Journal of Plant Diseases and Protection (2022) 129:521–532
https://doi.org/10.1007/s41348-022-00589-5

ORIGINAL ARTICLE

Early detection and identification of grape diseases using convolutional neural networks

RajinderKumar M. Math[1,2] · **Nagaraj V. Dharwadkar**[3]

Received: 21 May 2021 / Accepted: 14 February 2022 / Published online: 1 March 2022

Abstract

Crop protection aims to develop an agriculture system that is resilient to common agricultural threats like diseases, pests, and weeds that result in sub-optimal growth of crops in terms of quality and quantity. Therefore, timely disease detection and identification is a crucial concern for farmers across the globe. At present, crop diseases are identified by farmers manually, which is time-consuming, subjective, and also error-prone due to human involvement. The early disease identification and detection would help the farmers reduce the use of pesticides, minimizing the environmental footprint while increasing the profits by reducing the losses. To precisely identify the crop diseases at the initial stages, the machine learning (ML) algorithms or, in more precise, deep learning (DL) algorithms are very helpful. The research aims to develop a deep convolutional neural network (DCNN) model to identify and classify grape diseases based on the RGB leaf images. The proposed model uses an image dataset of grape crops from the Plant Village dataset publicly available for researchers and engineers. The specialty of the developed model is that the CNN classification model is built from scratch that provides accuracy close to or even more significant than the accuracy obtained for some pre-trained models using transfer learning. The model achieved an accuracy of 99.34% and equal values for precision, recall, and an F1 score of 0.9934. These results indicate models' capability to accurately identify and classify grapes' common diseases based on the RGB leaf images. The trained model is converted and saved in TensorFlow tflite format, and it can be readily deployed to mobile devices to provide real-time disease identification in precision agriculture (PA) application.

Keywords Crop protection · Precision agriculture · Machine learning · Deep learning · Disease detection

Introduction

Precision agriculture aims to provide a sustainable solution to the farmers who heavily rely on agriculture for their only livelihood. This sustainability is brought about by strategically managing the agricultural resources to bring down the agricultural losses while enhancing profit by improved quantity and quality of yields. One of the critical factors that can bring sustainability in the agricultural domain is adopting suitable crop management schemes.

Crop management strategies, particularly crop protection against diseases, play a vital role in the PA system's successful implementation. Crop protection against diseases involves identifying and detecting crop diseases when they are first sited on the crops. Thus, early disease detection can help farmers control the spread of the disease, thereby avoiding crop loss.

One of the reasons why farming in countries like India is considered risky is that most of the farmers' losses are natural calamities like floods and drought (due to poor monsoons). Even if everything goes in favour of farmer (sufficient water for irrigation), normally invisible disease-causing organisms such as fungi, nematodes, bacteria, and viruses might attack the crops (either in the early stage or

✉ RajinderKumar M. Math
ec.math@bldeacet.ac.in

Nagaraj V. Dharwadkar
nagaraj.dharwadkar@ritindia.edu

[1] Department of Electronics and Communication Engineering, B.L.D.E. Association's V.P Dr. P.G. Halakatti College of Engineering and Technology, Ashram Road, Vijayapur, India

[2] Research Scholar at VTU- Regional Research Center (RRC), Belagavi, Karnataka, India

[3] Department of Computer Science and Engineering, Rajarambapu Institute of Technology, Islampur, Maharashtra, India

at the harvesting stage), resulting in the loss of the total yield or its majority. Commonly, infestation by pathogenic microbes visually appears on the crop or plant. Ensuring precise crop protection from diseases requires a precise disease detection system. The state-of-the-art technologies are precise enough to detect the diseases at the inception, reducing the requirements of many pesticides or herbicides at later stages. The availability of high-resolution field images of crops (leaves and fruits) to the researchers has led to the successful implementations using the Convolutional Neural Networks (CNNs) (Bhatia et al. 2020; Trivedi et al. 2020), where disease detection is possible over a click on a mobile phone. The implementations can be obtained by developing the DL models starting from scratch or using the transfer learning approach where a pre-trained model is trained on the new dataset to avoid significant training times and high computational resource requirements.

Deep learning models find versatile usage due to their ability to handle vast amounts of data, especially when working with image data corresponding to the crops. Computer-vision is another technology that has gained momentum in real-time crop disease detection and diagnosis utilizing smartphones and computers (Thangaraj et al. 2021; Tian et al. 2020).

Crop diseases have always been the enemy of farmers ever since the inception of farming practices. Globally, farmers lose nearly 40% of their crops to pests and diseases. Researchers involved in Precision Agriculture systems are continuously striving to develop state-of-the-art systems capable of improving protection against crop diseases while reducing the environmental footprint. Crops mainly develop diseases due to either biotic or abiotic factors (Ahmad et al. 2019). Abiotic factors consist of living organisms (pathogens) of fungi, bacteria, viruses, and nematodes. While the abiotic factors essentially consist of non-living things such as temperature, humidity, wind drought, nutrient imbalance, under irrigation, over-irrigation.

In conventional agriculture, crop disease detection and identification were very challenging for the farmers as they relied on the manual methods of visually inspecting every crop to identify the diseases. This manual disease detection method was time-consuming, where the farms are widespread. It was also prone to gross errors due to human intervention. To avoid losses due to crop diseases, farmers started using pesticides and fertilizers in excess quantities. This excessive usage of pesticides soon started to pose another problem for the farmers, a problem where the soil started to degrade and the environment (Rahman and Zhang 2018), which adversely impacted the yields.

Researchers could use the variable rate application (VRA) on the farm (Kempenaar et al. 2017). With VRA's introduction, using agricultural tools and machines optimized the agricultural inputs (water, fertilizers, pesticides, and herbicides) while automation reduced the farmers' burden. Soon, the contributions of cutting-edge technologies such as IoT (Dasig 2020), big data analytics (Shastry and Sanjay 2020), artificial intelligence (AI) (Patrício and Rieder 2018), remote sensing and satellite imagery (Yang 2020), and UAV (Tsouros et al. 2019) took agriculture altogether to a new level. Climate change (Hatfield et al. 2020) is yet another serious issue that has a significant impact on the sustainability of agriculture. The biotic factors affecting agricultural productions are controllable, but abiotic factors heavily depend on the climate and are uncontrollable directly. As the global population is rising, the agricultural product demand will always rise. Thus, the agriculture domain is always happening, where the bulk of the research takes place, and researchers contribute immensely. Any technological advancement should make farmers' lives easy, which is otherwise a pity in most countries.

Materials and methods

This section highlights the materials and methods used to develop a CNN model that can precisely identify three disease classes of grape and one healthy. The first sub-section provides insights into the dataset, and the second subsection is concerned with the exploratory data analysis (EDA), followed by the third subsection dealing with image preprocessing and data augmentation. The pre-trained models used in research utilizing the transfer learning approach were adopted (Keras: The Python Deep Learning API, 2021).

Dataset description

The dataset used to develop the CNN model for grapevine leaf-based disease detection and classification is derived from the well-known PlantVillage (Mohanty et al. 2016) dataset of 54,303 images divided into 38 leaf images comprising healthy and diseased leaves. This dataset is viral among beginners in machine learning and researchers working in the agriculture domain. It has pre-processed dimensions $256 \times 256 \times 3$, where 256 corresponds to height and width, and 3 indicates the number of channels (RGB). These images are pre-processed and are ready to be inputted into the machine learning algorithm. Only the images of grape leaves were selected, which belonged to four categories: black rot, healthy, black measles, and leaf blight. These three categories are the major threats to grape cultivation and contribute to most losses due to diseases. Image augmentation was adopted to increase the number of images belonging to each dataset class.

It is clear from (Table 1) that the dataset is highly imbalanced, consisting of images belonging to multiclass with a variable number of images corresponding to each class. The

Table 1 Grape image dataset with training and testing split of 70% and 30%, respectively

Sl. no.	Class	Training set (70%)	Testing set (30%)	Total images per class/ (%)
1	Black Rot	826	354	1180 (29.0%)
2	ESCA (Black Measles)	968	415	1383 (34.0%)
3	Healthy	296	127	423 (10.4%)
4	Leaf Blight	753	323	1076 (26.5%)
	Total images	2843	1219	4062

dataset has a majority class as Esca (34.0%, 1383 images), followed by Black Rot (29.0%, 1180 images), Leaf Blight (26.5%, 1076), and Healthy (10.4%, 423 images). The train-test split is also depicted in (Table 1) with four image classes (70% train and 30% test).

Exploratory data analysis

Exploratory data analysis (EDA) is a widely used technique that analyses the data and simplifies the model-building process. When dealing with images, extracting meaningful features that define the image classes when solving classification problems is necessary. Thus, it is always required to understand the data's crucial features to help build the model that best suits the data's features. The dataset consists of pre-processed images having dimensions of 256 × 256 with three RGB channels. The colour images form an essential feature that helps build feature maps during convolution operation, providing accurate class activation.

The proposed model classifies healthy leaves against diseased ones. The green channel is more prominent in the images corresponding to the healthy leaves than the diseased ones. The image class belonging to healthy has a higher distribution of green channel values than the other classes.

Image pre-processing and augmentation

Developing a highly generalizable image classification model involving CNN's requires an enormous image dataset and suitable image pre-processing techniques considering the deployment or production environment and real-world image scenarios where the trained model has to provide accurate inferences. Some of the pre-processing steps typically used in practice are image resizing, rescaling, and colour corrections. Image augmentation artificially increases the number of images in the dataset using some functions on the original dataset images. The Keras library helped increase the number of training images in the dataset using image augmentation. Apart from expanding the training set images, data augmentation also helps build a robust model that avoids under-fitting the model due to insufficient images and aids the model to generalize well on unseen data. Multiple images are generated by image augmentation on images. Some techniques include width and height shift, shear, brightness, random zoom, horizontal and vertical flips, and random rotations. After pre-processing and augmentation, the images are inputted to the CNN model as indicated in the architecture (Fig. 1).

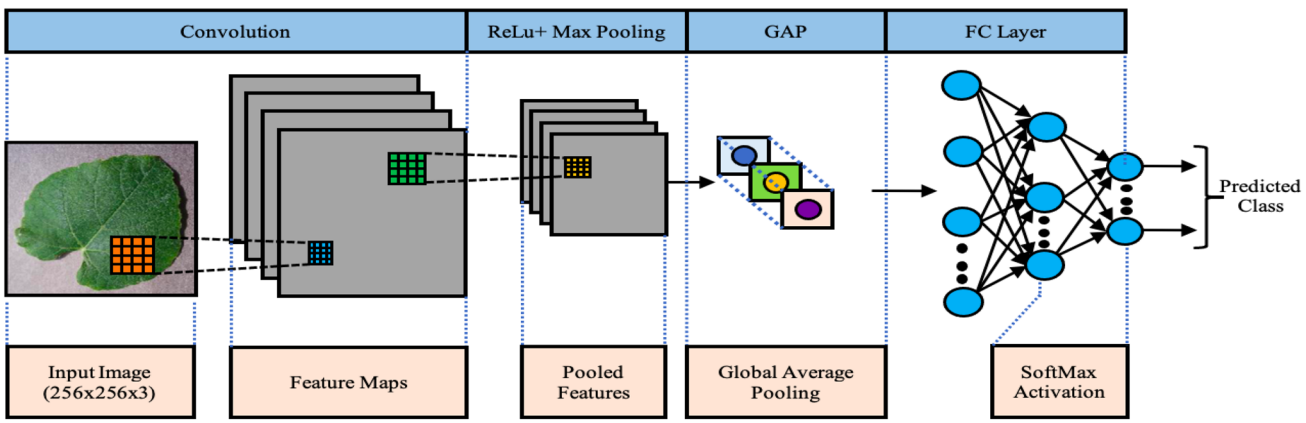

Fig. 1 Architecture of the proposed CNN model indicating various layers involved

Architecture of the proposed model

The architecture of the proposed CNN model is as shown in (Fig. 1). The architecture resembles a standard CNN architecture in which the Global Average Pooling (GAP) layer replaces the *flattening* layer. The activation function used throughout is rectified linear unit (ReLU), which adds a non-linearity as the images are highly non-linear.

Apart from activation function at each layer, the architecture consists of convolutional layers (for generating feature maps), max-pooling layers (for reducing the image dimension, also known as down sampling), global average pooling layer (for dimensionality reduction), and dense layers (for classifying image into a label).

The convolutional layer forms the core functionality of any CNN model. The convolutional layer is responsible for building feature maps using the convolutional filters. The filters also referred to as kernels, help extract some abstract information from the images. The first convolutional layer filters extract features such as edges, colours, and lines. As the CNN's depth increases, the convolutional layer filters can extract very complex features from the images necessary in classifying them into respective classes. The convolutional filter size is usually selected as odd size 3×3, 5×5, 7×7, or 11×11. The choice of the filter depends on the input image dimensions. The depth of the filter is selected to be the same as that of the image depth (Channels). As the filter size increases, the classification accuracy will improve at the cost of higher computational time. The implementation here uses eight convolutional layers for extracting low, medium, and high-level features corresponding to the four image classes.

Normally, the performance of any deep learning model increases with the increase in the number of images in the dataset and improves upon the classification and generalization capability. With the increase in the number of images, the training time and computational power requirement shoot up. The convolutional layers responsible for generating feature maps tend to increase in number with an increasing number of filters, resulting in higher training time and becoming computationally intensive. To reduce the number of computations and training time, the dimensions of images are down sampled using pooling layers. A max-pooling layer always follows a convolutional layer that acts like a down sampler where the input image dimensions are reduced to meet these requirements. For example, if the image dimension is 4×4 after convolution, the max-pooling layer output will result in an image dimension of 2×2 that drastically reduces the model training time. Other available pooling options include min-pooling and average pooling. The proposed system uses four max-pooling layers to reduce the image dimensions while retaining the critical features necessary for accurate classification.

Dropout is essentially a technique used to provide regularization when building deep learning models. The dropout layer in the deep learning model serves two critical purposes. Firstly, it avoids overfitting during model training by randomly dropping out a few neurons during the forward pass, and weights are not updated during the backward pass. Secondly, it enhances the generalization capability of the model. The proposed model uses four dropout layers with a dropout value set to 0.2 (20%) throughout the CNN model.

The global average pooling (GAP) layer is usually a substitute for the flatten and FC layer in the CNN model. The flatten layer transforms the input tensor of any shape to a 1-D vector, while the GAP layer is capable of reducing the spatial dimension to 1. Since the proposed model uses GAP layer instead of flatten layer, the model has high possibility of avoiding overfitting and reducing spatial image dimensions that speed up the training and inference processes. In the absence of GAP layer, a flatten layer would have resulted in $8\times8\times256=16{,}384$ vectors which means that the first layer in FC layer would require 16,384 neurons to process the input. This would have not only drastically increased the parameters, but also would have resulted in increased training time of the model. Thus, with GAP layer, the resulted vector is $1\times1\times256=256$, which shows a significant improvement in training and processing speeds.

A fully connected (FC) layer, also known as a dense layer, usually follows a flatten layer where each neuron in the FC layer connects to every neuron in the previous layer. The FC layer's output goes to the output layer with the SoftMax activation function that produces a multi-class probability distribution corresponding to the number of classes. The class that gets the highest probability will have the label corresponding to that class. The output layer uses four neurons corresponding to the grape leaves' four classes.

Experimental environment and setup

The process of model building, training, and evaluation uses the open-source framework Keras, which runs on TensorFlow 2.4.0. The entire process of developing the CNN model makes use of Google Colaboratory (Bisong 2019), Google's free cloud service that provides free GPUs to the researchers and developers for developing ML or DL based models. Apart from providing free GPU, Google Colaboratory comes with pre-installed popular libraries such as TensorFlow, PyTorch, Keras, and OpenCV.

Model performance and evaluation metrics

After successful model training and validation process, suitable metrics were identified to evaluate trained model's performance. Since the dataset used for building the model is highly imbalanced, merely having high accuracy is not

enough to evaluate the developed model's performance. Apart from accuracy, important metrics considered for model evaluation are Precision, Recall, F1-score, Receiver Operating Characteristics (ROC) curve, and Area under the Curve (AUC). Before diving more in-depth to understand these metrics, a prerequisite is to know the confusion matrix. Here, as the model deals with the imbalanced multi-classes, knowledge of the confusion matrix will simplify understanding other dependent metrics. If all the entries occupy only the diagonal elements and non-diagonal elements have zero entries in a confusion matrix, the model has an accuracy of 100%. The off-diagonal (image numbers) indicate the classifier's incapability in classifying images correctly.

1) *Accuracy:* overall model accuracy for a multi-class image classification problem is defined as the number of correctly classified images in the dataset to the total number of images, given by Eq. (1)

$$\text{Accuracy} = \frac{\text{Correctly classified images}}{\text{Total images}} \tag{1}$$

In terms of True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN), the accuracy for n classes can be defined in Eq. (2):

$$\text{Accuracy} = \frac{\sum_{i=1}^{n} \left(\text{TP}_i + \text{TN}_i\right)}{\sum_{i=1}^{n} \left(\text{TP}_i + \text{TN}_i + \text{FP}_i + \text{FN}_i\right)}. \tag{2}$$

2) *Precision:* precision for multi-class imbalanced dataset is defined as the ratio of sum of TP (all classes) to the sum of TP and FP (all classes) as is represented in Eq. (3)

$$\text{Precision} = \frac{\sum_{i=1}^{n} \text{TP}_i}{\sum_{i=1}^{n} \left(\text{TP}_i + \text{FP}_i\right)} \tag{3}$$

3) *Recall:* recall for the multi-class imbalanced dataset is defined as the ratio of the sum of TP (all classes) to the sum of TP and FN (all classes) and denoted as in Eq. (4)

$$\text{Recall} = \frac{\sum_{i=1}^{n} \text{TP}_i}{\sum_{i=1}^{n} \left(\text{TP}_i + \text{FN}_i\right)} \tag{4}$$

4) *F1 score:* F1 score for the multi-class imbalanced dataset is defined as twice the ratio of Precision and Recall's product to the sum of Precision and Recall represented as in Eq. (5):

$$\text{F1 Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5}$$

Or by Eq. (6):

$$\text{F1 Score} = \frac{2 \sum_{i=1}^{n} \text{TP}_i}{\sum_{i=1}^{n} \left(2\text{TP}_i + \text{FP}_i + \text{FN}_i\right)} \tag{6}$$

3) *Receiver Operating Characteristics (ROC) curve:* ROC curve is an evaluation metric that evaluates a binary classifier's performance. For multi-class classification, the ROC curve uses the One-verses-All (OvA) scheme that involves a plot of True Positive Rate (TPR) against False Positive Rate (FPR) at various threshold levels.

6) *Area Under the Curve (AUC)* The AUC conveys a classifier's ability to distinguish between classes and serves as a summary of the ROC curve.

Results

After successful training of the model, the results were evaluated to validate the performance of the model. The results obtained are represented in terms of classification results (training and testing performance visualization, confusion matrix, top predictions, worst predictions, ROC and AUC plots) and CNN visualization.

Training and testing performance visualization

During training, Keras Callback API was used to provide three functionalities to ease the model training process that includes early stopping (in case if validation loss does not improve), Check pointer (saving model to drive), and reduce LR (learning Rate) (reducing learning rate if no improvements are observed). The model learning was configured with the optimizer as Adam, with a loss function of categorical cross-entropy, training metric as accuracy, and the learning rate set to a value of 0.001. The model training was set for 300 epochs, but the training stopped after 154 epochs due to early stopping.

Model training and validation accuracy plots are represented in (Fig. 2a), while model losses are shown in (Fig. 2b), respectively. During model training, the model reached the highest validation accuracy of 99.34% at the end of the 154th epoch. Similarly, the validation loss reaches around 0.0914 after reaching 154 epochs. The plots indicate no sign of overfitting as the validation accuracy closely follows the training accuracy throughout.

Confusion matrix

The confusion matrix shown in (Fig. 3) helps in visualizing the overall model's performance. The entries in the confusion matrix might belong to any one of the categories (TP, TN, FP, and FN). The diagonal entries of the matrix

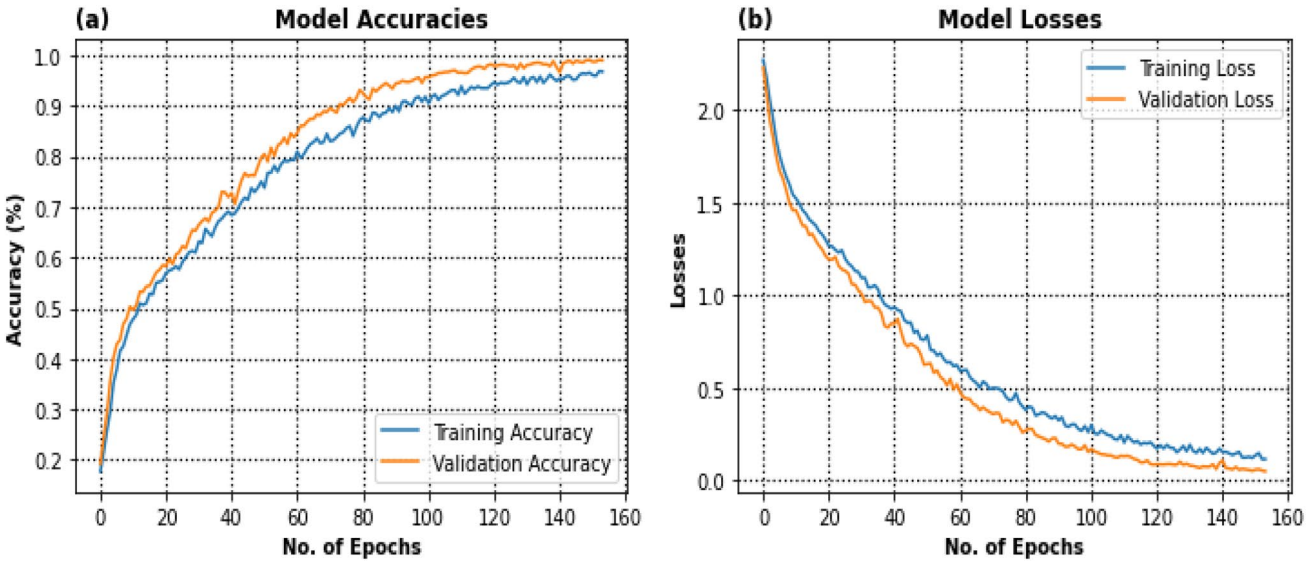

Fig. 2 Model training and validation plots trained for 154 epochs with Keras Callback API with a learning rate of 0.001 shows the increase in accuracy and decrease in loss per epoch **a** accuracy plot **b** loss plot

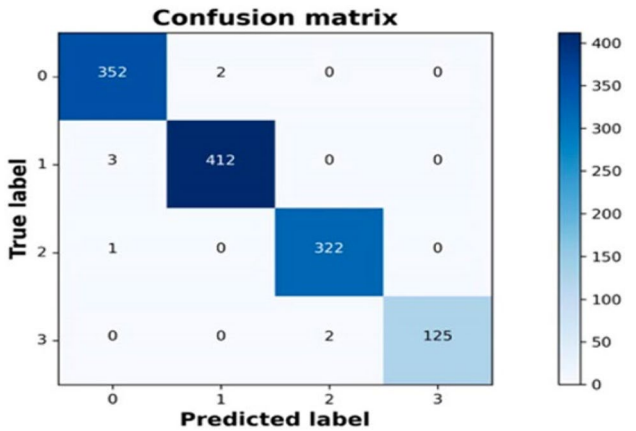

Fig. 3 A 4×4 Confusion Matrix indicative of model performance in classifying images into four classes

correspond to the TP's (correctly predicted) for each class. For example, the TP for class 0 is 352, for class 1, it is 412, while for class 2, the TP value is 322, and finally, for class 3, it has a value of 125. The observation reveals that the model's best performance in terms of accuracy resulted for class 2 (leaf blight), where out of the total 323 total test images, 322 were correctly classified, giving an accuracy of 99.69%. Also, for class 3 (healthy), the model's accuracy was lowest at 98.42%, while class 0 (black rot) and class 1 (black measles) obtained an accuracy of 99.43% and 99.27%, respectively. The overall validation accuracy for the combined classes was 99.34%.

Top predictions

The models' best five predictions are shown in (Fig. 4), which illustrates the model's predicted label and the true labels. The values in brackets indicate the class probability obtained using the SoftMax function at the output layer of the CNN model.

Worst predictions (false classification)

The confusion matrix of (Fig. 3) provides the per-class accuracy attained by the model in classifying images and indicates its misclassification where the model failed to identify and differentiate between the classes. The off-diagonal entries indicate the model's inability to classify the images into correct classes (FN and FP). The off-diagonal column entries correspond to FPs, while the off-diagonal row entries indicate the FNs. For example, for class 0 (column 1), the column entries 3 and 1 indicate that the 3 images belonging to class 1 and 1 image belonging to class 2 were falsely predicted to be belonging to class 0. Hence, the total FPs for class 0 is $(3+1+0=4)$. Similarly, for class 1 (column 2), entry 2 is an FP which shows that 2 images belonging to class 0 were mis-predicted to be of class 1. Hence, total FPs for class 1 are $(2+0+0=2)$., for class 2 (column 3), entry 2 is an FP that indicates that 2 images belonging to class 2 were mis-predicted to be of class 3. Hence, the total FPs for class 2 is $(0+0+2=2)$, and for class 3 (column 4), there are no entries, indicating no FPs for class 3. To obtain FNs for class 0, consider first row entries. Here entry 2 means that 2 images belonging to class 0 were predicted to be belonging to class 1.

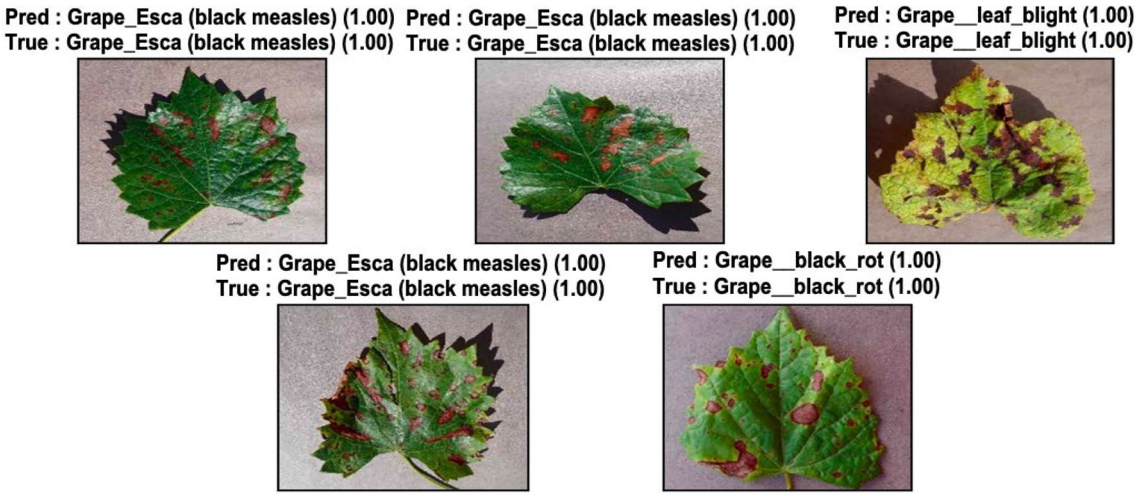

Fig. 4 Top five predictions by the proposed model with class probabilities indicated in brackets

Therefore, class 0 has 2 FNs. Likewise, for classes 1, 2, and 3, the FNs are 3, 1, and 2, respectively. As a result, the total FNs for class 0 is $(2+0+0=2)$, for class 1 is $(3+0+0=3)$, for class 2 is $(1+0+0=1)$, and for class 3, it is $(0+0+2=2)$. The sum of all the off-diagonal entries is equal to 8 $(2+3+1+2)$. Hence, a total of eight predictions went wrong, as indicated in (Fig. 5). The model wrongly predicted two black rot images to be black measles with a probability of 0.58 and 0.97 (FPs), respectively. Similarly, three black measles images were misclassified and predicted as black rot with probabilities of 0.87, 0.76, and 1.00 (FPs), respectively. One image of leaf blight was wrongly predicted as black rot with a probability of 1.00 (FP), and two images of the healthy class were predicted to be leaf blight with probabilities as 0.78 and 0.85 (FPs).

Receiver Operating Characteristics (ROC) and Area Under the Curve (AUC)

The last performance metric evaluated for the model is ROC curves and AUC as shown in (Fig. 6). The ROC curve resemble the ideal ROC curve for the combined classes. The AUC score of 1 indicates the best possible results that an image classifier model can obtain.

The developed CNN model was not only trained and evaluated as an individual custom model. Transfer learning with pre-trained models yielded new models with base layers intact, and the output layer was replaced with a dense layer having four neurons.

Since accuracy alone cannot justify the model's performance when dealing with imbalanced classes, the evaluation includes precision, recall, and F1 score. The precision,

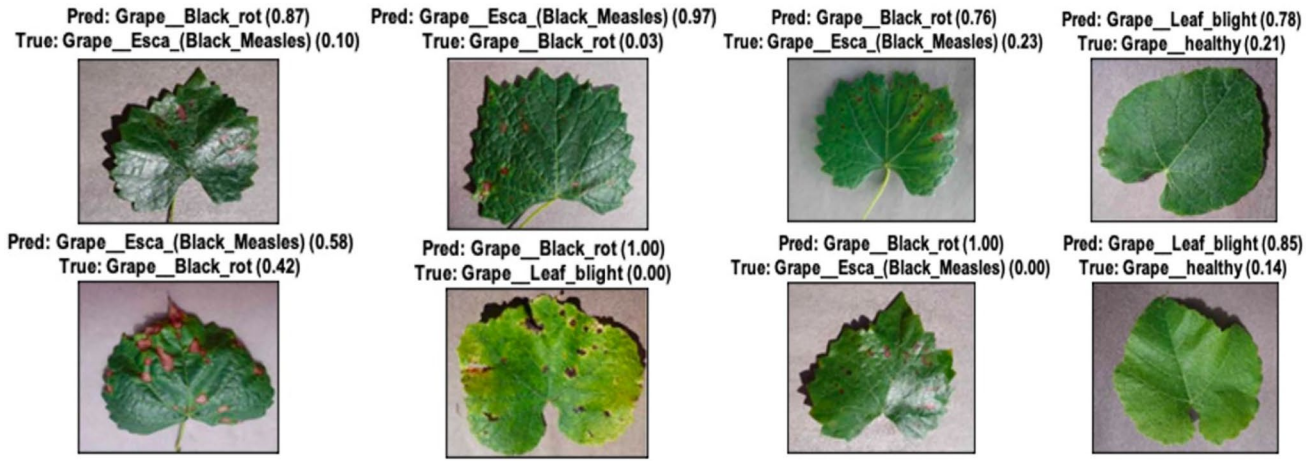

Fig. 5 Worst predictions by the model in terms of False Positives (FP) and False Negatives (FN)

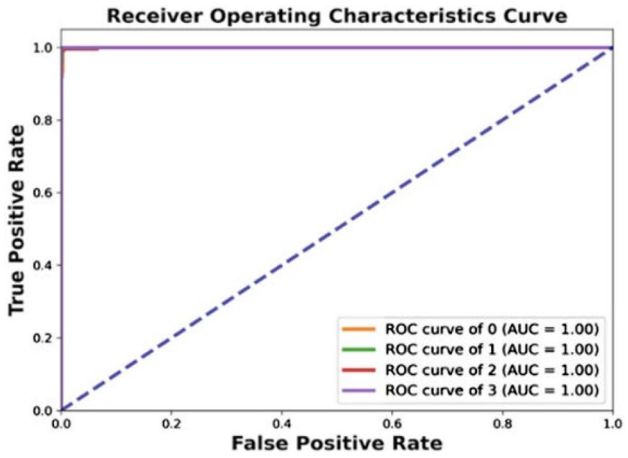

Fig. 6 Combined ROC curve plots for the four classes of grape RGB leaves under different threshold values

recall, and F1 score for the model were the same, with a value of 0.9934.

Two important concepts in developing disease detection CNN model were explored. In the first concept, the CNN model was built from scratch, which would be beneficial if the model has to be trained on custom datasets, where pre-trained models are not available. On the other hand, model building from scratch would require more time training and also computational power, but once trained, can provide good performance in terms of accuracy and generalization. As indicated in the (Table 2), a custom CNN model trained from scratch provides results comparable to the results obtained from other state of the art models trained using transfer learning. Some of the existing pre-trained models are InceptionV3, ResNet50, VGG16, VGG19 and Xception. All these models are pre-trained on ImageNet and were trained for 50 epochs for evaluation and comparison. Table 2 highlights a class-wise comparison of all the pre-trained models with custom CNN. The comparison is based on four performance metrics accuracy, precision, recall and F1 score. The best results were obtained for class 2 (Leaf Blight) both for transfer learning and custom CNN models. The accuracy obtained for pre-trained models was 100%, while the custom model yielded an accuracy of 99.69%, very close to the accuracy obtained from transfer learning (100%). In the evaluation of transfer learning models, for class 0 (Black Rot), the highest value of accuracy was obtained by VGG19 (0.9972), while the lowest value was obtained by ResNet50 (0.9859). The custom CNN model yielded an accuracy value of 0.9943, close to the highest value of 0.9972. Likewise, the metrics values of precision, recall, and F1 score for other classes are indicated in detail using (Table 2).

It can be observed that the transfer learning models are capable of performing better when compared to the custom CNN model, but a stringent requirement for using a pre-trained model is that the image dataset used to train the model should exactly match with the dataset on which the pre-trained model is trained. If the datasets do not match, it might result in negative transfer degrading the model's model performance. Another issue that might arise while using the transfer learning model is that the model may overfit the limited dataset to which the model is trained. These limitations of transfer learning might be avoided using

Table 2 Class-wise metric evaluation for pretrained and custom CNN models

Classes	Performance metrics	Pretrained models (transfer learning)					Proposed CNN model	Support
		InceptionV3	ResNet50	VGG16	VGG19	Xception		
Black Rot (0)	Accuracy	0.9943	0.9859	0.9661	0.9972	0.9943	0.9943	354
	Precision	1.0000	1.0000	0.9913	0.9916	0.9972	0.9888	
	Recall	0.9943	0.9859	0.9661	0.9972	0.9943	0.9943	
	F1 score	0.9972	0.9929	0.9785	0.9944	0.9957	0.9915	
ESCA (Black Measles) (1)	Accuracy	1.0000	1.0000	0.9904	0.9928	0.9976	0.9927	415
	Precision	0.9952	0.9904	0.9880	0.9976	0.9952	0.9952	
	Recall	1.0000	1.0000	0.9904	0.9928	0.9976	0.9928	
	F1 score	0.9976	0.9952	0.9892	0.9952	0.9964	0.9940	
Leaf Blight (2)	Accuracy	1.0000	1.0000	1.0000	1.0000	1.0000	0.9969	323
	Precision	1.0000	1.0000	0.9844	1.0000	1.0000	0.9938	
	Recall	1.0000	1.0000	0.9921	1.0000	1.0000	0.9969	
	F1 score	1.0000	1.0000	0.9893	1.0000	1.0000	0.9954	
Healthy (3)	Accuracy	0.9984	0.9959	0.9921	1.0000	1.0000	0.9842	127
	Precision	1.0000	0.9922	0.9844	1.0000	1.0000	1.0000	
	Recall	1.0000	1.0000	0.9921	1.0000	1.0000	0.9842	
	F1 score	1.0000	0.9961	0.9893	1.0000	1.0000	0.9921	

custom training of the CNN model from scratch. The overall performance of the custom CNN model with the pre-trained models is as indicated in Table 3. The macro averaged values of Precision, Recall, and F1 score is indicated in which all classes (0 through 3) are considered to contribute to the final average of the metric equally. In the weighted averaged scenario, the contribution of each class to the average is weighted by each class's size. It is evident from the comparison that the overall performance of the custom model is close to that obtained from transfer learning while improving the generalizable capability of the model without overfitting the data.

CNN filter visualizations

No doubt, machine learning (ML) and deep learning (DL) models can simplify the complicated problems involving object detection, localization, identification, and classification. However, can we understand how the models can do this? It is not easy to understand how a particular function gets modelled while training the Deep Neural Network (DNN) model. ML and DL model's complex nature has given them the name "black boxes". In other words, the models are complicated to interpret as the number of features grows. Various techniques are in use to understand what exactly happens during model training. One technique widely used to visualize and interpret CNNs is filter or kernel visualization. This visualization helps to understand how the image information passes from one layer to another and how basic features get converted into classes at CNN's output.

Figure 7 shows the filter visualization only for the first eight filters corresponding to each layer for simplicity.

The CNN model consists of four blocks, out of which the first three blocks have a repeating sequence of two convolution layers followed by the max- pooling and finally, the dropout layers. Block 1 filters are responsible for providing clear visualizations that are highly interpretable without any noisy patterns. Like filters 3 and 7 of the first convolution layer, some filters indicate that the filters are not activated. The first convolutional layer filters almost retain the input image (as the grape leaf image is visible).

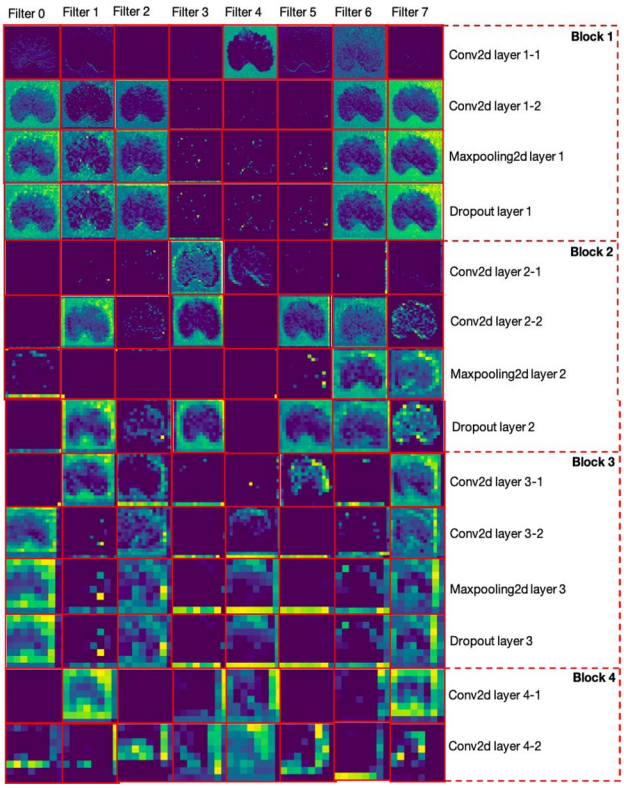

Fig. 7 Visualizations of first eight filters of CNN model corresponding to four blocks: Block 1 through Block 4

The other filters in the first convolutional layer try to find the edges and colours in the image. The max-pooling layer reduces the image dimension by 2, resulting in a significant decrease in the computation time due to a reduction in the image dimension. The third layer is the dropout layer, which randomly drops a few neurons while training to avoid

Table 3 Custom model performance comparison with pre-trained (transfer learning) models

Performance metrics	Pretrained models (transfer learning)					Proposed CNN model
	InceptionV3	ResNet50	VGG16	VGG19	Xception	
Accuracy	0.9983	0.9959	0.9860	0.9967	0.9975	0.9934
Precision						
Macro	0.9988	0.9957	0.9856	0.9973	0.9981	0.9944
Weighted	0.9984	0.9959	0.9861	0.9967	0.9975	0.9934
Recall						
Macro	0.9986	0.9965	0.9871	0.9975	0.9980	0.9921
Weighted	0.9983	0.9959	0.9860	0.9967	0.9975	0.9934
F1 score						
Macro	0.9987	0.9960	0.9863	0.9974	0.9980	0.9932
Weighted	0.9983	0.9959	0.9860	0.9967	0.9975	0.9934

overfitting. A close observation shows that when the convolutional layers go deeper from block 2 to block 4, the prominent activations visible in Block 1 become very abstract, and image visualization is impossible.

The visualization complexity shows up because the filters tend to look for high-level features such as corners, borders, and angles when going more in-depth.

These high-level features help classify the images into the corresponding classes or labels rather than looking for low-level features as Block 1 filters do. The output layer uses the SoftMax activation function with four neurons corresponding to four classes of grape leaves.

Discussion

After evaluation of the trained CNN model, it showed some outstanding results corresponding to the evaluation metrics. Similar research on deep learning models showed model evaluation based on a limited set of metrics of which accuracy is one. When dealing with balanced datasets (each class has an equal number of images), accuracy can be considered for evaluating model performance. Hence, in building robust models, the metrics like precision, recall, and F1 score is essential. The CNN model for grape disease identification and detection used two approaches, one with transfer learning, and the other approach showed how the CNN model could be built starting from scratch.

In literature, similar implementations for crop disease detection use either a transfer learning approach or build a model from scratch. Both these approaches come with their advantages and limitations. This research describes an implementation of both these approaches and provides a detailed comparison between them in terms of model performance, considering metrics that genuinely define the model performance. The dataset greatly influences the model selection for the disease classification. Some of the state-of-the-art implementations use public image datasets like PlantVillage Dataset. Most of the research either uses an entire dataset or part of it, depending upon the problem.

A transfer learning approach based on the RestNet50 pre-trained model (Mukti and Biswas 2019) was developed utilizing all the 38 classes of the PlantVillage dataset to classify leaf images into healthy or diseased classes and obtained an accuracy of 99.80%. The research (Rao et al. 2021) used a subset of the PlantVillage dataset to detect diseases corresponding to the mango and gapes. They built the CNN model using a pre-trained AlexNet model and obtained 99% and 89% accuracy for grapes and mango leaves, respectively. Similarly Sagar and Dheeba (2020) used the same dataset with 19 classes of diseased and healthy leaves to develop a disease classification system based on pre-trained models like VGG16, ResNet50, InceptionV3, InceptionResNet, and DenseNet16. The results obtained with four metrics were accuracy of 98.20%, the precision of 0.94, recall of 0.94, and F1 score of 0.94. Another research utilizing the PlantVillage dataset with four classes of grape leaves (Ji et al. 2020) obtained 99.17%, 0.9905, 0.9888, and 0.9896, corresponding to accuracy, precision, recall, and F1 score for a United Model, respectively. A custom CNN model by Trivedi et al. (2020) used an entire PlantVillage dataset having 54,305 images corresponding to 38 different classes of diseased and healthy leaves and obtained an accuracy of 95.81%. The experimentation by Hassan et al. (2021) used the entire PlantVillage dataset in three different forms coloured images, segmented images, and grayscale images. Trained InceptionV3, InceptionResNetV2, MobileNetV2, and EfficientNetB0 pre-trained models to achieve accuracies of 98.42%, 99.11%, 97.02%, and 99.56%, respectively. Research of Islam et al. (2019) used 3 classes of potatoes (2 diseased and 1 healthy) from the PlantVillage dataset to train two models, one based on sequential CNN model and the other using a transfer learning approach. The CNN model obtained an accuracy of 86.31% for the test set, while the transfer learning model achieved an accuracy of 99.43% on the testing set. A custom CNN model (Militante et al. 2019) used 32 classes from the PlantVillage dataset for disease detection and obtained an accuracy as high as 96.5%. A fruit classification and disease grading system (Nikhitha et al. 2019) using a transfer learning approach (using InceptionV3 model) was proposed using Fruits 360 (fruit classification dataset).

Some implementations developed disease classification models by developing their custom datasets. A transfer learning approach was developed to classify diseases into five classes by combining DenseNet and Xception pre-trained models (Chao et al. 2020) on a custom dataset consisting of Apple images. The model achieved an overall classification accuracy of 98.82%. The research work (Kusrini et al. 2020) developed a deep learning model (modified VGG16) to detect diseases caused by pests on some part of the mango tree, including leaf, stem, root, or fruit pertaining to16 classes. The model obtained 73% and 76% accuracy on validation and testing sets, respectively. A hybrid approach (Singh et al. 2019) (using PlantVillage dataset + Custom dataset) was utilized to develop a model based on AlexNet pre-trained model to classify mango (custom) and other diseases of leaves (PlantVillage). The model achieved an accuracy of 97.13%, corresponding to four classes of leaves. Another similar hybrid approach was proposed by Arya and Singh (2019) to detect diseases corresponding to mango and potato leaves. The mango leaves were obtained from a field located at Pantnagar. Two models were developed and compared, the first custom CNN model and the other based on AlexNet pre-trained model. The CNN-based custom model achieved an overall accuracy of 90.85%, while the AlexNet

model obtained 98.33%. An SVM (Support Vector Machine) based approach (Mia et al. 2020) developed a model for mango leaf disease detection that could detect mango leaf diseases corresponding to five classes (including healthy leaves). This approach achieved an average classification accuracy of 80%. Another CNN model (Jadhav et al. 2021) based on AlexNet and GoogleNet pre-trained models was developed to identify and classify disease of soybean leaves into four classes including healthy leaves. The custom dataset was obtained from soybean fields in Kolhapur district. The models were able to achieve an accuracy of 98.75% and 96.25% for AlexNet and GoogleNet models, respectively.

Referring to the aforementioned state-of-the-art implementations, it is evident that the proposed approach of training a custom CNN model from scratch obtained results that outperform some of the models implemented to classify diseases of crops using the transfer learning approach. Also, the performance metrics used to test the developed model were based on the imbalanced nature of the dataset. These metrics provide critical analysis of the model and ease fine-tuning before the model is deployed in real-time situations like the agricultural field. Thus, the developed custom CNN model provides the performance that can typically be obtained only by transfer learning approach, but at the same time uses less number of trainable parameters that improve inference time drastically in real-time situations.

Conclusion and future scope

The developed CCN model for early detection of diseases of the grapes uses the self-feature extracting property inherent in CNNs, circumventing using a classifier that follows a DNN stage, simplifying the design requirements and complexity. The proposed method develops a highly generalizable CNN model for disease detection and classification and introduces the concept of transfer learning by training five models (Inception V3, ResNet50, VGG16, VGG19, and Xception) and thereby comparing them with the custom model.

The key concepts such as data pre-processing, data augmentation, feature visualization, performance metrics selection, and hyper parameter tuning lay the foundation in building models with high generalization capability. The model evaluation's performance parameters are based on the dataset's imbalanced nature (multi-class with varying number of images in each class) as in the real-time dataset. Also, the model's conversion to TensorFlow light version TensorFlow tflite format makes the model readily deployable on to mobile devices to provide real-time disease identification.

The knowledge gained can further be applied in understanding how the concept of transfer learning in pre-trained models can help develop classifiers capable of providing very high classification accuracy while using low computational resources. The future work would incorporate a few more classes corresponding to other crops like potato and pepper with possible model deployment to provide a real-time solution to the farmers right from the agricultural field.

Data availability The data used for the study can be obtained from the corresponding author on request.

Declarations

Conflict of interest The authors declare that they have no conflict of interest for the manuscript and approve it for the publication.

References

Ahmad B, Raina A, Khan S (2019) Impact of biotic and abiotic stresses on plants, and their responses. In: Wani SH (ed) Disease resistance in crop plants: molecular, genetic and genomic perspectives. Springer, pp 1–19. https://doi.org/10.1007/978-3-030-20728-1_1

Arya S, Singh R (2019) A comparative study of CNN and AlexNet for detection of disease in potato and mango leaf. In: 2019 International conference on issues and challenges in intelligent computing techniques (ICICT), vol 1, pp 1–6. https://doi.org/10.1109/ICICT46931.2019.8977648

Bhatia GS, Ahuja P, Chaudhari D, Paratkar S, Patil A (2020) Plant disease detection using deep learning. In: Smys S, Senjyu T, Lafata P (eds) Second international conference on computer networks and communication technologies. Springer, pp 408–415. https://doi.org/10.1007/978-3-030-37051-0_47

Bisong E (2019) Google colaboratory. In: Bisong E (ed) Building machine learning and deep learning models on google cloud platform: a comprehensive guide for beginners. Apress, pp 59–64. https://doi.org/10.1007/978-1-4842-4470-8_7

Chao X, Sun G, Zhao H, Li M, He D (2020) Identification of apple tree leaf diseases based on deep learning models. Symmetry 12(7):1065. https://doi.org/10.3390/sym12071065

Dasig DD (2020) Implementing IoT and wireless sensor networks for precision agriculture. In: Pattnaik PK, Kumar R, Pal S (eds) Internet of Things and analytics for agriculture, vol 2. Springer, pp 23–44. https://doi.org/10.1007/978-981-15-0663-5_2

Hassan SM, Maji AK, Jasiński M, Leonowicz Z, Jasińska E (2021) Identification of plant-leaf diseases using CNN and transfer-learning approach. Electronics 10(12):1388. https://doi.org/10.3390/electronics10121388

Hatfield JL, Antle J, Garrett KA, Izaurralde RC, Mader T, Marshall E, Nearing M, Philip Robertson G, Ziska L (2020) Indicators of climate change in agricultural systems. Clim Change 163(4):1719–1732. https://doi.org/10.1007/s10584-018-2222-2

Islam F, Hoq MN, Rahman CM (2019) Application of transfer learning to detect potato disease from leaf image. In: 2019 IEEE international conference on robotics, automation, artificial-intelligence and Internet-of-Things, RAAICON 2019. https://doi.org/10.1109/RAAICON48939.2019.53

Jadhav SB, Udupi VR, Patil SB (2021) Identification of plant diseases using convolutional neural networks. Int J Inf Technol 13(6):2461–2470. https://doi.org/10.1007/s41870-020-00437-5

Ji M, Zhang L, Wu Q (2020) Automatic grape leaf diseases identification via UnitedModel based on multiple convolutional neural

networks. Inf Process Agric 7(3):418–426. https://doi.org/10.1016/j.inpa.2019.10.003
Kempenaar C, Been T, Booij J, van Evert F, Michielsen JM, Kocks C (2017) Advances in variable rate technology application in potato in The Netherlands. Potato Res. https://doi.org/10.1007/s11540-018-9357-4
Keras: The Python deep learning API (2021) Retrieved 15 Jan 2022. https://www.keras.io. Accessed 15 Jan 2021
Kusrini K, Suputa S, Setyanto A, Agastya IMA, Priantoro H, Chandramouli K, Izquierdo E (2020) Data augmentation for automated pest classification in Mango farms. Comput Electron Agric 179:105842. https://doi.org/10.1016/j.compag.2020.105842
Mia MdR, Roy S, Das SK, Rahman MdA (2020) Mango leaf disease recognition using neural network and support vector machine. Iran J Comput Sci 3(3):185–193. https://doi.org/10.1007/s42044-020-00057-z
Militante SV, Gerardo BD, Dionisio NV (2019) Plant leaf detection and disease recognition using deep learning. In: 2019 IEEE Eurasia conference on IOT, communication and engineering (ECICE), pp 579–582. https://doi.org/10.1109/ECICE47484.2019.8942686
Mohanty SP, Hughes DP, Salathé M (2016) Using deep learning for image-based plant disease detection. Front Plant Sci. https://doi.org/10.3389/fpls.2016.01419
Mukti IZ, Biswas D (2019) Transfer learning based plant diseases detection using ResNet50. In: 2019 4th International conference on electrical information and communication technology (EICT), pp 1–6. https://doi.org/10.1109/EICT48899.2019.9068805
Nikhitha M, Sri SR, Maheswari BU (2019) Fruit recognition and grade of disease detection using inception V3 model. In: Proceedings of the 3rd international conference on electronics and communication and aerospace technology, ICECA 2019. https://doi.org/10.1109/ICECA.2019.8822095
Patrício DI, Rieder R (2018) Computer vision and artificial intelligence in precision agriculture for grain crops: a systematic review. Comput Electron Agric 153:69–81. https://doi.org/10.1016/j.compag.2018.08.001
Rahman K, Zhang D (2018) Effects of fertilizer broadcasting on the excessive use of inorganic fertilizers and environmental sustainability. Sustainability 10(3):759. https://doi.org/10.3390/su10030759
Rao US, Swathi R, Sanjana V, Arpitha L, Chandrasekhar K, Naik PK (2021) Deep learning precision farming: grapes and mango leaf disease detection by transfer learning. Global Transit Proc 2(2):535–544. https://doi.org/10.1016/j.gltp.2021.08.002
Sagar A, Dheeba J (2020) On using transfer learning for plant disease detection. BioRxiv. https://doi.org/10.1101/2020.05.22.110957
Shastry KA, Sanjay HA (2020) Data analysis and prediction using big data analytics in agriculture. In: Pattnaik PK, Kumar R, Pal S (eds) Internet of things and analytics for agriculture, vol 2. Springer, pp 201–224. https://doi.org/10.1007/978-981-15-0663-5_10
Singh UP, Chouhan SS, Jain S, Jain S (2019) Multilayer convolution neural network for the classification of mango leaves infected by anthracnose disease. IEEE Access 7:43721–43729. https://doi.org/10.1109/ACCESS.2019.2907383
Thangaraj R, Anandamurugan S, Kaliappan VK (2021) Automated tomato leaf disease classification using transfer learning-based deep convolution neural network. J Plant Dis Prot 128(1):73–86. https://doi.org/10.1007/s41348-020-00403-0
Tian H, Wang T, Liu Y, Qiao X, Li Y (2020) Computer vision technology in agricultural automation—a review. Inf Process Agric 7(1):1–19. https://doi.org/10.1016/j.inpa.2019.09.006
Trivedi J, Shamnani Y, Gajjar R (2020) Plant leaf disease detection using machine learning. In: Gupta S, Sarvaiya JN (eds) Emerging technology trends in electronics, communication and networking, vol 1214. Springer, pp 267–276. https://doi.org/10.1007/978-981-15-7219-7_23
Tsouros DC, Bibi S, Sarigiannidis PG (2019) A review on UAV-based applications for precision agriculture. Information 10(11):349. https://doi.org/10.3390/info10110349
Yang C (2020) Remote sensing and precision agriculture technologies for crop disease detection and management with a practical application example. Engineering 6(5):528–532. https://doi.org/10.1016/j.eng.2019.10.015