El Colegio de Veracruz

RINDERESU
Revista Internacional de Desarrollo Regional Sustentable

ISSN 2448-5527

Detección de enfermedades foliares con arquitecturas de redes neuronales convolucionales

Eduardo A. Huerta-Mora[1*], Víctor González-Huitrón[2], Héctor Rodríguez-Rangel[1], Leonel Ernesto Amabilis-Sosa[2]

[1]Tecnológico Nacional de México / Instituto Tecnológico de Culiacán, División de Estudios de Posgrado e Investigación, Av. Juan de Dios Bátiz, No. 310, Culiacán 80014, México.

[2]CONACYT. Tecnológico Nacional de México / Instituto Tecnológico de Culiacán.

Av. Juan de Dios Bátiz, No. 310, Culiacán, México

*Autor de correspondencia: eduardo_huerta@itculiacan.edu.mx

Recibido 18 de mayo de 2020; aceptado 21 de junio de 2020

RESUMEN

Cuando los diferentes tipos de cultivos empiezan a verse afectados por diversas enfermedades, dicha afectación se manifiesta en su crecimiento y desarrollo, por eso la importancia de la detección temprana de enfermedades. Se describe un enfoque para desarrollar modelos de deep learning capaces de reconocer las enfermedades de plantas de interés comercial por medio de la clasificación de imágenes foliares, para lo cual se utilizó una microcomputadora Raspberry Pi 4 para su implementación en hardware. Algunos modelos empleados de un subconjunto de la inteligencia artificial, denominado deep learning, se utilizaron para la detección de plagas por medio de transfer learning con ajuste fino, con el fin de obtener altos índices de precisión con base en el conjunto de datos de Plant Village, que contiene treinta y ocho clases diferentes, que incluyen hojas enfermas y saludables. El objetivo principal del estudio es clasificar las diferentes clases de plantas con alta precisión, utilizando redes neuronales convolucionales con transfer learning implementado en un dispositivo hardware; los modelos fueron evaluados a través de un análisis basado en precisión,

sensibilidad, F-Score y exactitud. Los resultados obtenidos presentan valores significativos obtenidos por la técnica VGG16, con 90% de sensibilidad y 90% de exactitud. Por lo anterior, es posible concluir que el modelo VGG16 es una herramienta útil para los agricultores en la identificación y protección de las plantas de las enfermedades mencionadas.

PALABRAS CLAVE: Detección temprana, enfermedades, Plant Village, deep learning, transfer learning.

ABSTRACT

When different types of crops begin to be affected by various diseases, this is reflected in their growth and development, which is where the importance of early disease detection comes in. An approach is described to develop deep learning models that recognize plant diseases of commercial interest through the classification of leaf images, for wich a Raspberry Pi 4 microcomputer was used for hardware implementation. Some models employed from a subset of artificial intelligence, called deep learning, were used for pest detection by fine-tuning transfer learning to obtain high accuracy rates, based on the Plant Village dataset containing thirty-eight different classes, including diseased and healthy leaves. The main objective of the study is to classify the different classes of plants with high precision, using convolutional neural networks with transfer learning implemented in a hardware device; the models were evaluated through a analysis based on precision, sensitivity, F-Score and accuracy. The results present significant values obtained by the VGG16 technique, with 90% of sensitivity and 90% of accuracy. For the above, it's possible to conclude that the VGG16 model is a useful tool for farmers to help and protect plants from the diseases mentioned.

KEY WORDS: Detection of diseases, Plant Village, transfer learning, deep learning models.

INTRODUCCIÓN

En el 2017, en Sinaloa se cultivaron 1 millón 149 mil 320 hectáreas, ocupando el cultivo de hortalizas 71 mil 014 hectáreas, lo que representó el 6.18% del total (CODESIN, 2018), por lo que el diagnóstico oportuno y preciso de las enfermedades en las plantas es uno de los pilares de la agricultura de precisión. Actualmente, se tienen diferentes formas para detectar patologías de plantas mediante diversos modelos que involucran el uso de inteligencia artificial (Mohanty et al., 2016). Al respecto, Sharma et al. (2018), reportan el uso de un clasificador con SVM y KNN desarrollado en Python; mientras que Toda et al. (2019), reportan que se aumentó la precisión de los modelos estudiados al establecer redes neuronales convolucionales de deep learning (CNN por sus siglas en inglés) como un clasificador que presenta mejor generalización de los datos.

En lo referente a la implementación en hardware, se establece un sistema de monitoreo para hortalizas, el cual es implementado mediante microcontroladores y sistemas de control clásico (Hemming et al., 2019); mientras que otros autores reportan sistemas de control mediante ML, los cuales son capaces de monitorear y ajustar parámetros de acuerdo con el tipo de cultivo y condiciones ambientales (Castro, 2016). Por su parte, Santos Senra (2017) presenta un sistema de monitoreo implementado con Raspberry Pi 3, el cual es capaz de almacenar la información y proporcionar interfaces gráficas, por lo que un modelo automatizado diseñado para apoyar en la detección de anomalías en las plantas por la apariencia y síntomas visuales será de gran ayuda para los agricultores y/o expertos en la agricultura en la verificación del diagnóstico de enfermedades.

Debido al avance progresivo en deep learning y su creciente adopción para el apoyo en tareas de detección y clasificación, es posible diseñar un sistema que sea capaz de integrar los avances realizados en deep learning para el tratamiento de imágenes, junto con hardware especializado con miras a una futura implementación en campo que logre apoyar y mejorar la detección temprana de plagas y enfermedades. Por lo anterior, el objetivo del presente trabajo fue detectar enfermedades que ocurren en diversos cultivos o en invernaderos mediante el programa deep learning, con el fin de detectar las diversas enfermedades en las hojas. Como objetivo adicional, se

determinó implementar el sistema en hardware específico, con el objetivo de obtener detecciones en tiempo real en una microcomputadora de Raspberry Pi 4.

De esta forma, el objetivo principal fue comparar los modelos de las diferentes arquitecturas de convolutional neural network, utilizando las métricas de matriz de confusión: precisión, recall, F-score.

MATERIALES Y MÉTODOS

El procedimiento se divide en cuatro pasos básicos: adquisición, capacitación, clasificación y evaluación de los datos, como se muestra en la figura 1, que se detallan a continuación:

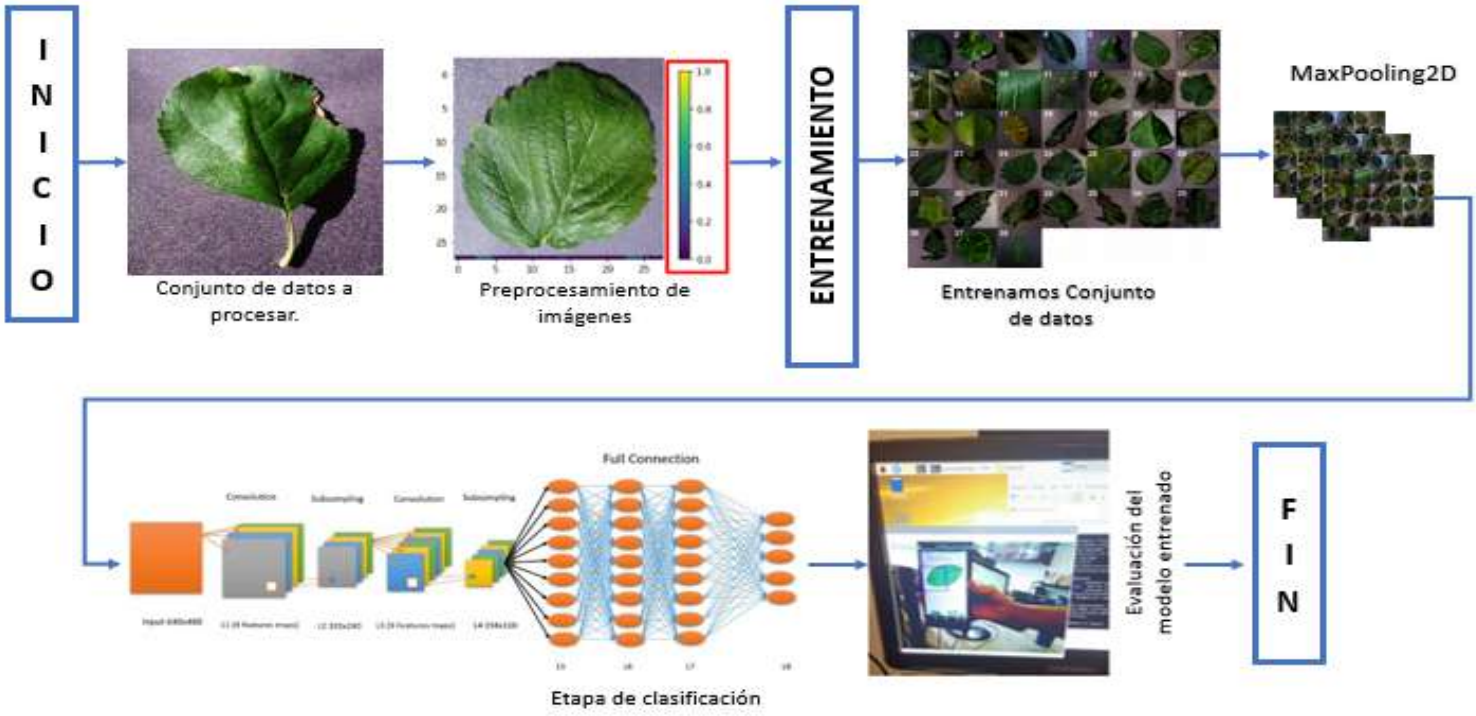

Figura 1. Diagrama de flujo del procedimiento.

2.1.- Adquisición de datos y preprocesamiento.

Conjunto de datos Plant Village. Es un repositorio abierto que contiene 54,485 imágenes, de 14 cultivos y 38 tipos de enfermedades de las plantas. De este conjunto de datos, se extrajeron las imágenes de las hojas de las diferentes clases (Hughes et al., 2016); la figura 2 muestra un ejemplo de cada clase, y la tabla 1 ofrece un

resumen del conjunto de datos. Todas las imágenes utilizadas se re-escalaron para ser de 224 × 224 o 299 × 299 pixeles, de acuerdo con el tamaño de entrada de la red. Los datos se separaron en dos conjuntos, que contienen el 70% de los datos en el conjunto de entrenamiento y el 30% restante en el conjunto de pruebas, y la elección de la división se basó en la propuesta de (Mohanty et al., 2016).

Figura 2. Imágenes de muestra de Plant Village - conjunto de datos de las 38 clases.

El rendimiento de las redes neuronales de deep learning a menudo mejoran con la cantidad de datos disponibles, y el realizar este aumento de manera artificial se denomina ‘data augumentation’ o ‘aumento de datos’, que es una parte primordial del preprocesamiento de datos.

Tabla 1. Conjunto de datos Plant Village

Núm.	**Clases**	**Núm. de imágenes**
1	Sarna del manzano	636
2	Pudrición negra del manzano	625
3	Óxido del cedro y del manzano	281
4	Manzana saludable	1641
5	Blueberry saludable	1502
6	Cereza saludable	855
7	Moho polvoriento de la cereza	1054
8	Mancha foliar gris del maíz	526
9	Moho común del maíz	1215
10	Maíz saludable	1164
11	Tizón de la hoja del norte de maíz	994
12	Pudrición negra de la uva	1188
13	Enfermedad del tronco de la uva	1383
14	Uva saludable	423

15	Tizón de la hoja de la uva	1079
16	Enfermedad del brote amarillo de la naranja	5522
17	Mancha bacteriana del durazno	2309
18	Durazno saludable	361
19	Mancha bacteriana del pimiento	1000
20	Pimiento saludable	1478
21	Tizón temprano de la papa	1005
22	Papa saludable	152
23	Tizón tardío papa	1003
24	Frambuesa saludable	372
25	Frijol de soja saludable	5090
26	Moho polvoriento de calabaza	1835
27	Fresa saludable	458
28	Quemadura de la hoja de fresa	1114
29	Tomate mancha bacteriana	2133
30	Tomate tizón temprano	1010
31	Tomate saludable	1595
32	Tomate tizón tardío	1916
33	Moho de la hoja del tomate	962
34	Tomate mancha bacteriana septoria	1779
35	Tomate araña roja	1677
36	Tizón corynespora del tomate	1412
37	Tomate virus mozaico	373
38	Virus del rizado amarillo del tomate	5358
	Total	**54,485**

La intención es expandir el conjunto de datos de capacitación con ejemplos nuevos y plausibles, lo cual significa las variaciones de las imágenes del conjunto de entrenamiento que es probable que vea el modelo (Brownlee, 2019, s.f.).

Las redes neuronales convolucionales de deep learning necesitan una gran cantidad de imágenes para que el modelo se entrene de manera efectiva, lo cual ayuda a aumentar el rendimiento del modelo al generalizar mejor y, por lo tanto, a reducir el sobreajuste. Los conjuntos de datos más populares para los datos de clasificación y detección de objetos tienen de miles a millones de imágenes (Khandelwal, 2019).

Para este proyecto se realizaron tres transformaciones, que son zoom, rotación de imagen, unos cuantos grados y cortar o recortar el conjunto de datos de Plant Village; en la tabla 2 se describen los valores utilizados en 'data augmentation'.

Tabla 2. Parámetros de data augmentation para el conjunto de datos de plantvillage.

Tipos de transformación	**Valores**
Rotation range	30°
Shear range	0.2
Zoom range	0.2
Rescale	1/255

2.2. Entrenamiento de datos

En este paso, la capacitación de las redes neuronales convolucionales se llevó a cabo con el conjunto de datos ImageNet (Gupta, 2017), que contiene 1.2 millones de imágenes de 1000 categorías, con el objetivo de inicializar los parámetros antes de la capacitación en el conjunto de datos de las treinta y ocho clases. En la siguiente etapa, se utilizó transfer learning (Pan et al., 2010), que tiene como objetivo transferir el conocimiento de uno o más dominios y aplicar el conocimiento a otro con una tarea objetivo diferente (al., 2016). Cabe señalar que el ajuste fino es un concepto de aprendizaje de transferencia, que consiste en reemplazar la capa de salida pre-entrenada con una capa que contiene el número de clases del conjunto de datos Plant Village.

Para este trabajo, se reemplazan las capas FC y la última capa, una capa de salida softmax de predicción. El objetivo principal del uso de modelos de redes neuronales convolucionales pre-entrenados está relacionado con el entrenamiento rápido y fácil, usando parámetros inicializados aleatoriamente (Deniz et al., 2018), así como el logro de un error de entrenamiento menor que las redes neuronales artificiales que no están pre-entrenados.

El rendimiento de las siguientes arquitecturas de redes neuronales convolucionales se ha evaluado para el problema de clasificación de las treinta y ocho clases del conjunto de datos Plant Village.

En este estudio, se utilizaron métodos de deep learning para detectar enfermedades; la selección de la arquitectura de deep learning fue el tema clave para la implementación. De modo que se evaluaron cinco

arquitecturas de red de deep learning diferentes, VGG16 (Sahu, 2019), InceptionV3 (Szegedy et al., 2016), MobileNetV2 (Velasco et al., 2019), Xception (Francois, 2017) y ResNet50V2 (Simonyan et al., 2014).

Para estas redes de deep learning, el entrenamiento y la validación se realizaron en un equipo de cómputo con procesador Intel i7-7700 y tarjeta de video NVIDIA GTX1070 con 8GB en RAM.

2.2.1. VGG16

VGG16 consta de 16 capas de parámetros, que incluyen trece capas convolucionales con un tamaño de filtro de 3 X 3, y el resto son capas completamente conectadas, mientras que las configuraciones de capas completamente conectadas en VGG-16 son las mismas con AlexNet. La zancada y el relleno de todas las capas convolucionales se fijan en 1 píxel. Todas las capas convolucionales se dividen en 5 grupos y cada grupo es seguido de una capa de agrupación máxima (Sahu, 2019).

2.2.2. InceptionV3

InceptionV3 (Szegedy et al., 2016) es una arquitectura convolucional profunda ampliamente utilizada para tareas de clasificación. La red Inception V3 tiene múltiples bloques de construcción simétricos y asimétricos, donde cada bloque tiene varias ramas de convoluciones, agrupación promedio, agrupación máxima, concatenado, deserciones y capas completamente conectadas. Esta red tiene 42 capas totales y posee 29,3 millones de parámetros (Chebet Too et al., 2018). Finalmente, se concluye que la combinación de un recuento de parámetros más bajo y una regularización adicional, con suavizadores de etiquetas de clasificadores auxiliares normalizados por lotes, permite el entrenamiento de una red de alta calidad en conjuntos de entrenamiento de tamaño relativamente modesto (Szegedy et al., 2016).

2.2.3. ResNet50V2

Red residual (ResNet) son modelos que fueron desarrollado por He, Zhang, Ren y Sun (2016), y se basan en arquitecturas profundas que han demostrado buenos comportamientos de convergencia y precisión convincente. ResNet fue construido por varias unidades residuales apiladas, y desarrollado con muchos números diferentes de capas: 18, 34, 50, 101, 152 y 1202, sin embargo, el número de operaciones puede variar según

las diferentes arquitecturas; para lo anterior, las unidades residuales se componen de convolucional, agrupación y capas. ResNet es similar a VGG net, pero ResNet es aproximadamente ocho veces más profundo que VGG (Simonyan et al., 2014), pues contiene 49 capas convolucionales y una capa totalmente conectada al final de la red. Finalmente, para ahorrar recursos informáticos y tiempo de capacitación, se eligió ResNet50V2 para el desarrollo del trabajo.

2.2.4. MobileNetV2

Introduce los conceptos de residuos invertidos (inverted residuals) y cuellos de botella lineales (linear bottlenecks), que permiten construir capas que toman entradas comprimidas con pocas dimensiones y las expanden a más dimensiones, permitiendo la aplicación de filtros separables en profundidad, para luego proyectar nuevamente la salida a un número bajo de dimensiones mediante una convolución lineal. Lo anterior aumenta la eficiencia del modelo, mientras se obtiene una exactitud comparable a la de la primera familia de MobileNets (Velasco et al., 2019).

2.2.5. Xception

Presentamos una interpretación de los módulos inception en redes neuronales convolucionales como un paso intermedio entre la convolución regular y la operación de convolución separable en profundidad (una convolución en profundidad seguida de una convolución puntual). Desde este punto de vista, una convolución separable en profundidad se entiende como un módulo de inicio con un número máximo de torres, observación que lleva a proponer una nueva arquitectura de convolutional neural network profunda inspirada en Inception, donde los módulos han sido reemplazados por convoluciones separables en profundidad. La arquitectura, denominada Xception, supera ligeramente a InceptionV3 en el conjunto de datos de ImageNet (para lo cual se diseñó InceptionV3), y supera de forma significativa a InceptionV3 en un conjunto de datos de clasificación de imágenes más grande que comprende 350 millones de imágenes y 17,000 clases. Dado que la arquitectura Xception tiene el mismo número de parámetros que Inception V3, las ganancias de rendimiento no se deben a una mayor capacidad, sino a un uso más eficiente de los parámetros del modelo (Francois, 2017).

2.2.6. Configuración de redes neuronales convolucionales

La configuración de redes neuronales convolucionales generalmente consiste en una serie de elementos específicos, que son los que presentan las variaciones en las diferentes arquitecturas. La figura 3 presenta de forma gráfica la arquitectura general de una red neuronal convolucional, con sus elementos principales, como son la capa de entrada, la capa convolucional, la capa de agrupación y un proceso de aplanamiento, donde la información se ingresa en un conjunto de capas densas que representan el resultado obtenido en la capa de salida (Maeda et al., 2020).

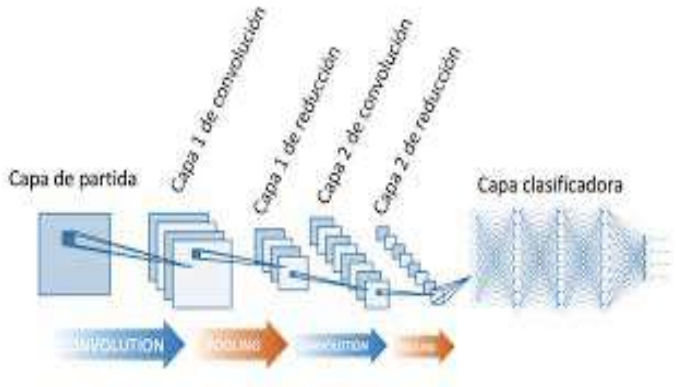

Figura 3. Representación de la arquitectura de una red neuronal convolucional.

Por lo tanto, las características de las arquitecturas utilizadas se describen en la tabla 3.

Tabla 3. Resumen de las arquitecturas utilizadas

Red	Profundidad	Parámetros (millones)	Tamaño de imagen de
VGG16	16	15.6	224x224
InceptionV3	48	23.7	299x299
MobileNetV2	19	2.3	224x224
ResNet50V2	50	27.4	224x224
Xception	71	20.9	299x299

Para permitir una comparación equitativa entre las diferentes arquitecturas, también se estandarizaron los hiperparámetros entre ellas; en la tabla 4 se detallan los utilizados.

Tabla 4. Hiperparámetros de entrenamiento.

Hiperparámetros	Valor
Algoritmo de optimización	Adam*
Épocas	10
Batch size	64
Loss	categorical_crossentropy

Adam es un algoritmo para la optimización basada en gradiente de primer orden de funciones objetivo-estocásticas, basado, a su vez, en estimaciones adaptativas de momentos de orden inferior. El método es fácil de implementar, así como computacionalmente eficiente; tiene pocos requisitos de memoria, es invariable para el cambio de escala diagonal de los gradientes y es muy adecuado para problemas que son grandes en términos de datos y/o parámetros. Los resultados empíricos demuestran que *Adam* funciona bien en la práctica y se compara favorablemente con otros métodos de optimización estocástica (Kingma et al., 2017).

Todos los experimentos se realizaron en una estación de trabajo, que se muestran en la tabla 5. El proceso de capacitación fue realizado con Python 3.7.6, Keras (2.2.4) y Tensorflow (1.15.0), que proporciona un marco para diseñar e implementar CNN, donde las aplicaciones y los gráficos mediante matplotlib ayudan a visualizar las activaciones de la red y monitorear el progreso de la capacitación en la red.

Tabla 5. Especificaciones del equipo utilizado para el entrenamiento de la red convolucional.

Hardware y software	Características
Memoria	16GB
Procesador	CPU Intel Core i7-7700 a 3.60 GHz
Gráficos	GeForce GTX 1070 X 4 GB
Sistema operativo	Windows 10, 64 bits

2.3. Clasificación de datos

El número de la capa de salida de clasificación es igual al número de las clases; cada salida tiene una probabilidad diferente para la imagen de entrada porque este tipo de modelos tienen la capacidad de aprender automáticamente las características durante la etapa de entrenamiento, por lo que el modelo elige la probabilidad más alta como su predicción de la clase. Finalmente, en este paso se determina qué enfermedad está presente en la hoja utilizando el conjunto pre-entrenado (Durmus et al., 2017).

2.4. Evaluación

El rendimiento del método propuesto se evaluó comparando los modelos pre-entrenados con diferentes

métricas. Además, la calidad de los algoritmos de aprendizaje generalmente se evalúa analizando su desempeño en los datos de una prueba (Montero et al., 2019).

La matriz de confusión es una de las métricas más intuitivas y sencillas que se utiliza para estimar la precisión y exactitud del modelo, por lo que se utiliza para el problema de clasificación donde la salida puede ser de dos o más tipos de clases (González, 2019).

Una matriz de confusión es un resumen de los resultados de predicción sobre un problema de clasificación, y el número de predicciones correctas e incorrectas se resume con valores de conteo y se desglosa por clase. También muestra las formas en que el modelo de clasificación se confunde cuando hace predicciones; además de dar una idea no solo de los errores que está cometiendo un clasificador, sino, más importante aún, de los tipos de errores que se están cometiendo (Geeks, 2020).

La precisión, definida por la ecuación 1, trata sobre la corrección, es decir, evalúa el poder predictivo del algoritmo; asimismo, la precisión es cuán 'preciso' es el modelo fuera de los positivos predichos y cuántos de ellos son realmente positivos.

$$Precision = \frac{TP}{TP+FP} \quad (1)$$

La sensibilidad o 'recall' corresponde a la precisión de los ejemplos positivos, y se refiere a cuántos ejemplos de las clases positivas se etiquetaron correctamente; esto se puede calcular con la ecuación 2, donde TP se refiere a los verdaderos positivos, que son el número de instancias que son positivas y están correctamente identificadas; mientras que FN representa los falsos negativos, que son el número de casos positivos que se clasifican erróneamente como negativos.

$$Sensibilidad\ o\ recall = \frac{TP}{TP+FN} \quad (2)$$

El F-score se determina como la precisión y recall media armónica, como se muestra en la ecuación 3, y se centra en el análisis de clase positiva. Un valor alto de esta métrica indica que el modelo funciona mejor en la clase positiva.

$$F - score = 2 * \frac{Presicion*recall}{Presicion+recall} \quad (3)$$

Comúnmente, la exactitud es la métrica más utilizada para evaluar el rendimiento de la clasificación. En la

etapa de evaluación, la exactitud se calculó cada 10922 iteraciones. Esta métrica calcula el porcentaje de muestras que se clasifican correctamente y se representa en la ecuación 4:

$$Exactitud\ o\ accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (4)$$

2.5. Implementación en hardware específico.

La microcomputadora Raspberry Pi es una placa computadora o SBC (del inglés: Single Board Computer) de bajo coste, desarrollada en Reino Unido con el objetivo de facilitar el acceso a la tecnología de computación a un público más amplio y potenciar la enseñanza de esta ciencia en las escuelas.

Para la clasificación de plantas y el procesamiento de imágenes, se ha optado por utilizar la microcomputadora Raspberry Pi modelo 4 debido a las características que ofrece el miniordenador, el cual se muestra en la tabla 6; fue gracias a este equipo que se pudo tomar en tiempo real imágenes mediante el módulo de cámara oficial de Raspberry Pi (Pi-Camera) y lograr así una clasificación de las mismas (Pérez, 2016).

El sistema operativo que posee la microcomputadora Raspberry Pi está basado en GNU/Linux, el cual dispone de una gran colección de software, que se distribuye en su mayor parte bajo una licencia libre o de código abierto (open source) (Pérez, 2016).

Tabla 6. Especificaciones de la microcomputadora Raspberry Pi 4 modelo B

Sistema de un chip	Broadcom BCM2711.
CPU	Procesador de cuatro núcleos a 1,5 GHz con brazo cortex-A72.
GPU	Videocore VI.
Memoria	4GB LPDDR4 RAM
Conectividad	802.11ac wi-fi / bluetooth 5.0, gigabit, ethernet.
Video y Sonido	2 x puertos micro-HDMI que admiten pantallas de 4K@60Hz a través de HDMI 2.0, puerto de pantalla MIPI DSI, puerto de cámara MIPI CSI, salida estéreo de 4

	polos y puerto de vídeo compuesto.
Puertos	2 x USB 3.0, 2 x USB 2.0
Alimentación	5V/3A vía USB-C, 5V vía cabezal GPIO.
Expansión	Cabezal GPIO de 40 pines.

RESULTADOS

En este trabajo, se realizó una evaluación de modelos pre-entrenados para la tarea de predicción de enfermedades de las plantas de las diferentes clases del conjunto de datos de Plant Village; los resultados se muestran en la tabla 7

Tabla 7. Medidas de rendimiento (%) de validación o test para cada modelo pre-entrenado.

Medidas de desempeño	**VGG16**	**InceptionV3**	**MobileNetV2**	**ResNet50V2**	**Xception**
Precision (test)	91%	71%	72%	85%	92%
Recall (test)	90%	55%	55%	75%	87%
F-score (test)	90%	54%	53%	75%	87%
Accuracy (test)	90%	55%	56%	75%	87%
Tiempo (mins)	220	192	102	207	238

Solo dos modelos mostraron resultados en la validación o test similares con las métricas de precisión; Xception tuvo el mejor resultado en la métrica de precisión con un 92%, seguido de VGG16 con 91%, ResNet50V2 con 85%, MobileNetV2 con 72%, y el más bajo fue InceptionV3 con 71%. Del mismo modo, el orden de los resultados de precisión fue igual a la medida anterior, y el porcentaje más bajo fue obtenido por InceptionV3 con 71%, y el más alto por Xception con 92% (tabla 7). Finalmente, en las mediciones de sensibilidad o recall y F-score, el nuevo rendimiento de InceptionV3 y MobileNetV2 fueron menores, con 55%, 54% y 55.24% por parte de InceptionV3, y 55%, 53% y 56.03% por MobileNetV2, en oposición a VGG16 y Xception, que obtuvieron los mejores porcentajes en las métricas anteriores con 91%, 90% y 90% por VGG16, y 92%, 87% y 87% por parte de Xception, respectivamente. De

hecho, como se muestra, estos dos últimos alcanzaron un rendimiento mayor en cada medida, pero la implementación de VGG16 alcanzó el porcentaje más alto analizando los resultados, lo que implica que esta arquitectura es mucho más eficiente para este trabajo en específico; la diferencia radica en que la cantidad de parámetros, debido a las distintas cantidades de capas, es la arquitectura con menor número de profundidad. Además, en función del tiempo de procesamiento que le tomó a cada red neuronal convolucional llevar a cabo el proceso de clasificación, MobileNetV2 mostró el mejor desempeño al tomarse el menor tiempo gracias a su menor cantidad de parámetros. lo que significa una mejor eficiencia en comparación con las otras arquitecturas, señalando que su desventaja en las medidas estadísticas no es significativa en comparación con los resultados de las otras convolutional neural netwoks, por lo que podría considerarse sacrificar una mayor disminución en la precisión a cambio de una mayor eficiencia de procesamiento.

La figura 4 presenta la matriz de confusión del modelo VGG-16; dependiendo de los resultados, es posible evaluar visualmente el rendimiento del clasificador y determinar qué clases están resaltadas por las neuronas del modelo VGG-16. Las filas están relacionadas con la clase de salida, mientras que las columnas están relacionadas con la clase verdadera. Por su parte, las celdas diagonales están asociadas con las observaciones que están clasificadas correctamente, y las celdas fuera de diagonal corresponden a las observaciones clasificadas incorrectamente.

Entre las treinta y ocho clases, de tres a nueve produjeron resultados de clasificación 100% correctos, ya que esas enfermedades tienen una apariencia y características distintivas en comparación con las otras clases. La implementación de la Raspberry Pi 4 modelo B en la clasificación de imágenes de plantas en tiempo real, utilizando una Pi Camera, como la que se muestra en la figura 5, se cargaba el modelo de entrenamiento generado en un archivo con extensión h5, que será para procesar ese modelo dentro del miniordenador R-Pi.

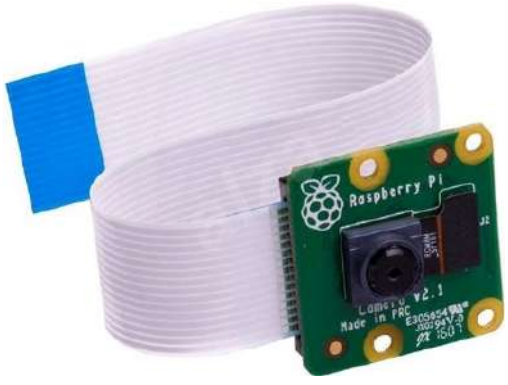

Figura 5. Pi-Cámara utilizada para la clasificación de imágenes en R-Pi.

En la figura 6 se muestran los resultados mediante la cámara en tiempo real. Cada arquitectura de transfer learning fue implementada en la microcomputadora Raspberry Pi 4, obteniendo una clasificación cada 20 segundos, por lo cual se probaron varias clases para comprobar el correcto funcionamiento de estas redes neuronales convolucionales dentro de la misma.

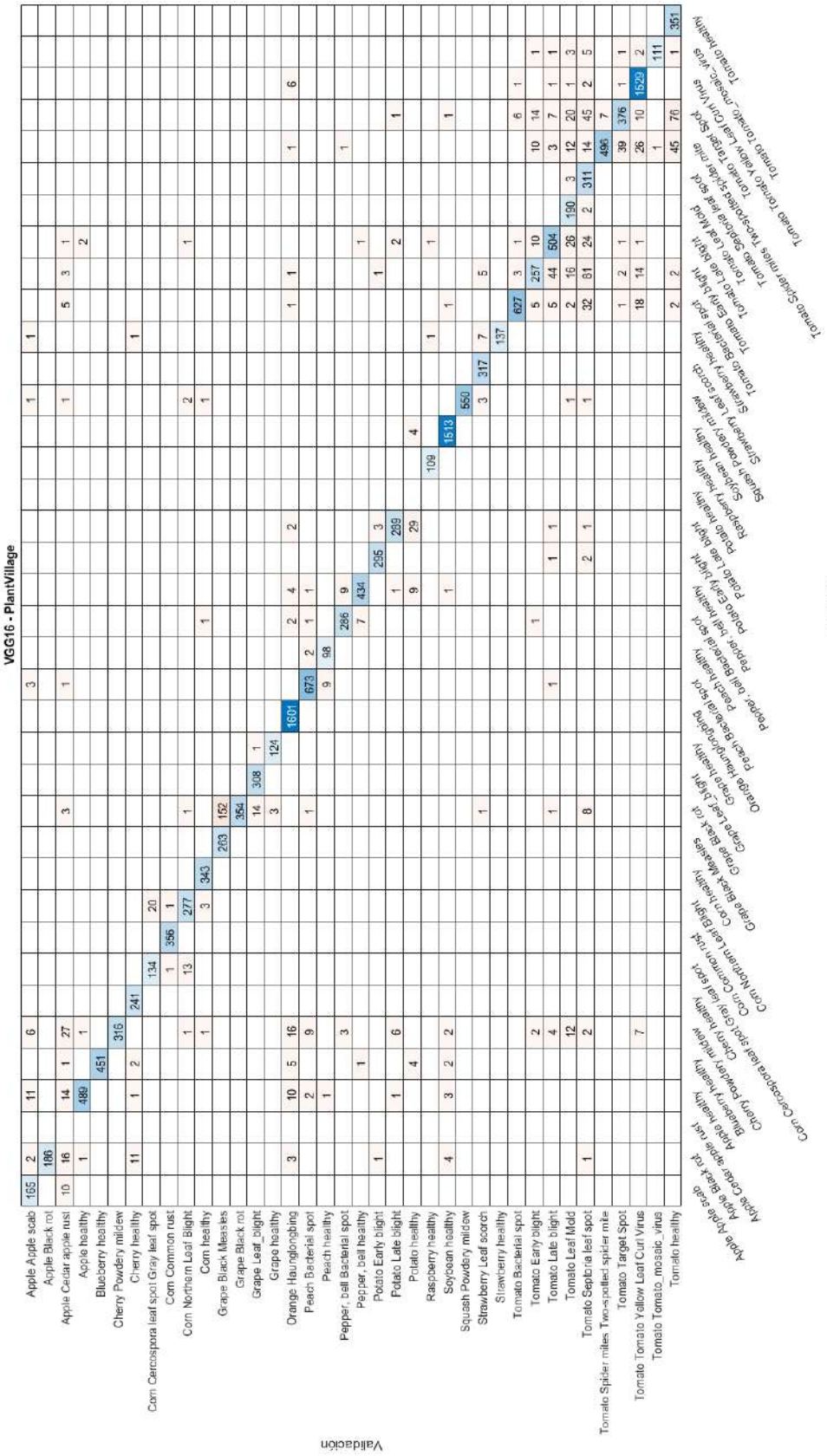

Figura 4. Matriz de Confusión del modelo VGG16

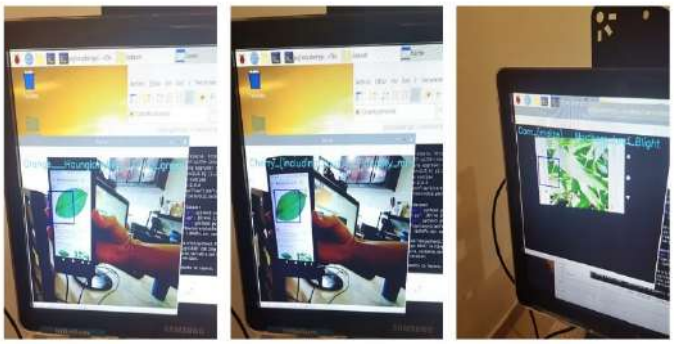

Figura 6. Representación en tiempo real de la clasificación de plantas con los diferentes modelos de transfer learning.

Discusión

Deep learning presenta una oportunidad para ampliar la investigación y la aplicación de la base para la clasificación y predicción de enfermedades de las plantas mediante imágenes digitales, pero se requieren modelos rápidos y precisos para detectar las enfermedades a tiempo (Velasco et al., 2019). Para este trabajo, el conjunto de datos utilizado presenta 26 clases de enfermedades de la diversidad de clases y 12 clases saludables del conjunto de datos de Plant Village, que tiene un total de 54, 485 imágenes. El conjunto de datos se dividió en 70% – 30% (entrenamiento-validación). Por lo tanto, cinco modelos pre-entrenados de última generación, a saber, VGG16, InceptionV3, MobileNetV2, ResNet50V2 y Xception, fueron entrenados por el concepto de ajuste fino, el cual consiste en reemplazar las últimas tres capas, y la capa de salida final debe ser compatible con el mismo número de clases entrenadas.

Todos estos modelos pre-entrenados fueron evaluados por varias métricas: precisión, sensibilidad (recall) y F-score, utilizando los mismos hiperparámetros. Según la tabla 7, que muestra un rendimiento general de resultados, Xception logró mejores resultados que las otras arquitecturas; además, Inception V3, que es uno de los modelos más profundos de redes neuronales convolucionales, muestra un bajo rendimiento en la tabla 3.

Es posible determinar que las clases con el menor error fueron ocho: tomate saludable, frambuesa saludable, durazno saludable, enfermedad del brote amarillo de la naranja, tizón de la hoja de la uva , maíz Saludable, moho común del maíz y la clase cereza saludable; las clases con mayor error fueron papa aaludable, y óxido del cedro y del manzano. Todos estos se basaron en la métrica de precisión.

Por otro lado, una de las limitaciones de este trabajo estuvo relacionada con la cantidad de imágenes utilizadas, por lo que podría ser interesante probar el modelo en un conjunto de imágenes tomadas en

condiciones controladas o poder clasificar imágenes de enfermedades, tal como se presentan en la planta.

Asimismo, podría ser beneficioso crear una aplicación móvil que implemente el modelo VGG16 para un diagnóstico automático de enfermedades de las plantas del conjunto de datos de PlantVillage, para que los usuarios o agricultores con poco o ningún conocimiento puedan usarla y realizar su trabajo de manera efectiva en la detección de plantas de las clases de PlantVillage.

También es importante mencionar que la capacitación de los modelos lleva mucho tiempo (alrededor de 1 o 2 horas en una computadora con CPU de alto rendimiento, o incluso más), mientras que la clasificación es muy rápida en una unidad de procesamiento de gráficos (GPU).

Uno de los problemas principales detectados se debe a la alta cantidad de parámetros, el bajo procesamiento de una Raspberry Pi 4 para predecir las diversas clases de hojas saludables y enfermas, trabajando con modelos de arquitectura de transfer learning, así como la baja cantidad en las métricas de rendimiento de algunas clases a causa de la poca cantidad de datos, provocando un déficit en la precisión de clases, como papa saludable, entre otras.

CONCLUSIONES

Como principales aportaciones, se enumeran las siguientes: 1) Metodología para el uso de imágenes de origen agrícola con las redes neuronales convolucionales y librería keras para tareas de clasificación de las diferentes clases con las que se entrenaron los modelos. 2) La segunda aportación se concentró en comparar y analizar los rendimientos de VGG16, InceptionV3, MobileNetV2, ResNet50V2 y Xception, con 4 diferentes métricas de rendimiento; la comparación de este tipo de modelos presenta diversas ventajas, pues las redes neuronales convolucionales no requieren ningún procesamiento previo exhaustivo porque se cambian sus últimas capas para adecuarlas a que realicen predicciones respecto a las clases que queremos predecir, además de que tienen una tasa de convergencia (junto a su rendimiento de entrenamiento) considerado alto. 3) La selección del modelo más adecuado para la tarea de predicción de enfermedades de las diferentes clases del conjunto de datos de PlantVillage, pues cada modelo utilizado fue capaz de predecir 26 enfermedades de 12 clases de hojas distintas.

Recursos

Los datos y el código utilizados en este documento están disponibles en las siguientes ubicaciones:

Código:https://github.com/EduardoHuerta/PlantVillage

Datos:https://github.com/EduardoHuerta/PlantVillage/tree/master/dataset/pv

BIBLIOGRAFÍA

Brownlee, J. (3 de 07 de 2017). *Machine Learning Mastery.* Obtenido de Gentle Introduction to the Adam Optimization Algorithm for Deep Learning: https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/

Brownlee, J. (12 de Abril de 2019). *Machine Learning Mastery*. Obtenido de How to Configure Image Data Augmentation in Keras: https://machinelearningmastery.com/how-to-configure-image-data-augmentation-when-training-deep-learning-neural-networks/

Castro Silva, J. A. (2016). *Sistema de riego autónomo basado en la Internet de las Cosas.* Neiva-Huila(Colombia). Recuperado el 13 de Mayo de 2020, de https://reunir.unir.net/bitstream/handle/123456789/3648/CASTRO%20SILVA,%20JUAN%20ANTONIO.pdf?sequence=1

CHATTERJEE, S. (2019). *Plant Disease Detection Using Convolutional Neural Network.* Indian Institute of Technology, Indian School of Mines , India.

Chebet Too, E., Yujian, L., Njuki, S., & Yingchun, L. (27 de Marzo de 2018). A comparative study of fine-tuning deep learning models for plant disease. *ELSEVIER*, 8. Recuperado el 13 de Mayo de 2020, de http://download.eline.ba/downloads/PowerUpdate/too2018.pdf

CODESIN. (04 de junio de 2018). *sinaloaennumeros*. Obtenido de unidad de estadística y análisis.: http://sinaloaennumeros.com/agricultura-en-sinaloa-2017/

Deniz, E., Şengür, A., Kadiroğlu, Z., Guo, Y., & Budak, Ü. (2018). Transfer learning based histopathologic image classifcation for breast cancer detection. *Health Information Science Journal*, 7. Recuperado el 13 de Mayo de 2020, de https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6162199/pdf/13755_2018_Article_57.pdf

Durmus, H., Gunes, E., & kircy, M. (2017). Detección de enfermedades en las hojas de las plantas de tomate mediante el aprendizaje profundo. *6a Conferencia Internacional de Agrogeoinformática* (págs. 1-5). Fairfax, VA, EE. UU.: IEEE.

Francois, C. (2017). *CVF.* Obtenido de Xception: aprendizaje profundo con convoluciones separables

en profundidad: http://openaccess.thecvf.com/content_cvpr_2017/html/Chollet_Xception_Deep_Learning_CVPR_2017_paper.html

geeks, G. A. (24 de 04 de 2020). *GeeksforGeeks*. Obtenido de https://www.geeksforgeeks.org/confusion-matrix-machine-learning/

González, L. (17 de 05 de 2019). Obtenido de https://www.youtube.com/watch?v=r5WIImKV1XA

Gupta, V. (26 de Diciembre de 2017). *Aprende OpenCV*. Obtenido de https://www.learnopencv.com/keras-tutorial-using-pre-trained-imagenet-models/

Hammad Saleem , M., Potgieter, J., & Mahmood Arif, K. (2019). Plant Disease Detection and Classification by. *MDPI*, 22.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. *CVF-Computer Vision Foundation*, 9. Recuperado el 13 de Mayo de 2020, de http://openaccess.thecvf.com/content_cvpr_2016/papers/He_Deep_Residual_Learning_CVPR_2016_paper.pdf

Hemming, S., de Zwart, F., Elings, A., Righini, I., & Petropoulou, A. (2019). Remote Control of Greenhouse Vegetable Production with Artificial Intelligence—Greenhouse Climate, Irrigation, and Crop Production. *MDPI*, 9. Recuperado el 14 de Mayo de 2020, de https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6515393/

Hughes, D. P., & Salathé, M. (2016). An open access repository of images on plant health to enable the development of mobile disease diagnostics. *arXiv*, 13. Recuperado el 14 de Mayo de 2020

Khandelwal, R. (10 de Diciembre de 2019). *Towards data science.* Obtenido de Data Augmentation techniques in python: https://towardsdatascience.com/data-augmentation-techniques-in-python-f216ef5eed69T

Kingma, D. P., & Ba, J. L. (2017). ADAM: a method for stochastic optimization. *arXiv*, 15. Recuperado el 18 de Mayo de 2020, de https://arxiv.org/pdf/1412.6980v9.pdf

Maeda-Gutiérrez, V., Galván-Tejada, C. E., Zanella-Calzada, L. A., Celaya-Padilla, J. M., Galván-Tejada, J. I., Gamboa-Rosales, H., . . . Olvera Olvera, C. A. (2020). Comparison of Convolutional Neural Network. *applied sciences*, 15.

Mamani, M., Villalobos, M., & Herrera, R. (2017). Sistema web de bajo costo para monitorear y controlar un invernadero agrícola. *Ingeniare. Rev. chil. ing. vol.25 no.4 Arica dic. 2017*, 20.

Mohanty, s. (23 de Septiembre de 2018). *Github.* Obtenido de PlantVillage-Dataset: https://github.com/spMohanty/PlantVillage-Dataset

Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016). Using Deep Learning for Image-Based Plant Disease Detection. *Frontiers in Plant Science*, 10.

Montero Rodríguez, J. C., Roshan Biswal, R., & Sánchez de la Cruz, E. (2019). Algoritmos de aprendizaje autom´atico de. *Research in Computing Science Journal*, 14. Recuperado el 13 de Mayo de 2020, de https://www.rcs.cic.ipn.mx/2019_148_7/Algoritmos%20de%20aprendizaje%20automatico%20de%20vanguardia%20para%20el%20diagnostico%20de%20enfermedades.pdf

Pan, S. J., & Yang, Q. (2010). A Survey on Transfer Learning. *IEEE*, 15. Recuperado el 13 de Mayo de 2020, de https://scholar.google.com/scholar_lookup?title=A+survey+on+transfer+learning&author=Pan,+S.J.&author=Yang,+Q.&publication_year=2010&journal=IEEE+Trans.+Knowl.+Data+Eng.&volume=22&pages=1345%E2%80%931359&doi=10.1109/TKDE.2009.191

Pérez González, C. (2016). Detección y seguimiento de objetos por colores en una plataforma raspberry Pi. 80.

Sahu, A. (5 de 06 de 2019). *quora.* Obtenido de https://www.quora.com/What-is-the-VGG-neural-network

Santos Senra, H. (2017). *Diseño e implementación de un sistema domótico.* Trabajo fin de grado (Tesis), Madrid (España). Recuperado el 13 de Mayo de 2020, de https://e-archivo.uc3m.es/bitstream/handle/10016/26313/TFG_Hector_Santos_Senra.pdf

Sharma, S., Varinderjit Kaur, E., & Dhillon, N. (2018). Plant disease classification with KNN-SVM Classification. *International Journal of Applied Engineering Research (IJAER), 5*, 7. Recuperado el 12 de Mayo de 2020, de http://www.ijaerd.com/papers/finished_papers/Plant_Disease_Classification_with_KNN-SVM_Classification-IJAERDV050590792.pdf

Simonyan, K., & Zisserman, A. (2014). Vvery deep convolutional networks for large-scale iamge recognition. *arXiv journal*, 14. Obtenido de https://arxiv.org/pdf/1409.1556.pdf%20http://arxiv.org/abs/1409.1556.pdf

Szegedy, C., Vanhoucke, V., Ioffe, S., & Shlens, J. (2016). Rethinking the Inception Architecture for Computer Vision. *CVF-Computer Vision Foundation*, 9. Recuperado el 13 de Mayo de 2020, de http://openaccess.thecvf.com/content_cvpr_2016/papers/Szegedy_Rethinking_the_Inception_CVPR_2016_paper.pdf

Toda, Y., & Okura, F. (2019). How Convolutional Neural Networks Diagnose Plant Disease. *Plant Phenomics*

A Science Partner Journal, 14. Obtenido de https://spj.sciencemag.org/plantphenomics/2019/9237136/

Velasco Zapata, J. A. (2019). *Diseño y desarrollo de un sistema prototipo de diagnóstico de afecciones en plantas.* Pontificia Universidad Javeriana Cali, Departamento de Electrónica y Ciencias de la Computación, Cali. Obtenido de http://vitela.javerianacali.edu.co/bitstream/handle/11522/11943/Diseno_Desarrollo_Sistema.pdf?sequence=1&isAllowed=y

Zwetsloot, R. (Abril de 2019). *The MagPi*. Obtenido de https://magpi.raspberrypi.org/articles/raspberry-pi-4-raspbian-buster