UTMACH

FACULTAD DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

DESARROLLO DE UNA APLICACIÓN PARA LA DETECCIÓN DE PLAGAS EN EL CULTIVO DE PAPAS APLICANDO REDES NEURONALES CONVOLUCIONALES

MACAS BERMEO JOSE RONALDO
INGENIERO DE SISTEMAS

MACHALA
2022

UTMACH

FACULTAD DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

DESARROLLO DE UNA APLICACIÓN PARA LA DETECCIÓN DE PLAGAS EN EL CULTIVO DE PAPAS APLICANDO REDES NEURONALES CONVOLUCIONALES

MACAS BERMEO JOSE RONALDO
INGENIERO DE SISTEMAS

MACHALA
2022

UTMACH

FACULTAD DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

TRABAJO TITULACIÓN
PROPUESTAS TECNOLÓGICAS

DESARROLLO DE UNA APLICACIÓN PARA LA DETECCIÓN DE PLAGAS EN EL CULTIVO DE PAPAS APLICANDO REDES NEURONALES CONVOLUCIONALES

MACAS BERMEO JOSE RONALDO
INGENIERO DE SISTEMAS

RIVAS ASANZA WILMER BRAULIO

MACHALA, 20 DE SEPTIEMBRE DE 2022

MACHALA
2022

josemacas

INFORME DE ORIGINALIDAD

7%	8%	1%	1%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE

FUENTES PRIMARIAS

1	vsip.info Fuente de Internet	1%
2	hdl.handle.net Fuente de Internet	1%
3	dspace.utb.edu.ec Fuente de Internet	1%
4	Submitted to Universidad Internacional de la Rioja Trabajo del estudiante	<1%
5	repositorio.escuelaing.edu.co Fuente de Internet	<1%
6	oa.upm.es Fuente de Internet	<1%
7	www.dykinson.com Fuente de Internet	<1%
8	www.elsevier.es Fuente de Internet	<1%
9	repositorio.ucv.edu.pe Fuente de Internet	

CLÁUSULA DE CESIÓN DE DERECHO DE PUBLICACIÓN EN EL REPOSITORIO DIGITAL INSTITUCIONAL

El que suscribe, MACAS BERMEO JOSE RONALDO, en calidad de autor del siguiente trabajo escrito titulado DESARROLLO DE UNA APLICACIÓN PARA LA DETECCIÓN DE PLAGAS EN EL CULTIVO DE PAPAS APLICANDO REDES NEURONALES CONVOLUCIONALES, otorga a la Universidad Técnica de Machala, de forma gratuita y no exclusiva, los derechos de reproducción, distribución y comunicación pública de la obra, que constituye un trabajo de autoría propia, sobre la cual tiene potestad para otorgar los derechos contenidos en esta licencia.

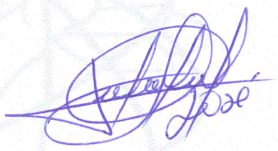
El autor declara que el contenido que se publicará es de carácter académico y se enmarca en las dispociones definidas por la Universidad Técnica de Machala.

Se autoriza a transformar la obra, únicamente cuando sea necesario, y a realizar las adaptaciones pertinentes para permitir su preservación, distribución y publicación en el Repositorio Digital Institucional de la Universidad Técnica de Machala.

El autor como garante de la autoría de la obra y en relación a la misma, declara que la universidad se encuentra libre de todo tipo de responsabilidad sobre el contenido de la obra y que asume la responsabilidad frente a cualquier reclamo o demanda por parte de terceros de manera exclusiva.

Aceptando esta licencia, se cede a la Universidad Técnica de Machala el derecho exclusivo de archivar, reproducir, convertir, comunicar y/o distribuir la obra mundialmente en formato electrónico y digital a través de su Repositorio Digital Institucional, siempre y cuando no se lo haga para obtener beneficio económico.

Machala, 20 de septiembre de 2022



MACAS BERMEO JOSE RONALDO
0706127743

DEDICATORIA

La presente tesis está dedicada en primer lugar a Dios, por brindarme la inteligencia, salud y sabiduría para terminar con éxito mi carrera como estudiante, además por estar conmigo en todo momento, permitiéndome vivir, ya que sin su presencia en mi vida nada de esto sería posible. En segundo lugar, voy a referirme a mis padres, por ser uno de los pilares importantes en mi vida, por todos sus sacrificios, enseñanzas, valores, amor incondicional, apoyo que, en todo momento recibí, y sobre todo dándome sus concejos para que cada día sea una mejor persona.

A mis queridos hermanos, por estar conmigo sin importar la situación, apoyándome y brindándome su mano amiga cada vez que lo he necesitado.

A mi amigo Wilson Yánez, que más que un amigo es como un hermano para mí, porque durante toda la carrera estuvo conmigo motivándome a seguir adelante, a no rendirme, bridándome su apoyo incondicional en todo momento.

Sr. José Ronaldo Macas Bermeo.

AGRADECIMIENTO

En primer lugar, quiero agradecer a Dios por darme la fortaleza todos los días para seguir adelante, y alcanzar mis sueños; por darme cada día la sabiduría y oportunidades para lograr desempeñarme exitosamente tanto en las actividades académicas como personales.

Mi familia, quienes con su amor, empeño y dedicación me han brindado su apoyo durante todas mis actividades académicas. Pero, sobre todo, quiero agradecer inmensamente a mi Madre y a mi Padre, quienes cada día me motivan a seguir adelante, me brindan su apoyo incondicional, y sobre todo son mi pilar fundamental, sabiéndome guiar con sus consejos tanto en mi vida personal como académica.

Y finalmente, quiero expresar mis agradecimientos hacia mi tutor académico, el Ing. Wilmer Braulio Rivas Asanza, quien con sus conocimientos me ha sabido guiar en la realización de mi tesis.

RESUMEN

En los últimos años el desarrollo de aplicaciones inteligentes se ha intensificado considerablemente, haciendo que resolver tareas complejas o tareas que requieren grandes esfuerzos sean más sencillos de realizar. Diferentes investigaciones muestran que la inteligencia artificial desenvuelve un papel muy importante dentro de las distintas áreas como la medicina, la robótica, la educación, la agricultura, la matemática, etc., que gracias a las ventajas que ofrece este tipo de tecnología, ha permitido encontrar soluciones más eficientes y eficaces a las necesidades de la población, mediante la automatización de procesos.

Recientes estudios realizados acerca del uso de la Inteligencia Artificial en la agricultura, establecen que su uso ha permitido mejorar e innovar los sistemas tradicionales de cultivo, riego, identificación de suelos, identificación de malezas, donde los beneficios se ven reflejados en el aumento de la producción de cultivos de calidad.

La agricultura es una de las áreas de estudio más antiguas en la humanidad y por lo tanto se la considera un pilar fundamental tanto en la alimentación de miles de personas como en la economía interna y externa de un país, por ende, el producir productos de calidad es una necesidad importante para los agricultores.

La papa es uno de los productos con mayor demanda en muchos países y es uno de los sembríos más cultivados a nivel nacional, tanto en pequeñas escalas como grandes escalas. Comúnmente el cultivo de papa se ve afectado por distintos factores, que causan la aparición de agentes patógenos, los mismos que provocan enfermedades severas. Por eso el identificar a tiempo las plagas en el cultivo de papa es clave para evitar pérdidas económicas. La forma más rápida y segura de identificación de estas plagas es por medio del uso de la tecnología de la inteligencia artificial.

Con la finalidad de apoyar a los agricultores a tener un mejor manejo saludable del cultivo se pretende implementar una aplicación, usando herramientas de inteligencia artificial para la identificación de plagas con gran precisión en el cultivo de papa mediante el uso redes neuronales convolucionales CNN.

Para llevar a cabo este proyecto se empleó la herramienta Jupyter Notebook para el diseño de la red neuronal convolucional, Visual Studio Code y PyCharm como editores de código fuente y librerías como TensorFlow, Uvicorn, FastApi entre otras.

Durante la construcción de la red neuronal se optó por dividir los datos, un 80% de los datos para entrenamiento, 10% para la validación y el 10% restante se empleó para la evaluación del modelo, con la finalidad de obtener resultados satisfactorios. Para el diseño de la aplicación en este caso una aplicación web se optó por usar React Js, la cual se acopla perfectamente a nuestras necesidades, ya que nos permite implementar interfaces de usuario de manera sencilla y rápida. Para la validación del modelo implementado se realizaron múltiples pruebas y mediante el uso de la matriz de confusión se logró calcular las métricas de rendimiento del prototipo, obteniendo un promedio de precisión del 90%.

Gracias a los resultados obtenidos de la aplicación con datos reales, se concluye que nuestro modelo presenta un nivel de precisión aceptable y confiable, con una precisión superior a la media necesaria para considerar valido el modelo.

Palabras claves: Redes neuronales convolucionales, plagas, Inteligencia artificial, aplicación web.

ABSTRACT

In recent years, the development of intelligent applications has intensified considerably, making it easier to solve complex tasks or tasks that require great effort. Different investigations show that artificial intelligence plays a very important role in different areas such as medicine, robotics, education, agriculture, mathematics, etc., which, thanks to the advantages offered by this type of technology, has allowed find more efficient and effective solutions to the needs of the population, by automating processes.

Recent studies carried out on the use of Artificial Intelligence in agriculture, establish that its use has allowed to improve and innovate traditional systems of cultivation, irrigation, soil identification, weed identification, where the benefits are reflected in the increase in production of quality crops.

Agriculture is one of the oldest areas of study in humanity and therefore it is considered a fundamental pillar both in the feeding of thousands of people and in the internal and external economy of a country, therefore, producing products of Quality is an important need for farmers.

The potato is one of the products with the highest demand in many countries and is one of the most cultivated crops at the national level, both on small and large scales.

Commonly, potato cultivation is affected by different factors that cause the appearance of pathogenic agents, the same ones that cause severe diseases. That is why identifying pests in potato crops in time is key to avoiding economic losses. The fastest and safest way to identify these pests is through the use of artificial intelligence technology.

In order to support farmers to have a better healthy management of the crop, it is intended to develop an application, using artificial intelligence tools for the identification of pests with great precision in the potato crop through the use of CNN convolutional neural networks.

To carry out this project, the Jupyter Notebook tool was used for the design of the convolutional neural network, Visual Studio Code and PyCharm as source code editors and libraries such as TensorFlow, Uvicorn, FastApi among others.

During the construction of the neural network, it was decided to divide the data, 80% of the data for training, 10% for validation and the remaining 10% was used for the evaluation of the model, in order to obtain satisfactory results. For the design of the application in this case, a web application was chosen to use React Js, which fits perfectly to our needs, since it allows us to implement user interfaces in a simple and fast way. For the validation of the implemented model, multiple tests were carried out and by using the confusion matrix it was possible to calculate the performance metrics of the prototype, obtaining an average accuracy of 90%.

Thanks to the results obtained from the application with real data, it is concluded that our model presents an acceptable and reliable level of precision, with a precision higher than the average necessary to consider the model valid.

Keywords: Convolutional neural networks, diseases, artificial intelligence, web application.

INTRODUCCIÓN

La aplicación de las tecnologías emergentes es cada vez más frecuentemente empleada en los diferentes ámbitos de estudio y con ello promueven a la investigación e innovación científica. Las tecnologías como la inteligencia artificial, hoy en día, permiten encontrar soluciones más óptimas a las necesidades del ser humano, mediante el uso de técnicas de Machine Learning y Deep Learning.

Las técnicas como el aprendizaje profundo o Deep Learning se emplean para abordar los problemas más complejos relacionados con el contenido visual como lo relacionado con la identificación y detección de objetos [1]. Actualmente, la identificación de plagas en los cultivos tradicionales se realiza por medio de una simple inspección es decir de forma visual, donde los resultados adquiridos se ven afectados por el razonamiento y juicio propio de cada persona, dando origen a una ambigüedad de identificación.

El presente trabajo de investigación tienes como finalidad el desarrollo de una aplicación para la detección de plagas en el cultivo de papa usando redes neuronales convolucionales, mediante herramientas de inteligencia artificial como Machine Learning y Deep Learning.

Con la ayuda de la inteligencia artificial se plantea dar solución al problema de detección mediante el desarrollo de una aplicación y un modelo de red neuronal convolucional con la capacidad de predecir correctamente las plagas.

Durante el desarrollo del proyecto se presentaron dificultades en la recolección de muestras para el entrenamiento de nuestro modelo, razón por la cual se optó por usar un conjunto de datos de internet para realizar nuestro entrenamiento.

El presente documento consta de tres capítulos:

Capítulo 1: se describe los hechos de interés, la problemática del proyecto y se describen los requerimientos

Capítulo 2: se enfoca en la fundamentación teórica del proyecto, el diseño del prototipo y la definición de objetivos de la investigación, así como también el ensamblaje del prototipo.

Capítulo 3: se detallan los resultados obtenidos de las diferentes pruebas realizadas.

CONTENIDO

Índice de Figuras

Índice de Tablas

1. CAPÍTULO I. **DIAGNÓSTICO DE NECESIDADES Y REQUERIMIENTOS**

1.1. Ámbito de Aplicación: descripción del contexto y hechos de interés

La población global crece de manera incontenible, por lo tanto, el sector agrícola se ve en la necesidad de duplicar su nivel de producción con la finalidad de satisfacer la demanda poblacional [2]. Tomando en consideración el limitado uso de recursos naturales y la erosión de los suelos agrícolas, el agricultor tiende la necesidad de encontrar formas más óptimas y sostenibles de producción. Una solución a este problema es el uso de la inteligencia artificial [3].

La inteligencia artificial durante las últimas décadas ha tenido un gran impacto dentro de la agricultura. Este tipo de tecnologías permite mejorar una serie de tareas relacionadas con la agricultura tales como la detección de plagas y enfermedades, producción de cultivos sanos, monitorear los estados del suelo, predicción de cambios de clima, entre otros [2].

Es por ello que se ha propuesto desarrollar una aplicación para la detección de plagas en el cultivo de papa por medio de imágenes aplicando redes neuronales convolucionales.

Las redes neuronales convolucionales son modelos matemáticos que simulan el comportamiento de las neuronas de un cerebro biológico, la misma que permite realizar múltiples tareas tales como clasificar e identificar imágenes o diagnosticar enfermedades con alta exactitud y precisión [4].

Para la detección de plagas se utilizan las imágenes de hojas del cultivo de papa, por ello es importante recolectar imágenes de fuentes confiables para obtener resultados satisfactorios [5]. Citando lo anterior se puede concluir que la toma de imágenes se debe hacer con un dispositivo de gama alta.

En la agricultura detectar las plagas y enfermedades a tiempo es un factor clave para reducir las pérdidas y aumentar su producción de manera sustentable [6]. En base a lo expuesto anteriormente podemos decir que el detectar las plagas y enfermedades nos permitirá aumentar nuestros ingresos de forma rentable.

1.2. Establecimiento de Requerimientos

La inteligencia artificial ofrece un sin número de beneficios tanto como automatizar tareas como hacer cálculos predictivos. El reconocimiento de plagas y enfermedades en los cultivos es una labor muy complicada que requiere amplio tiempo, mano de obra y otros factores de propios de la agricultura. Para diseñar un modelo de red neuronal convolucional con la capacidad de reconocer las plagas mediante imágenes se necesitará de los siguientes requerimientos:

- API para el aprendizaje profundo.
- Imágenes de hojas de papas para crear nuestro dataset de entrenamiento.
- Variedad de muestras.

En caso de no poseer una dataset completa se necesitará una cámara de alta resolución para realizar fotografías.

En cuanto a los requerimientos necesarios para la aplicación destacan los siguientes:

- Un editor de texto
- Un entorno de desarrollo integrado como PyCharm.
- Desarrollo de una Interfaz de usuario amigable.
- Funcionar correctamente de forma local.
- La aplicación debe recibir como entrada una imagen.

Como requisitos complementarios, pero no menos importantes se destacan los siguientes:

- Las muestras deben tener un solo formato.
- Una herramienta para el testeo de la API durante el desarrollo.

En cuanto al modelo final su aprendizaje será limitado ya que solo se usará tres tipos de plagas que son los más comunes en el cultivo de papa para su entrenamiento. Pero se puede hacer nuevos entrenamientos con una dataset más robusta, es decir con más tipos de plagas.

1.3. Justificación de Requerimientos a Satisfacer

Uno de los grandes retos que enfrenta hoy en día el sector agrícola son las enfermedades causadas por plagas presentes en el cultivo, lo cual conduce a una baja productividad [7]. El detectar plagas y enfermedades en los sembríos de manera tradicional requiere de conocimientos, experiencia y tiempo. Normalmente en su mayoría las plagas y enfermedades se presentan en las hojas [8].

En las hojas se puede observar las plagas y enfermedades presentes en los sembríos de papas, convirtiéndola en una pieza de información clave para nuestra investigación. Considerando que para tener resultados más exactos es conveniente realizar un estudio más a profundidad.

El reconocer las enfermedades y plagas en el cultivo de papa desde un enfoque tradicional es decir a simple vista resulta una tarea complicada, y más aún para los nuevos productores y para aquellas personas con poco conocimiento sobre el tema por lo tanto, son ineficaces lo cual reduce la precisión en la identificación, llevando a subministrar el tratamiento equivocado, afectando directamente al crecimiento del cultivo y por ende afectando en su producción [9]. Además, el identificar plagas de manera visual se puede ver afectado por el grado de visión del ser humano.

En base a lo expuesto anteriormente, se presenta una buena justificación para desarrollar el presente trabajo de titulación, ya que se propone resolver el problema de detección e identificación de plagas en el cultivo de papá.

Además, el modelo implementado se aplica la técnica de aumento de datos para obtener características más complejas, lo que lo hace más preciso a la hora de detectar la plaga.

2. CAPÍTULO II. **DESARROLLO DEL PROTOTIPO**

2.1. Definición del prototipo tecnológico

El presente trabajo tecnológico se desarrolló en tres fases:

- En la primera fase se realizó la implementación de la estructura del modelo de red neuronal.
- En la segunda fase consiste en realizar pruebas para validar nuestro modelo.
- En la segunda fase se realizó la implementación de una aplicación web con una interfaz sencilla y fácil de usar.

El modelo de la red neuronal que se implementó, está basado en la estructura de las redes neuronales de tipo convolucional (CNNs). Y que se obtendrá, como resultado final un archivo keras, el mismo que contendrá todos los parámetros de entrenamiento, es decir tanto las variables como sus pesos de entrenamiento.

Una de las características presentes en el modelado es la robustez con la que se implementó, haciéndolo un modelo completo que se puede emplear para nuevas investigaciones.

Para la presentación de resultados se implementará una aplicación web de manera local, pero diseñado para trabajar en tiempo real en un futuro. Para la comunicación entre la aplicación web y nuestro modelo es mediante una API rápida que hoy en día nos ofrece las tecnologías actuales.

2.2. Fundamentación teórica del prototipo.

2.2.1. Marco Teórico

2.2.1.1. Plagas y enfermedades en el cultivo de papa

Las plagas y enfermedades representan una gran amenaza para los cultivos que llegan afectar significativamente a una parcela y en casos severos a toda la plantación, causando pérdidas tanto económicas como productivas [10]. Los cambios climáticos, las bajas temperaturas, el exceso de agua y el mal uso de plaguicidas, son algunos de los factores claves que favorecen el aumento y aparición de nuevas plagas y enfermedades dentro de la población de cultivos.

El suelo como recurso natural es uno de los más importantes, y considerado parte esencial en la producción agrícola, ya que influye directamente sobre el crecimiento de

las plantas [11]. Por ende, las condiciones de los suelos deben ser más los óptimos para la agricultura. Debido a que suelos muy húmedos son zonas adecuadas para el desarrollo de agentes patógenos y enfermedades.

Una de las enfermedades más comunes en el cultivo de papas es el tizón tardío que por décadas ha venido causando graves pérdidas de cultivos. Esta enfermedad ataca directamente al follaje de la planta siendo muy notorio un color amarillento o marrón oscuro en la hoja, sin el tratamiento adecuado para controlar esta enfermedad podría afectar en un 100% de la plantación [12].

2.2.1.2. Las tecnologías en la fitopatología.

Investigaciones recientes sobre el estudio de las enfermedades en las plantas, demuestran las diferentes formas de aplicar las tecnologías emergentes actuales.

- El Internet de las cosas (IoT), recopila información en tiempo real sobre las condiciones climáticas, humedad de los suelos o de la aparición de plagas y enfermedades en los cultivos, etc., con finalidad de aumentar el nivel y la calidad de producción agrícola [13].
- El aprendizaje profundo conjuntamente con las redes neuronales facilita el identificar y detectar las características de las plantas haciendo uso de imágenes multiespectrales para una mejor segmentación, logrando una precisión de hasta un 94% [14].
- Mediante Machine Learning (**ML**), predecir el estado de salud del banano y detectar las enfermedades en sus hojas mediante la fotogrametría y la teledetección [15].

2.2.1.3. La inteligencia artificial.

La inteligencia artificial (IA), se la puede definir como el conjunto de algoritmos que permiten a las máquinas tener la capacidad de entender y aprender a realizar tareas propias de una mente humana, de tal manera que puedan tomar decisiones propias en base a su aprendizaje adquirido. En pocas palabras la IA busca hacer posible que las máquinas puedan actuar y pensar como humanos [16].

El desarrollo de la inteligencia artificial involucra dos conceptos importantes como el razonamiento y el desarrollo cognitivo. Con la IA se pretende desarrollar herramientas informáticas que tenga la capacidad de tomar decisiones por iniciativa propia. En el ser humano el cerebro constantemente recibe miles y miles de datos, para luego

procesarlos, almacenarlos y en base a esos datos tomar decisiones en situaciones futuras. La toma de decisiones conlleva a un proceso lógico y secuencial, que se puede plasmar a algoritmos mediante un lenguaje, de tal manera que las máquinas puedan entender, interpretar y ejecutar con mucha más rapidez y mayor precisión que el ser humano [17].

Hoy en día el desarrollar dispositivos que tenga la capacidad de reflejar la inteligencia del ser humano ya no solo es un cuento es una realidad, que poco a poco se va adentrando en nuestro diario vivir, haciendo que realizar tareas complejas sea más sencillo. Son muchos los ámbitos en los que la inteligencia artificial ha tenido gran auge, entre ellos la robótica, la salud, videojuegos, entre otros.

Tabla 1 Principales ámbitos de aplicación de los sistemas de inteligencia artificial

Área	Aplicaciones
Medicina	Ayuda al diagnóstico Análisis de imágenes biomédicas Procesado de señales fisiológicas
Ingeniería	Organización de la producción Optimización de procesos Cálculo de estructuras Planificación y logística Diagnóstico de fallos Toma de decisiones
Economía	Análisis financiero y bursátil Análisis de riesgos Estimación de precios en productos derivados Minería de datos Marketing y fidelización de clientes
Biología	Análisis de estructuras biológicas Genética médica y molecular
Informática	Procesado de lenguaje natural Criptografía Teoría de juegos Lingüística computacional
Robótica y automática	Sistemas adaptativos de rehabilitación Interfaces cerebro-computadora Sistemas de visión artificial Sistemas de navegación automática
Física y matemáticas	Demostración automática de teoremas Análisis cualitativo sistemas no-lineales Caracterización de sistemas complejos

***Fuente**: Inteligencia artificial avanzada de R. Benítez, G. Escudero y S. Kanaan [18].*

2.2.1.4. Aprendizaje Automático (Machine Learning).

El Machine Learning o ML, tecnología que se ubica en el centro de la IA, es una de las ramas de la Inteligencia artificial enfocada al aprendizaje automático, que, por medio de un entrenamiento previo, provee la capacidad a las computadoras de inferir y aprender de manera independiente. Su principal ventaja es que permite la identificación de patrones en un conjunto de datos creando un modelo para luego realizar procesos como la predicción o la clasificación [19] [20].

Este tipo de tecnología es similar al proceso de aprendizaje cognitivo humano ya que tiene la facilidad de mejorar automáticamente en base a las experiencias de datos, sin la necesidad de la intervención humana [21].

Actualmente existen tres tipos de ML clasificados de la siguiente manera:

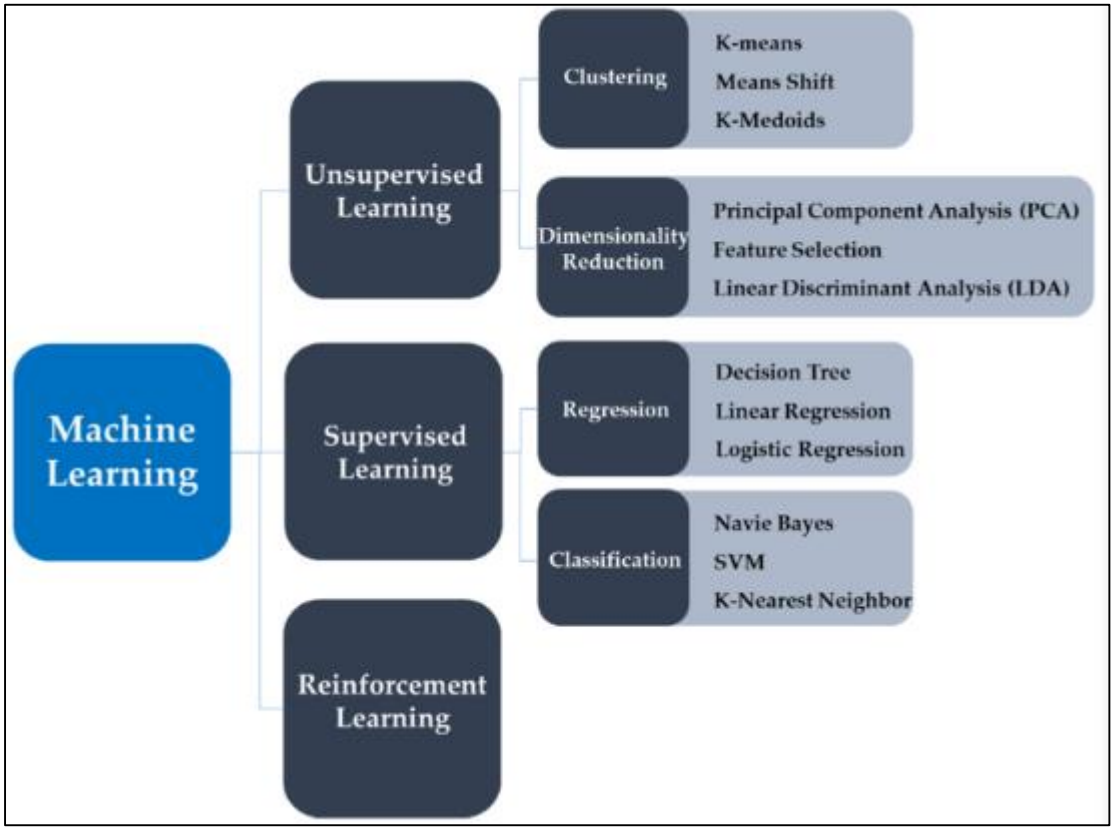


Figura 1 Tipos de Algoritmos Machine Learning

Fuente: *Energy Consumption Forecasting in Korea Using Machine Learning Algorithms de S.-Y. Shin y a.-G. Woo, 2022* [22].

En Machine Learning la validación de un modelo entrenado se hace mediante el uso de la matriz de confusión, la misma que permite visualizar el número de predicciones acertadas positivamente y el número de veces acertadas erróneamente [23].

2.2.1.5. Aprendizaje Profundo (Deep Learning).

El Deep Learning (DL) o aprendizaje profundo es una de tecnologías de la IA claves para el desarrollo a la percepción humana, es una forma avanza de Machine Learning en la cual su estructura está formada por algoritmos específicos conocidos como redes neuronales convolucionales, mejorando la capacidad de clasificar, identificar, detectar y de reconocimiento [24]. En términos simples el Deep Learning engloba a Machine Learning.

Investigadores de todas partes del mundo optaron por emplear Deep Learning para las investigaciones por los buenos resultados obtenidos en la predicción, y segmentación de imágenes [25].

2.2.1.6. Redes neuronales convolucionales.

Las redes neuronales en la IA son modelos computacionales que simulan el comportamiento de las neuronas del cerebro humano, en el Deep Learning la usan para resolver problemas como clasificación de imágenes, segmentación de imágenes, detección de objetos, reconocimiento de patrones entre otros. Están compuestas básicamente por tres capas una capa de entrada, una capa oculta y una capa de salida [26].

El modelo de las redes neuronales convolucionales (CNN ***Convolutional Neural Network***) es de las más conocidas y empleadas por muchos investigadores en diferentes áreas. Se la puede definir como un modelo matemático que se caracteriza principalmente por la aplicación de una capa en la que se realiza una operación matemática denominada convolución. Su función principal es realizar procesos que permitan recibir pixeles de una imagen, extraer sus características para luego clasificar en capas posteriores [27]. Es decir, al momento de ingresar una imagen se aplica una serie de procesos convolucionales dando como resultado un conjunto de mapas de características, donde salida (output) de una capa se convierte en la entrada (input) de la siguiente capa.

Además, este tipo de red neuronal es capaz de aprender un proceso de extracción de características por sí misma.

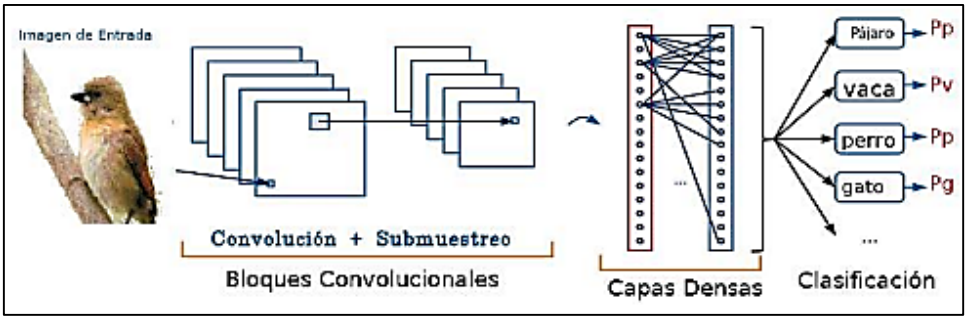


Figura 2 Arquitectura de una Red Neuronal Convolucional

Fuente: *Diseño de una arquitectura de Red Neuronal Convolucional para la clasificación de objetos de M. G. Villanueva y L. R. Muñoz, 2020 [28].*

2.2.1.7. Recursos para implementar IA.

Los recursos usados para la IA son en su mayoría gratuitos. En la siguiente figura se muestra los frameworks gratuitos disponibles.

Tabla 2 Frameworks para Implementar IA

Framework	License	Programming interfaces
Apache SINGA [420]	Apache 2.0 [421]	C++, Java, Python
BigDL [422]	Apache 2.0 [423]	Python, Scala
Caffe [424, 425]	BSD [426]	C++, MATLAB, Python
Chainer [427]	MIT [428]	Python
Deeplearning4j [429]	Apache 2.0 [430]	Clojure, Java, Kotlin, Python, Scala
Dlib [431, 432]	BSL [433]	C++
Flux [434]	MIT [435]	Julia
MATLAB Deep Learning Toolbox [436]	Proprietary [437]	MATLAB
Microsoft Cognitive Toolkit [438]	MIT [439]	BrainScript, C++, Python
Apache MXNet [440]	Apache 2.0 [441]	C++, Clojure, Go, JavaScript, Julia, Matlab, Perl, Python, R, Scala
OpenNN [442]	GNU LGPL [443]	C++
PaddlePaddle [444]	Apache 2.0 [445]	C++
PyTorch [446]	BSD [447]	C++, Python
TensorFlow [448, 449]	Apache 2.0 [450]	C++, C#, Go, Haskell, Julia, MATLAB, Python, Java, JavaScript, R, Ruby, Rust, Scala, Swift
Theano [451, 452]	BSD [453]	Python
Torch [454]	BSD [455]	C, Lua
Wolfram Mathematica [456]	Proprietary [457]	Wolfram Language

Fuente: *Deep Learning in electron microscopy de Jeffrey M Ede, 2021* [29].

TensorFlow es la librería más popular dentro de la comunidad científica, es compatible con muchos interfaces de programación entre ellos se encuentra Python, es un lenguaje de código abierto, que permite desarrollar proyectos de inteligencia artificial [30].

2.2.1.8. Aplicaciones web.

Las aplicaciones web son programas alojadas en la nube, que no requiere una instalación en nuestro dispositivo, donde todos los datos se almacenan en grandes servidores ubicados en diferentes partes a través del mundo. Y que además su única forma de acceder es por medio de un navegador web [31].

Al crear aplicaciones web se debe considerar la seguridad de los datos, facilidad de manejo, entre otras características. Por ende, es recomendable realizar pruebas piloto antes de almacenarlo en la web. Una forma de hacer estas pruebas es de manera local.

2.2.1.9. React Js.

React Js, es una librería que permite implementar el Frontend de una aplicación web, en términos más simples es una librería para el desarrollo de interfaces web. Es la parte gráfica Frontal de una aplicación web mediante la cual se interactúa con el cliente.

Una de las características más relevante de React Js, es la reutilización de componentes dentro del proyecto ya que cada componente se crea de forma independiente, es decir al diseñar una app en React, lo que realmente estamos creando son componentes independientes que nos van a permitir implementar interfaces web más complejas con mucha facilidad y de manera organizada. Además, React Js, se la puede ver con un modelo de programación orientada a objetos [32].

2.2.1.10. Metodología CRISP-DM.

CRISP-DM (Cross-Industry Standard Process for Data Mining) es una metodología empleada en la ciencia de datos, la cual proporciona una descripción de las fases del ciclo de vida de un proyecto estándar, así como las tareas necesarias y las relaciones entre las diferentes tareas [33]. Esta metodología está compuesta de 6 fases relacionadas entre sí, que indican las dependencias más importantes entre fases. Una de las ventajas de esta metodología es su alta flexibilidad [34].

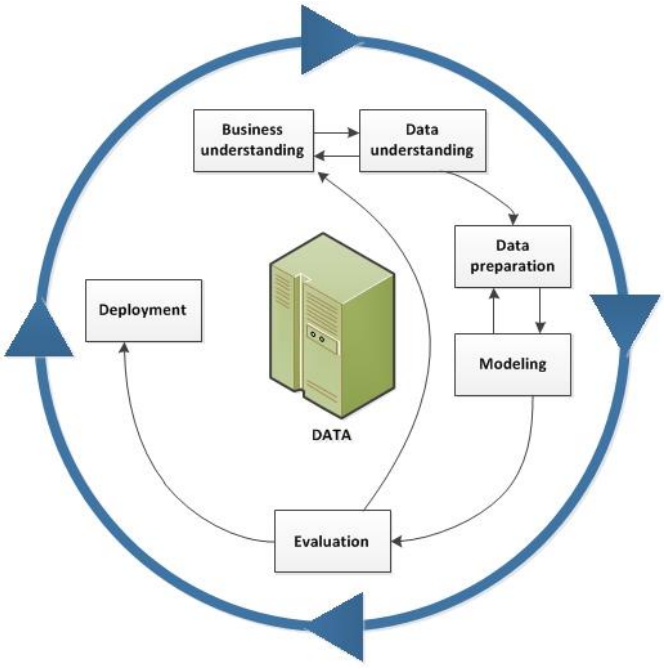


Figura 3 Metodología CRISP-DM

Fuente: *Conceptos básicos de ayuda de CRISP-DM, 2021* [34].

2.2.2. Marco Metodológico.

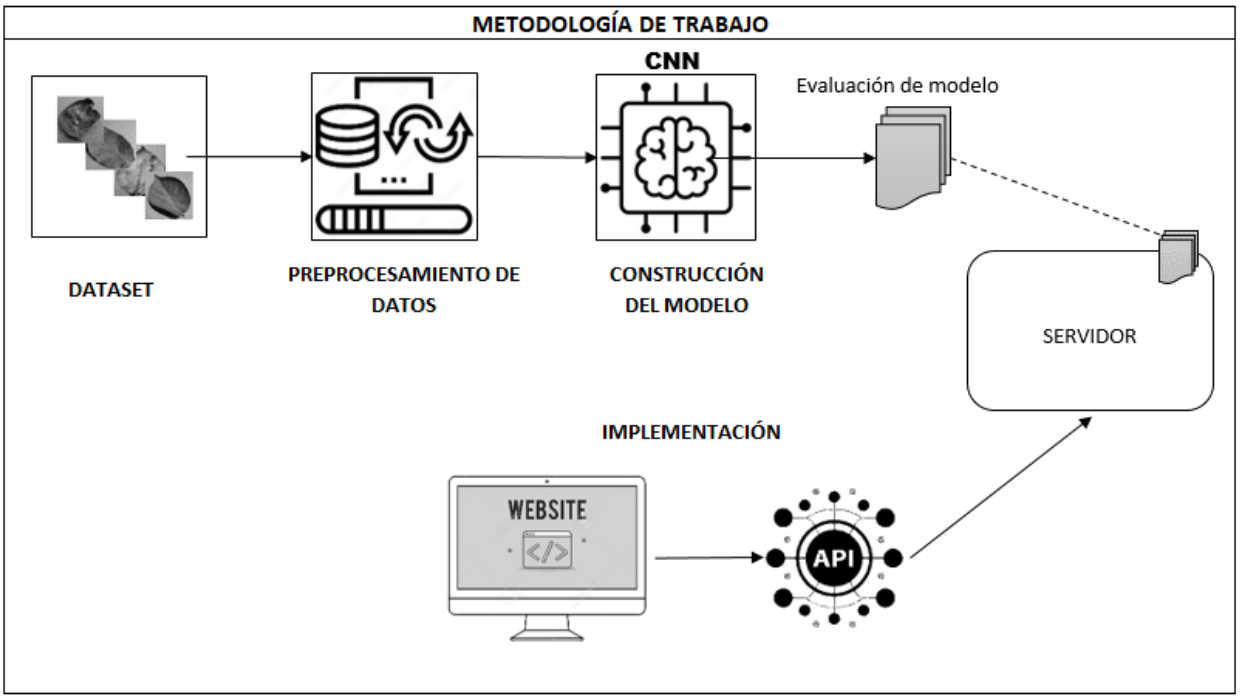


Figura 4 Metodología de trabajo.

Fuente: *Elaboración propia.*

La metodología de trabajo aplicada para realizar este proyecto es la CRISP-DM la cual consiste como trabajo inicial el obtener imágenes de las hojas del cultivo de papa que formaran nuestro dataset. Posterior a esto se preparó las herramientas necesarias para implementar el trabajo de IA con la creación del modelo, en la cual se realiza los procesos de entrenamiento, validación, optimización y evaluación del prototipo. Como último paso se desarrolla una interfaz web que permita el ingreso de datos en este caso que permita el ingreso de imágenes.

El conjunto de imágenes o dataset utilizado para el entrenamiento de nuestro modelo se lo obtuvo del conjunto de datos PlantVillage, actualmente es el conjunto de datos más popular a nivel mundial. Múltiples estudios recientemente realizados usan conjuntos datos públicos es decir que están a disposición de manera gratuita para entrenar modelos con Deep Learning para la detección e identificación de plagas y enfermedades en las diferentes áreas de cultivo de la agricultura. Aunque también se pueden usar dataset personalizados.

PlantVillage es una comunidad dedica al estudio de la aplicación de Inteligencia artificial dentro de la agricultura con la finalidad de aumentar el rendimiento y la rentabilidad de miles de productores agrícolas a nivel mundial [35]. Su más reciente investigación se trata de un asistente de inteligencia artificial denominado NURU. Su desarrollo tiene

como objetivo el diagnóstico de enfermedades en los cultivos mediante el uso de la visión artificial [36].

En la fase de desarrollo de trabajo se aplicó la técnica de aumento de datos que permite incrementar la diversidad de nuestro conjunto de datos.

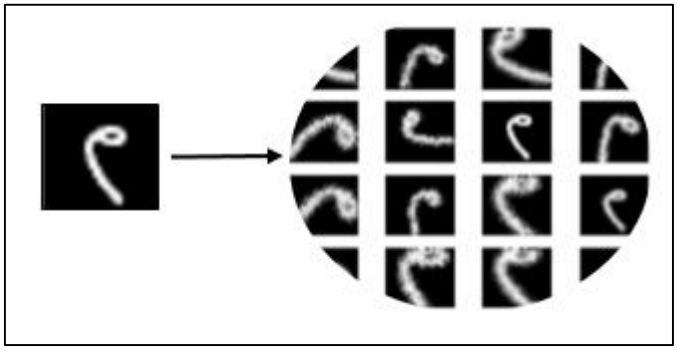


Figura 5 Técnica de Aumento de Datos (Data Augmentation)

Fuente: *Elaboración propia.*

En cuanto a la metodología para la red neuronal convolucional se basó en la siguiente imagen

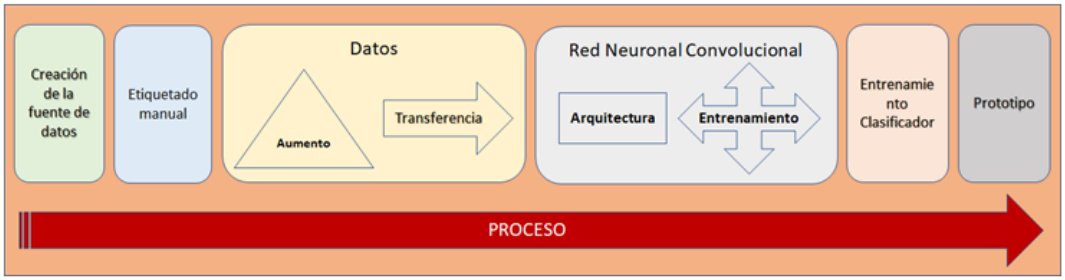


Figura 6 Metodología de una red neuronal convolucional CNN

Fuente: *Clasificación automática de anastomosis mediante redes neuronales convolucionales en video fetoscópico de Ángel Lerma Sánchez, Adriana Santoyo, Fabiola Villalobos, Pedro Reyes,2021* [37].

Ya en la parte final de nuestro proyecto se implementó el Frontend de la aplicación web mediante React Js, que permitirá poner en uso el funcionamiento de nuestra red neuronal convolucional.

2.3. Objetivos del prototipo.

2.3.1. Objetivo general

- Desarrollar una aplicación web para la detección de plagas en el cultivo de papa aplicando redes neuronales convolucionales.

2.3.2. Objetivos Específicos

- Implementar un modelo de red neuronal convolucional.
- Aplicar técnicas de aumento de datos en el entrenamiento para obtener una mayor precisión.
- Evaluar el modelo obtenido mediante el uso de métricas de rendimiento.
- Desarrollar una aplicación web que permita poner en funcionamiento el modelo obtenido.

2.4. Diseño del prototipo.

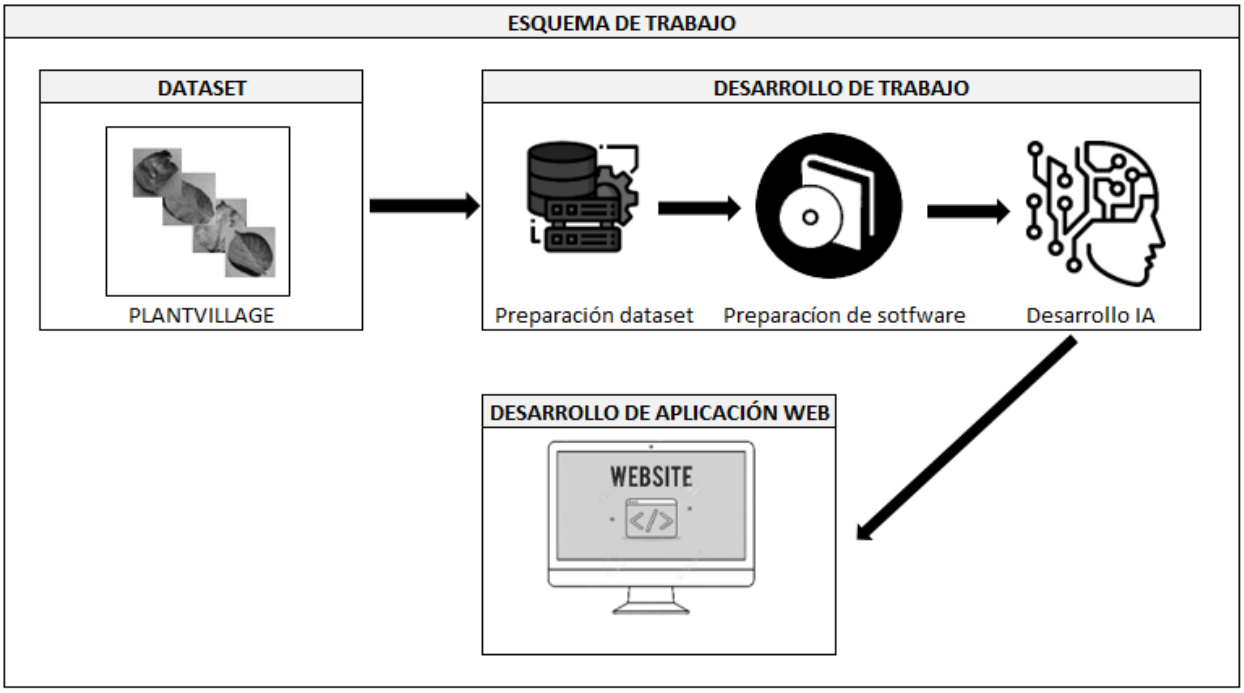


Figura 7 Esquema de Trabajo

Fuente: *Elaboración propia*

Para el diseño del prototipo se basó en el ciclo de vida que proporciona la metodología CRISP-DM, la cual nos permite obtener una vista general del diseño del producto y que se adapta a nuestras necesidades.

2.4.1. Estructura de proyecto

El proyecto está estructurado de la siguiente forma:

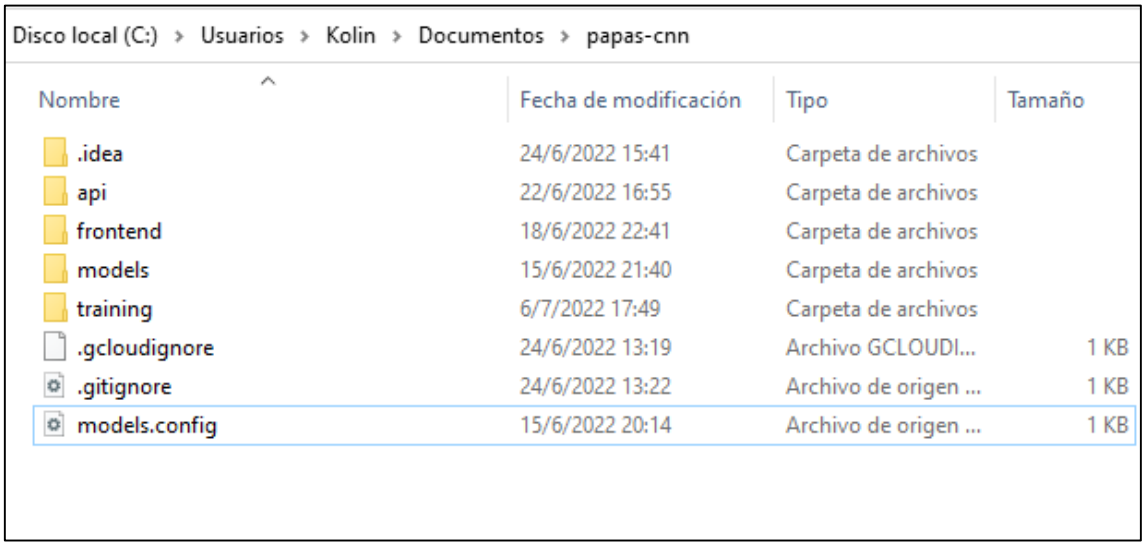


Figura 8 Estructura de archivos del proyecto

Fuente: *Elaboración propia*

Dentro de la carpeta denominada **papas-cnn** se ubican las siguientes carpetas y archivos de proyecto.

- **Training**

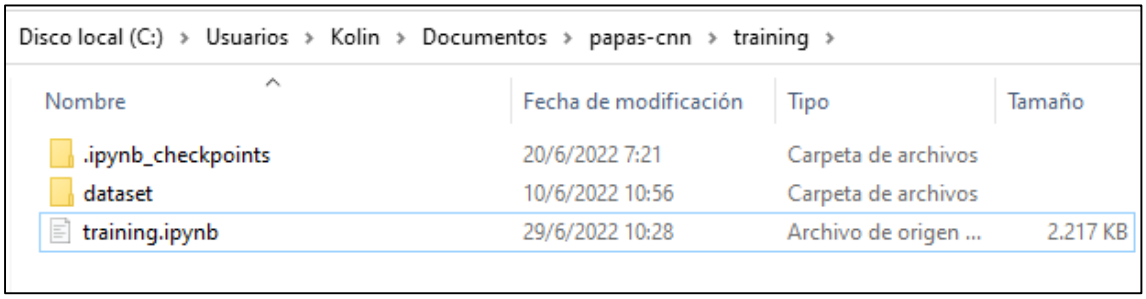


Figura 9 Estructura carpeta Training

Fuente: *Elaboración propia*

Esta carpeta contiene lo siguiente:

- **Dataset**: carpeta contenedora del conjunto de datos que nos servirán para entrenar nuestro modelo.
- **Training.ipynb:** archivo que contiene el código fuente para el entrenamiento y creación de nuestro modelo.

- **Models**

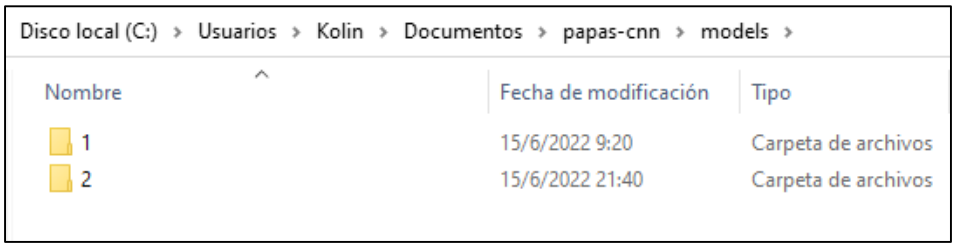


Figura 10 Estructura carpeta Models

Fuente: *Elaboración propia*

Contiene los modelos generados durante el entrenamiento.

- **Api**

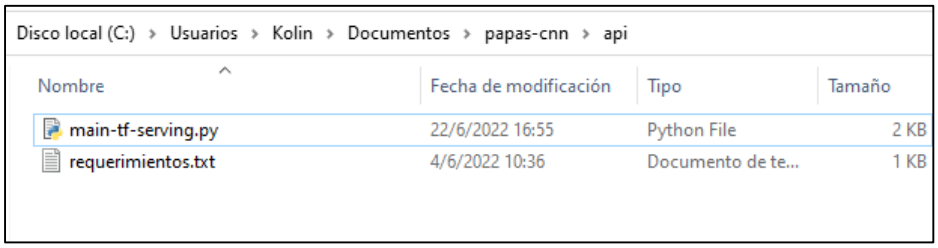


Figura 11 Estructura carpeta Api

Fuente: *Elaboración propia*

- **Main-tf-serving.py:** contiene el código fuente para levantar un servidor.
- **Requerimientos.txt:** contiene el nombre de las librerías necesarias, que se deben instalar.

- **Frontend**

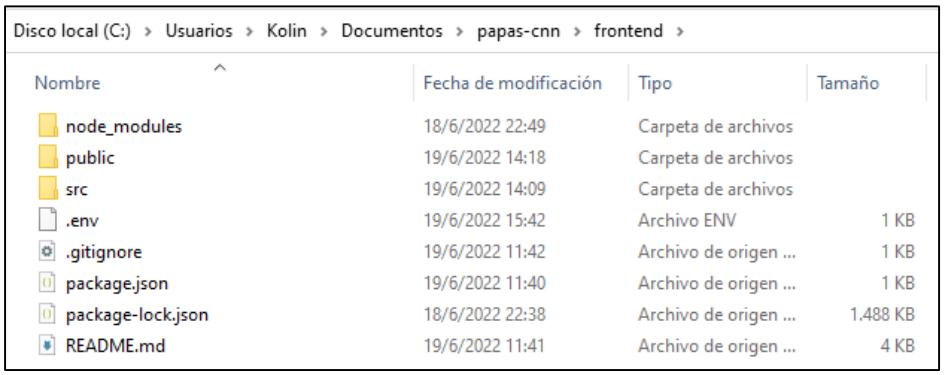


Figura 12 Estructura carpeta Frontend

Fuente: *Elaboración propia*

En esta carpeta contiene lo siguiente:

- **Node_modules**: contiene las librerías para desarrollar aplicaciones web.
- **Src**: contiene el condigo fuente de la aplicación web.

2.5. Ejecución y/o ensamblaje del prototipo

2.5.1. Obtención del conjunto de datos

Para el desarrollo del presente trabajo de titulación se utilizó el conjunto de datos del sitio web PlantVillage. El dataset usado consta de 3 carpetas, dos de ellas contienen imágenes con las plagas en el cultivo de papa y la otra carpeta contiene las imágenes de hojas sanas del cultivo de papa, ubicada en la carpeta raíz del proyecto.

2.5.2. Preparación del dataset

En este proyecto se usó dimensiones 256x256 por ende se debe verificar las propiedades de la imagen, para evitar problemas durante el entrenamiento.

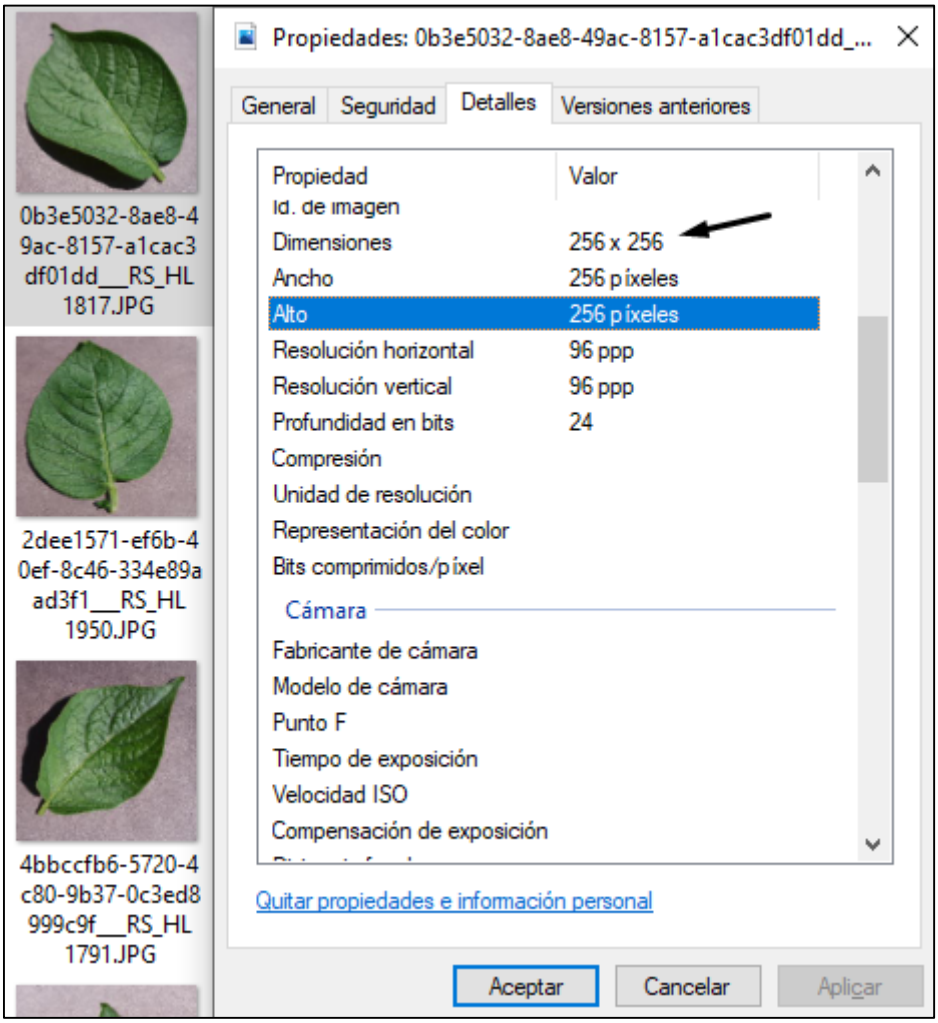


Figura 13 Propiedades de la imagen

Fuente: *Elaboración propia*

Como se puede apreciar en la imagen anterior, las dimensiones de nuestra imagen es de 256 x 256 pixeles. En caso que las dimensiones de las imágenes de nuestro conjunto

de datos varíen, es recomendable cambiar las dimensiones para mantener uniformidad en los datos.

2.5.3. Preparación de Software.

La preparación de software consiste en instalar las herramientas necesarias para realizar procesos de IA con Deep Learning. A continuación, se presenta la lista de herramientas y librerías necesarias para desarrollar este proyecto.

Como lenguaje de programación:

- Python v3.8

Como Editor de texto:

- Microsoft Visual Studio Code v1.68.1

Como Servidor:

- Uvicorn

Como entorno de ejecución:

- Node JS v16.16.0-x64.msi

Como Aplicación cliente servidor:

- Jupyter Notebook v6.4.12.

Como Marco de trabajo:

- FastApi v 0.78.0

Como librerías:

- TensorFlow v2.8.0
- Python-multipart
- Pillow
- TensorFlow-serving-api v2.8.0
- Matplotlib
- Numpy

2.5.4. Desarrollo de IA.

Como primer paso se importó las librerías necesarias para el desarrollo de nuestro modelo, así también se definió las variables que van a usar durante el entrenamiento.

```
In [1]: import tensorflow as tf
        from tensorflow.keras import models,layers
        import matplotlib.pyplot as plt

In [2]: BATCH_SIZE = 32
        CHANNELS=3
```

Figura 14 Importación de Librerías

Fuente: *Elaboración propia.*

Se importó las librerías de tensorflow para realizar el ML y las librerías Matplotlib para realizar gráficos a partir de los datos que se produzcan en el entrenamiento.

El tamaño de lote (BATCH_SIZE) que se usó para entrenar nuestra red neuronal convolucional es de 32, con número de canales (CHANNELS) de 3 por los colores RGB.

2.5.4.1. Pre-procesamiento de Datos

A continuación, se realizó un pre-procesamiento del conjunto de datos que se usara en el entrenamiento. Llamamos a la función `image_dataset_from_directory` esta nos devuelve un conjunto de datos grande que produce lotes de imágenes de los subdirectorios.

```
In [3]: dataset = tf.keras.preprocessing.image_dataset_from_directory(
            "dataset",
            shuffle =True,
            image_size = (256,256),
            batch_size =BATCH_SIZE,
        )

        Found 2152 files belonging to 3 classes.
```

Figura 15 Pre-procesamiento de Datos

Fuente: *Elaboración propia.*

Esta función recibe como argumentos lo siguiente:

- ***directory*** nombre del directorio donde se encuentra almacenado nuestro conjunto de imágenes.
- ***shuffle,*** este se establece en verdadero para mezclar los datos.
- ***image_size*** para modificar el tamaño de las imágenes, debido a que se procesa por lotes de imágenes. El valor predeterminado es de 256 x 256.
- ***batch_size*** tamaño de los lotes de 32.

Como resultado de ejecutar esta función nos muestra que ha encontrado 2152 imágenes pertenecientes a 3 clases.

Para comprobar que nuestro dataset se ha cargado correctamente, se procedió a realizar la visualización de los datos, tal y como se muestra en la siguiente imagen.

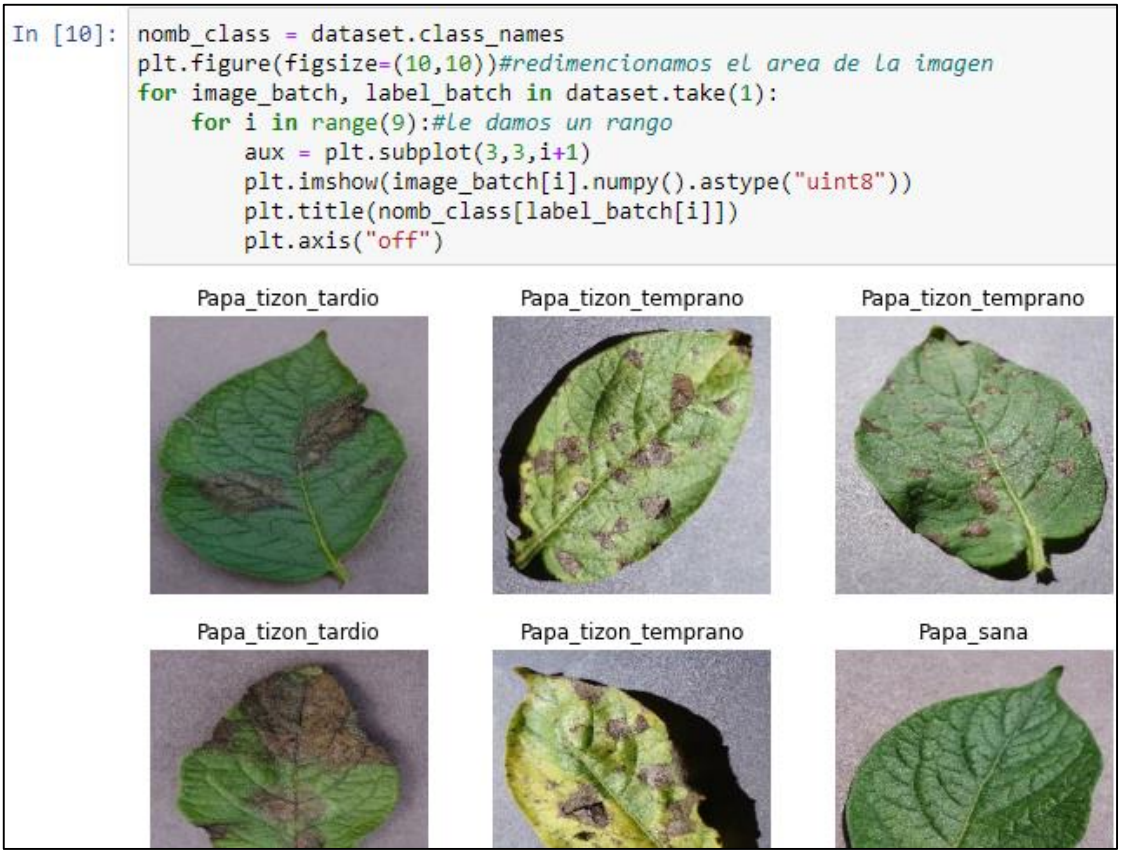


Figura 16 Visualización de Datos

Fuente: *Elaboración propia*

2.5.4.2. División de Datos

La división de datos se refiere a determinar el porcentaje necesario para el realizar el entrenamiento, la validación y el test.

El 80% de los datos se usaron para el entrenamiento, el 10% para la validación al final de cada época y el 10% restante para medir la precisión de nuestro modelo final.

En la siguiente imagen se muestra las operaciones realizadas para determinar los valores de cada uno.

```
In [12]: train_size = 0.8
         len(dataset)*train_size

Out[12]: 54.400000000000006

In [13]: train_ds = dataset.take(54)
         len(train_ds)

Out[13]: 54
```

Figura 17 Valor de Entrenamiento

Fuente: *Elaboración propia*

```
In [14]: val_size=0.1
         len(dataset)*val_size

Out[14]: 6.800000000000001

In [15]: val_ds = test_ds.take(6)
         len(val_ds)

Out[15]: 6

In [16]: test_ds = test_ds.skip(6)
         len(test_ds)

Out[16]: 8
```

Figura 18 Valor de test y validación

Fuente: *Elaboración propia}*

Tabla 3 División de datos para entrenamiento, validación y test

Nombre	Porcentaje		Valor
Entrenamiento	80%	0.8	54
Validación	10%	0.1	6
Test	10%	0.1	8

Fuente: *Elaboración propia*

Posterior a esto se aplicó la técnica de aumento de datos para incrementar la diversidad de nuestro conjunto de datos. El aumento de datos se hace por medio de operaciones básicas como rotación, zoom, volteretas o recortes a la imagen original.

```
In [21]: data_augmentation = tf.keras.Sequential([
           layers.experimental.preprocessing.RandomFlip("horizontal_and_vertical"),
           layers.experimental.preprocessing.RandomRotation(0.2),
         ])
```

Figura 19 Técnica Data Augmentation

Fuente: *Elaboración propia.*

2.5.4.3. Diseño de la red neuronal Convolucional

```
In [31]: input_shape = (32, 256, 256, 3)
         n_classes = 3

         model = models.Sequential([
             resize_and_rescale,
             data_augmentation,
             layers.Conv2D(32, (3,3), activation='relu', input_shape=input_shape),
             layers.MaxPooling2D((2, 2)),
             layers.Conv2D(64,  kernel_size = (3,3), activation='relu'),
             layers.MaxPooling2D((2, 2)),
             layers.Conv2D(64,  kernel_size = (3,3), activation='relu'),
             layers.MaxPooling2D((2, 2)),
             layers.Conv2D(64, (3, 3), activation='relu'),
             layers.MaxPooling2D((2, 2)),
             layers.Conv2D(64, (3, 3), activation='relu'),
             layers.MaxPooling2D((2, 2)),
             layers.Flatten(),
             layers.Dense(64, activation='relu'),
             layers.Dense(n_classes, activation='softmax'),
         ])
         model.build(input_shape=input_shape)
```

Figura 20 Diseño de la Red Neuronal Convolucional

Fuente: *Elaboración propia*

En la imagen anterior se muestra la arquitectura de la red neuronal ya terminada. Consta de 32 filtros en la primera capa, con una función de activación relu que es la más popular para las redes neuronales CNNs, un kernel de 3x3, seguida de una capa de MaxPooling de 2×2. Ya para las siguientes capas se usó filtros de 64. El tamaño de kernel puede variar es decir se puede usar un kernel de 5x5, 7x7, pero lo ideal es usar un kernel 3x3.

Para compilar el modelo se usó como optimizador Adam, definimos nuestra función de pérdida y la métrica la precisión.

```
In [23]: model.compile(
             optimizer='adam',
             loss= tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False),
             metrics=['accuracy']
         )
```

Figura 21 Compilar modelo

Fuente: *elaboración propia*

Existen otros optimizadores como el SGD, rmsprop, que se pueden usar para compilar.

2.5.4.4. Entrenamiento de la red neuronal convolucional

Para el entrenamiento del modelo se ajustó los siguientes parámetros:

- Numero de épocas 20.
- Tamaño de lote 32
- Tamaño de la validación de los datos.

```
In [24]: history = model.fit(
             train_ds,
             epochs = 20,
             batch_size = 32,
             verbose =1,
             validation_data = val_ds
         )
```

Figura 22 Ajuste del Modelo para Entrenamiento

Fuente: *Elaboración propia*

Luego de entrenarlo se aplicó un test donde se realizó una prueba a un conjunto de datos donde se obtuvo un resultado de 0.97 de precisión.

Ya con el modelo entrenado se procedió a realizar pruebas con datos reales para comprobar el correcto funcionamiento de nuestro modelo implementado. En la siguiente imagen se presenta las pruebas que se realizó, donde se obtuvo un promedio de un 99% en la detección del tipo de enfermedades en el cultivo de papa.

Como se puede apreciar el grado de confianza de la predicción se mantiene entre un rango de 97% al 100% de veracidad de los resultados en la predicción.

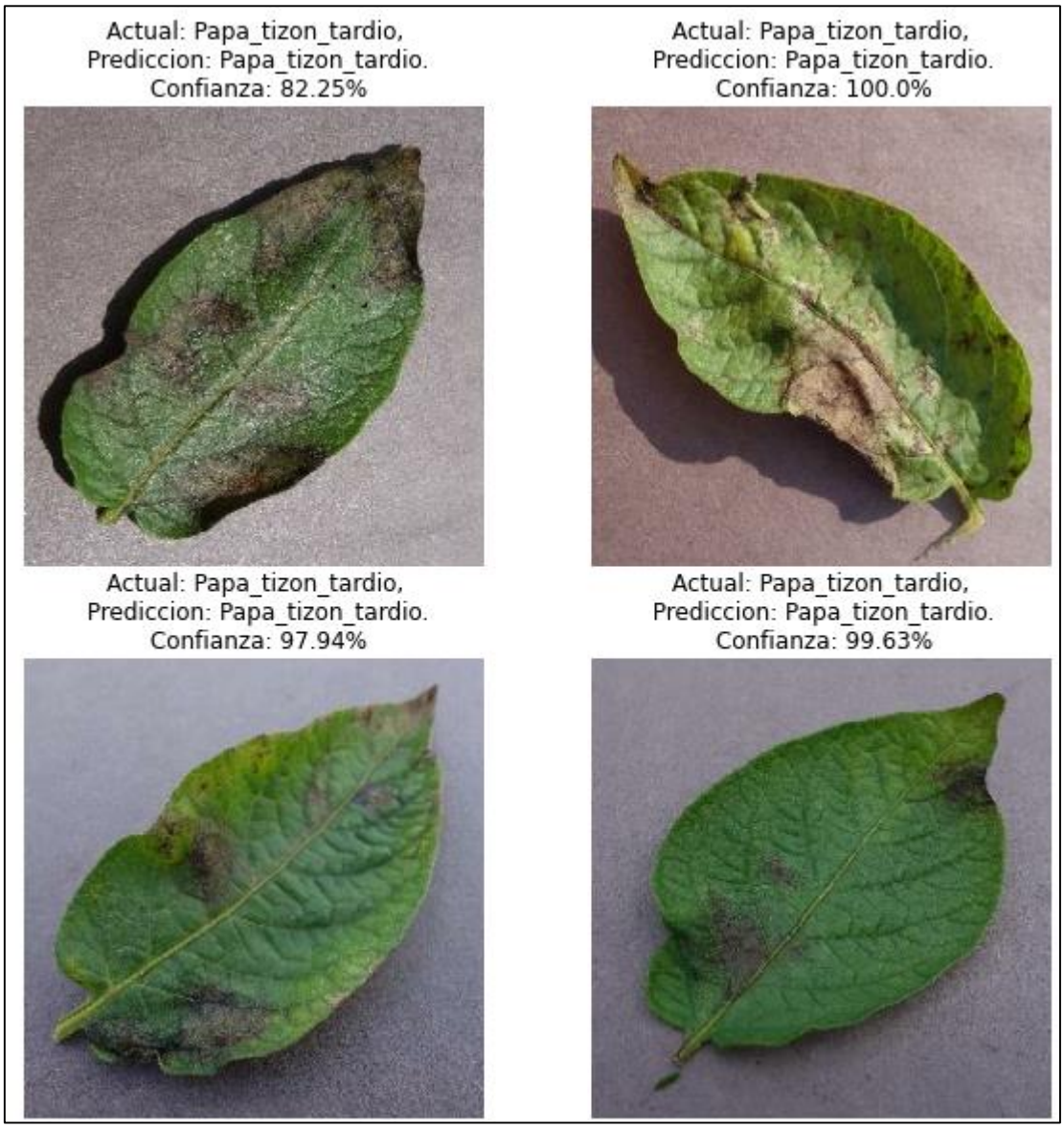


Figura 23 Resultados de Predicción

Fuente: *Elaboración propia*

En base a los valores de precisión y de pérdida obtenidos durante el entrenamiento se realizó los siguientes gráficos de líneas.

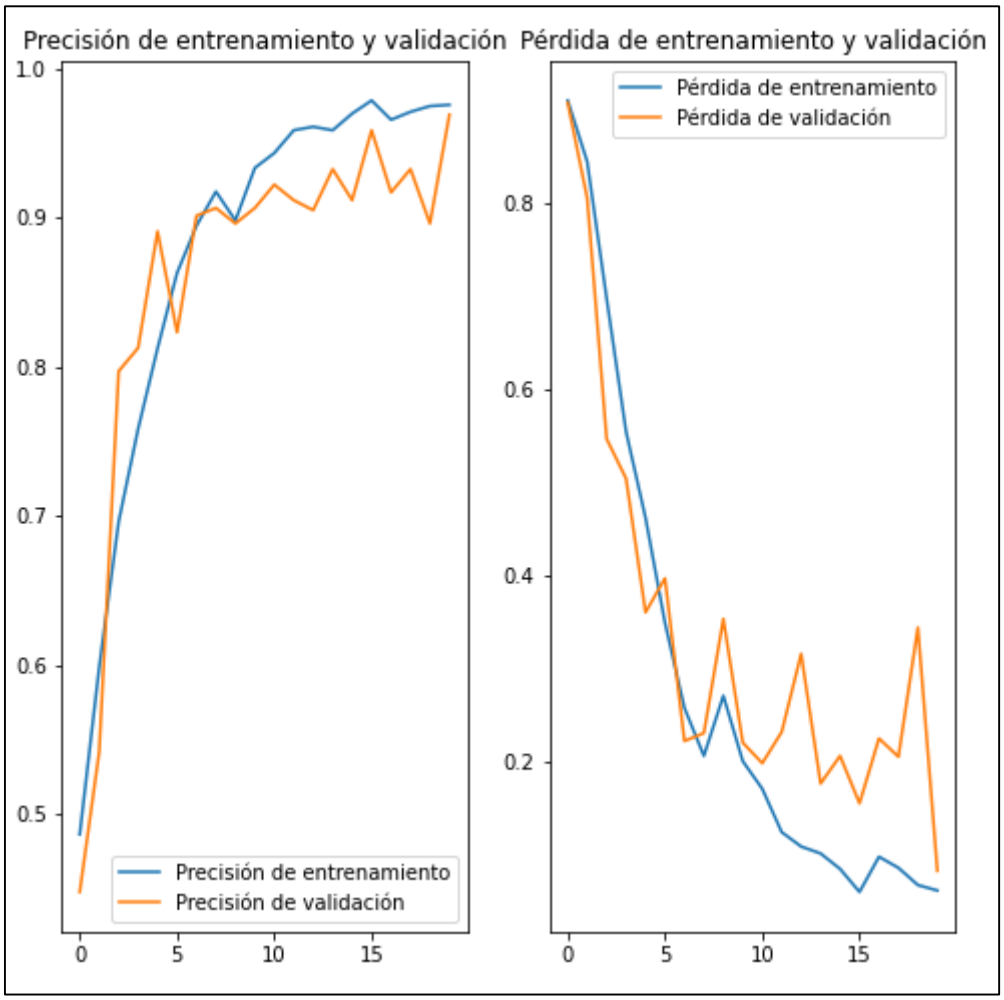


Figura 24 Gráfico de Lineas de pérdida y precisión

Fuente: *Elaboración propia.*

Donde el primer grafico muestra los valores de precisión en el entrenamiento y validación. Al mismo tiempo se puede apreciar que en la primera época de entrenamiento empezó con una precisión igual 50% pero al final del ciclo de las épocas se obtuvo un porcentaje aproximadamente de 97% de precisión.

El siguiente grafico refleja los valores de perdida durante el entrenamiento donde el valor inicial de perdida es igual 0.90 en la primera época, y en cada época del modelo el valor de perdida va disminuyendo.

2.5.5. Desarrollo de la aplicación web

2.5.5.1. Implementación del servidor y FastApi.

Para poder ejecutar una aplicación FastApi se necesita de un servidor, mediante el uso de la librería de Uvicorn se implementó un servidor web asincrónico de alto de rendimiento ASGI dedicado al trabajo con Python.

El archivo main-tf-serving.py, contiene tanto las librerías como las funciones necesarias para implementar y ejecutar un servidor Uvicorn, tales como FastApi, keras, tensorflow, entre otras librerías.

```
1   import keras
2   from fastapi import FastAPI, File, UploadFile
3   import numpy as np
4   import uvicorn
5   from io import BytesIO
6   from PIL import Image
7   import tensorflow as tf
8   from keras.models import load_model
9   import requests
```

Figura 25 Importación de Librerías

Fuente: *Elaboración propia*

El siguiente fragmento de código es la funcion read_file_as_image de tipo numpy que permite convertir el archivo de entrada (imagen) a una matriz numpy.

```
18  def read_file_as_image(archivo) -> np.ndarray:
19      # convertimos a una matriz numpy
20      image = np.array(Image.open(BytesIO(archivo)))
21      return image  # retornamos image
```

Figura 26 Función read_file_as_image

Fuente: *Elaboración propia*

Para realizar la predicción se implemento un método asincrónico predict. Esta función recibe como parámetro datos de tipo file, que por medio de la funcion read_file_as_image convertimos nuestro file a una matriz numpy para realizar el proceso de indentificacion de la enfermdad y devuelve el tipo de enfermedad junto con el grado de confianza.

```python
# metodo predict
@app.post("/predict")
async def predict(
        file: UploadFile = File(...)
):
    image = read_file_as_image(await file.read())
    image_bach = np.expand_dims(image, 0)
    json_data = {
        "instances": image_bach.tolist()
    }
    response = requests.post(endpoint, json=json_data)
    prediction = np.array(response.json()["predictions"][0])
    class_predict = CLASS_NAMES[np.argmax(prediction[0])]
    confidence = np.max(prediction)


    return {
        'enfermedad': class_predict,
        'Probabilidad': float(confidence)
    }
```

Figura 27 Función Predict

Fuente: *Elaboración propia*

Al momento de levantar nuestro servidor Uvicorn lo hará en la siguiente ruta http://localhost:8000.

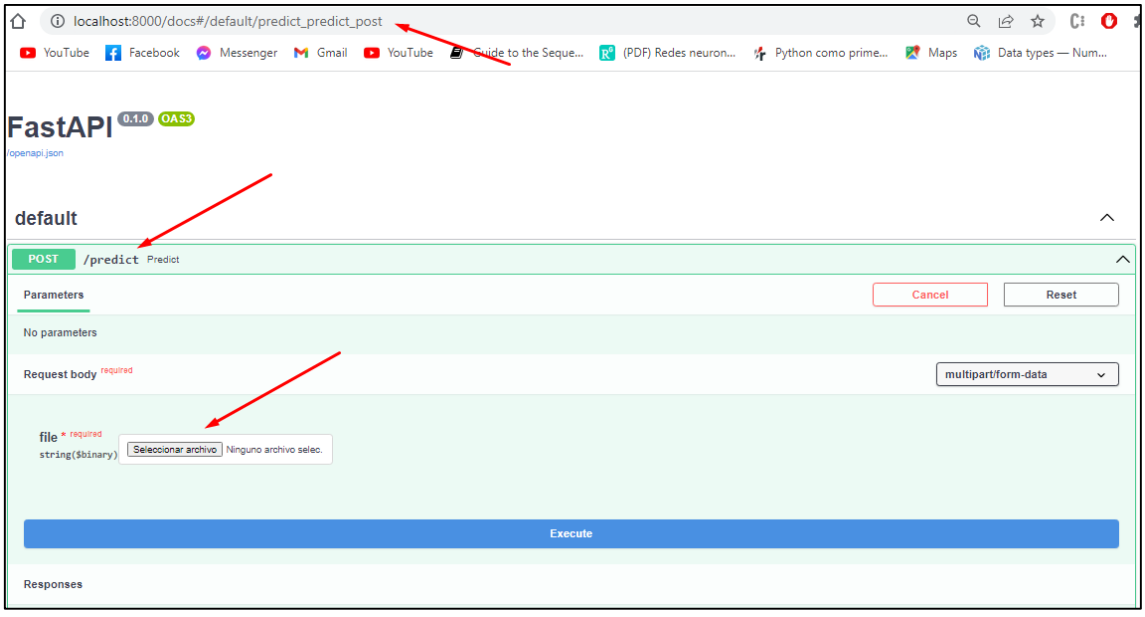


Figura 28 Serving Uvicorn

Fuente: *Elaboración propia*

2.5.5.2. Diseño de la aplicación web.

Para diseñar el Frontend de nuestra aplicación web se utilizó la biblioteca de React Js, es una de las más utilizas para el diseño de aplicaciones web y una de las ventajas que es código abierto. Al crear una app mediante React Js, se crean varios archivos por defecto.

Hay que tener una visión clara de lo que vamos a diseñar y los componentes que vamos a usar y en base a eso importar las librerías necesarias para comenzar a trabajar. En la siguiente imagen refleja las librerías que se importaron para diseñar nuestra interfaz gráfica.

```
1   import { useState, useEffect } from "react";
2   import { makeStyles, withStyles } from "@material-ui/core/styles";
3   import AppBar from "@material-ui/core/AppBar";
4   import Toolbar from "@material-ui/core/Toolbar";
5   import Typography from "@material-ui/core/Typography";
6   import Avatar from "@material-ui/core/Avatar";
7   import Container from "@material-ui/core/Container";
8   import React from "react";
9   import Card from "@material-ui/core/Card";
10  import CardContent from "@material-ui/core/CardContent";
11  import { Paper, CardActionArea, CardMedia, Grid, TableContainer,
12    Table, TableBody, TableHead, TableRow, TableCell, Button,
13     CircularProgress } from "@material-ui/core";
14  import cblogo from "./cblogo.PNG";
15  import imagen from "./bg.png";
16  import { DropzoneArea } from 'material-ui-dropzone';
17  import { common } from '@material-ui/core/colors';
18  import Clear from '@material-ui/icons/Clear';
```

Figura 29 Clase home.js

Fuente: *Elaboración propia*

Posteriormente se implementó constantes necesarias para enviar los datos de entrada al servidor donde se encuentra almacenado nuestra función de predicción, también se creó otras constantes para personalizar los componentes tales como el color, el alienado del texto, entre otras características.

El siguiente fragmento de código permite enviar el archivo de entrada (imagen) a nuestro servidor Uvicorn. Para el envío de los datos lo hace por medio de url que conecta con nuestro y aplicando un método de envió post.

```
155   const sendFile = async () => {
156     if (img) {
157       let formData = new FormData();
158       formData.append("file", selectedFile);
159       let res = await axios({
160         method: "post",
161         url: process.env.REACT_APP_API_URL,
162         data: formData,
163       });
164       if (res.status === 200) {
165         setData(res.data);
166       }
167       setIsloading(false);
168     }
169   }
```

Figura 30 Declaración constante sendFile

Fuente: *Elaboración propia*

La constante Colorbutton mediante el uso de withStyles permite definir el estilo de un componente mediante accesorios como por ejemplo el color del boton.

```
21  const ColorButton = withStyles((theme) => ({
22    root: {
23      color: theme.palette.getContrastText(common.white),
24      backgroundColor: common.white,
25      '&:hover': {
26        backgroundColor: '#ffffff7a',
27      },
28    },
29  }))(Button);
```

Figura 31 Personalizar componentes ColorButton Js..

Fuente: *Elaboración propia*

3. CAPÍTULO III. EVALUACIÓN DEL PROTOTIPO.

3.1. Plan de evaluación.

Como plan de evaluación del prototipo se optó por usar un método de evaluación como la matriz de confusión también conocida como matriz de error, la cual permite determinar el rendimiento de un modelo mediante métricas como la precisión [38].

En Machine Learning una matriz de confusión es una matriz cruzada que mayormente es usada para medir la precisión de la predicción de nuestro modelo [39]. Es decir, que en base a los resultados de predicción que se obtiene de distintas pruebas y análisis se puede determinar con exactitud la efectividad del modelo.

Tabla 4 Tabla de Matriz de Confusión

Table 4. Conceptualization of a binary classification confusion matrix. TP = True Positive, TN = True Negative, FP = False Positive, and FN = False Negative.

		Reference Data	
		Positive	**Negative**
Classification Result	Positive	TP	FP
	Negative	FN	TN

Fuente: *"Accuracy Assessment in Convolutional Neural Network-Based Deep Learning Remote Sensing Studies" de Aaron E. Maxwell, Timothy A. Warner and Luis Andrés Guillén* [40]***.***

Tabla 5 Descripción de las métricas de una Matriz de Confusión

Matriz de confusión

Verdadero positivo (True Positive) TP	*Número de veces que el modelo clasifica correctamente una muestra positiva.*
Falso negativo (False Negative) FN	*Número de veces que el modelo clasifica erróneamente una muestra positiva como negativa.*
Falso positivo (False Positive) FP	*Número de veces que el modelo clasifica erróneamente una muestra negativa como positiva.*
Verdadero negativo (True Negative) TN	*Número de veces que el modelo clasifica correctamente una muestra negativa*

Fuente: *Elaboración propia*

3.1.1. Métricas de evaluación.

Mediante lo parámetros establecidos en la matriz de confusión podemos calcular los principales indicadores de asertividad del modelo.

Tabla 6 Métricas de rendimiento de clasificación

Definition of classification performance metrics.

Symbol	Metric	Defined as
SNS	Sensitivity	$\frac{TP}{TP+FN}$
SPC	Specificity	$\frac{TN}{TN+FP}$
PRC	Precision	$\frac{TP}{TP+FP}$
NPV	Negative Predictive Value	$\frac{TN}{TN+FN}$
ACC	Accuracy	$\frac{TP+TN}{TP+FN+TN+FP}$
F_1	F_1 score	$2\frac{PRC \cdot SNS}{PRC+SNS}$

Fuente: *"The impact of class imbalance in classification performance metrics based on the binary confusion matrix" de Amalia Luque, Alejandro Carrasco; Alejandro Martín; Ana de las Heras,2019* [41].

3.1.2. Calculo de Métricas.

En la siguiente tabla se describe como se calcula las métricas de rendimiento para evaluar nuestro modelo [42].

Tabla 7 Cálculo de Métricas de Rendimiento

MÉTRICA	CÁLCULO
Sensibilidad ***(****Sensivity****)***	Consiste en el número de verdaderos positivos dividido para la suma de los verdaderos positivos y falsos negativos.
Especificidad ***(****Specificity****)***	Consiste en la división del número de verdaderos negativos para la suma de verdaderos negativos y falsos positivos.
Precisión ***(****Precision****)***	Consiste en la división de verdaderos positivos para la suma de verdaderos positivos + falsos positivos.
Exactitud ***(****Accuracy****)***	consiste en sumar los verdaderos positivos + verdaderos negativos y dividirlos para la suma de verdaderos positivos + falsos negativos + verdaderos negativos + falsos positivos.
F1 Score	F1 Score es la media de la precisión y la sensibilidad.

Fuente: *Elaboración propia*

3.1.3. Descripción de Métricas.

- **Sensibilidad:**

La sensibilidad hace referencia a la eficacia que presenta un modelo en la detección de casos positivos [43].

- **Especificidad:**

La especificidad es lo contrario a la sensibilidad ya que esta presenta la tasa de verdaderos negativos predichos por el modelo [43].

- **Precisión:**

La precisión se refiere al porcentaje de casos verdaderos positivos identificados por el modelo. Es decir, mientas más alto sea el valor del porcentaje de precisión más efectivo es el modelo [43].

- **Exactitud:**

La exactitud se refiere al grado de cercanía del valor predicho con el valor real. Esta métrica no se emplea para modelos entrenados con dataset desequilibrados [44].

- **F1 Score:**

El F1 Score o F1 Puntuación se la puede definir como la media entre la precisión y la sensibilidad. Es una de las métricas más empleadas en la evaluación de modelos [44].

3.2. Resultados de la evaluación

Las pruebas se realizaron usando imágenes de hojas del cultivo de papa reales que no fueron sometidas a entrenamiento. La evaluación del modelo se realizó en base a las 3 categorías aprendidas durante el entrenamiento, mediante un total de 60 imágenes, es decir 20 imágenes de cada categoría.

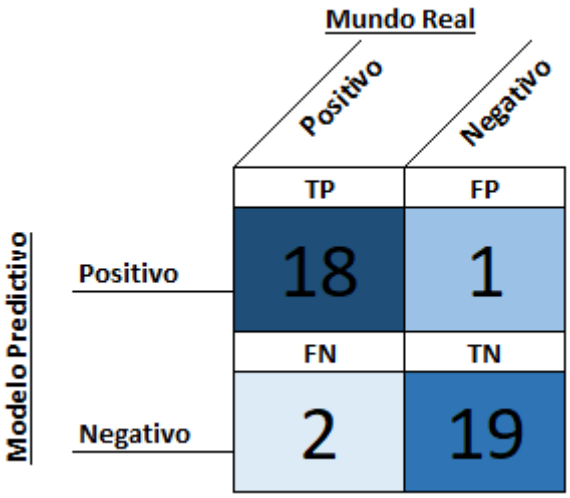
Para la evaluación de cada categoría se empleó un total 20 imágenes de muestras verdaderas y un total de 20 imágenes de muestras falsas seleccionadas aleatorias.

Mediante la ayuda de la tabulación se plasmó los resultados obtenidos para una mejor perspectiva de los datos.

3.2.1. Evaluación de muestras Sanas.

Tabulación de los datos en la matriz de confusión.

Tabla 8 Matriz de Confusión - Muestras Sanas



Fuente: *Elaboración propia*

A partir de los datos que se obtengan en la matriz de confusión se calculan las métricas de rendimiento.

Tabla 9 Cálculo de Métricas de Rendimiento

Métrica	Fórmula	Cálculo	%
Sensibilidad	$S = \frac{TP}{TP + FN}$	0,90	90%
Especificidad	$E = \frac{TN}{TN + FP}$	0,95	95%
Precisión	$P = \frac{TP}{TP + FP}$	0,95	95%
Exactitud (Accurary)	$E = \frac{TP + TN}{TP + FN + TN + FP}$	0,93	93%
F1 Score	$F1 = 2\frac{P * S}{P + S}$	0,92	92%

Fuente: *Elaboración propia*

Análisis rápido de resultados.

Tabla 10 Análisis - Muestras Sanas

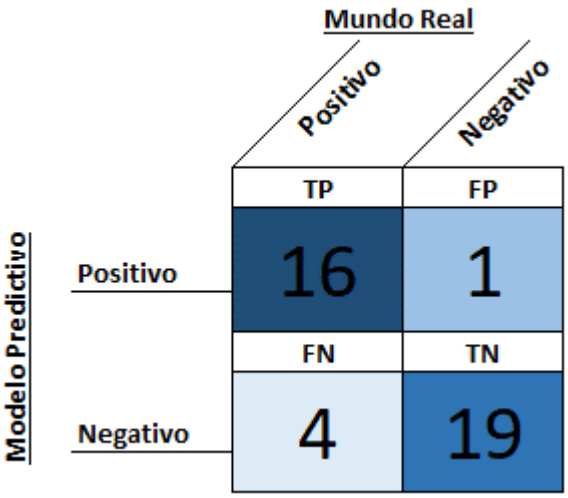
Métrica	Porcentaje %	Análisis
Sensibilidad	90	El 90% de las muestras positivas fueron identificadas correctamente.
Especificidad	95	El modelo presenta un 95% de confianza en la identificación de casos negativos.
Precisión	95	Las muestras identificadas por el modelo tienen una precisión en la predicción del 95%.

Fuente: *Elaboración propia.*

3.2.2. Evaluación de muestras con Tizón Tardío

Tabulación de los datos en la matriz de confusión.

Tabla 11 Matriz de Confusión - Muestras Tizón Tardío



Fuente: *Elaboración propia*

A partir de los datos que se obtengan en la matriz de confusión se calculan las métricas de rendimiento.

Tabla 12 Cálculo de Métricas de Rendimiento

Métrica	Fórmula	Cálculo	%
Sensibilidad	$S = \frac{TP}{TP + FN}$	0,80	80%
Especificidad	$E = \frac{TN}{TN + FP}$	0,95	95%
Precisión	$P = \frac{TP}{TP + FP}$	0,94	94%
Exactitud (Accurary)	$E = \frac{TP + TN}{TP + FN + TN + FP}$	0,88	88%
F1 Score	$F1 = 2\frac{P * S}{P + S}$	0,86	86%

Fuente: *Elaboración Propia.*

Análisis rápido de resultados

Tabla 13 Análisis - Muestras Tizón Tardío

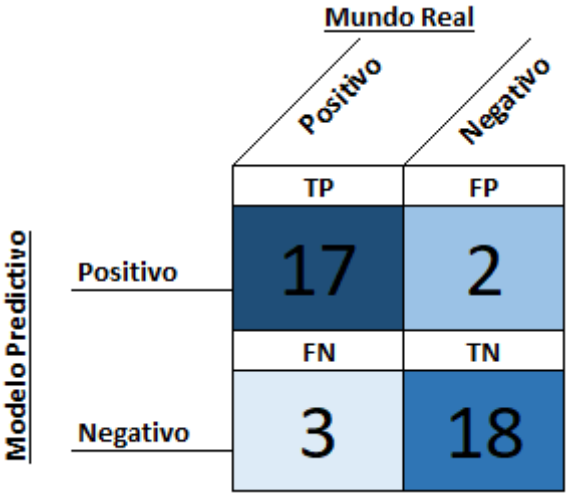
Métrica	Porcentaje %	Análisis
Sensibilidad	80	El 80% de los casos fueron identificadas correctamente positivos.
Especificidad	95	El modelo presenta un 95% de confianza en la identificación de casos negativos.
Precisión	94	Las muestras identificadas por el modelo tienen una precisión en la predicción del 94%.

Fuente: *Elaboración propia.*

3.2.3. Evaluación de muestras con Tizón Temprano

Tabulación de los datos de predicción en la matriz de confusión.

Tabla 14 Matriz de Confusión - Muestras Tizón Temprano



Fuente: *Elaboración propia*

A partir de los datos que se obtengan en la matriz de confusión se calculan las métricas de rendimiento.

Tabla 15 Cálculo de Métricas de Rendimiento

Métrica	Fórmula	Cálculo	%
Sensibilidad	$S = \frac{TP}{TP + FN}$	0,85	85%
Especificidad	$E = \frac{TN}{TN + FP}$	0,90	90%
Precisión	$P = \frac{TP}{TP + FP}$	0,89	89%
Exactitud (Accurary)	$E = \frac{TP + TN}{TP + FN + TN + FP}$	0,88	88%
F1 Score	$F1 = 2\frac{P * S}{P + S}$	0,87	87%

Fuente: *Elaboración propia*

Análisis rápido de resultados

Tabla 16 Análisis - Muestras Tizón Temprano

Métrica	Porcentaje %	Análisis
Sensibilidad	85	El 85% de los casos fueron identificadas correctamente positivos.
Especificidad	90	El modelo presenta un 90% de confianza en la identificación de casos negativos.
Precisión	89	Las muestras identificadas por el modelo tienen una precisión en la predicción del 89%.

Fuente: *Elaboración propia*

3.3. Resultados de la validación del modelo

Para validar un modelo de IA, se debe considerar que las métricas de rendimiento como el F1 Score estén por encima del porcentaje mínimo de 0.7 es decir de un 70% para poder validar el modelo desarrollado.

Durante las pruebas realizadas se obtuvieron puntajes satisfactorios ya que valor de F1 Score que es la media entre la precisión y la exactitud fue mayor al 86%, por lo tanto, se puede considerar que nuestro modelo tiene la capacidad para ser aplicado en el mundo real.

4. CONCLUSIONES

Luego de realizar las investigaciones pertinentes para el desarrollo de una aplicación para la detección de plagas en el cultivo de papa aplicado técnicas de inteligencia artificial se llegaron a las siguientes conclusiones.

- Mediante la investigación científica se pudo diseñar y desarrollar una aplicación web para la detección de plagas en el cultivo de papa aplicando un modelo de red neuronal de tipo convolucional.
- Con el uso de Python, librerías keras y tensorflow se logró desarrollar un modelo de red neuronal convolucional para el reconocimiento de plagas en el cultivo de papa.
- Mediante la aplicación de técnicas de aumento de datos (Data Augmentation) se pudo obtener una diversidad de datos durante el entrenamiento de la red, para aumentar el nivel de precisión en la identificación de las plagas.
- Se realizó una evaluación de rendimiento del modelo para la cual se empleó un total 60 imágenes de pruebas, donde los resultados obtenidos se tabulo es un matriz de confusión, que gracias a las métricas que se pudieron identificar se pudo determinar la validación de nuestro modelo de red neuronal.
- Mediante el uso de la biblioteca de React Js, se logró implementar una aplicación web que permitió poner en funcionamiento la red neuronal convolucional diseñada.

5. RECOMENDACIONES

- Es recomendable que el dataset que se vaya a usar durante el entrenamiento tenga un solo formato y que las imágenes sean de buena calidad para evitar errores predictivos durante su funcionamiento.
- Se recomienda usar imágenes de tamaño reducido no superior a los 256 píxeles tanto en altura y en ancho.
- En caso de no poseer una dataset publica es recomendable realizar las fotografías con un bueno equipo tecnológico y que las imágenes.
- La técnica de aumento de datos se debe aplicar siempre y cuando el dataset que se va a usar para el entrenamiento no presenta uniformidad en los datos, es decir que las cantidades entre los diferentes datos varíen.

6. BIBLIOGRAFÍA

[1] D. A. Sánchez, H. R. G. Diez y Y. H. Heredia, «Revisión de algoritmos de detección y seguimiento de objetos con redes profundas para videovigilancia inteligente,» *Revista Cubana de Ciencias Informáticas,* vol. 14, nº 03, pp. 165-195, 2020.

[2] G. A. S. Megeto, A. G. d. Silva, R. F. Bulgarelli, C. F. Bublitz, A. C. Valente y D. A. G. d. Costa, «Artificial intelligence applications in the agriculture 4.0,» *Revista Ciência Agronômica,* vol. 51, 2020.

[3] J. S. B. Segovia, F. A. D. Rojas y M. W. V. Quishpe, «Estudio del uso de técnicas de inteligencia artificial aplicadas para análisis de suelos para el sector agrícola,» *RECIMUNDO: Revista Científica de la Investigación y el Conocimiento,* vol. 5, nº 1, pp. 4-19, 23 Enero 2021.

[4] J. P. C. T. Jaime Israel Izquierdo Valladarez, «Reconocimiento de objetos del hogar, usando redes neuronales convolucionales para personas con discapacidad visual,» *Polo del Conocimiento,* vol. 5, nº 1, pp. 563-580, 2019.

[5] K. R. S. J. Abirami Devaraj, «Identification of Plant Disease using Image Processing Technique,» *2019 International Conference on Communication and Signal Processing,* pp. 749-753, 2019.

[6] I. Zepeda-Jazo, «Manejo sustentable de plagas agrícolas en México,» *Agricultura, sociedad y desarrollo,* vol. 15, nº 1, pp. 99-108, 2018.

[7] A. Sharma, M. Georgi, M. Tregubenko, A. Tselykh y A. Tselykh, «Enabling smart agriculture by implementing artificial intelligence and embedded sensing,» *Computers & Industrial Engineering,* vol. 165, 2022.

[8] T. T. M. J Bhuvana, «An approach to plant disease detection using deep learning techniques,» *ITECKNE: Innovación e Investigación en Ingeniería,* vol. 18, nº 2, p. 169, 2021.

[9] W. A. L. Portilla:, M. J. S. Barón y E. A. Fernández, «Aplicación de redes neuronales convolucionales para la detección del tizón tardío Phytophthora infestans en papa

Solanum tuberosum,» *Revista U.D.C.A Actualidad & Divulgación Científica,* vol. 24, nº 2, 2021.

[10] G. Gabardo, M. D. Pria, H. L. d. Silva y M. G. Harms, «Iternative products to control late season diseases in soybeans,» *Ciência Rural, Santa Maria,* vol. 52, nº 2, 2022.

[11] S. Marín, F. Bertsch y L. Castro, «Efecto del manejo orgánico y convencional sobre propiedades bioquímicas de un Andisol y el cultivo de papa en invernadero,» *Agronomía Costarricense,* vol. 41, nº 2, pp. 27-46, 2017.

[12] R. A. N. Odalys, C. R. J. Andrés, A. O. M. Salvador, Y. V. Marco y L. R. Samuel, «Detección de enfermedades en cultivos de Papa usando procesamiento de imágenes.,» *Cumbres,* vol. 6, nº 1, pp. 43 - 52, 2020.

[13] E. A. Q. Montoya, S. F. J. Colorado, W. Yesid y G. E. C. Golondrino, «Propuesta de una Arquitectura para Agricultura de Precisión Soportada en IoT,» *Revista Ibérica de Sistemas e Tecnologias de Información,* nº 24, pp. 39-54, 2017.

[14] M. J. Längkvist, A. Kiselev, M. Alirezaie y A. Loutfi, «Classification and Segmentation of Satellite Orthoimagery Using Convolutional Neural Networks,» *Remote Sensing,* vol. 8, 2016.

[15] M. Selvaraj, AlejandroVergara, FrankMontenegro, H. Ruiz y NancySafari, «Detection of banana plants and their major diseases through aerial images and machine learning methods: A case study in DR Congo and Republic of Benin,» *ISPRS Journal of Photogrammetry and Remote Sensing,* vol. 169, pp. 110-124, 2020.

[16] M. R. Chamorro y A. M. Gamero, «Artificial intelligence at the service of medicine,» *Atención Primaria Práctica,* vol. 4, 2022.

[17] J.F.Avila-Tomás, M.A.Mayer-Pujadas y V.J.Quesada-Varela, «La inteligencia artificial y sus aplicaciones en medicina I: introducción antecedentes a la IA y robótica,» *Atención Primaria,* vol. 52, nº 10, pp. 778-784, 2020.

[18] R. Benítez, G. Escudero y S. Kanaan, Inteligencia artificial avanzada, vol. 3, Editorial Uoc; 1er edición, 2013.

[19] M. J. A. Chuchucaa, S. A. S. Lunac, B. G. Suárez, K. E. Suárez, A. Gelrudf y T. M. Berzin, «El papel emergente de la inteligencia artificial en la endoscopia gastrointestinal: una revisión de la literatura,» *Gastroenterología y Hepatología,* vol. 45, pp. 492-497, 2022.

[20] W. Zhu, X. Wang y P. Xie, «Self-directed Machine Learning,» *AI Open,* pp. 58-70, 2022.

[21] A. Martorell, A. Martin-Gorgojo, E. Ríos-Viñuela, J. R. Carnero, F. Alfageme y R. Taberner, «Artificial Intelligence in Dermatology: A Threat or an Opportunity?,» *Actas Dermo-Sifiliográficas,* vol. 113, nº 1, pp. 30-46, 2021.

[22] S.-Y. Shin y a.-G. Woo, «Energy Consumption Forecasting in Korea Using Machine Learning Algorithms,» *Energies,* vol. 15, nº 13, 2022.

[23] V. Kryzhanivskyya, R. M'Saoubi, M. Bhallamudi y M. Cekal, «Machine Learning based Approach for the Prediction of Surface Integrity in Machining,» *Procedia CIRP,* vol. 108, pp. 537-542, 2022.

[24] A. Ahmad, D. Saraswat y A. E. Gamal, «A survey on using deep learning techniques for plant disease diagnosis and recommendations for development of appropriate tools,» *Smart Agricultural Technology,* 16 Junio 2022.

[25] D. Gibert, J. Planes, C. Mateu y Q. Le, «Fusing feature engineering and deep learning: A case study for malware classification,» *Expert Systems With Applications,* vol. 207, 2022.

[26] E. Portillo, I. Cabanes, M. Marcos y A. Zubizarreta, «Aplicación de Redes Neuronales en la Detección de Regímenes Degradados en el Proceso Wedm,» *Revista Iberoamericana de Automática e Informática Industrial,* vol. 6, nº 1, pp. 39-50, 2009.

[27] D. López-Betancur, R. B. Durán, C. Guerrero-Méndez, R. Z. Rodríguez y T. S. Anaya, «Comparación de arquitecturas de redes neuronales convolucionales para el diagnóstico de COVID-19,» *Computación y Sistemas,* vol. 25, nº 3, 2021.

[28] M. G. Villanueva y L. R. Muñoz, «Diseño de una arquitectura de Red Neuronal Convolucional para la clasificación de objetos,» *Ciencia Nicolaita,* nº 81, pp. 46-61, 2020.

[29] J. M. Ede, «Deep learning in electron microscopy,» *Machine Learning: Science and Technology,* vol. 2, nº 1, 2021.

[30] J. Christmann, S. Rohn y P. Weller, «gc-ims-tools – A new Python package for chemometric analysis of GC–IMS data,» *Food Chemistry,* vol. 394, 2022.

[31] H. Samad, S. Hanizan, R. Din, R. Murad y A. Tahir, «Performance Evaluation of Web Application Server based on Request Bit per Second and Transfer Rate Parameters,» *Journal of Physics: Conference Series,* 2018.

[32] A. Henson, «How to Use the React Profiler Component to Measure Render Performance,» Life at Paperless Post, Septiembre 2019. [En línea]. Available: https://medium.com/life-at-paperless/how-to-use-the-react-profiler-component-to-measure-performance-improvements-from-hooks-d43b7092d7a8. [Último acceso: 03 Julio 2022].

[33] J. J. E. Zúñiga, «Aplicación de metodología CRISP-DM para segmentación geográfica de una base de datos pública,» *Ingeniería, Investigación y Tecnología,* vol. 21, nº 1, 2020.

[34] IBM, «Conceptos básicos de ayuda de CRISP-DM,» 17 08 2021. [En línea]. Available: https://www.ibm.com/docs/es/spss-modeler/saas?topic=dm-crisp-help-overview. [Último acceso: 25 08 2022].

[35] B. N. Naik, R.Malmathanraj y P.Palanisamy, «Detection and classification of chilli leaf disease using a squeeze-and-excitation-based CNN model,» *Ecological Informatics,* vol. 69, 2022.

[36] M. Salathé y D. Hughes, «PlantVillage: A deep-learning app diagnoses crop diseases,» Escuela Politécnica Federal de Lausana, 04 Octubre 2016. [En línea]. Available: https://actu.epfl.ch/news/plantvillage-a-deep-learning-app-diagnoses-crop-di/. [Último acceso: 6 Julio 2022].

[37] Á. L. Sánchez, A. Santoyo, F. Villalobos y P. Reyes, «Clasificación automática de anastomosis mediante redes neuronales convolucionales en video fetoscópico,»

RIDE. Revista Iberoamericana para la Investigación y el Desarrollo Educativo, vol. 11, nº 22, 2021.

[38] C. Das, A. K. Sahoo y C. Pradhan, «Chapter 12 - Multicriteria recommender system using different approaches,» de *Cognitive Big Data Intelligence with a Metaheuristic Approach*, Academic Press, 2022, pp. 259-277.

[39] A. V. S. Caldas y A. F. d. A. d. S. Júnior, «COMPARATIVE EVALUATION OF INVESTMENT ANALYSIS METHODS: AN APPLICATION IN RENEWABLE ENERGY AUCTIONS BETWEEN 2011 E 2015,» *Revista De Administração Da UFSM,* vol. 14, nº 3, pp. 693-715., 2021.

[40] A. E. Maxwell, T. A. Warner y L. A. Guillén, «Accuracy Assessment in Convolutional Neural Network-Based Deep Learning Remote Sensing Studies—Part 1: Literature Review,» *Remote Sensing,* vol. 13, nº 13, p. 2450, 2021.

[41] A. Luque, A. Carrasco, A. Martín y A. d. l. Heras, «The impact of class imbalance in classification performance metrics based on the binary confusion matrix,» *Pattern Recognition,* vol. 91, pp. 216-231, 2019.

[42] T. J. Liu, M. Christian, Y.-C. Chu, Y.-C. Chen, C.-W. Chang, F. Lai y H.-C. Tai, «A pressure ulcers assessment system for diagnosis and decision making using convolutional neural networks,» *Journal of the Formosan Medical Association,* 2022.

[43] «Gray level co-occurrence matrix and extreme learning machine for Covid-19 diagnosis,» *International Journal of Cognitive Computing in Engineering,* vol. 2, pp. 93-103, 2021.

[44] P. Asgari, M. Mohammad y M. Asgari, «The comparison of selected machine learning techniques and correlation matrix in ICU mortality risk prediction,» *Informatics in Medicine Unlocked,* vol. 31, 2022.

Anexos

- **Prueba test muestra sana.**

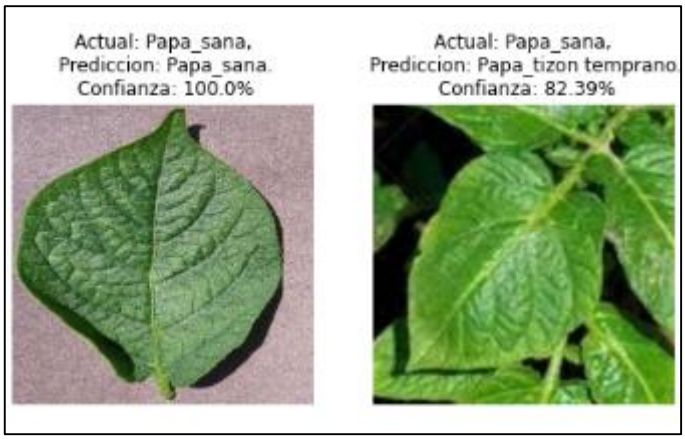


Figura 32 Test Muestra Sana

Fuente: *Elaboración propia*

- **Prueba test muestra tizón tardío.**

Figura 33 Test Muestra Tizón Tardío

Fuente: *Elaboración propia*

- **Prueba test muestra tizón temprano.**

Figura 34 Test Muestra Tizón Temprano

Fuente: *Elaboración propia*

- **Ejecución de la aplicación.**

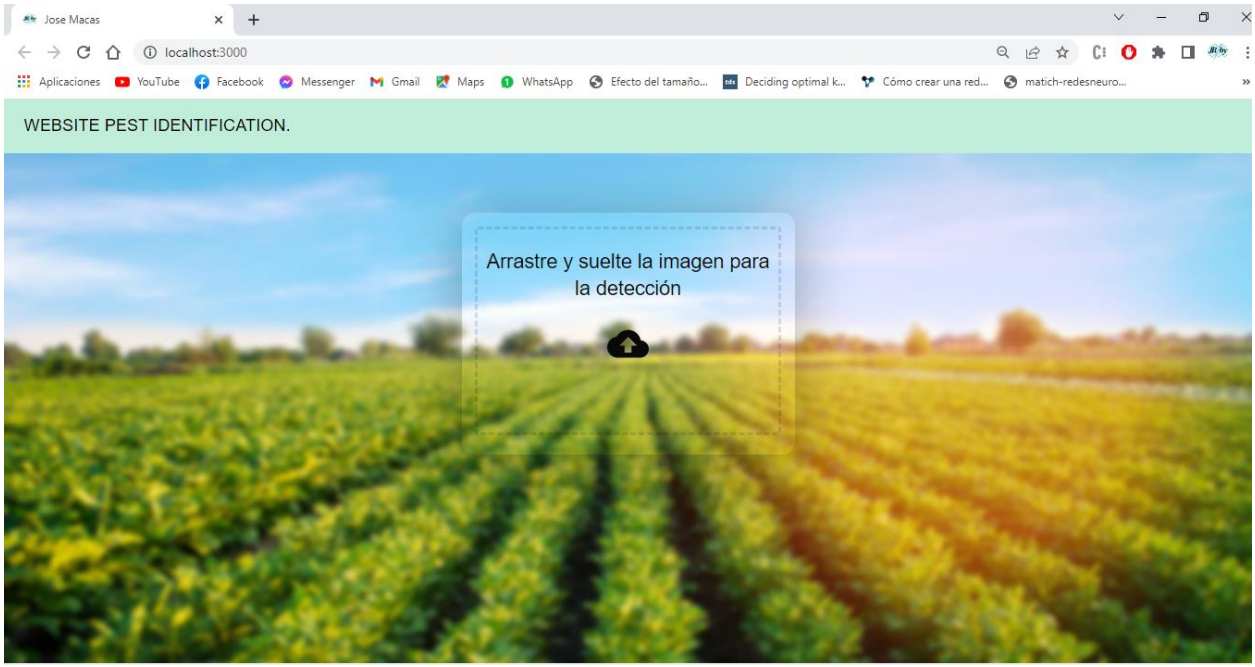


Figura 35 Interfaz Gráfica

Fuente: *Elaboración propia*

- **Test Realizado**

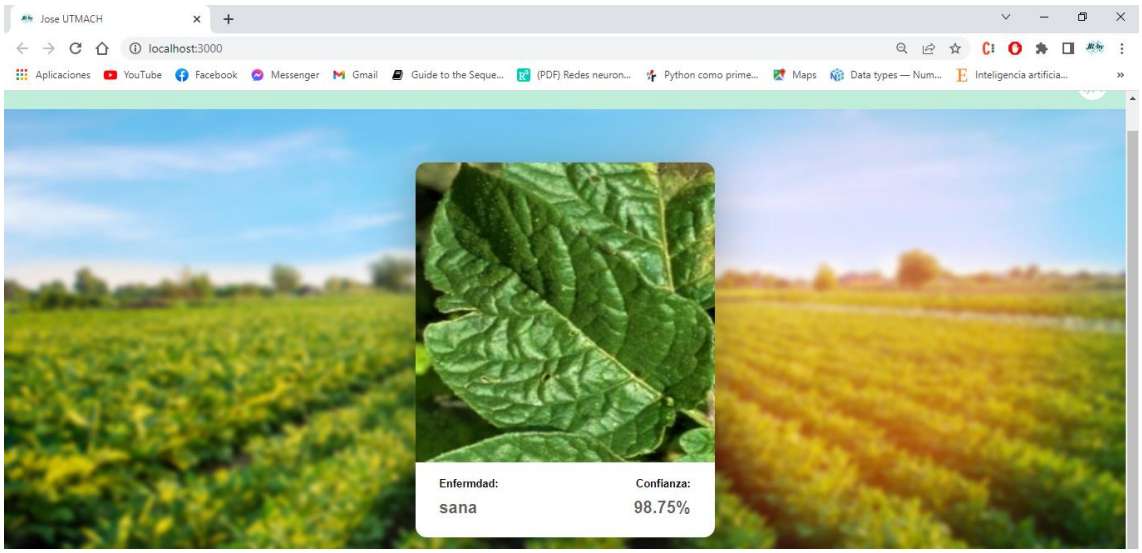


Figura 36 Resultado de la Aplicación

Fuente: *Elaboración propia*