Disease Identification in Crop Plants based on Convolutional Neural Networks

Orlando Iparraguirre-Villanueva[1], Victor Guevara-Ponce[2], Carmen Torres-Ceclén[3], John Ruiz-Alvarado[4], Gloria Castro-Leon[5], Ofelia Roque-Paredes[6], Joselyn Zapata-Paulini[7], Michael Cabanillas-Carbonell[8]
Facultad de Ingeniería y Negocios, Universidad Privada Norbert Wiener, Lima, Perú[1]
Escuela de Posgrado, Universidad Ricardo Palma, Lima, Perú[2, 6]
Facultad de Ingeniería, Universidad Católica los Ángeles de Chimbote, Ancash, Perú[3]
Facultad de Ingeniería, Universidad Tecnológica del Perú, Lima, Perú[4]
Facultad de Ingeniería y Gestión, Universidad Nacional Tecnológica de Lima Sur, Lima, Perú[5]
Escuela de Posgrado, Universidad Continental, Lima, Perú[7]
Facultad de Ingeniería, Universidad Privada del Norte, Lima, Perú[8]

***Abstract*—The identification, classification and treatment of crop plant diseases are essential for agricultural production. Some of the most common diseases include root rot, powdery mildew, mosaic, leaf spot and fruit rot. Machine learning (ML) technology and convolutional neural networks (CNN) have proven to be very useful in this field. This work aims to identify and classify diseases in crop plants, from the data set obtained from Plant Village, with images of diseased plant leaves and their corresponding Tags, using CNN with transfer learning. For processing, the dataset composing of more than 87 thousand images, divided into 38 classes and 26 disease types, was used. Three CNN models (DenseNet-201, ResNet-50 and Inception-v3) were used to identify and classify the images. The results showed that the DenseNet-201 and Inception-v3 models achieved an accuracy of 98% in plant disease identification and classification, slightly higher than the ResNet-50 model, which achieved an accuracy of 97%, thus demonstrating an effective and promising approach, being able to learn relevant features from the images and classify them accurately. Overall, ML in conjunction with CNNs proved to be an effective tool for identifying and classifying diseases in crop plants. The CNN models used in this work are a very good choice for this type of tasks, since they proved to have a very high performance in classification tasks. In terms of accuracy, all three models are very accurate in image classification, with an accuracy of over 96% with large data sets.**

***Keywords*—CNN; identification; models; pathogen; plant; classification; machine learning**

I. Introduction

The identification and classification of plant diseases is essential for agriculture, as the presence of pathogens can cause significant damage and reduce production [1]. Some of the most common diseases include: root rot, which is caused by fungi that attack the roots of the plant; powdery mildew, which is caused by fungi that affect the leaves and flowers of the plant; mosaic, which is caused by viruses that affect the growth and appearance of the leaves of the plant; leaf spot, which is caused by bacteria that affect the leaves of the plant; and fruit rot, which is caused by fungi or bacteria that affect the fruit of the plant [2], [3]. It is important to identify the disease as early as possible to take measures to combat it, either by fungicide treatments or traditional measures that help prevent the spread of the disease. Since ancient times, agriculture has been an important source of food [4], hence the importance of identifying diseases and being able to treat them, since diseases are the cause of a great loss of crop production, since about 36% of crops are lost because of not identifying diseases at early stages [5]. Early-stage identification of plant diseases can greatly reduce their considerable impact. This involves the use of computer technology in agriculture to help identify these diseases [6]. ML technology has proven to be useful in plant disease identification. One specific technique that has proven to be effective is transfer learning (TL) supported by CNN [7]. The idea behind TL allows models to leverage what they have previously learned to improve their performance on new tasks [8]. CNNs can be used to identify plant diseases by analyzing plant images. The CNN is trained on a labeled image dataset, where labels correspond to different types of diseases [3], [9]. During the training process, the CNN learns to recognize patterns in the images that are indicative of a particular disease. Once trained, the CNN can be used to classify new plant images as carrying or not carrying a particular disease [10]. This approach can be very effective in identifying plant diseases, as it allows the use of visual information, which can be very useful in identifying certain types of diseases that may not be evident from other types of data [11]. In this work we use the transfer learning models: DenseNet-201, ResNet-50 and Inception-v3. DenseNet-201 is a CNN with 201 layers deep. ResNet-50 is a CNN with 50 layers deep; Inception-v3 is a CNN with 48 layers deep. The models work with their own training architecture; therefore, they deliver different results. The models have been trained with Google's ImageNet dataset. Both models are characterized by training with large volumes of images and achieving a high level of accuracy [12]. This is due to their ability to learn specific image features through convolution and pooling layers [13], [14]. However, the specific performance of a model depends on several factors, such as the quality and quantity of training data, model architecture, and parameter optimization. This work aims to identify and classify crop plant diseases from PlantVillage dataset, diseased plant leaf images and their corresponding labels, using CNN with Transfer Learning.

II. Related Work

Lately, researchers and scientists related to the agricultural industry and computer science have conducted different studies related to plant diseases using ML techniques, specifically with CNN. For example, in [15] they developed a model to detect, quantify the severity and classify plant images in different species. To illustrate this, in [16] they developed a hybrid application with web technology and CNN models to detect plant diseases in real time. The model used in the web application achieved an accuracy performance rate of 0.9935 for which they used 9914 training arguments. Also, in [17], [18] implemented a CNN for automatic identification and classification of weeds using a mixed crop land, this model was trained in five different epochs; 10k, 20k, 100k, 200k and 242k images, with the purpose of automatic identification and classification of diseases. The results achieved for each epoch: in 10k the weed accuracy reached 0.67; in 20k the weed accuracy reached 0.962; in 100k the accuracy reached 0.983; in 200k the accuracy reached 0.95 and in 242k the accuracy improved significantly. Similarly, the authors in [18], [19] proposed a model for classifying disease affected leaves, for which they used images and obtained directly from cotton crop fields. They used the CNN DarkNet-19, which obtained a performance rate in accuracy of 0.91. Plant diseases directly affect food security, thus decreasing production. The identification of these diseases is the first important step for their treatment. That is why, in [20] they developed a mobile application with CNN to diagnose in real time 26 diseases of 14 crop species. It was validated and tested with 87k images, divided into 38 classes. The model achieved a classification accuracy of 0. 957. Similarly, in [21] they proposed a CNN with EfficientNet model to identify and classify images with diseases into categories. The model obtained an optimal performance rate of 0.9972 accuracy. Also, in [22] through the ResNet-50 model they classified diseases, applying CNN techniques for image prediction. In the same line, [23] used computer vision and ML technology to detect diseases in field crops. The proposed model achieved an accuracy of 0.978 in detecting four crop plant diseases. CNNs have contributed significantly to the detection of plant pathogens. As such, in [24] performed a work with CNNs using different optimization algorithms for detection. They used 5571 manually collected images, the model achieved an accuracy performance of 0.9259. Image recognition through ML is an active area by scientists and researchers. For example, in [25] they proposed a semi-automatic algorithm to detect diseases in crop plants, making use of CNNs. They used the KijaniNet model, who achieved an accuracy performance of 0.8448 and 0.6257.

III. Methodology

This section presents the theoretical basis of the DenseNet-201, ResNet-50 and Inception-v3 models and the development process to identify and classify crop plant diseases. For this purpose, initially the image dataset of infected and healthy plants, collected from PlantVillage, is loaded. This is followed by data cleaning, exploratory analysis and then training and validation of the model using the CNN architecture, which is particularly well suited for image recognition tasks.

A. CNN Architecture

The architecture of a CNN consists of several layers, including input layers, convolution layers, pooling layers, fully connected layers, and an output layer. For this case, the input layer is where the input image is provided to the CNN. Convolution layers perform filtering operations on the input data, using filters that are slid over the image to detect specific patterns. Pooling layers reduce the dimensionality of the data by grouping values from a subregion of the image and replacing them with a single value, such as the maximum or average value. The fully connected layers are used to classify the patterns detected in the previous layers. The output layer provides the final predictions of the CNN, as shown in Fig. 1. In general, CNNs are used for computer vision tasks.

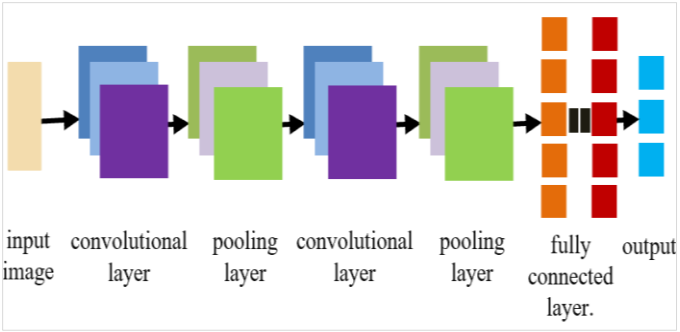

Fig. 1. CNN architecture.

B. Convolutional Layer

A convolution layer is a layer in a CNN that performs filtering operations on the input images. The layer is composed of a set of filters, each of which is slid over the input image and applied to a subregion of the input image [26]. Each filter looks for specific patterns in the image, such as edges or textures [27]. A filter is a small matrix of numbers that is multiplied with the image pixels in a specific window, and the products are summed. In general, convolution layers are used to detect patterns in the input data and to extract relevant features from these data.

C. Activation Functions

In CNNs, the activation function is a mathematical equation that is defined at the output layers of a neuron before being sent to the next layers of the network. The activation function aims to regulate the output of a neuron, allowing the network to learn more complex patterns [28] [29]. Some commonly used activation functions are: Sigmoid, ReLU, tanh and Softmax. Each activation function has its own characteristics and advantages, and the choice of the appropriate activation function depends on the specific problem and data set.

D. Pooling Layer

A network layer in a CNN architecture that is used to group a set of inputs into similar groups. This is achieved through the use of clustering algorithms. Pooling layer is commonly used in unsupervised learning tasks [30], such as image segmentation.

E. Fully Connected Layer

It is known as the densely connected layer. In this layer, all neurons are connected to all neurons in the previous layer, and this layer is part of a neural network layer [31]. In other words, each neuron in a fully connected layer receives an input from each neuron in the previous layer and produces an output that is passed to each neuron in the next layer. Fully connected layers are commonly used in neural networks for tasks such as image classification and natural language processing.

F. Reducing Overfitting

The overfitting technique was applied to reduce the complexity of the model and to regulate and thus prevent the model weights from becoming too large. For this case dropout was used in the neural network layers [32]. This is for the purpose of optimizing performance since overfitting is a common problem in ML in which a model is overfitted to the training data and, as a result, may exhibit performance deficiency.

G. Transfer Learning

TL is a process in which knowledge acquired in one task is used to improve performance in a different task. It is a form of ML in which the model reuses what it has learned to improve its performance rate on another task [33]. TL is especially useful in ML as it allows models to leverage what they have previously learned to improve their performance on new tasks; the process of TL is seen in Fig. 2.

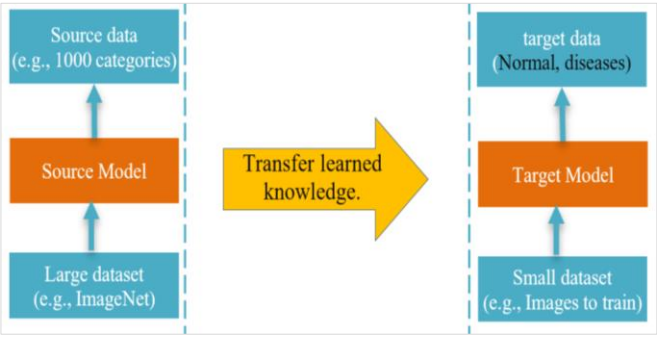

Fig. 2. Transfer learning process.

H. Model architecture

Three models were trained with images with and without disease. Each of the models used in this work is described below.

I. DenseNet-201

DenseNet-201 is characterized by the use of dense blocks, which are layered blocks in which each layer is connected to all other layers in the block [27]. This allows feature reuse and helps mitigate the problem of leakage gradients that can occur in deep neural networks. The number "201" refers to the number of convolutional layers in the network. The model architecture is shown in Fig. 3.

J. ResNet-50

ResNet-50 is a variant of the CNN ResNet. The key innovation of the ResNet-50 is the use of residual connections, which allow the network to create new residual functions for input to the layers, rather than attempting to learn the original unreferenced functions [34], this helps mitigate the problem of gradient loss that can occur in deep neural networks [30] [35]. The main use of this network is in image classification. The number "50" refers to the number of layers as presented in Fig. 4.

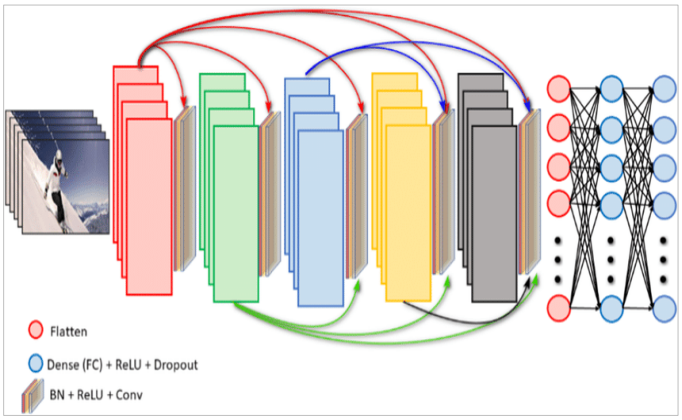

Fig. 3. Architecture of the DenseNet-2021 model.

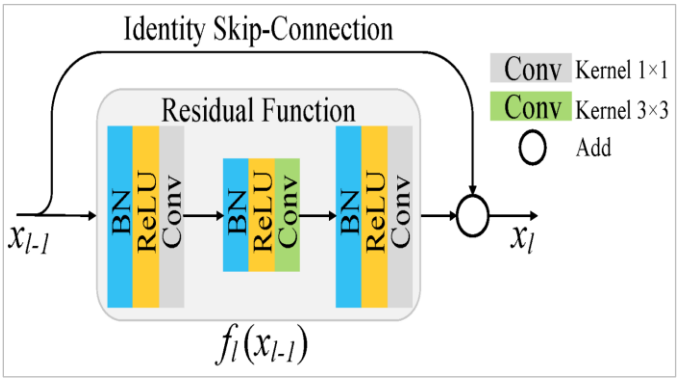

Fig. 4. ResNet-50 architecture.

K. Inception-v3

Inception-v3 is a CNN designed for object recognition in images. It uses a combination of filter layers of different sizes and depths to capture features at different scales and levels of abstraction in the input images [36] [37]. The architecture, as depicted in Fig. 5, includes an "Inception" mechanism that combines multiple filtering operations in a single layer to reduce the number of attributes and improve computational efficiency. Inception-v3 has been trained on a large set of images (ImageNet) and has been widely used in different cases of image classification and identification, object recognition, object detection, etc.

L. Understanding Data

There are a variety of diseases that affect plants, among the most common are: Mildew: caused by fungi that appear as white spots on the leaves; powdery mildew: caused by fungi that appear as a kind of powder on the leaves; Rust and Mosaic: are two diseases that are caused by fungi or insects that cause spots and deformations on the leaves, as shown in Fig. 6. These diseases can be treated with fungicides and pesticides. Also, it is important to take preventive measures

such as good ventilation or good soil drainage. In this work we used an original dataset, obtained from the PlantVillage repository, and this is composed of approximately 87 thousand images of healthy and diseased crop leaves that are organized in 38 different categories. For training and validation the dataset is divided in a proportion of 80% for training and 20% for validation, and then a directory with 33 test images is created for prediction purposes.

The image set is composed of 38 disease types. Each of the diseases associated with the plant types is described below: [Tomato with blight, healthy tomato, healthy grape, orange with virus, healthy soybean, healthy peach, pumpkin with downy mildew, healthy potato, corn with blight leaf, young tomato with blight, tomato with leaf spot, strawberry with frost, apple with scab, tomato with yellow leaf, healthy cherry, healthy blueberry, healthy corn, tomato with bacterial spot, rotting apple, cherry with downy mildew, healthy apple, peach with bacterial spot, royal apple, tomato with objective spot, healthy bell pepper, grape with blight, tomato with mosaic virus, potato with blight, healthy strawberry, black rotten grape, early potato with blight, common corn, beetle grape, healthy raspberry, tomato with mildew, tomato with spider mites, bell pepper with bacterial spot].

From the types of diseases described above, the name of 14 plants is obtained, excluding healthy leaves: [corn with gray leaf spot, corn with rust, corn with fever, healthy potato, rice with leaf spot, potato with fever, healthy corn, potato, healthy rice, healthy rice, rice with hispa, rice with leaf blast, healthy wheat, young potato, red brown wheat and red yellow wheat] and 26 types of images with disease. With this, we proceed to import the dataset; then we process and perform an exploratory analysis, train the DenseNet-201, ResNet-50 and Inception-v3 models and then apply the optimizers.

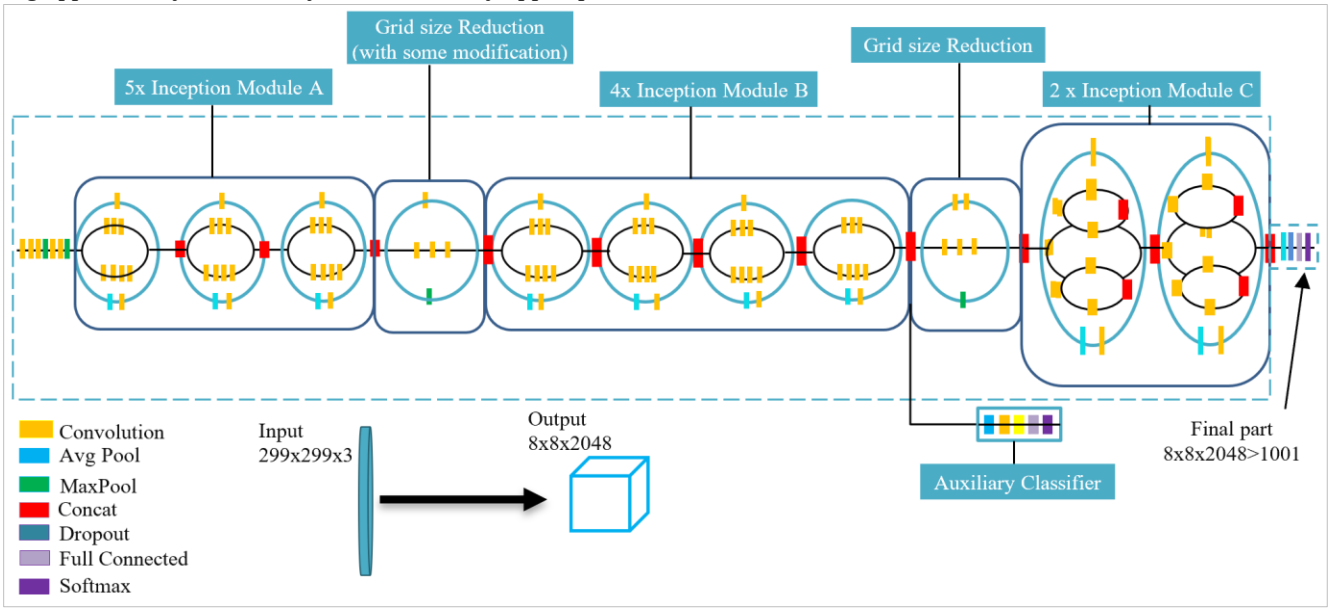

Fig. 5. Architecture of Inception-v3.

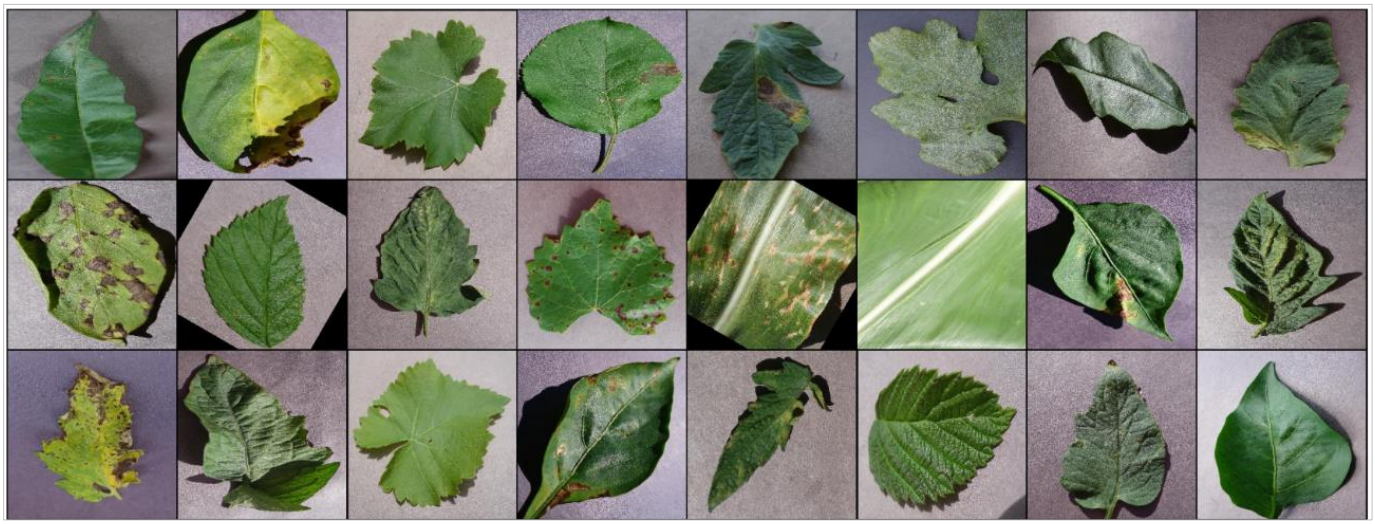

Fig. 6. Images for training.

M. Data Cleaning and Processing

The cleaning and processing process are important tasks in data analysis. Cleaning involves reviewing the data to detect and correct errors, duplicates, and missing values. Data processing involves transforming the data into a more suitable format for analysis, such as grouping, combining, or creating new variables. These tasks are essential to ensure that the results of the analysis are accurate and reliable. For the development of this work we used tools and libraries such as: the Python programming language, libraries such as numpy: for numerical calculations; pandas: to work with data frames; the pytorch module; matplotlib: to process information in graphs and images using tensors; torch.nn: to create neural networks; torch. utils.data: to load data; PIL: to load images; torch.nn.functional: function to calculate loss; torchvision.transforms: to transform images into tensors; torchvision.utils: for data verification; torchvision.datasets: to work with classes and images and torchsummary: to get the model summary.

N. Exploratory Data Analysis

Exploratory data analysis (EDA) is a process of investigating data using statistical tools and visualizations. The main objective of EDA is to understand the structure and distribution of the data, as well as to detect patterns and trends. EDA is an essential step in the data analysis process, as it provides a better understanding of the data before applying statistical and ML techniques. It can include techniques such as histograms, boxplots, scatterplots, among others, to visualize the data and detect patterns and trends. In addition, EDA can also help identify problems in the data, such as outliers or missing values, and make decisions on how to clean and process them before continuing with the analysis. Table I and Fig. 7 show the number of images for each disease.

The next step is to prepare the data for model training, this step is critical to ensure that the model performs optimally. This may include tasks such as normalization, scaling, transformation, and splitting into training and test sets. It is also very important to ensure that the data is in the correct format and is representative of the production data to avoid problems such as overfitting. It is at this stage that libraries become relevant, for example, torchvision.datasets(), is used to load the image dataset. After loading the data, the pixel values of each image (0-255) are transformed to 0 - 1, since CNNs understand much better when the data is normalized. The pixel matrix is converted to a torchTensor and then divided by 255. For example, the image can have the following form (3,256,256), where 3 is the number of channels (RGB) and 256x256 is the width and height of the image. Also, we have the batch_size library (), it allows to count the total number of images given in the CNN input. For example, if we have 1500 image samples for training and we need to set up a batch size of 150 samples from the training dataset, then what the network does is to randomly take the 150 samples and retrains the network, this process continues until all the samples in the network are propagated. Similarly, dataLoader (), the Liberian dataLoader that comes with PyTorch load data in parallel from the dataset. It also provides a convenient way to iterate over the dataset, in small batches, with the ability to shuffle the data before each epoch with the option to use multiple threads for data loading. This library is also accompanied by the num_workers function, which allows to calculate the number of processes that generate batches in parallel. Another very important point is in the configuration, the variable 'shuffle=true', this is very important so that the batches between epochs do not resemble each other.

TABLE I. NUMBER OF IMAGES PER DISEASE

Disease	No. of images
tomato blight	1851
healthy tomato	1926
healthy grapes	1692
orange greening	2010
terveellistä soijaa	2022
Squash mildew	1736
Potato healthy	1824
Northern Corn	1908
Tomato early	1920
Tomato with spots	1745
corn with leaf spot on leaves	1642
strawberry leaf with virus	1774
Peach healthy	1728
apple with scab on leaves	2016
Tomato with yellow leaves due to virus	1961
tomato with bacterial spot	1702
black apple	1987
Blueberry healthy	1816
Moldy cherry	1683
Peach with bacteria	1838
Apple Cedar	1760
tomato target point	1827
Healthy peppers	1988
grapes with leaf spot	1722
potato with Tizon	1939
tomato with virus	1790
Healthy strawberry	1824
Apple healthy	2008
rotten grapes	1888
young pope	1939
Cherry healthy	1826
Corn Common	1907
Grape black	1920
Healthy raspberry	1781
Tomato with mushroom	1882
Tomato with mites	1741
Bell pepper with bacterial spots	1913
Young corn	1859

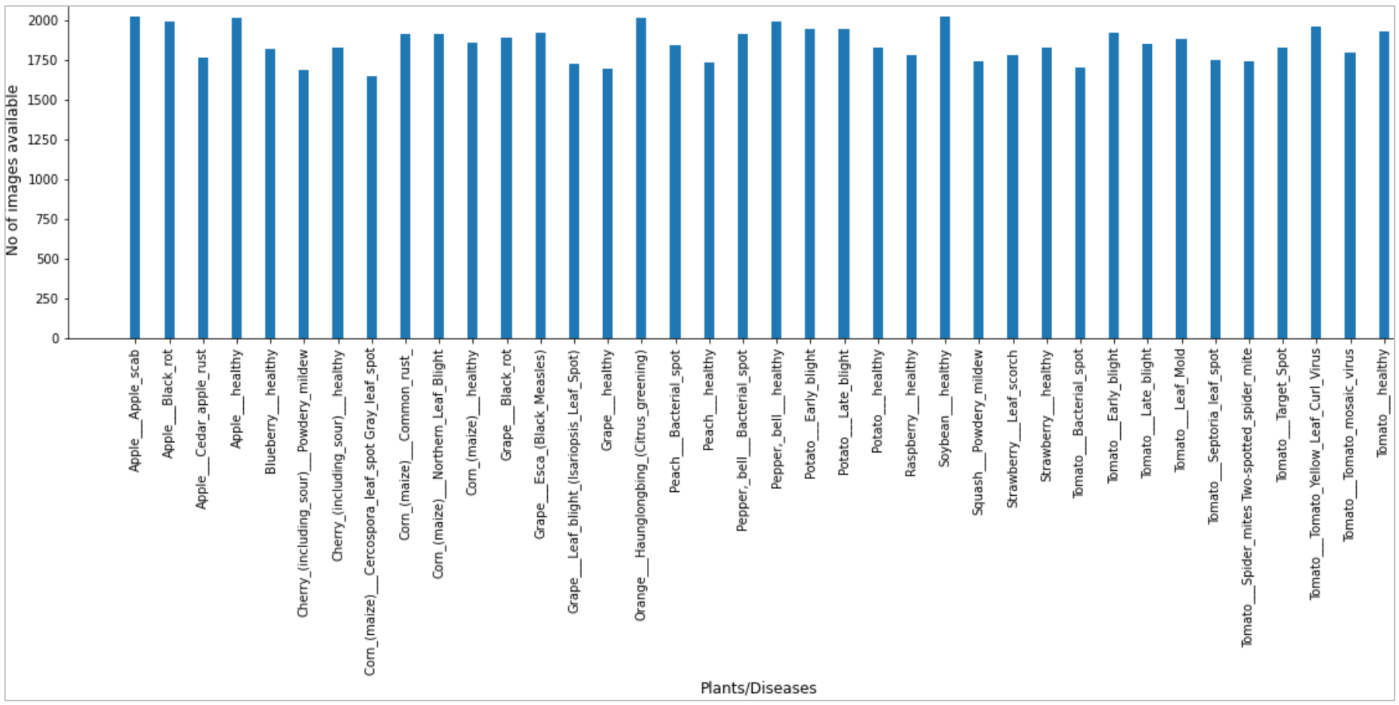

Fig. 7. Images for each type of plant disease.

O. Training and Testing of the Model

For the model training process, we worked with the training data set, in this case 80% of the images in the data set. During model training, the internal parameters were adjusted to minimize the loss function between model predictions. The training ends when it reaches a certain level of accuracy or when it reaches the maximum number of iterations. Meanwhile, to test the models, we worked with the difference of the training data set (20%). This stage is very important, since the aim is to evaluate the model's capability and to measure its accuracy. For which the following metrics were used, such as accuracy, recall Score-F1 and the performance curve (ROC). It is important to point out that both the training and testing processes are performed on different datasets, and another dataset was also used for validation. For training we chose to use GPUs instead of CPUs, considering that the volume of data is large, and a CPU is not able to respond to the demands, and GPUs are optimized to perform such tasks, as they can process multiple calculations simultaneously. In addition, they have a large number of cores to perform calculations and handle large amounts of data, which makes the memory bandwidth of a GPU the most suitable. Also, functions that help with the training are used, such as: Training_step(): this was used to find out how erroneous the model turned out to be after training. This function is not only an accuracy metric, but is necessary for the model to improve during training; Validation-setp(): this function is used to measure the accuracy across the threshold and is counted if the difference between the model prediction and the actual label is less than the threshold; Validation_epoch_end(): is used to track losses in validation accuracies and training losses after each epoch. However, it must be ensured not to be tracking the gradient; Epoch_end(): Used to print the validation accuracy losses and training losses and the learning rate after each epoch. We also use the accuracy function to calculate the overall accuracy of the models in the batch of the results. For example, in Fig. 8, we can see the training summary of the ResNet-50 model using Keras.

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1         [-1, 64, 256, 256]           1,792
       BatchNorm2d-2         [-1, 64, 256, 256]             128
              ReLU-3         [-1, 64, 256, 256]               0
            Conv2d-4        [-1, 128, 256, 256]          73,856
       BatchNorm2d-5        [-1, 128, 256, 256]             256
              ReLU-6        [-1, 128, 256, 256]               0
         MaxPool2d-7          [-1, 128, 64, 64]               0
            Conv2d-8          [-1, 128, 64, 64]         147,584
       BatchNorm2d-9          [-1, 128, 64, 64]             256
             ReLU-10          [-1, 128, 64, 64]               0
           Conv2d-11          [-1, 128, 64, 64]         147,584
      BatchNorm2d-12          [-1, 128, 64, 64]             256
             ReLU-13          [-1, 128, 64, 64]               0
           Conv2d-14          [-1, 256, 64, 64]         295,168
      BatchNorm2d-15          [-1, 256, 64, 64]             512
             ReLU-16          [-1, 256, 64, 64]               0
         MaxPool2d-17          [-1, 256, 16, 16]               0
           Conv2d-18          [-1, 512, 16, 16]       1,180,160
      BatchNorm2d-19          [-1, 512, 16, 16]           1,024
             ReLU-20          [-1, 512, 16, 16]               0
         MaxPool2d-21            [-1, 512, 4, 4]               0
           Conv2d-22            [-1, 512, 4, 4]       2,359,808
      BatchNorm2d-23            [-1, 512, 4, 4]           1,024
             ReLU-24            [-1, 512, 4, 4]               0
           Conv2d-25            [-1, 512, 4, 4]       2,359,808
      BatchNorm2d-26            [-1, 512, 4, 4]           1,024
             ReLU-27            [-1, 512, 4, 4]               0
         MaxPool2d-28            [-1, 512, 1, 1]               0
          Flatten-29                  [-1, 512]               0
           Linear-30                   [-1, 38]          19,494
================================================================
Total params: 6,589,734
Trainable params: 6,589,734
Non-trainable params: 0
```

Fig. 8. ResNet-50 model training summary.

In the training process, some functions that are very useful should be taken into account, such as evaluate (), a function used for the validation process, and the function fit_one_cycle(), a function that performs the entire training process. Some techniques were also used such as: 1) Learning Rate Scheduling: allows using a learning rate after each training, instead of using a fixed rate; 2) Weight decay: regulation technique that prevents the weights from becoming too large to add a term to the loss function; 3) Gradient Clipping: this technique allows limiting the values of the decays to a small range to avoid undesired changes in the parameters due to large decay values, it is a simple but very effective technique. After training the models, the confusion matrix of each model is constructed, as in the case of the ResNet-50 model shown in Fig. 9. The confusion matrix is very important, since it helps to understand and compare the predicted data with the real data; however, for a better understanding, four basic concepts must be known: true and false positives and true and false negatives. That is, true positives and true negatives are simply the hits, while false positives and false negatives are the misses. This tool is important since it allows visualizing the performance of each of the models. Each column represents the number of predictions of each class, while the rows represent the actual values. Finally, the hits and misses are placed in each of the cells, Fig. 10, we can see a confusion matrix of 15 x 15, because we have 15 classes of plants. In the main diagonal we have the hits (true positives and true negatives) and in the remaining cells the number of misses (false positives and false negatives). Now, with the confusion matrix constructed, the performance of each model can be seen in detail, such is the case of the resNet-50 model in Fig. 9. This process is repeated for the other models (DenseNet-201 and Inception-v3).

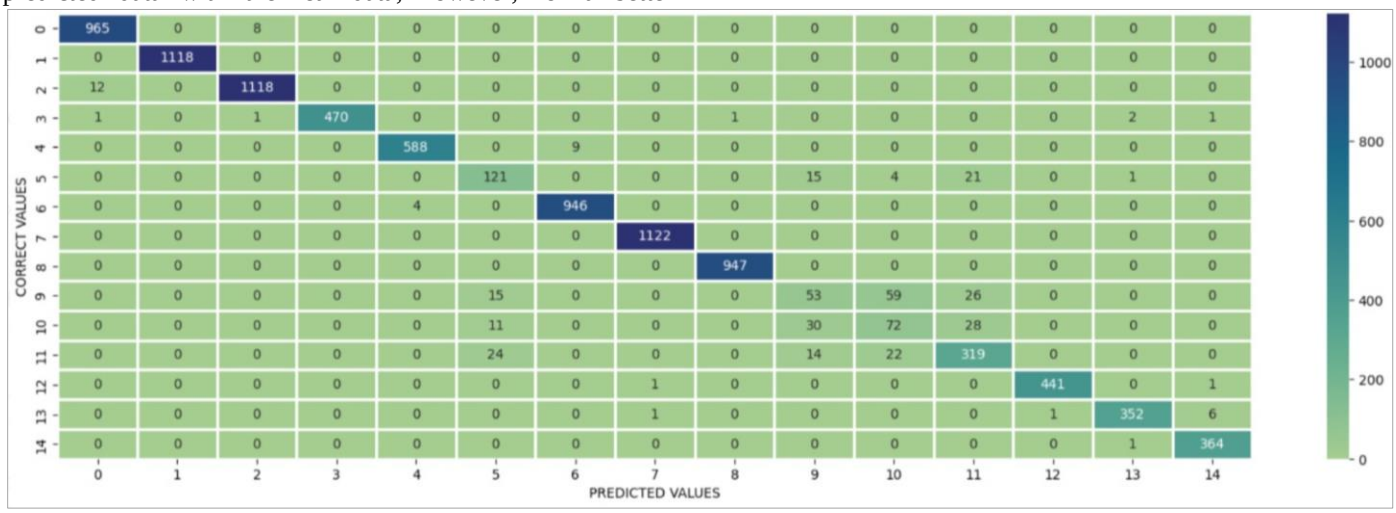

Fig. 9. Confusion matrix of the ResNet-50 model.

IV. Results and Discussion

This section presents the training results of the DenseNet-201, Inception-v3 and ResNet-50 models. For which an original dataset, obtained from the PlantVillage repository, was used, and is composed of approximately 87 thousand images of healthy and diseased crop leaves. The performance of each of the models is subject to the following metrics such as accuracy, precision, recall, F1 score and ROC curve as presented in Table II. The results of the DenseNet-201, Inception-v3 and ResNet-50 models vary depending on the dataset and the specific tasks they perform, as shown in Table II.

Table II shows that the DenseNet-201 and Inception-v3 models achieved better performance in image classification, showing slightly better accuracy than ResNet-50. In terms of accuracy, DenseNet-201, has been characterized as one of the best models for image classification, and it is thanks to its density block design that allows greater propagation of information between layers and better feature extraction. However, the DenseNeet-201 model is usually slower due to its larger number of parameters. On the contrary, the ResNte-50 and Inception-v3 models are faster due to their efficient design. A very important point to take into account is the available computational power and memory limitations. Fig. 10, Fig. 11 and Fig. 12 show the performance of the models. For example, in Fig. 10, graph (Results Loss), shows the performance of the loss and accuracy of the DenseNet-201 model, the blue line represents the training data and the red line the test, as can be seen in the first two epochs the error decreases very significantly, then the error decreases as it goes through the epochs or iterations, and in turn makes the comparison with the test data, however, as it progresses the two lines are separated, this means that it is entering an over fit, ideally the two lines should be joined. With respect to the graph (Results Accuracy) in Fig. 10, which is the result of the Accuracy, it is also evident that from the second epoch there is a very significant increase of hits, and from the third epoch onwards it has been improving its accuracy, however, the red line as the epochs progress is separating from the blue line, which is what should not happen, because it means that the test data are separated from the training. Similarly, it can be seen in Fig. 11, graph (Results Loss), shows the loss and accuracy performance of the ResNet-50 model, the blue line represents the training data and the red line the test, as can be seen in the first two epochs the error decreases very significantly, and progressively decreases the error as it advances the iterations in the epochs, and in turn makes the comparison with the test data, however, as it advances the iterations the two lines are

separated, this means that it is entering in an over adjustment, the ideal is that the two lines are joined. With respect to the graph (Results Accuracy) of Fig. 11, it is the result of the Accuracy, it is also evident that from the first epoch there is a very significant increase of hits, and from the second epoch it has been improving its accuracy, however, the red line as it advances the iterations of the epochs is separating from the blue line, it is what should not happen, because it means that the test data is separated from the training. Similarly in Fig. 12, graph (Results Loss), the performance of the loss and accuracy of the Inception-v3 model is presented, where the results are very similar to the Densenet-201 and ResNet-50 models. And with respect to the graph (Results Accuracy) of Fig. 12, is the result of the Accuracy, its performance of this model is very similar to the Densenet-201 and ResNet-50 models, these similarities are seen in Fig.12 graph (Results Accuracy).

TABLE II. MODEL EVALUATION RESULTS

DenseNet-201				
	accuracy [%]	**recall [%]**	**f1-score [%]**	**support**
accuracy			0.98	9316
macro avg	0.93	0.93	0.93	9316
weighted avg	0.98	0.98	0.98	9316

ResNet-50				
	accuracy [%]	**recall [%]**	**f1-score [%]**	**support**
accuracy			0.97	9316
macro avg	0.89	0.89	0.89	9316
weighted avg	0.96	0.97	0.97	9316

Inception-v3				
	accuracy [%]	**recall [%]**	**f1-score [%]**	**support**
accuracy			0.98	9316
macro avg	0.93	0.93	0.93	9316
weighted avg	0.98	0.98	0.98	9316

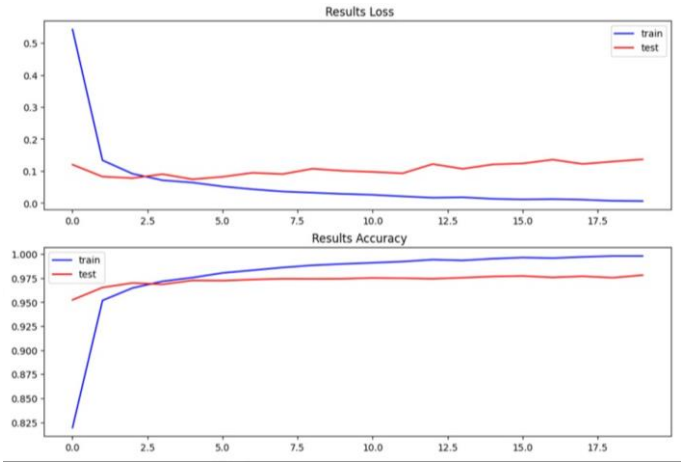

Fig. 10. Comparison of results of the denseNet-201 model between loss and accuracy.

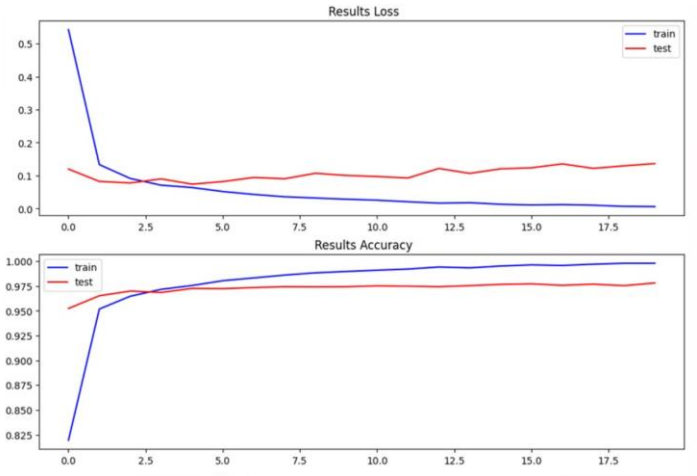

Fig. 11. Comparison of results of the ResNet-50 model between loss and accuracy.

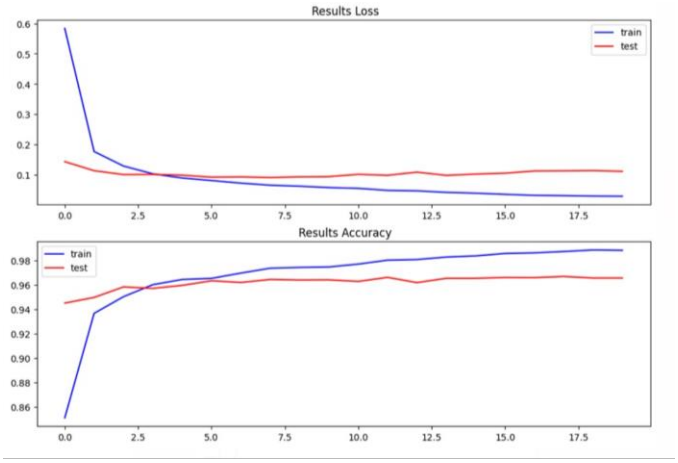

Fig. 12. Comparison of results of the Inception-3 model between loss and accuracy.

The results presented in Table II, Fig. 10, Fig. 11 and Fig. 12, allow an analysis of the performance of each of the trained models (DenseNet-201, ResNet-50 and Inception-v3). In general terms, the performance of the three models has been successful in the identification and classification of diseases in crop plants. For example, the DenseNet-201 model achieved an accuracy of 98%, slightly higher than the performance of the ResNet-50 model. This result is superior to that obtained in [20] where they used this model to classify images, it is important to specify that the accuracy rate will depend on different factors, such as the volume of data with which it was processed, as in [18] where this model achieved an accuracy of 99%, higher than the result obtained in this work. The ResNet-50 model achieved an accuracy level of 97%, slightly lower than the performance of the DenseNet-201 and Inception-v3 models. However, the level obtained does not mean that the model has weaknesses in identifying and classifying images; on the contrary, 97% accuracy is significantly very good. Considering that, for example in [24] this model obtained an accuracy of 92%, it is also considered an optimal rate for its level of accuracy, in [22], [23] used this model for image prediction where it reached an accuracy level of 97.8%, slightly higher than that obtained in this work. This indicates

that the performance of the model varies depending on the volume of the data set and the quality of images, among other factors. With respect to the Inception-v3 model, in this work it reached an accuracy level of 98% as shown in Table II, slightly higher than the ResNet-50 model, the Inception-v3 model is characterized by having multiple filters and kernel sizes in each layer, it is a lighter model than DenseNet-201 and ResNet-50, it has fewer parameters, which makes it ideal for identifying and classifying diseases in planes through artificial vision. The level of accuracy obtained is considered optimal, considering that it is close to 100%, this model also obtained a good performance in [16], where it reached an accuracy of 99% in the classification of images of different species, slightly higher than that obtained in this work. Similarly, in [17] they used this model to identify and classify weeds using mixed cropland, where it reached an accuracy level of 92%, a figure much lower than that obtained in this work. However, this does not mean that the model does not perform well in these tasks; on the contrary, the accuracy will depend on the volume of data and the quality of images with which it is processed. Finally, the three models used in this work have achieved optimal performance and accuracy in the process of identification and classification of diseases in crop plants. It is important to point out that the DenseNet-201 and Inception-v3 models obtained the best results in accuracy with 98%, so they would be a viable alternative in technological terms for the identification of diseases in crop plants.

V. CONCLUSION

The three CNN models (DenseNet-201, ResNet-50 and Inception-v3) used in this work have demonstrated an effective and promising approach, being able to learn relevant features from the images and classify them accurately. A dataset consisting of more than 87 thousand images of healthy and diseased crop leaves, categorized into 38 different categories, was used. For the purpose of identifying and classifying crop plant diseases, using CNN with Transfer Learning. The performance of each of the models was analyzed, as shown in Table II, Fig. 10, Fig. 11 and Fig. 12. The models used are characterized by excellent performance in image identification and classification. For example. The DenseNet-201 and Inception-v3 models achieved an accuracy of 98% in the identification and classification of plant diseases, slightly higher than the ResNet-50 model, but this does not mean that the ResNet-50 model does not have an optimal performance, on the contrary, it achieved 97% accuracy. Likewise, Fig. 11 (Results Loss) shows the training results of the DenseNet-50 model, where it is evident that the blue line from the second iteration drastically reduces the error and progressively decreases as the iterations advance in the epochs. However, at the end it separates from the red line, this is not good for the prediction, because it is an indication of an over adjustment to a greater number of iterations. In the case of Results Accuracy, it is also evident that from the second epoch onwards there is a very significant increase in accuracy, and from the third epoch onwards its accuracy has been improving. The ResNet-50 model in Fig. 12 and the Inception-v3 model in Fig. 12 have a very similar behavior to the DenseNet-201 model.

Finally, the DenseNet-201 and Inception-v3 models achieved the best results in the identification and classification of diseases in crop plants; therefore, they are the two models that are recommended for implementation for this type of task. In the future, a possible complement to this work would include the development of a mobile application so that the implemented model can be consumed.

REFERENCES

[1] N. Bevers, E. J. Sikora, and N. B. Hardy, "Soybean disease identification using original field images and transfer learning with convolutional neural networks," Comput Electron Agric, vol. 203, p. 107449, Dec. 2022, doi: 10.1016/J.COMPAG.2022.107449.

[2] B. Dey, M. Masum Ul Haque, R. Khatun, and R. Ahmed, "Comparative performance of four CNN-based deep learning variants in detecting Hispa pest, two fungal diseases, and NPK deficiency symptoms of rice (Oryza sativa)," Comput Electron Agric, vol. 202, p. 107340, Nov. 2022, doi: 10.1016/J.COMPAG.2022.107340.

[3] S. Janarthan, S. Thuseethan, S. Rajasegarar, and J. Yearwood, "P2OP—Plant Pathology on Palms: A deep learning-based mobile solution for in-field plant disease detection," Comput Electron Agric, vol. 202, p. 107371, Nov. 2022, doi: 10.1016/J.COMPAG.2022.107371.

[4] A. Hussain and P. Balaji Srikaanth, "Disease Classification and Detection Techniques in Rice Plant using Deep Learning," 8th International Conference on Smart Structures and Systems, ICSSS 2022, 2022, doi: 10.1109/ICSSS54381.2022.9782162.

[5] V. K. Shrivastava and M. K. Pradhan, "Rice plant disease classification using color features: a machine learning paradigm," Journal of Plant Pathology, vol. 103, no. 1, pp. 17–26, Feb. 2021, doi: 10.1007/S42161-020-00683-3/METRICS.

[6] Y. Lu, S. Yi, N. Zeng, Y. Liu, and Y. Zhang, "Identification of rice diseases using deep convolutional neural networks," Neurocomputing, vol. 267, pp. 378–384, Dec. 2017, doi: 10.1016/J.NEUCOM.2017.06.023.

[7] Rukhsar and S. K. Upadhyay, "Deep Transfer Learning-Based Rice Leaves Disease Diagnosis and Classification model using InceptionV3," Proceedings of International Conference on Computational Intelligence and Sustainable Engineering Solution, CISES 2022, pp. 493–499, 2022, doi: 10.1109/CISES54857.2022.9844374.

[8] D. Xiao et al., "Citrus greening disease recognition algorithm based on classification network using TRL-GAN," Comput Electron Agric, vol. 200, p. 107206, Sep. 2022, doi: 10.1016/J.COMPAG.2022.107206.

[9] H. Yu, J. Liu, C. Chen, A. A. Heidari, Q. Zhang, and H. Chen, "Optimized deep residual network system for diagnosing tomato pests," Comput Electron Agric, vol. 195, p. 106805, Apr. 2022, doi: 10.1016/J.COMPAG.2022.106805.

[10] S. Kendler et al., "Detection of crop diseases using enhanced variability imagery data and convolutional neural networks," Comput Electron Agric, vol. 193, p. 106732, Feb. 2022, doi: 10.1016/J.COMPAG.2022.106732.

[11] K. Paul et al., "Viable smart sensors and their application in data driven agriculture," Comput Electron Agric, vol. 198, p. 107096, Jul. 2022, doi: 10.1016/J.COMPAG.2022.107096.

[12] R. Gajjar, N. Gajjar, V. J. Thakor, N. P. Patel, and S. Ruparelia, "Real-time detection and identification of plant leaf diseases using convolutional neural networks on an embedded platform," Visual Computer, vol. 38, no. 8, pp. 2923–2938, Aug. 2022, doi: 10.1007/S00371-021-02164-9/METRICS.

[13] S. B. Jadhav, V. R. Udupi, and S. B. Patil, "Identification of plant diseases using convolutional neural networks," International Journal of Information Technology (Singapore), vol. 13, no. 6, pp. 2461–2470, Dec. 2021, doi: 10.1007/S41870-020-00437-5/METRICS.

[14] G. Yogeswararao, V. Naresh, R. Malmathanraj, and P. Palanisamy, "An efficient densely connected convolutional neural network for identification of plant diseases," Multimed Tools Appl, vol. 81, no. 23, pp. 32791–32816, Sep. 2022, doi: 10.1007/S11042-022-13053-1/METRICS.

[15] J. G. Arnal Barbedo, "Digital image processing techniques for detecting, quantifying and classifying plant diseases," Springerplus, vol. 2, no. 1, pp. 1–12, Dec. 2013, doi: 10.1186/2193-1801-2-660/TABLES/1.

[16] P. Bedi and P. Gole, "Plant disease detection using hybrid model based on convolutional autoencoder and convolutional neural network," Artificial Intelligence in Agriculture, vol. 5, pp. 90–101, Jan. 2021, doi: 10.1016/J.AIIA.2021.05.002.

[17] X. Kang, C. Huang, L. Zhang, M. Yang, Z. Zhang, and X. Lyu, "Assessing the severity of cotton Verticillium wilt disease from in situ canopy images and spectra using convolutional neural networks," Crop J, Dec. 2022, doi: 10.1016/J.CJ.2022.12.002.

[18] O. G. Ajayi and J. Ashi, "Effect of varying training epochs of a Faster Region-Based Convolutional Neural Network on the Accuracy of an Automatic Weed Classification Scheme," Smart Agricultural Technology, vol. 3, p. 100128, Feb. 2023, doi: 10.1016/J.ATECH.2022.100128.

[19] A. S. Paymode and V. B. Malode, "Transfer Learning for Multi-Crop Leaf Disease Image Classification using Convolutional Neural Network VGG," Artificial Intelligence in Agriculture, vol. 6, pp. 23–33, Jan. 2022, doi: 10.1016/J.AIIA.2021.12.002.

[20] S. Shrimali, "PlantifyAI: A Novel Convolutional Neural Network Based Mobile Application for Efficient Crop Disease Detection and Treatment," Procedia Comput Sci, vol. 191, pp. 469–474, Jan. 2021, doi: 10.1016/J.PROCS.2021.07.059.

[21] X. Sun, G. Li, P. Qu, X. Xie, X. Pan, and W. Zhang, "Research on plant disease identification based on CNN," Cognitive Robotics, vol. 2, pp. 155–163, Jan. 2022, doi: 10.1016/J.COGR.2022.07.001.

[22] R. G. Dawod and C. Dobre, "ResNet interpretation methods applied to the classification of foliar diseases in sunflower," J Agric Food Res, vol. 9, p. 100323, Sep. 2022, doi: 10.1016/J.JAFR.2022.100323.

[23] H. Liu and J. S. Chahl, "Proximal detecting invertebrate pests on crops using a deep residual convolutional neural network trained by virtual images," Artificial Intelligence in Agriculture, vol. 5, pp. 13–23, Jan. 2021, doi: 10.1016/J.AIIA.2021.01.003.

[24] E. M. Raouhi, M. Lachgar, H. Hrimech, and A. Kartit, "Optimization techniques in deep convolutional neuronal networks applied to olive diseases classification," Artificial Intelligence in Agriculture, vol. 6, pp. 77–89, Jan. 2022, doi: 10.1016/J.AIIA.2022.06.001.

[25] L. C. Ngugi, M. Abdelwahab, and M. Abo-Zahhad, "A new approach to learning and recognizing leaf diseases from individual lesions using convolutional neural networks," Information Processing in Agriculture, Oct. 2021, doi: 10.1016/J.INPA.2021.10.004.

[26] M. A. Murti, C. Setianingsih, E. Kusumawardhani, and R. Farhan, "Cedarwood Quality Classification using SVM Classifier and Convolutional Neural Network (CNN)," International Journal of Advanced Computer Science and Applications, vol. 13, no. 11, pp. 101–111, 2022, doi: 10.14569/IJACSA.2022.0131111.

[27] F. B. Mofrad and G. Valizadeh, "DenseNet-based Transfer Learning for LV Shape Classification: Introducing a Novel Information Fusion and Data Augmentation using Statistical Shape/Color Modeling," Expert Syst Appl, p. 119261, Mar. 2022, doi: 10.1016/J.ESWA.2022.119261.

[28] A.-A. Nayan et al., "A deep learning approach for brain tumor detection using magnetic resonance imaging," International Journal of Electrical and Computer Engineering (IJECE), vol. 13, no. 1, p. 1039, Feb. 2023, doi: 10.11591/IJECE.V13I1.PP1039-1047.

[29] N. Razmjooy, S. Razmjooy, Z. Vahedi, V. V. Estrela, and G. G. de Oliveira, "Skin Color Segmentation Based on Artificial Neural Network Improved by a Modified Grasshopper Optimization Algorithm," Lecture Notes in Electrical Engineering, vol. 696, pp. 169–185, 2021, doi: 10.1007/978-3-030-56689-0_9/COVER.

[30] L. Nahhas, M. Albahar, A. Alammari, and A. Jurcut, "Android Malware Detection Using ResNet-50 Stacking," Computers, Materials & Continua, vol. 74, no. 2, pp. 3997–4014, 2023, doi: 10.32604/CMC.2023.028316.

[31] T. Zheng, Q. Wang, Y. Shen, and X. Lin, "Gradient rectified parameter unit of the fully connected layer in convolutional neural networks," Knowl Based Syst, vol. 248, Jul. 2022, doi: 10.1016/J.KNOSYS.2022.108797.

[32] F. Martínez, H. Montiel, and F. Martínez, "A Machine Learning Model for the Diagnosis of Coffee Diseases," International Journal of Advanced Computer Science and Applications, vol. 13, no. 4, pp. 968–974, 2022, doi: 10.14569/IJACSA.2022.01304110.

[33] O. Iparraguirre-Villanueva et al., "Text prediction recurrent neural networks using long short-term memory-dropout," Indonesian Journal of Electrical Engineering and Computer Science, vol. 29, no. 3, pp. 1758–1768, Mar. 2023, doi: 10.11591/IJEECS.V29.I3.PP1758-1768.

[34] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition", Accessed: Jan. 17, 2023. [Online]. Available: http://image-net.org/challenges/LSVRC/2015/.

[35] X. Cai, X. Li, N. Razmjooy, and N. Ghadimi, "Breast Cancer Diagnosis by Convolutional Neural Network and Advanced Thermal Exchange Optimization Algorithm," Comput Math Methods Med, vol. 2021, 2021, doi: 10.1155/2021/5595180.

[36] S. Faizal, C. A. Rajput, R. Tripathi, B. Verma, M. R. Prusty, and S. S. Korade, "Automated cataract disease detection on anterior segment eye images using adaptive thresholding and fine tuned inception-v3 model," Biomed Signal Process Control, vol. 82, Apr. 2023, doi: 10.1016/J.BSPC.2022.104550.

[37] Z. Guo, L. Xu, Y. Si, and N. Razmjooy, "Novel computer-aided lung cancer detection based on convolutional neural network-based and feature-based classifiers using metaheuristics," Int J Imaging Syst Technol, vol. 31, no. 4, pp. 1954–1969, Dec. 2021, doi: 10.1002/IMA.22608.