema puede ser mejorado e implementado en nuevas áreas:

1. **Ampliación del Dataset:** Incorporar nuevas clases de enfermedades y tipos de cultivos permitirá un sistema más robusto y generalizable.
2. **Despliegue en Campo Real:** Validar el modelo en entornos operativos permitirá ajustar su rendimiento y evaluar su impacto en tiempo real.
3. **Optimización del Modelo:** Implementar técnicas más avanzadas, como **transfer learning** o modelos más complejos como YOLO, para mejorar aún más la precisión.
4. **Integración con otros sensores:** Combinar la detección visual con datos de sensores (pH, temperatura, humedad) permitirá diagnósticos más completos y precisos.

Aunque el modelo CNN ha demostrado un alto desempeño en un entorno simulado, la siguiente etapa implica su validación en condiciones operativas reales. La arquitectura planteada en el **Anexo C** representa un flujo técnico detallado para lograr una implementación escalable y eficiente en sistemas de producción.

REFERENCIAS BIBLIOGRÁFICAS

Indicarlas en orden alfabético por autor, por ejemplo:

Banco Mundial. (2023). *Agricultura, silvicultura y pesca, valor agregado (% del PIB) - Chile*. Recuperado de <https://datos.bancomundial.org/indicador/NV.AGR.TOTL.ZS?locations=CL>

Diario del Agro. (2023). *Hidroponía y Resistencia a Enfermedades: Estrategias Preventivas*. Recuperado de <https://www.diariodelagro.cl/hidroponia-y-resistencia-a-enfermedades-estrategias-preventivas/>

Food and Agriculture Organization of the United Nations. (n.d.). *Manejo de plagas y enfermedades en cultivos hidropónicos e invernaderos*. Portal Frutícola. Recuperado de <https://www.portalfruticola.com/noticias/2016/12/04/manejo-de-plagas-y-enfermedades-en-cultivos-hidroponicos-e-invernaderos/>

González-Muñiz, A. (2023). *Aplicación de técnicas de aprendizaje profundo (deep learning) al análisis y mejora de la eficiencia en sistemas de ingeniería* (Tesis de doctorado). Universidad de Oviedo. <https://doi.org/10.13140/RG.2.2.29722.82888>

Hughes, D. P., & Salathé, M. (2015). An open access repository of images on plant health to enable the development of mobile disease diagnostics. PlantVillage. Retrieved from <https://plantvillage.psu.edu>

Silva, J. C., & González, A. (2021). La agricultura chilena en el contexto internacional: desafíos y oportunidades. *Revista Chilena de Agricultura Sostenible*, 13(3), 15-25. Recuperado de https://www.scielo.cl/scielo.php?pid=S2452-57312021000300171&script=sci_arttext

Universidad de Puerto Rico. (n.d.). *Clínica de diagnóstico de enfermedades en hidroponía*. Centro de Información de la Agricultura. Recuperado de <https://www.uprm.edu/ciag/sea/clinica/CIdiaEnfHidrop.pdf>

ANEXOS

Anexo A: Configuración del Entorno de Ejecución

Información del Sistema

- **Sistema Operativo:** Linux 5.19.0-45-generic
- **Arquitectura:** 64-bit
- **Procesador:** x86_64
- **Núcleos CPU:** 8 (Lógicos y Físicos)
- **Memoria RAM Total:** 31.53 GB

Información de la GPU

- **Nombre:** Quadro P4000
- **Memoria Total:** 8192.0 MB
- **Memoria Libre:** 175.0 MB

Información de TensorFlow

- **Versión de TensorFlow:** 2.15.0
- **Disponibilidad CUDA:** Sí
- **Número de GPUs Disponibles:** 1 GPU

Librerías Instaladas

Tabla 5. Librerías Instaladas

Fuente: Elaboración Propia

LIBRERÍA	VERSIÓN
TENSORFLOW	2.15.0
KERAS	2.15.0
NUMPY	1.26.3
MATPLOTLIB	3.7.3
SCIKIT-LEARN	1.3.0
PANDAS	2.2.0
TORCH	2.1.1+cu121
TORCHVISION	0.16.1+cu121
WANDB	0.15.10
OPENCV	4.8.0.76
GPUUTIL	1.4.0

Notas sobre el Entorno:

- El entorno de ejecución fue configurado en Paperspace Gradient, utilizando Python 3.11 con soporte para CUDA 12.1.
- La GPU NVIDIA Quadro P4000 fue utilizada para el entrenamiento del modelo CNN, permitiendo el procesamiento eficiente de imágenes.
- La gestión de paquetes se realizó a través de pip, asegurando la instalación de todas las librerías necesarias.

Este anexo detalla la configuración técnica utilizada para el desarrollo y ejecución del modelo CNN propuesto, permitiendo replicar el entorno de trabajo para validaciones adicionales.

Anexo B: Pipeline de Preprocesamiento

Este anexo complementa la descripción del pipeline presentada en el Capítulo 2, proporcionando el código implementado y el esquema detallado del proceso.

Preprocesamiento de Imágenes

El preprocesamiento es fundamental para garantizar la calidad de las imágenes antes de ser ingresadas al modelo. En esta etapa, se aplican transformaciones tanto al conjunto de entrenamiento como al conjunto de validación:

Aumentos de Datos (solo en entrenamiento):

- Volteo horizontal.
- Rotación aleatoria hasta 10%.
- Zoom aleatorio hasta 10%.
- Ajuste de contraste aleatorio.

Normalización (entrenamiento y validación):

- Escalamiento de los valores de píxeles al rango [0, 1] mediante la fórmula:

$$X_{\text{Normalizado}} = X/255$$

Implementación del Pipeline

El código utilizado para el preprocesamiento es el siguiente:

```
import tensorflow as tf
from tensorflow.keras import layers

# Configuración del dataset
IMG_SIZE = (128, 128)
BATCH_SIZE = 32

def load_and_preprocess_data(dataset_path):
    """
    Carga y preprocesa el dataset aplicando aumentos de datos y normalización.
    """
    # Cargar el dataset
    train_ds = tf.keras.utils.image_dataset_from_directory(
        dataset_path, validation_split=0.2, subset="training", seed=123,
        image_size=IMG_SIZE, batch_size=BATCH_SIZE)

    val_ds = tf.keras.utils.image_dataset_from_directory(
        dataset_path, validation_split=0.2, subset="validation", seed=123,
        image_size=IMG_SIZE, batch_size=BATCH_SIZE)

    # Preprocesamiento: aumentos y normalización
    aug_layer = tf.keras.Sequential([
        layers.RandomFlip("horizontal"),
        layers.RandomRotation(0.1),
        layers.RandomZoom(0.1),
        layers.RandomContrast(0.1),
    ])
    norm_layer = layers.Rescaling(1./255)

    # Aplicar al conjunto de entrenamiento y validación
    train_ds = train_ds.map(lambda x, y: (norm_layer(aug_layer(x)), y))
    val_ds = val_ds.map(lambda x, y: (norm_layer(x), y))

    return train_ds, val_ds
```

Figura 11. Código del Preprocesamiento

Fuente: Elaboración Propia

Resumen

Cargar el Dataset:

- División: 80% entrenamiento y 20% validación.

Aumentos de Datos (solo en entrenamiento):

- Volteo horizontal, rotación, zoom y ajuste de contraste.

Normalización:

- Valores de píxeles escalados al rango [0, 1].

El siguiente ejemplo muestra imágenes preprocesadas con aumentos aplicados:

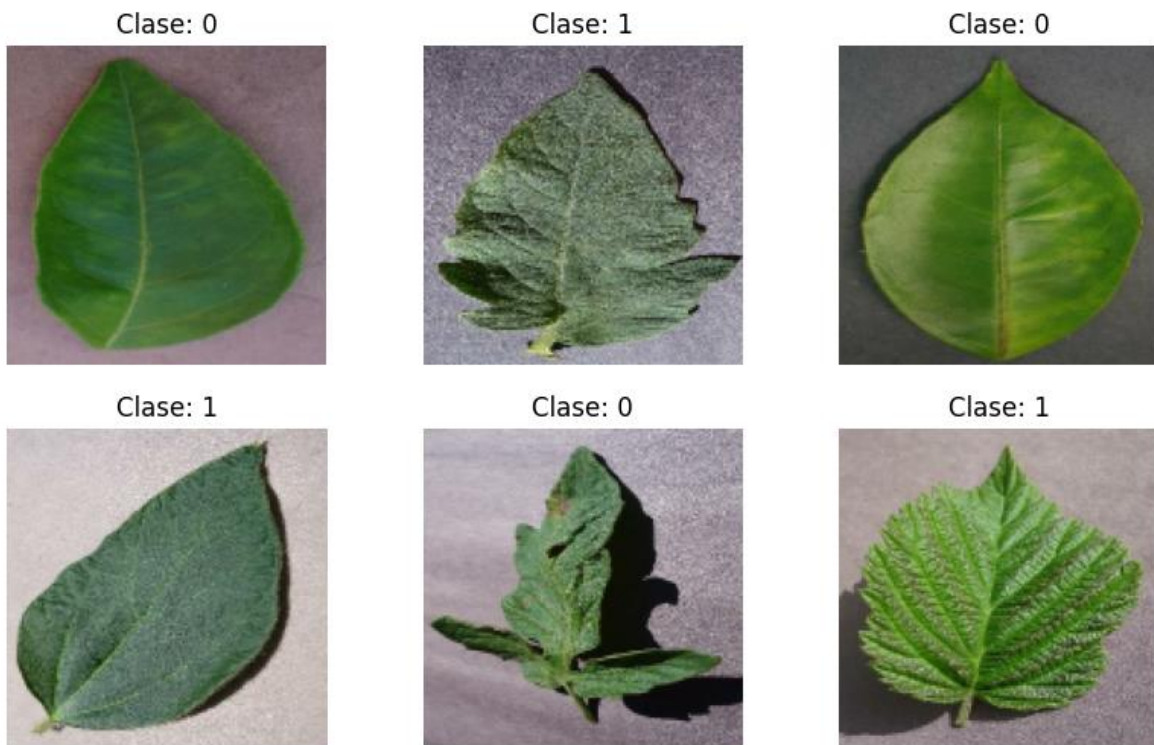


Figura 12. Imágenes Preprocesadas

Fuente: Elaboración Propia

Este pipeline garantiza un flujo eficiente desde la carga de datos hasta su preprocesamiento para el modelo. La inclusión de aumentos de datos permite mejorar la **generalización** del modelo y reducir el riesgo de **sobreajuste**.

Anexo C: Arquitectura del Modelo CNN en GreenGuardian

El presente anexo ofrece un desglose técnico y visual del flujo completo propuesto para la integración del modelo CNN en la arquitectura GreenGuardian. Se detalla la comunicación entre sensores, servicios en la nube, el modelo de predicción y la aplicación móvil.

Explicación del Flujo

Infraestructura Cloud:

- Los datos se originan desde los sensores ESP32, que capturan imágenes y otros datos relevantes (temperatura, pH, etc.) desde el entorno hidropónico.
- Los datos son enviados mediante el protocolo MQTT hacia la nube, donde se almacenan temporalmente en una base de datos AWS IoT.
- La información es preprocesada y preparada para ser consumida por el microservicio de predicción.

Microservicio de Predicción (CNN GreenGuardian):

- Procesamiento de Imágenes: Las imágenes son preprocesadas utilizando normalización y redimensionamiento, acorde a lo explicado en el pipeline del Capítulo 2.
- Modelo CNN: El modelo CNN es cargado para realizar la clasificación de las imágenes entre "sanas" y "enfermas".
- Registro y Monitoreo: Las predicciones generadas son almacenadas en un microservicio encargado de llevar un historial.
- Bucle de Retroalimentación: El sistema puede ser reajustado con nuevas imágenes etiquetadas para mejorar continuamente el rendimiento del modelo.

Base de Datos GreenGuardian:

La información de las predicciones es almacenada en la base de datos PostgreSQL, permitiendo:

- Gestión del historial de predicciones.
- Validación y análisis de datos a futuro.

Los datos son protegidos utilizando encriptación y mecanismos de autenticación basados en JWT.

Servicios Front-End:

- Los resultados de las predicciones se envían al microservicio de App Móvil para generar alertas y mostrar la información en tiempo real.
- Las notificaciones son procesadas en la infraestructura de GreenGuardian y visualizadas en aplicaciones móviles y web.
- Los usuarios pueden acceder al historial de predicciones y otros datos relevantes mediante interfaces intuitivas.

API Gateway:

- Actúa como el punto de acceso centralizado, permitiendo la comunicación entre los microservicios y garantizando la correcta gestión de solicitudes y respuestas.

Visualización de la Arquitectura

La arquitectura completa del flujo de datos e integración del modelo CNN en el sistema **GreenGuardian** se encuentra representada en la **Figura 2.3** del **Capítulo 2**.

Es importante señalar que esta arquitectura corresponde a un **diseño propuesto** para la integración final del modelo en el entorno GreenGuardian. Actualmente, el modelo CNN fue validado en un entorno controlado y simulado, y **no ha sido implementado aún en el flujo completo de producción**. El objetivo de esta propuesta es mostrar la **viabilidad técnica** y los pasos necesarios para su futura implementación.

Beneficios de la Arquitectura

- **Escalabilidad:** Permite manejar múltiples sensores y grandes volúmenes de datos.
- **Eficiencia:** Clasificación rápida y automatizada de imágenes, reduciendo el tiempo de diagnóstico.
- **Monitoreo en Tiempo Real:** Proporciona alertas automáticas a los usuarios para intervenciones oportunas.
- **Historial de Predicciones:** Facilita el análisis de tendencias y el ajuste del modelo en el futuro