Online-Journals.org

iJOE International Journal of Online and Biomedical Engineering

iJOE | eISSN: 2626-8493 | Vol. 19 No. 12 (2023) | OPEN ACCESS

https://doi.org/10.3991/ijoe.v19i12.40329

PAPER

A Model Proposal for Enhancing Leaf Disease Detection Using Convolutional Neural Networks (CNN): Case Study

Moulay Hafid Aabidi[1](✉), Adil EL Makrani[2], Brahim Jabir[3], Imane Zaimi[1]

[1]Multidisciplinary Research Laboratory for Science, Technology and Society, Higher School of Technology, Sultan Moulay Slimane University, Khenifra, Morocco

[2]Computer Science Research Laboratory, Faculty of Science, Ibn Tofail University, Kenitra, Morocco

[3]LIMATI Laboratory, Sultan Moulay Slimane University, Beni Mellal, Morocco

myhafidaabidi@yahoo.fr

ABSTRACT

Deep learning has gained significant popularity due to its exceptional performance in various machine learning and artificial intelligence applications. In this paper, we propose a comprehensive methodology for enhancing leaf disease detection using Convolutional Neural Networks (CNNs). Our approach leverages the power of CNNs and introduces innovative techniques to improve accuracy and provide insights into the inner workings of the models. The methodology encompasses multiple stages. We describe the methodology as follows: Firstly, we employ advanced preprocessing techniques to enhance the leaf image dataset, including data augmentation methods to augment the training data and improve model accuracy. Secondly, we design and implement a robust Convolutional Neural Network architecture with multiple layers and ReLU activation, enabling the network to effectively learn complex patterns and features from the input images. To facilitate monitoring and control of the CNN processes, we introduce a novel network visualization module. This module offers a filter-level 2D embedding view, providing real-time insights into the inner workings of the network and aiding in the interpretation of the learned features. Additionally, we develop an interactive module that enables real-time model control, allowing researchers and practitioners to fine-tune the model parameters and optimize its performance. To evaluate the effectiveness of our proposed methodology, we conduct extensive experiments using the PlantVillage dataset, which contains a diverse range of plant diseases captured through a large number of leaf images. Through rigorous analysis and evaluation, we demonstrate the superior performance of our approach, achieving classification accuracy exceeding 99%.

KEYWORDS

deep learning, CNN models, computer vision, VGG, leaf disease detection

1 INTRODUCTION

Precision agriculture is an innovative farming technique that uses technology to optimize crop yields while minimizing waste and reducing environmental impact. Deep learning, a subset of machine learning, has shown great potential in enhancing precision agriculture by providing a more accurate analysis of the complex

Aabidi, M.H., Makrani, A.E., Jabir, B., Zaimi, I. (2023). A Model Proposal for Enhancing Leaf Disease Detection Using Convolutional Neural Networks (CNN): Case Study. *International Journal of Online and Biomedical Engineering (iJOE)*, 19(12), pp. 127–143. https://doi.org/10.3991/ijoe.v19i12.40329

Article submitted 2023-04-10. Resubmitted 2023-06-07. Final acceptance 2023-06-09. Final version published as submitted by the authors.

relationships between crop growth, weather conditions, and soil health [1]. By leveraging deep learning algorithms, farmers can collect and process vast amounts of data from various sources, such as satellite imagery, sensors, and drones. This data can then be used to develop models that are more appropriate for accurate and predictive analytics, enabling farmers to make more informed decisions about crop management, irrigation, fertilization, and pest control. Conventional neural networks consist of several layers, each consisting of multiple nodes that compute a linear combination of the nodes from the previous layer and are subsequently subjected to a non-linear transformation such as Sigmoid, Tanh, or Softmax function [2]. However, challenges in training, such as long computation time, sensitivity to initialization and hyperparameters, have limited the use of neural networks. Techniques like dropout, batch normalization, and alternative nonlinear functions like rectified linear unit have been proposed to overcome these challenges. Convolutional neural networks (CNNs), in particular, have gained significant attention due to their exceptional performance in image classification tasks. While initially proposed in the early 1990s, CNNs were not widely utilized until 2012 [3], [4], when Krizhevsky et al. [5] achieved remarkable results on image classification tasks using a deep architecture model. Despite the impressive progress in CNN performance, little attention has been paid to comprehending the underlying processes in these models. This lack of clarity has resulted in a need for tools and techniques to explore and understand their inner workings. However, deep learning structures are intricate and difficult to comprehend, making the process of selecting a suitable model and determining appropriate hyperparameter values non-intuitive. Moreover, the training process requires an extensive amount of time, making it quite disconnected. In this article, we build upon the existing research by presenting our approach to leaf disease detection using CNNs. We extend the methodology by incorporating novel preprocessing techniques, optimizing the model architecture, and utilizing advanced data augmentation strategies. Through comprehensive experiments and analysis, we demonstrate the effectiveness and robustness of our proposed approach. The results obtained contribute to the growing body of knowledge in precision agriculture and provide valuable insights for practical implementation. In the following sections, we will present the related work and discuss the dataset used, the image preprocessing techniques applied, the details of our model architecture, and the relevant terminologies in the Methods and Tools section. Subsequently, we present our proposed approach, including the specific modifications and enhancements implemented. The Results and Discussion section provides a comprehensive analysis of our experimental findings and a comparison with existing approaches. Finally, we conclude the article by summarizing our contributions and highlighting potential avenues for future research in the field of leaf disease detection using deep learning techniques.

2 RELATED WORKS

In recent years, there has been significant research conducted on the topic of leaf disease detection using Convolutional Neural Networks (CNNs). Numerous studies have proposed methodologies and techniques to enhance the accuracy and efficiency of disease detection in plants. This section provides a summary of the related works in this field.

One notable study by Prasanna et al. [6] focused on detecting plant leaf diseases using a Deep Learning approach. They developed a system that employed CNNs for disease classification and remedy recommendation based on diseased leaf images from the Plant Village Dataset. Their CNN model achieved an impressive accuracy of 96% after training for 8 epochs using TensorFlow, demonstrating the effectiveness of CNNs in accurately classifying plant diseases. Another study by Andrew et al. [7] explored the use of pre-trained CNN models, such as DenseNet-121, ResNet-50, VGG-16, and Inception V4, for plant disease identification. They fine-tuned the hyperparameters of these models and focused on efficient disease recognition. Through experiments conducted on the PlantVillage dataset, which comprised a large number of images of different plant disease species, they achieved remarkable classification accuracy. DenseNet-121, in particular, demonstrated superior performance with a classification accuracy of 99.81%. Guerrero-Ibañez et al. [8] proposed a model for identifying and classifying tomato leaf diseases using CNNs. They utilized a public dataset and supplemented it with additional photographs taken in the fields. To address the issue of overfitting, they incorporated generative adversarial networks into their methodology. The proposed model achieved a high level of accuracy, surpassing 99% for both the training and test datasets, highlighting the potential of CNNs for accurately detecting and classifying tomato leaf diseases.

In a different approach, Sunil et al. [9] employed machine learning and image processing techniques for plant leaf disease detection, focusing on tomato plants. They utilized various descriptors, such as Discrete Wavelet Transform, Principal Component Analysis, and Grey Level Co-occurrence Matrix, to extract informative features from resized tomato leaf samples. The extracted features were then classified using different algorithms, including Support Vector Machine (SVM), CNN, and K-Nearest Neighbor (K-NN). Their results showed that CNN achieved the highest accuracy of 99.6% for detecting tomato leaf diseases. Karthik et al. [10] addressed the challenge of plant disease detection and diagnosis using CNNs. They proposed two deep architectures: one based on residual learning and another incorporating an attention mechanism. The models were trained and evaluated using the Plant Village Dataset, achieving an overall accuracy of 98% on the validation sets. Furthermore, Sardogan et al. [11] proposed a CNN model combined with the Learning Vector Quantization (LVQ) algorithm for the detection and classification of tomato leaf diseases. Their dataset consisted of 500 images of tomato leaves with four different disease symptoms. The CNN model was utilized for automatic feature extraction, while the LVQ algorithm trained the network. The experimental results demonstrated the effectiveness of the proposed methodology in recognizing and classifying different types of tomato leaf diseases. Deepalakshmi et al. [12] aimed to identify diseased and healthy leaves of various plants by extracting features from input images using the CNN algorithm. The authors observed that the proposed system achieved an average identification time of 3.8 seconds with an accuracy of more than 94.5%. Kshyanaprava et al. [13] developed a CNN model for the detection of corn leaf diseases. They improved the accuracy of disease detection by incorporating rectified linear unit activation functions, adjusting parameters, pooling operations, and reducing the number of trainable parameters. Another study conducted by Moyazzoma et al. [14] exploited the power of transfer learning. They developed a model for plant leaf disease detection using CNN with a pre-trained feature extraction method called MobileNetV2. They focused on major crops in Bangladesh and aimed to classify leaf diseases using computer vision techniques. The proposed approach achieved a validation accuracy of 90.38% and aimed to benefit farmers by minimizing crop damage and reducing costs.

In terms of model architectures, CNNs have emerged as a popular choice for leaf disease detection due to their exceptional performance in image classification tasks. Researchers have proposed various CNN architectures tailored specifically for leaf disease detection. These architectures often involve multiple convolutional and pooling layers followed by fully connected layers. Additionally, techniques like transfer learning, where pre-trained models on large-scale datasets are fine-tuned for leaf disease detection, have been explored to leverage the knowledge learned from other domains.

Evaluation metrics play a crucial role in assessing the performance of leaf disease detection models. Studies have employed various metrics such as accuracy, precision, recall, F1-score, and area under the curve (AUC) to measure the effectiveness of the models. Additionally, researchers have investigated the impact of different factors on model performance, including the size and quality of the dataset, class imbalance, and the presence of multiple diseases on a single leaf. These studies collectively highlight the significance of CNNs in leaf disease detection. Some researchers focused solely on CNN-based classification, while others incorporated additional techniques such as pre-trained models, attention mechanisms, or transfer learning. These methodologies contributed to achieving high accuracy in detecting and classifying various types of leaf diseases. The advancements made in this field have the potential to revolutionize precision agriculture, minimize crop losses, and improve the overall quality of food production.

3 MATERIALS AND METHODS

3.1 Dataset

The study utilized the PlantVillage dataset, which contains a diverse range of plant diseases captured through around 54,305 images of plant leaves. The dataset features 14 different plant species, such as Apple, Blueberry, Cherry, Corn, Grape, Orange, Peach, Bell Pepper, Potato, Raspberry, Soybean, Squash, Strawberry, and Tomato. In Figure 1, you can find a random selection of eight pairs of plant and diseases [15].

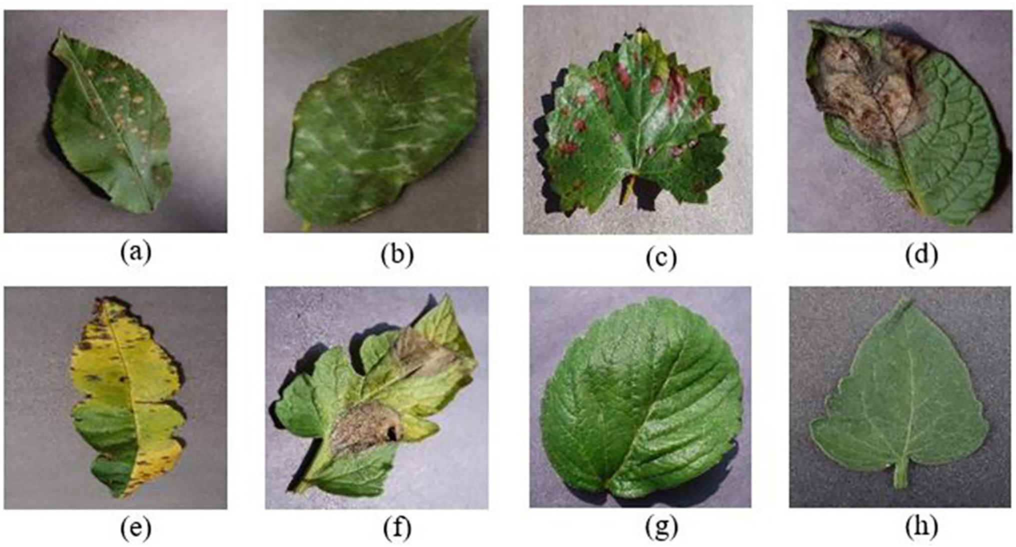

Fig. 1. Eight distinct pairs of plant-diseases that were extracted from the dataset

Table 1 presents the 38 classes in the dataset along with the number of images in the Training, Test, and Validation folders. Additionally, there is a class called "background without leaves" in the dataset to distinguish between the leaves and the background. The Training folder consists of 35,249 images, while the Test and Validation folders have 10,770 and 8,286 images, respectively. The distribution of images in each folder is expressed as a percentage. The Training folder has the highest number of images at 64.90%, with the Test and Validation folders at 19.84% and 15.26%, respectively.

Table 1. Summary of classes and number of images in the plant disease dataset

Leaf Class	Number of Images In		
	Training Folder	Test Folder	Validation Folder
Apple Black Rot	400	124	97
Apple Cedar Rust	164	55	56
Apple healthy	1053	329	263
Apple Scab	403	127	100
background without leaves	808	229	106
Bell Pepper Bacterial Spot	635	201	161
Bell Pepper healthy	947	295	236
Blueberry healthy	959	301	242
Cherry healthy	546	171	137
Cherry Powdery Mildew	676	211	165
Corn Common Rust	762	239	191
Corn Gray Leaf Spot	328	103	82
Corn healthy	800	232	130
Corn Northern Leaf Blight	630	197	158
Grape Black Measles	856	277	250
Grape Black Rot	756	236	188
Grape healthy	270	85	68
Grape Leaf Blight	688	216	172
Orange Haunglongbing	3604	1002	901
Peach Bacterial Spot	1469	460	368
Peach healthy	230	72	58
Percentage	64,90%	19,84%	15,26%
Potato Early Blight	640	200	160
Potato healthy	96	31	25
Potato Late Blight	640	200	160
Raspberry healthy	237	74	60
Soybean healthy	3561	1019	510
Squash Powdery Mildew	1189	367	279
Strawberry Healthy	292	91	73
Strawberry Leaf Scorch	710	222	177
Tomato Bacterial Spot	1360	426	341
Tomato Early Blight	640	200	160
Tomato healthy	1117	318	156
Tomato Late Blight	1222	381	306
Tomato Leaf Mold	609	191	152
Tomato Mosaic Virus	238	75	60
Tomato Septoria Leaf Spot	1132	355	284
Tomato Target Spot	890	280	234
Tomato Two Spotted Spider Mite	1072	335	269
Tomato Yellow Leaf Curl Virus	3428	1072	857
Total	35249	10770	8286

3.2 Preprocessing

In order to improve the accuracy of our models, we utilized image preprocessing techniques [16]. We employed data augmentation methods on the dataset [17] and selected the following parameters: a batch size of 128, three layers of convolutional neural networks with filter sizes of 3×3, 32 filters in the first two layers, and 64 filters in the third layer, and eight epochs. The final architecture consists of three layers of 3×3 convolution with ReLU activation and 2×2 Max-Pooling with 32 filters for the first two layers and 64 filters for the third layer. Additionally, a fully connected (FC) layer was added to classify the 39 classes.

3.3 Architecture and related terms

The architecture used is a Convolutional Neural Network (CNN) approach, which is a neural network consisting of several layers, including a convolutional layer, pooling layer, and activation layer. The convolutional layer applies a kernel to an input image, multiplying the values in the kernel with the corresponding values in the image to produce a filtered image that highlights patterns and features [18]. The filters used in this layer are vectors of weights adjusted during training to recognize different patterns and features in images. Pooling layers are similar to convolutional layers, but they apply a different function to the kernel and image window [19]. The most common types of pooling are max pooling and average pooling, which are useful for reducing the size of the output and simplifying the network. Activation layers apply a function to the output of the previous layer, squashing the values into a specific range [20]. The Rectified Linear Unit (ReLU) shown in Figure 2 is the most commonly used activation function in CNNs, setting all negative values to zero and leaving positive values unchanged. Activation layers typically squash output values into a range of [0,1] or [–1,1].

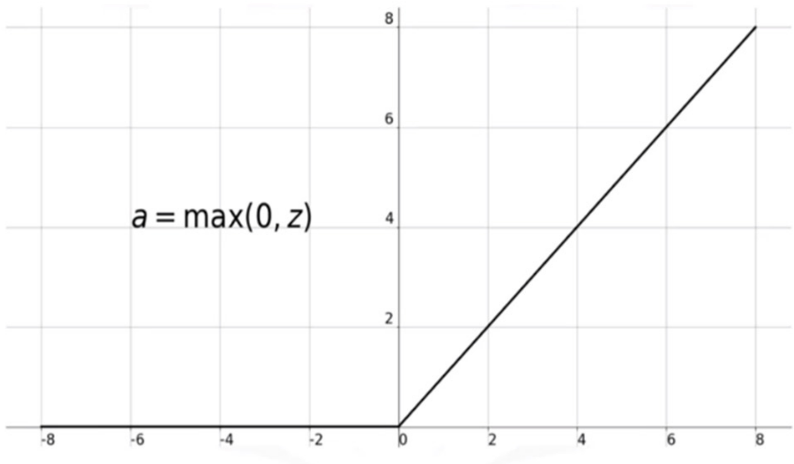

Fig. 2. ReLu function

The Fully Connected layer is a Multi-Layer Perceptron that uses a Softmax activation function in the output layer. This layer is "Fully Connected" because every neuron in the previous layer is connected to every neuron in the next layer (refer to Figure 3). The output obtained from the convolutional and pooling layers represents high-level features of the input image, and the Fully Connected layer is responsible for using these features to classify the input image into different categories based on the training dataset. The Softmax activation function in the output layer ensures that the sum of output probabilities from the Fully Connected layer is 1, by squashing a vector of arbitrary real-valued scores to a vector of values between zero and one that add up to one. Although other classifiers such as SVM can also be used, the Softmax function is commonly employed in this layer [21].

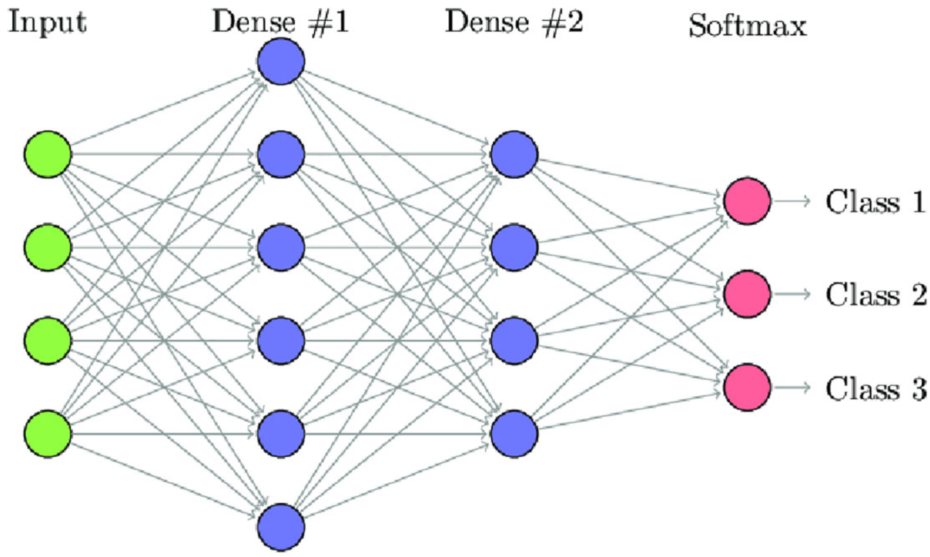

Fig. 3. CNN architecture with fully connected layer

There are various other ConvNet architectures besides the ones already mentioned. AlexNet is a deeper and wider version of LeNet, developed by Alex Krizhevsky and others, and won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012 [22]. ZFNet, the ILSVRC 2013 winner, improved upon AlexNet by tweaking the hyperparameters of the architecture [23]. GoogleNet, developed by Szegedy and others from Google, won the ILSVRC 2014 with its main contribution being the Inception Module that reduced the number of network parameters significantly [24]. VGGNet, the runner-up in ILSVRC 2014, showed that the depth of a network (number of layers) is critical to good performance [25]. Residual Network, developed by Kaiming He and others, won the ILSVRC 2015 and is currently the state-of-the-art Convolutional Neural Network model [26]. Finally, the Densely Connected Convolutional Network (DenseNet), published by Gao Huang and others, has each layer directly connected to every other layer in a feed-forward fashion and has shown significant improvements over previous architectures in highly competitive object recognition benchmarks [27].

3.4 Approach proposed

Our study utilized TensorFlow and various Python libraries, such as Keras, NumPy, Matplotlib, among others, to execute all experiments detailed in this paper. To accurately categorize our dataset, we employed the models illustrated in Figure 4, including VGGNet Model, Three Layer ConvNet Model, and Five Layer ConvNet Model [28].

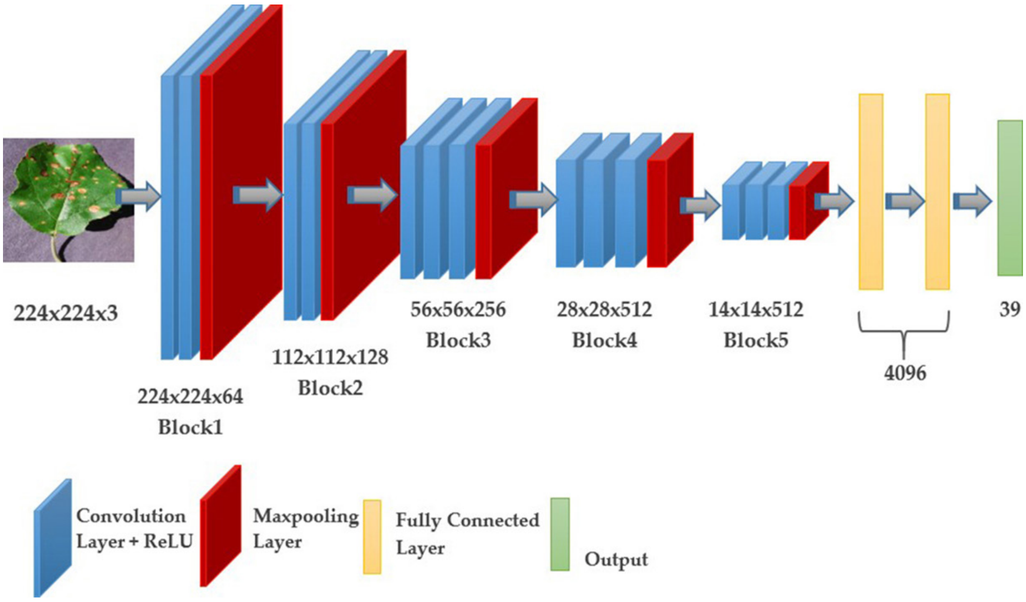

Fig. 4. VGG16 deep learning & three layer convnet architecture

The VGGNet architecture utilizes 3×3 convolution filters to increase depth while minimizing computational overhead compared to larger filters [25], [29]. The architecture comprises eight sections, with the first five sections featuring two pairs of convolution layers with ReLU activation, followed by max-pooling. The last three sections consist of fully-connected layers. The max-pooling size is 4×4 with a stride of 4 in the first section and 2×2 with a stride of 2 in the remaining sections. The number of filters in the convolution layer varies per section, with 64, 128, 256, 512, and 512 filters in the first five sections, respectively, and 4096 and 39 neurons for the fully-connected layers in the last section. During training, the VGGNet employs stochastic gradient descent (SGD) with Adam optimization and dropout for regularization [30].

The Three Layer ConvNet model consists of three sections [31], with the first two sections having a convolutional layer followed by ReLU activation and max-pooling. The third section comprises a fully-connected layer with a ReLU activation and a linear affine layer. The model uses 32 3×3 filters in the convolution layers with a stride of 1, and 2×2 max-pooling with a stride of 1. During training, the model employs SGD optimization with dropout for regularization to avoid overfitting. The input for the Three Layer ConvNet model is a raw image with dimensions of 224×224×3, and it can classify the input image into a specific tag.

Our 5-layer ConvNet architecture is similar to the 3-layer ConvNet, with the addition of two more convolutional-ReLU-pooling layers [32]. The convolutional layer parameters, max-pooling, and fully-connected layers remain the same. We also use SGD with Adam optimization and dropout for regularization during training [33].

3.5 Implementation of the model

To develop our classifier, we utilized the VGG16 Deep Learning Architecture [34]. We followed a set of steps to implement and train the model, beginning with importing the necessary libraries and loading the dataset. Next, we designed our convolutional neural network (CNN) model and generated diagnostic learning curves to track its progress. We also calculated the number of images in the train and test folders, and printed the class and label dictionary. Finally, we trained our model on the dataset.

As illustrated in Figure 5, the VGG16 model is composed of 13 convolutional layers and 3 fully connected layers. The first 13 layers are responsible for feature extraction, while the last 3 layers perform classification. The input shape is (None, 224, 224, 3), meaning it can accept an input image with a height and width of 224 pixels and 3 color channels (RGB).

The output is a tensor of shape (None, 39), corresponding to the number of output classes. The model has a total of 134,453,095 parameters, of which 126,801,447 are trainable parameters and 7,651,648 are non-trainable parameters. The trainable parameters are updated during training to optimize the model for the given task, while the non-trainable parameters are used to extract features from the input image.

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 224, 224, 3)]	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
flatten (Flatten)	(None, 25088)	0
dense (Dense)	(None, 4096)	102764544
dropout (Dropout)	(None, 4096)	0
batch_normalization (BatchNo	(None, 4096)	16384
dense_1 (Dense)	(None, 4096)	16781312
dropout_1 (Dropout)	(None, 4096)	0
batch_normalization_1 (Batch	(None, 4096)	16384
dense_2 (Dense)	(None, 39)	159783

Total params: 134,453,095
Trainable params: 126,801,447
Non-trainable params: 7,651,648

Fig. 5. The summary of the model

4 RESULTS AND DISCUSSION

In a Convolutional Neural Network (CNN), accuracy and loss are commonly used metrics to evaluate the performance of the model during training and testing phases.

Accuracy: The accuracy of a CNN is a measure of how well the model predicts the correct class or category of the input data. It is calculated as the ratio of correctly predicted samples to the total number of samples in the dataset. The accuracy equation is as follows (formula 1):

$$\text{Accurcy} = \frac{(\text{Number of correctly predicted samples})}{(\text{Total number of samples})} \tag{1}$$

Loss: Loss, also known as the cost function or objective function, represents the error or mismatch between the predicted output of the CNN and the actual target output. The loss function quantifies how well the model is performing in terms of minimizing the error. The choice of the loss function depends on the specific problem being solved. Some commonly used loss functions for classification tasks in CNNs include cross-entropy loss and mean squared error (MSE) loss.

The equation for these loss function is as in formula 2:

$$Cross-entropy\ loss: Loss = -\sum\left(y \times \log(p) + (1-y) \times \log(1-p)\right) \tag{2}$$

Where:

- y represents the true or actual target output.
- p represents the predicted output of the CNN.
- n represents the total number of samples.

During the training process, the CNN aims to minimize the loss function by adjusting the weights and biases of the network through optimization techniques such as gradient descent. By minimizing the loss, the CNN learns to make more accurate predictions over time.

The outcomes presented in Figures 6 and 7 demonstrate that the image classifier model was effective in categorizing object images. The plots exhibit a steady increase in both training and validation accuracies over each epoch. The training accuracy began at 83.75% and reached 99.73% by the eighth epoch, while the validation accuracy started at 91.75% and rose to 97.80% at the end of training. The model was also tested on a new dataset, and it yielded an accuracy of 97.8%, indicating that it has the capacity to generalize to unseen data. The training process took around 31 minutes to complete, with an average of 235 seconds per epoch. These findings indicate that the model is functioning well and can be employed for object image classification tasks.

```
Epoch 1/8
282/282 - 318s - loss: 0.5554 - accuracy: 0.8375 - val_loss: 0.2597 - val_accuracy: 0.9175
Epoch 2/8
282/282 - 235s - loss: 0.1347 - accuracy: 0.9561 - val_loss: 0.1755 - val_accuracy: 0.9430
Epoch 3/8
282/282 - 236s - loss: 0.0647 - accuracy: 0.9790 - val_loss: 0.1079 - val_accuracy: 0.9648
Epoch 4/8
282/282 - 235s - loss: 0.0373 - accuracy: 0.9879 - val_loss: 0.1031 - val_accuracy: 0.9693
Epoch 5/8
282/282 - 235s - loss: 0.0261 - accuracy: 0.9929 - val_loss: 0.0908 - val_accuracy: 0.9711
Epoch 6/8
282/282 - 235s - loss: 0.0191 - accuracy: 0.9953 - val_loss: 0.0839 - val_accuracy: 0.9750
Epoch 7/8
282/282 - 236s - loss: 0.0150 - accuracy: 0.9965 - val_loss: 0.0812 - val_accuracy: 0.9753
Epoch 8/8
282/282 - 235s - loss: 0.0131 - accuracy: 0.9973 - val_loss: 0.0764 - val_accuracy: 0.9780
```

Fig. 6. Learning results

```
summarize_diagnostics(history)
```

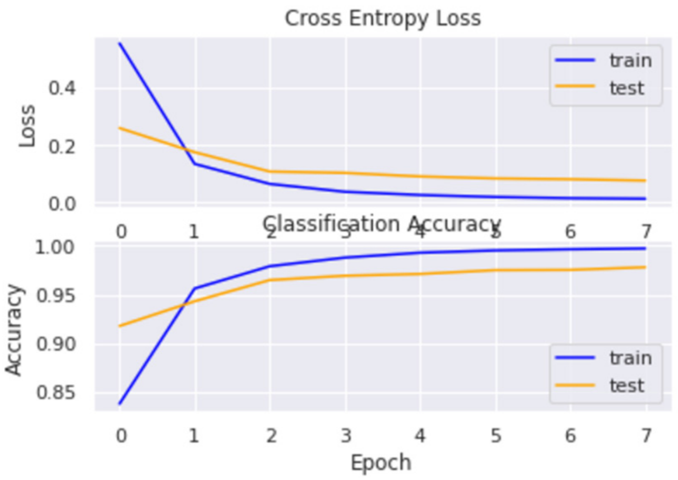

Fig. 7. Learning curves

Evaluating the predictions of the model is crucial in assessing its effectiveness in image classification tasks. The ability of the model to accurately identify and classify images is an indication of how well it has learned the relevant features of the input images. The prediction results showcase its strong performance in accurately identifying various types of plant diseases, as depicted in the input images. This proves the ability of the model to classify images and make it a dependable tool for identifying plant diseases in real-world situations. Algorithm 1 shows that the image classification model can successfully detect Pepper Bell Bacterial Spot with high accuracy. The model loads an image of a pepper leaf with bacterial spots from a validation set of a plant disease dataset and uses a pre-trained VGG16 convolutional neural network model, saved as *'plantdisease_vgg16model.h5'*, to predict the class label of the image.

Algorithm 1: Prediction of the correct class
Input: Image path **Output:** Detected leaf class name **Procedure:** **1.** Load the image from the specified path. **2.** Print the label: "Pepper, bell_Bacterial_spot:" **3.** Predict the leaf class using the trained model: a. Obtain the prediction array by passing the image through the new_model. b. Find the index corresponding to the maximum value in the prediction array. c. Use the index to retrieve the class label from the *'labels'* array. **4.** Print the detected leaf class name.

The model proposed in our study achieved impressive results on the PlantVillage dataset, with a training accuracy exceeding 99%. To evaluate the performance of our model, we compared it with other studies that utilized the same dataset and similar models. Additionally, we considered findings from articles 1 to 10 to provide a comprehensive comparison. Sladojevic et al. [35] conducted a study on the PlantVillage dataset, employing various deep learning models such as VGG16, InceptionV3, and ResNet50. They reported an accuracy of 98.8% using the InceptionV3 model, which is slightly lower than the accuracy obtained by the VGG16 model in our study. Mohanty et al. [36] also utilized the PlantVillage dataset and applied different machine learning algorithms, including Random Forest and Support Vector Machines. They reported an accuracy of 99.35% using the Random Forest algorithm, which is marginally lower than the accuracy achieved by our deep learning model. Researchers at the Indian Institute of Technology Roorkee conducted a study using a dataset of 38 plant disease classes and the VGG16 architecture. Their deep learning model achieved an accuracy of 96.29% on the test set [37]. While this accuracy is lower than the results achieved in our study, it is important to note that their dataset included a higher number of disease classes. Aravind et al. [38] employed GoogleNet and a pretrained VGG16 as a feature extractor to identify different diseases in plants. Their model achieved an accuracy of 97.3% on the test set, which is comparable to the accuracy achieved in our study. Jiang et al. [39] used a deep learning model based on the ResNet-50 architecture to classify plant diseases from a dataset of 15 classes. Their model achieved an accuracy of 96.8% on the test set, slightly lower than the accuracy achieved in our study. Expanding our comparison beyond the PlantVillage dataset, we considered additional studies for a broader perspective. Prasanna et al. [6] proposed a leaf disease detection system using a CNN algorithm, achieving an accuracy of 96% for 8 epochs on the PlantVillage Dataset. Sunil et al. [9] developed a plant leaf disease detection system using computer vision and machine learning algorithms. They achieved accuracies of 88% using Support Vector Machine (SVM), 97% using K-Nearest Neighbor (K-NN), and 99.6% using Convolutional Neural Network (CNN) on tomato disordered samples. Andrew et al. [7] employed CNN-based pre-trained models, including DenseNet-121, ResNet-50, VGG-16, and Inception V4, to identify plant diseases. Their DenseNet-121 model achieved an impressive classification accuracy of 99.81%. Guerrero-Ibañez et al. [8] proposed a model based on convolutional neural networks to identify and classify tomato leaf diseases, achieving an accuracy greater than 99% in both the training and test datasets. Shelar et al. [40] used image processing with a CNN for plant disease detection, achieving accurate results without providing specific accuracy values. Sardogan et al. [11] presented a CNN model combined with the Learning Vector Quantization (LVQ) algorithm for tomato leaf disease detection and classification. Their model effectively recognized four different types of tomato leaf diseases. Deepalakshmi et al. [12] aimed to identify diseased and healthy leaves of different plants using a CNN algorithm, achieving an accuracy of more than 94.5% with an average identification time of 0.15 seconds per leaf. Kshyanaprava et al. [13] developed a CNN approach for corn leaf disease detection, achieving an average detection accuracy of 98.78% for identifying various corn leaf diseases. Karthik et al. [10] proposed an attention-embedded residual CNN for disease detection in tomato leaves, achieving an overall accuracy of 98% on the validation sets. Moyazzoma et al. [14] utilized transfer learning with CNN and MobileNetv2 for plant leaf disease detection, achieving a validation accuracy of 90.38%. In comparison with the aforementioned studies, our proposed model demonstrates competitive performance with high accuracy on the PlantVillage dataset. When comparing results across different studies, it is essential to consider factors such as the number of disease classes, dataset size, and specific implementation details. Overall, the results of our model in the current study demonstrate its ability to effectively classify images of diseased plants, which can be used to support plant pathology research and disease management strategies. However, it is important to note that the performance of the model can vary depending on various factors, such as the quality and size of the dataset, the choice of hyperparameters, and the specific task at hand. Therefore, further research is needed to explore the optimal models and techniques for plant disease classification using deep learning.

Limitations of the study include the use of a single dataset, the PlantVillage dataset, which may not generalize well to other plant disease datasets. Additionally, the study focused solely on the classification of plant disease images, and future research could explore other aspects of plant disease detection, such as early detection or prediction of disease outbreaks. Furthermore, the study did not explore the interpretability of the model, which is an essential aspect of understanding how the model makes decisions. Finally, while the proposed model achieved impressive results, it is important to note that the performance may vary in real-world scenarios, and further testing and validation is required to determine the practical usefulness of the model. To address the limitations of the study, future research could focus on collecting and using multiple plant disease datasets to improve the generalizability of the model. Additionally, exploring methods for early detection or prediction of disease outbreaks could have a significant impact on reducing crop losses and improving agricultural productivity. Furthermore, investigating the performance of the proposed model in real-world scenarios and testing its practical usefulness could be a valuable avenue for future research.

5 CONCLUSION

Deep learning has made significant advancements in various machine learning problems such as speech recognition and computer vision. However, the intricate structure of deep learning models makes it difficult for users to understand the underlying processes. Moreover, selecting suitable models and determining hyperparameters is a challenging task. In this article, we present a deep learning model for image classification and discuss the results obtained. Our deep learning model demonstrated remarkable accuracy in classifying images of plant diseases, achieving a training accuracy of over 99%. The ability of the proposed model to generalize to new data was verified by its accuracy on the test dataset. Additionally, the predictions showed excellent performance in identifying different types of plant diseases, making it a reliable tool for real-world scenarios. While the accuracy achieved was slightly lower than some other studies that used the same dataset, it still performed exceptionally well. Therefore, our model can be used as an effective tool for image classification tasks involving images of objects, particularly for identifying plant diseases.

6 REFERENCES

[1] L. Geng and T. Dong, 'An agricultural monitoring system based on wireless sensor and depth learning algorithm', *Int. J. Onl. Eng.*, vol. 13, no. 12, pp. 127–137, 2017, https://doi.org/10.3991/ijoe.v13i12.7885

[2] M. Shivanandappa and M. M. Patil, 'Extraction of image resampling using correlation aware convolution neural networks for image tampering detection', *IJECE*, vol. 12, no. 3, pp. 3033–3043, 2022, https://doi.org/10.11591/ijece.v12i3.pp3033-3043

[3] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, 'Gradient-based learning applied to document recognition', *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998, https://doi.org/10.1109/5.726791

[4] Y. Lecun *et al.*, 'Handwritten digit recognition with a back-propagation network', in *Advances in Neural Information Processing Systems (NIPS 1989), Denver, CO*, D. Touretzky, Ed., Morgan Kaufmann, 1990.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, 'ImageNet classification with deep convolutional neural networks', *Commun. ACM*, vol. 60, no. 6, pp. 84–90, 2017, https://doi.org/10.1145/3065386

[6] P. L. Kompalli, K. R. Mekala, V. S. R. S. Modala, V. Devalla, and A. B. Kompalli, 'Leaf disease detection and remedy recommendation using CNN algorithm', *Int. J. Onl. Eng.*, vol. 18, no. 07, pp. 85–100, 2022, https://doi.org/10.3991/ijoe.v18i07.30383

[7] J. Andrew, J. Eunice, D. E. Popescu, M. K. Chowdary, and J. Hemanth, 'Deep learning-based leaf disease detection in crops using images for agricultural applications', *Agronomy*, vol. 12, no. 10, p. 2395, 2022, https://doi.org/10.3390/agronomy12102395

[8] A. Guerrero-Ibañez and A. Reyes-Muñoz, 'Monitoring tomato leaf disease through convolutional neural networks', *Electronics*, vol. 12, no. 1, p. 229, 2023, https://doi.org/10.3390/electronics12010229

[9] S. S. Harakannanavar, J. M. Rudagi, V. I. Puranikmath, A. Siddiqua, and R. Pramodhini, 'Plant leaf disease detection using computer vision and machine learning algorithms', *Global Transitions Proceedings*, vol. 3, no. 1, pp. 305–310, 2022, https://doi.org/10.1016/j.gltp.2022.03.016

[10] R. Karthik, M. Hariharan, S. Anand, P. Mathikshara, A. Johnson, and R. Menaka, 'Attention embedded residual CNN for disease detection in tomato leaves', *Applied Soft Computing*, vol. 86, p. 105933, 2020, https://doi.org/10.1016/j.asoc.2019.105933

[11] M. Sardogan, A. Tuncer, and Y. Ozen, 'Plant leaf disease detection and classification based on CNN with LVQ algorithm', in *2018 3rd International Conference on Computer Science and Engineering (UBMK)*, Sarajevo: IEEE, 2018, pp. 382–385. https://doi.org/10.1109/UBMK.2018.8566635

[12] P. Deepalakshmi, T. Prudhvi Krishna, S. Siri Chandana, K. Lavanya, and P. N. Srinivasu, 'Plant leaf disease detection using CNN algorithm', *International Journal of Information System Modeling and Design*, vol. 12, no. 1, pp. 1–21, 2021, https://doi.org/10.4018/IJISMD.2021010101

[13] K. P. Panigrahi, A. K. Sahoo, and H. Das, 'A CNN approach for corn leaves disease detection to support digital agricultural system', in *2020 4th International Conference on Trends in Electronics and Informatics (ICOEI) (48184)*, Tirunelveli, India: IEEE, 2020, pp. 678–683. https://doi.org/10.1109/ICOEI48184.2020.9142871

[14] R. Moyazzoma, Md. A. A. Hossain, Md. H. Anuz, and A. Sattar, 'Transfer learning approach for plant leaf disease detection using CNN with pre-trained feature extraction method Mobilnetv2', in *2021 2nd International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST)*, DHAKA, Bangladesh: IEEE, 2021, pp. 526–529. https://doi.org/10.1109/ICREST51555.2021.9331214

[15] D. P. Hughes and M. Salathe, 'An open access repository of images on plant health to enable the development of mobile disease diagnostics'. arXiv, Apr. 11, 2016. Accessed: Mar. 27, 2023. [Online]. Available: http://arxiv.org/abs/1511.08060

[16] B. Jabir, N. Falih, and K. Rahmani, 'Accuracy and efficiency comparison of object detection open-source models', *Int. J. Onl. Eng.*, vol. 17, no. 05, pp. 165–184, 2021, https://doi.org/10.3991/ijoe.v17i05.21833

[17] K. Ould Babaali, E. Zigh, M. Djebbouri, and O. Chergui, 'A new approach for road extraction using data augmentation and semantic segmentation', *IJEECS*, vol. 28, no. 3, pp. 1493–1501, 2022, https://doi.org/10.11591/ijeecs.v28.i3.pp1493-1501

[18] M. E. Hoque and K. Kipli, 'Deep learning in retinal image segmentation and feature extraction: A review', *Int. J. Onl. Eng.*, vol. 17, no. 14, pp. 103–118, 2021, https://doi.org/10.3991/ijoe.v17i14.24819

[19] R. Nirthika, S. Manivannan, A. Ramanan, and R. Wang, 'Pooling in convolutional neural networks for medical image analysis: A survey and an empirical study', *Neural Comput & Applic*, vol. 34, no. 7, pp. 5321–5347, 2022, https://doi.org/10.1007/s00521-022-06953-8

[20] T. Szandała, 'Review and comparison of commonly used activation functions for deep neural networks', in *Bio-inspired Neurocomputing*, A. K. Bhoi, P. K. Mallick, C.-M. Liu, and V. E. Balas, Eds., in Studies in Computational Intelligence, vol. 903. Singapore: Springer Singapore, 2021, pp. 203–224. https://doi.org/10.1007/978-981-15-5495-7_11

[21] F. I. Salaheddine, 'Convolutional neural networks (CNN): Models and algorithms', in *Advances in Healthcare Information Systems and Administration*, H. A. Bouarara, Ed., IGI Global, 2022, pp. 97–120. https://doi.org/10.4018/978-1-6684-5656-9.ch005

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, 'ImageNet classification with deep convolutional neural networks', in *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, P. L. Bartlett, F. C. N. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., 2012, pp. 1106–1114. [Online]. Available: https://proceedings.neurips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html

[23] M. D. Zeiler and R. Fergus, 'Visualizing and understanding convolutional networks'. arXiv, Nov. 28, 2013. Accessed: Mar. 27, 2023. [Online]. Available: http://arxiv.org/abs/1311.2901

[24] C. Szegedy *et al.*, 'Going deeper with convolutions'. arXiv, Sep. 16, 2014. Accessed: Mar. 28, 2023. [Online]. Available: http://arxiv.org/abs/1409.4842

[25] K. Simonyan and A. Zisserman, 'Very deep convolutional networks for large-scale image recognition'. arXiv, Apr. 10, 2015. Accessed: Mar. 28, 2023. [Online]. Available: http://arxiv.org/abs/1409.1556

[26] K. He, X. Zhang, S. Ren, and J. Sun, 'Deep residual learning for image recognition'. arXiv, Dec. 10, 2015. Accessed: Mar. 28, 2023. [Online]. Available: http://arxiv.org/abs/1512.03385

[27] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, 'Densely connected convolutional networks', in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI: IEEE, 2017, pp. 2261–2269. https://doi.org/10.1109/CVPR.2017.243

[28] K. O'Shea and R. Nash, 'An introduction to convolutional neural networks'. arXiv, Dec. 02, 2015. Accessed: Mar. 28, 2023. [Online]. Available: http://arxiv.org/abs/1511.08458

[29] X. Chen *et al.*, 'A VGGNet-based method for refined bathymetry from satellite altimetry to reduce errors', *Remote Sensing*, vol. 14, no. 23, p. 5939, 2022, https://doi.org/10.3390/rs14235939
[30] Y. Wang, P. Zhou, and W. Zhong, 'An optimization strategy based on hybrid algorithm of adam and SGD', *MATEC Web Conf.*, vol. 232, p. 03007, 2018, https://doi.org/10.1051/matecconf/201823203007
[31] J. A. Pandian *et al.*, 'A five convolutional layer deep convolutional neural network for plant leaf disease detection', *Electronics*, vol. 11, no. 8, p. 1266, 2022, https://doi.org/10.3390/electronics11081266
[32] G. Liu, Z. Wang, H. Zhang, X. Guo, Y. Wang, and C. Zhang, 'A novel violent video detection method based on improved C3D and transfer learning', in *CIBDA 2022; 3rd International Conference on Computer Information and Big Data Applications*, 2022, pp. 17.
[33] R. Chauhan, K. K. Ghanshala, and R. C. Joshi, 'Convolutional neural network (CNN) for image detection and recognition', in *2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC)*, Jalandhar, India: IEEE, 2018, pp. 278–282. https://doi.org/10.1109/ICSCCC.2018.8703316
[34] N. H. Alkurdy, H. K. Aljobouri, and Z. K. Wadi, 'Ultrasound renal stone diagnosis based on convolutional neural network and VGG16 features', *IJECE*, vol. 13, no. 3, pp. 3440–3448, Jun. 2023, https://doi.org/10.11591/ijece.v13i3.pp3440-3448
[35] S. Sladojevic, M. Arsenovic, A. Anderla, D. Culibrk, and D. Stefanovic, 'Deep neural networks based recognition of plant diseases by leaf image classification', *Computational Intelligence and Neuroscience*, vol. 2016, pp. 111, 2016, https://doi.org/10.1155/2016/3289801
[36] S. P. Mohanty, D. P. Hughes, and M. Salathé, 'Using deep learning for image-based plant disease detection', *Front. Plant Sci.*, vol. 7, p. 1419, 2016, https://doi.org/10.3389/fpls.2016.01419
[37] M. Polling *et al.*, 'Neural networks for increased accuracy of allergenic pollen monitoring', *Sci Rep*, vol. 11, no. 1, p. 11357, 2021, https://doi.org/10.1038/s41598-021-90433-x
[38] K. Rangarajan Aravind and P. Raja, 'Automated disease classification in (Selected) agricultural crops using transfer learning', *Automatika*, vol. 61, no. 2, pp. 260–272, 2020, https://doi.org/10.1080/00051144.2020.1728911
[39] F. Jiang, Y. Lu, Y. Chen, D. Cai, and G. Li, 'Image recognition of four rice leaf diseases based on deep learning and support vector machine', *Computers and Electronics in Agriculture*, vol. 179, p. 105824, 2020, https://doi.org/10.1016/j.compag.2020.105824
[40] N. Shelar, S. Shinde, S. Sawant, S. Dhumal, and K. Fakir, 'Plant disease detection using CNN', *ITM Web Conf.*, vol. 44, p. 03049, 2022, https://doi.org/10.1051/itmconf/20224403049

7 AUTHORS

Moulay Hafid Aabidi received his Ph.D. degree in Computer Science in 2016 from Moulay Ismail University, Faculty of Science and Technology, Morocco. he is currently a Research Professor in Computer Sciences at Higher School of Technology – Khenifra, Moulay Slimane University, Morocco. He is affiliated to Multidisciplinary Research Laboratory for Science, Technology and Society (L.P.S.T.S.) and the Department of Computer Engineering and Mathematics, Higher School of Technology, Khenifra, Sultan Moulay Slimane University, Morocco. His research interests include Software Engineering, Requirements Engineering, Code Generation, Reverse Engineering, Artificial Intelligence, Big Data Analytics and Data Science (email: myhafidaabidi@yahoo.fr; h.aabidi@usms.ma).

Adil EL Makrani is a Research Professor in Computer Science at the Faculty of Science, Ibn Tofail University, Kenitra. He received his Master in Computer Science, Computer Graphics and Imagery, and his Ph.D. in Computer Science, from Sidi Med Ben Abdellah University in 2009 and 2015, respectively. He is affiliated to Computer Science Research Laboratory (L@RI), Faculty of Science, Ibn Tofail University, Kenitra, Morocco. His research is currently focuses on Artificial Intelligence technologies, Big Data Analytics, and their applications (email: adil.elmakrani@uit.ac.ma).

Brahim Jabir is a professor of computer science at the Teacher Training College of Beni Mellal Khenifra. He received his degree in computer engineering from the Faculty of Sciences and Techniques at Sultan Moulay Slimane University in Beni Mellal, Morocco. Since 2021, he is the Leader of the LANDitic Research Group, he has also been responsible for the ICT department at the Teacher Training College. He is an associate editor at the international journal information technologies systems of IGI Global, and his research focuses on artificial intelligence (AI) and its various applications in different fields (email: ibra.jabir@gmail.com).

Imane Zaimi received her Ph.D. degree in 2018 in Computer Sciences from Mohammed V University, Faculty of sciences in Rabat, Morocco. She is currently Research Professor in Computer Sciences at Higher School of Technology, Khenifra, Moulay Slimane University, Morocco. She is affiliated to Multidisciplinary Research Laboratory for Science, Technology and Society (L.P.S.T.S.) and the Department of Computer Engineering and Mathematics, Higher School of Technology, Khenifra, Sultan Moulay Slimane University, Morocco. Her research interests include Performance Evaluation, Quality of Service, Quality of Experiment, Multimedia Communications, Topology Control Traffic Optimization and Fuzzy Logic Applications in Wireless Ad hoc Networks (email: imanzaimi@gmail.com; i.zaimi@usms.ma).