Journal of Scientific & Industrial Research
Vol. 82, May 2023, pp. 536-545
DOI: 10.56042/jsir.v82i05.1089

Hyperparameter Optimization for Transfer Learning-based Disease Detection in Cassava Plants

Kalyani G[1]*, Sai Sudheer K[2], Janakiramaiah B[3] & Narendra Kumar Rao B[4]

[1]Velagapudi Ramakrishna Siddhartha Engineering College, Vijayawada 520 007, Andhra Pradesh, India

[2]Tejas Networks, Bengaluru 560 100, India

[3]Prasad V. Potluri Siddhartha Institute of Technology, Vijayawada 520 007, Andhra Pradesh, India

[4]School of computing, Mohan Babu University, Tirupati 517102, Andhra Pradesh, India

Received 29 August 2022; revised 03 October 2022; accepted 07 October 2022

Cassava is quite possibly the most widely recognized staple food crop. It is a nutty-flavored, starchy root vegetable that is a primary energy source and carbs for individuals. During crop cultivation, cassava plant infections can influence the leaf and root, bringing about a tremendous loss to the harvest and financial market esteem. Hence, it is vital to detect diseases in cassava plants. But it requires enormous labor, longer time planning, and thorough plant-specific knowledge. If disease detection is possible at the initial stages, then actions can be taken on time. Hence, there is a need to develop automatic detection methods for monitoring different parts of cassava plants. This study evaluates the efficiency of applying transfer learning to the pre-trained models for identifying diseases in cassava plants. The pre-trained EfficientNet model detects the disorders using data augmentation, fine-tuning the hyperparameters, cross-validation, and transfer learning. The experimentation is done with the cassava dataset provided by Kaggle, which contains cassava plant leaf images belonging to five classes. An experimental investigation shows that EfficientNet with transfer learning attains up to 89% accuracy. The effect of transfer learning is significant; consider getting the results of high accuracy and less dispersion; in very few cases, the model forecasts the wrong class labels. The outcomes give a promising strength to the objective of this work, i.e., a model trained explicitly for agriculture with transfer learning can assist the farmers with highly accurate results during farming to get a high yield.

Keywords: Cassava leaf diseases, Deep learning, EfficientNet, Plant disease detection, Precision agriculture

Introduction

India is an agronomic country whose economy largely depends on the agriculture sector. Agriculture accounts for 16% of India's Gross Domestic Product (GDP) and 10% of its trade. About 75% of India relies upon the farming domain either straightforwardly or in a roundabout way.[1] Henceforth, Agriculture has been a significant wellspring of economic development in India. Because of the rising populace, climate changes, and political vulnerability, farming enterprises started to search for new strategies to enhance the quantity of food produced, which permits specialists to search for novel high-efficiency advancements that are successful and precise. The farmer chooses the necessary harvest depending on the dirt sort, the climate condition of an area, and financial worth. Using precise agriculture with technological advances, farmers may gather the information to decide the correct choice for high production. Precise Agriculture (PA) is a cutting-edge innovation that provides advanced procedures for enhancing production in farming. By utilizing these refined advances, Economic improvement in agriculture can be accomplished. PA can be used in different applications, for example, identifying plant bugs, weed identification, crop yield estimation, disease identification in plants, etc. Farmers generally use pesticides to control pests and diseases and increase crop yield. Harvest infections are messing-up farmers with low yields and financial misfortunes. Along these lines, disease detection and estimating its seriousness should be characterized as suitable.[2]

With the advancement of computational frameworks, specifically Graphical Processing Units (GPU) installed processors and Machine Learning associated Artificial Intelligence applications have accomplished outstanding development, prompting the improvement of novel systems and models, and Deep Learning.[3] The computationally plausible deep learning models have reformed areas, for example,

*Author for Correspondence
E-mail: kalyanichandrak@gmail.com

image recognition, speech recognition, and some complex applications which involve the investigation of vast volumes of data, giving ample scope to the applications such as self-driving vehicles and translation. Applying these deep learning methods to the agriculture domain, precisely in the arena of plant disease determination, has been ongoing over the recent years and to a slightly restricted degree.[4–9] Numerous global scientists introduced diverse cutting-edge frameworks for automatically detecting plant diseases with different methods like Machine Learning and Deep Learning.[10–19]

In Sub-Saharan Africa, cassava is possibly the most well-known vital food crop. Greenery and starchy roots are frequently ingestible plant components. The roots are usually burned-through because they are an essential source of energy that can be consumed as crude, bakedin a single coal oven, and cooked or handled in various methods for consumption.[20,21] The greeneries and delicate sprouts are good sources of proteins and nutrients; hence, they are used as vegetables in numerous districts.[22] African farmers are developing cassava in tiny, medium, and enormous scopes in extensive coverage of ecological and weather situations to contribute to food production and industrial harvest. However, the significant issue is cassava plants are easily affected by various diseases. Sometimes the cure to those diseases is to burn the plants to stop the further spread whenit is not identified at the early stages. It indicates a need for better-computerized methodologies that can support farmers in locating cassava diseases at early stages and counter acting them.

For the cassava image dataset, transfer learning was performed. Therefore, this paper employs transfer learning to retrain an Efficient Net, a pre-trained Convolutional Neural Network (CNN) model, with an ImageNet dataset. The core objectives of this work are listed below:

- Classifying cassava leaf diseases with Deep learning methods (retrained EfficientNet) to determine whether the cassava plant is healthy or suffers from any disease.
- For the classification of disease, 5-fold cross-validation is used. The folds are randomly selected, comprising almost equal proportions of the five different class labels.
- After training, if new images are submitted as input to the system, it forecasts the disease category in the early stage. It supports taking corresponding measures to reduce its effect on the plant.
- The agriculture sector is one of the critical domains of the nation, and this work supports improving the revenue from the field by recognizing infected plants at the initial stages.

Analysis of Existing Works

In farming, diseases in the plant cause a decrease in the yield bringing about monetary misfortune. Different researchers have projected various strategies to put beware of these contaminations. Some analysts have been using deep learning approaches such as CNNs to identify the characteristics of diseases and categorize disorders. A model based on Faster Region-based CNN, Region-based Fully CNN, and Single Shot Detector for locating plant diseases.[23–26] Their dataset comprises pictures of fruit & vegetable harvests and cereal yields. The authors apply data augmentation techniques like rotations, contrast enhancement, viewpoint change, and relative change. Adhikari *et al*. projected a methodology for automatically identifying plant diseases for tomato plants, such as late blight, grey spots, and bacterial canker.[27] The deep learning model created 24 convolutional layers and two eventually coupled layers to do this. The model attains a precision of 89% on the standard dataset called Plant Village and 76% on their private dataset. The model is planned using the familiar YOLO model.[28] Karthik *et al.* introduced two deep models validated with the Plant Village dataset to distinguish three types of infections in tomato plants. In the model design, feed-forward CNN is used, followed by CNN with the instrument and leftover learning.[29] The consideration-based CNN design introduced a remarkable precision of 98%. To discriminate ten classes (nine diseases and one solid) in tomato plants, Agarwal *et al.* designed a CNN with three convolutional, three max-pooling, and two related layers that had a general precision of 91.2%.[(30)]

To diagnose nine pathogens in tomato plants, a novel portable program using MobileNet has been considered an efficient application.[31–32] The application used 7,176 images of tomato leaves from the Plant Village dataset with a 90.3% accuracy rate. A CNN model was created by Widiyanto *et al.* to distinguish between healthy leaves and four tomato plant illnesses, including septoria leaf spot, late blight, yellow leaf twist infections, and mosaic disease.[33] The created model trained on 1,000 pictures from the Plant Village dataset for every class. The classification model has attained 96.6% accuracy for five classes. Disease identification in cassava plants dependent on computerized picture acknowledgment through feature extraction has shown promising outcomes.[34–36] Yet, feature extraction is computationally expensive and requires domain knowledge to achieve high performance. The reviewed works on plant disease detection is shown in Table 1.

Although there has been a large amount of development over the past few years, still specific gaps are detailed here:

- The Plant Village dataset was typically utilized in studies to assess the efficacy and performance of the corresponding DL architectures.
- Since the severity of plant diseases varies over time, DL models should be enhanced or adjusted to allow for the recognition and classification of diseases throughout their entire sequence of existence.
- The DL architecture needs to be effective under various lighting circumstances; therefore, the datasets must also include pictures taken under different field conditions.

A plant-specific dataset was used to retrain the EfficientNet model to address the gaps. The reason to select the EfficientNet model is that the model can succeed with greater efficiency and accuracy than current CNNs, lowering parameter size and FLOPS by a factor of ten.

Materials and Methods

Dataset Description

Images used for the experiment were available on Kaggle.[45] The dataset consisted of five cassava leaf classes and 10,000 labeled images collected during a formal review in Uganda. Farmers who took photographs of their nurseries contributed most of the images. Experts tagged the pictures at the National Crops Resources Research Institute and the Artificial Intelligence lab at Makerere University, Kampala. For practical training of the model, images need to be preprocessed to enhance the clarity of the images, and the class imbalance shown in Fig. 1 is also to be

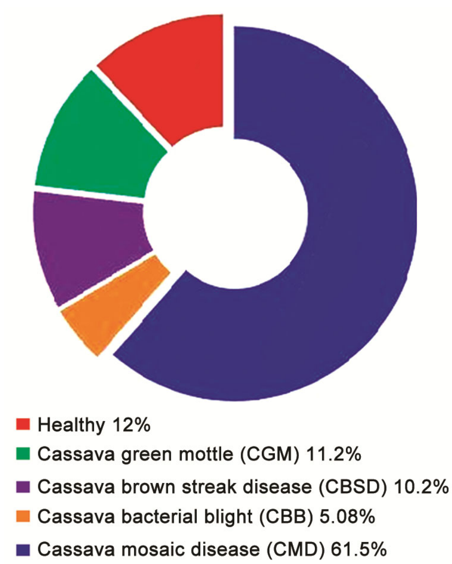

Fig. 1 — Distribution of the number of instances per class

Table 1 — Remarks of the reviewed works on plant disease detection

DL Architecture	Dataset Images	Plant type	Performance Metrics
Capsule Networks[37]	Created a new Experimental field dataset	Mango leaf diseases	Classification Accuracy (97%)
R-FCN, SSD, Faster R-CNN with ResNet[38]	5,000 images taken in the fields	Nine diseases in tomato	85.98% Precision with ResNet-50 and Region-based Fully Convolutional Network
Artificial Neural Network[39]	9000 leave images inthe Plant Village dataset	Five disease classes in tomato	Accuracy 99.84%
AlexNet, Google Net, and ResNet ss[40]	5,560 Tomato plant images from Plant Village dataset	Eight disease classes	Accuracy 97.28%
AlexNet SqueezeNet[41]	Plant Village dataset	Ten disease classes of tomato leaves	Accuracy 95.65%
AlexNet, GoogLeNet, VGG[42]	Plant Village and in-field pictures of various plants	Apple, blueberry, banana, cabbage, etc.	The success rate of VGG is the finest among all
AlexNet, VGG16, SqueezeNet, GoogLeNet, InceptionResNetv2, ResNet50[43]	Created a dataset with Real field images	Plant diseases in Walnut, Apricot, Cherry, Peach	ResNet-F1score(97.14), Accuracy (97.86 ±1.56)
LeNet[44]	Plant Village	Banana	Classification Accuracy-98.6%, F1-score-98.6%

handled. The sample images of each class from the Cassava Dataset is shown in Fig. 2. The class names corresponding to classes 0 to 4 are Class-0: Cassava Bacterial Blight (CBB), Class-1: Cassava Brown Streak Disease (CBSD), Class-2: Cassava Green Motile (CGM), Class-3: Cassava Mosaic Disease (CMD) and Class-4: Healthy.

Transfer Learning

State-of-the-art models can be copied and utilized straightforwardly or combined with other models for customized tasks. Deep neural network models might require days or even a long time to train on enormous datasets. An approach to alternate route this procedure is to reutilize the pre-trained model weights produced with computer vision benchmark datasets, for example, ImageNet.

Deep learning is a model initially trained on a task like an issue currently being addressed. Transfer learning uses the trained models of a particular task for another task. At least one layer from the pre-trained model is exploited for another problem of interest. Transfer learning reduces the required time for preparing a model with a lower error rate. Transfer learning is beneficial when the task on which the model was trained includes much more labeled information than the task to be solved, and the properties of the two tasks are similar.

Need for Transfer Learning

Transfer learning gives some help in recognizing essential features of the given image, which are helpful in classification tasks as opposed to the beginning without any preparation. For example, ImageNet loads trained on many pictures help us

Fig. 2 — Sample images of each class from the Cassava Dataset (a) class 0, (b) class 1, (c) class 2, (d) class 3, and (e) class 4

recognize patterns like edges, shapes, and so forth, with no earlier learning on our dataset.

Pre-trained Models

The pre-trained models can be used as feature extractors for the model we want to design. In such cases, the pre-trained model output depends on the task characteristics to be solved. For instance, if the pictures involved in the task are entirely different from the images in the ImageNet, then the pre-trained model's output is only valid up to some layers. On the other hand, if the images of the task are similar to images in the ImageNet, then, at that point, the output from layers is more valid in utilizing the deeper layers of the pre-trained network. In such cases, the pre-trained model can be used as a feature extraction task. The features extracted by the pre-trained model are given as input to the new model further to train the model with the new image features.

On the other hand, the pre-trained or required part of the model can be incorporated straightforwardly into another model. In sucha case, the weights of the pre-trained models are to be fixed so that they are not refreshed when the new model is prepared. Then again, the consequences might be restored during the preparation of the new model, maybe with a lesser learning rate, permitting the pre-trained model to behave like a weight introduction conspire in preparing the new model. We can sum up these strategies as follows:

- Classifier: The pre-trained model is utilized straightforwardly to predict the new pictures class label.
- Separate Feature Extractor: The complete pre-trained model or part of it is utilized to retrieve and concentrate on the required features.
- Integrated Feature Extractor: The complete model or part is incorporated with another model. However, layers of the pre-trained model are fixed when training the network.
- Weight Initialization: The weights of the pre-trained model are considered as the weights of the new model, and the model is trained with new images.

All the above-stated approaches are effective and save massive time in creating and preparing a deep CNN model. The new progressions in the domain of Computer Vision offer us a chance to initialize the model weights with the well-trained model weights obtained by training the deep CNN models with massive datasets.

Transfer Learning with EfficientNet Network

Tan Mingxing and Quoc V. Le proposed a well-known architecture for image segmentation called EfficientNet.[46] The EfficientNet has built up the idea of scaling up convolutional neural networks (CNN). The authors started their work with a question: is there any process to scale up CNNs to attain improved accuracy and efficiency? The authors stated the importance of balancing the network's three dimensions, i.e., width, depth, and resolution. The three dimensions can be offset by just scaling every size with a constant ratio. For this, the authors have applied the compound scaling method, which consistently scales network width, depth, and resolution with a fixed set of coefficients. For instance, if we decide to use 2^X times more computational resources, then simply upturn the network depth by α^X, width by β^X and image size by γ^X, where $\alpha, \beta,$ and γ are fixed coefficients determined by a grid search on the original model.

The summarized architecture of the EfficientNet-B0 is shown in Table 2. Individual row defines a stage' i' with Li layers, with input resolution Hi × Wi and output channels Ci. By considering this as a base, the authors of EfficientNet have scaled the architecture with Compound Scaling to obtain EfficientNet-B1-B7**.**

The critical idea of bottleneck design used in ResNet is 1×1 convolution usage to bring down the number of channels and perform the convolution operation with 3×3 or 5×5 kernel size to the reduced channels to retrieve the features for classification. Lastly, use an extra 1×1 convolution operation to enhance the number of channels to the original value. The bottleneck design used in ResNets has been displayed in Fig. 3(a).

Table 2 — The base architecture of EfficientNet-B0

Stage i	Operator	Resolution Hi × Wi	#Channels Ci	# Layers Li
1	Conv 3 × 3	224 × 224	32	1
2	MBConv1, k3×3	112 × 112	16	1
3	MBConv6, k3×3	112 × 112	24	2
4	MBConv6, k5×5	56 × 56	40	2
5	MBConv6, k3×3	28 × 28	80	3
6	MBConv6, k5×5	14 × 14	112	3
7	MBConv6, k5×5	14 × 14	192	4
8	MBConv6, k3×3	7 × 7	320	1
9	Conv 1 × 1 & Pooling & FC	7 × 7	1280	1

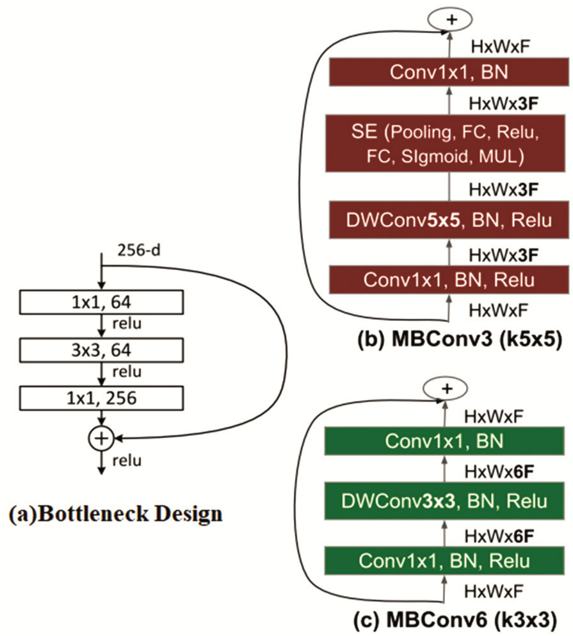

Fig. 3 — (a) Bottleneck design of inception model (b) and (c) MBConv layers of EfficientNet

The MBConv layer is an inverted bottleneck block with excitation and squeeze connections. The inverted bottleneck, as in MBConv, does the reverse, i.e., instead of decreasing the channels, the 1×1 Conv layer maintains the channels to 3 times the original. Note that using a regular convolution operation may be computationally heavy. Hence, a Depth-wise Convolution is used to retrieve the output feature map. Lastly, the second 1×1 Conv layer is used to down sample the number of channels to the initial value. It has been exemplified in Figs 3(b & c). The compound scaling of EfficientNet-B0 uses a compound coefficient $'\emptyset'$ to scale the network width, depth, and resolution consistently. For EfficientNet-B0, the value $\emptyset$ is taken as 1 and α = 1.2, β =1.1, γ = 1.15. For EfficientNet-B1 to B7, the values of α, β, γ are fixed to the values of B0, and different values $\emptyset$ are used to decide the depth, width, and resolution of the network.

For this research, the cassava leaf images are used by resizing all to 512 × 512 images from the dataset by considering the trained model EfficientNet-B7. While applying transfer learning, the pre-trained weights of the model, except the last layer, were loaded. The model was trained with stochastic gradient descent optimization for faster convergence by considering the pre-trained weights as primary weights. The learning rate was reduced by 5% if the validation loss did not change in the sequence's ten epochs. If the model is trained with all the training images once, it is considered as one epoch in the training process. The reduction in the learning rate increased the model's efficiency by adjusting to reach its local minimum.

Analysis of Experimental Results

This section explores the metrics used for assessing the efficiency of the trained model with transfer learning, the protocol design used in the experimentation, and the discussion on the results.

Evaluation Metrics

It is essential to assess how exactly it forecasts the appropriate label when building a classification model. However, this forecasting alone is not always enough because the model may sometimes deliver the wrong results. Hence, additional measures become imperative to decide the more important estimations and assessing the trained deep learning model.

The efficiency metrics assessed primarily are accuracy, recall, precision, F1-score with the confusion matrix, and the maximum distinguished one AUC-ROC (Area Under Curve ROC) curve. The measures can be evaluated individually for the entire data or every class label. To validate the retrained EfficientNet of this study, recall, precision, and F1-score are considered class labels wise, and the model's accuracy is evaluated. In the second case, the average of all class labels is viewed as the model's performance.

Accuracy is a vital efficiency parameter for the task of classification. It is simple to apprehend and easy to relate to binary and multiclass classification issues. Accuracy shows the fraction of correctly classified instances in the dataset's total quantity of samples. Accuracy may also infer wrong information in the case of an imbalanced dataset. Hence, other measures are to be evaluated in that case. Precision shows the fraction of the reliable positives in estimated positives, and recall gives the fraction of complete positive samples efficiently forecasted as positive. The two metrics can be combined to create another measurement called the F1score if a trade-off between recall and precision is necessary. The harmonic mean of precision and recall between zero and one is the F1-score. The computational guidance for evaluating the standard measurements is given by Eqs (1–4).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad \ldots (1)$$

$$Precision = \frac{TP}{TP+FP} \quad \ldots (2)$$

$$Recall = \frac{TP}{TP+FN} \quad \ldots (3)$$

$$F1 - Score = 2 * \frac{Precision*Recall}{Precision+Recall} \quad \ldots (4)$$

In Eq (1–4), TP, TN, FN, and FP indicate true positive, true negative, false negative, and false positive, respectively. A model is good if trained with recall and precision as 1, which thus gives F1-score as 1. But generally, 100% accuracy isn't possible for any deep learning task. So the trained model ought to have higher accuracy, precision, and recall.

Experimental Protocol

Due to the extensive accessibility of the deep learning libraries and frameworks, Python® was used to implement the expected transfer learning of EfficientNet. The TensorFlow framework is used in the backend to build deep learning architectures. Experiments were done with Tensor Processing Unit (TPU) support, and to have parallel processing, we considered eight replicas of TPU for running the code.

The challenging part of this dataset is the lack of good resolution and contrast images. The Contrast Limited Adaptive Histogram Equalization (CLAHE) technique is applied to enhance image quality.[47] The CLAHE can support low-resolution computer vision models with poor contrast images to achieve much better performance.

In the training process of EfficientNet, two kinds of parameters require concentration. First, the parameters learned in the model, such as weights in the model, and the second was hyperparameters, such as learning rates, epochs, batch size, input shape, etc. So, when the model was trained, the original weights of the EfficientNet-B7 were initially considered. Compared to the other well-known computer vision models, it has a precise good trade-off between the number of parameters and the accuracy o ImageNet. The initial weights were updated with gradient descent-based backpropagation. The model was trained with hyperparameters. However, the initial parameter values were enhanced by fine-tuning the importance of these hyperparameters with tuning procedures. The main parameters used were Batch Size as 16 * Replicas, Learning Rate as 3e-5 * Replicas, Epochs as 20 with 167 steps in each epoch, loss function as cross-entropy loss, and optimizer as SGD.

The suggested methodology is applied using the 5-fold cross-validation method. The experiment has been done five times. The five portions of the dataset were chosen at random. Every run includes a factor specified for testing, with the remaining four parts combined for training. The outcomes of the five runs are combined to provide the final results.

Discussion of the Results

The model's training started with 0.00000001 as the Learning rate; during the training process, it went up to 0.00024. The experimentation has been done with a 5-fold cross-validation technique to ensure that every image will participate in training and testing. For fold-1, the sparse categorical cross-entropy loss starts at 1.6291, and at the end of 20 epochs, it reduces to 0.2313. Similarly, the loss starts at 1.6364, 1.6321, 1.6304, and 1.6217 and ends at 0.2354, 0.2352, 0.2377, and 0.235 for fold-2 to fold-5, respectively. After the 20 epochs, the accuracy obtained is 0.889, 0.896, 0.82, 0.89, and 0.892 for fold-1 to fold-5, respectively.

The confusion matrix obtained from the dataset gives a more specified evaluation by different shades of color on how model performance alters with the disease representations in the images. The rows of the confusion matrix were taken as the actual class label, and columns were taken as the predicted class label. Diagonal cells display the proportion of the instances for which the actual and predicted classes are matched. The off-diagonal cells convey the proportion of the instances in which misclassification happens. The trained model's confusion matrix given in Table 3 reported 70%, 81%, 82%, 97%, and 73% of CBB, CBSD, CGM, CMD, and Healthy class images respectively. The off-diagonal cells in the Confusion matrix show that most misclassification happens in healthy and CBB images. The majority of misclassification in CBSD images was done as CBB diseased images. Most

Table 3 — Confusion matrix for Multiclass classification

CBB	0.7	0.05	0.03	0.04	0.17
CBSD	0.05	0.81	0.03	0.04	0.07
CGM	0.01	0.02	0.82	0.1	0.05
CMD	0	0.01	0.02	0.97	0.01
Healthy	0.09	0.06	0.06	0.07	0.73
	CBB	CBSD	CGM	CMD	Healthy

Table 4 — Evaluated performance metrics-class label wise

Class Labels	Precision	Recall	F1-Score	Support
Class-0 (CBB)	0.65	0.70	0.68	1086
Class-1 (CBSD)	0.84	0.81	0.82	2189
Class-2 (CGM)	0.81	0.82	0.82	2386
Class-3 (CMD)	0.96	0.97	0.96	13158
Class-4 (Healthy)	0.76	0.73	0.75	2576
Accuracy	—	—	0.89	21395

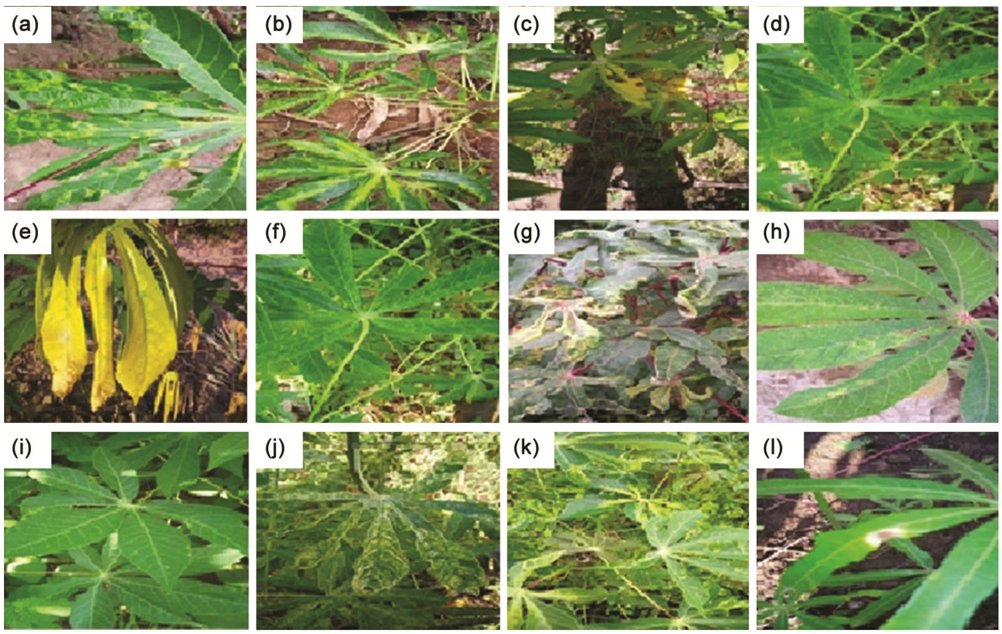

Fig.4 — (a) Class 3 (True), (b) Class 3 (True), (c) Class 0 (True), d)Class 4 (False, should be class 1), (e) Class 4 (False, should be class 0), (f) Class 1 (True), (g) Class 3 (True), (h) Class 3 (False, should be class 4), (i) Class 0 (False, should be class 4)), (j) Class 2 (True), (k) Class 3 (True), and (l) Class 0 (True)

misclassification of CGM images happens with CMD and vice versa. Misclassification in healthy images happens majorly as CBB diseased images.

The classification report with the performance measures as precision, recall, and F1-score separately for every class label in the dataset are illustrated in Table 4. Recall achieved the highest of 97% for CMD disease and a minimum of 70% for CBB disease. Maximum and minimum precision values for CMD and CBB diseases are 96% and 65%, respectively. Similarly, 96% and 68% are maximum and minimum F1-score values for CMD and CBB diseases, respectively. Finally, the model has achieved an accuracy of 89% with the support of 21395 images.

The images in Fig. 4 are randomly picked from the dataset, along with predictions made by the model. These images are random images of the dataset. These predictions show that applying transfer learning for EfficientNet is good for forecasting the leaf disease classifications in cassava plants.

Conclusions

The work can conclude that the effect of transfer learning is significant and considerably enhances the results of existing methods with a high percentage of accuracy and low loss rate. In some cases, the model failed to detect the diseases in the cassava plant leaves. The results exemplify a promising advance in automated disease detection in agriculture harvests. Subsequently, the next step is to take the issue of detected diseases by applying transfer learning on the EfficientNet model as the classification technique, under the hypothesis that a more precise detection allows for high classification accuracy. If that is the case, the entire framework could be a supporting tool for the farmer in the agriculture domain. This work is planned to explore and experiment with optimizing the EfficientNet architecture confined to a precise task: detecting specified four diseases with leaf images instead of using the whole architecture that considers any part of the plant which can be affected by conditions. Integrating this automated disease detection system with another system suggests the farmer's corresponding actions based on the identified disease. Finally, as said before, we are interested in assessing the work results to enhance the existing disease identification methods. Deep learning models for plant disease identification at early stages provide transcendental results in farming and research.

References

1 Himani G, An analysis of agriculture sector in Indian economy, *IOSR J Humanit Soc Sci (IOSR-JHSS)* **19(1)** (2014) 47–54.
2 Shruthi U, Nagaveni V & Raghavendra B K, A review on machine learning classification techniques for plant disease detection, *Proc.2019 5th Int Conf Adv Comput Commun Syst (ICACCS)*, IEEE (2019) 281–284.
3 LeCun Y, Bengio Y & Hinton G, Deep learning, *Nature*, **521** (2015) 436–444.
4 LeCun Y, Bottou L, BengioY & Haffner P, Gradient-based learning applied to document recognition, *Proc IEEE*, **86(11)** (1998) 2278–2324.
5 Dan C, Meier U, Masci J, Gambardella L M & Schmidhuber J, Flexible, high performance convolutional neural networks for image classification, *Proc 22nd Int Joint Conf Artif Intell*, **2** (2011) 1237–1242.
6 Hinton G, Deng L, Yu D, Dahl G E, Mohamed A R, Jaitly N, Senior A, Vanhoucke V, NguyenP, Sainath T N & Kingsbury B, Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups, *IEEE Signal processmag*, **29(6)** (2012) 82–97.
7 LeCun Y & Bengio Y, Convolutional networks for images, speech, and time series, in *The handbook of brain theory and neural networks*, **3361(10) (**1995).
8 Carranza-Rojas J, Goeau H, Bonnet P, Mata-Montero E & Joly A, Going deeper in the automated identification of Herbarium specimens, *BMC Evol Biol*, **17**, 181 (2017), https://doi.org/10.1186/s12862-017-1014-z.
9 Yang X, Guo T, Machine learning in plant disease research, *Eur J Bio Med Res* **3(1)** (2017) 6–9, DOI:10.18088/ ejbmr.3.1.2017.pp6-9.
10 Ahmed K, Shahidi T R, Alam I & Momen S, Rice leaf disease detection using machine learning techniques, *2019 Int Conf Sustain Technol Ind 4.0 (STI)*, *IEEE*, Dhaka, Bangladesh, (2019) 1–5.
11 Naik M R & Sivappagari C M R, Plant leaf and disease detection by using HSV features and SVM classifier, *Int J Eng Sci*, **6(12)** (2016) 1–4.
12 Panigrahi K P, Das H, Sahoo A K & Moharana S C, Maize leaf disease detection and classification usingmachine learning algorithms, in *Progress in Computing, Analytics and Networking. Advances in Intelligent Systems and Computing* edited by H Das, P Pattnaik, S Rautaray, K C Li, **1119** (Springer, Singapore), 659–669, https://doi.org/10.1007/978-981-15-2414-1_66.
13 Rao D R, Krishna M & Ramakrishna B, Smart ailment identification system for Paddy crop using machine learning, *Int J Innov Eng Manag Res*, **9(3)** (2020) 96–100.
14 Ashraf T & Khan Y N, Weed density classification in rice crop using computer vision, *Comput Electron Agric*, **175** (2020) 105590.
15 Chen J, Chen J, Zhang D, Sun Y & Nanehkaran Y A, Using deep transfer learning for image-based plant disease identification, *Comput Electron Agric*, **173** (2020) 105393.
16 Karlekar A & Seal A, SoyNet: soybean leaf diseases classification, *Comput Electron Agric*, **172** (2020) 105342.
17 Kim W S, Lee D H & Kim Y J, Machine vision-based automatic disease symptom detection of onion downy mildew, *Comput Electron Agric*, **168** (2020) 105099.
18 Janakiramaiah B, Kalyani G, Prasad L V, Karuna A & Krishna M, Intelligent system for leaf disease detection using capsule networks for horticulture, *J Intell Fuzzy Syst*, **41(6)** (2021) 6697–6713, 10.3233/JIFS-210593.
19 Subramanian M, Narasimha Prasad L V, Janakiramaiah B, Mohan Babu A & Sathishkumar V E, Hyperparameter optimization for transfer learning of VGG16 for disease identification in corn leaves using bayesian optimization, *Big Data*, **10(3)** (2022) 215–229, https://doi.org/10.1089/ big.2021.0218.
20 Taiwo K A, Utilization potentials of cassava in Nigeria: the domestic and industrial products, *Food Rev Int*, **22(1)** (2006) 29–42.
21 Singh V & Misra A K, Detection of plant leaf diseases using image segmentation and soft computing techniques, *Inf process Agric*, **4(1)** (2017) 41–49.
22 Ufuan A A, Ajayi O A, Bokanga M & Maziya-Dixon B, The use of cassava leaves as food in Africa, *Ecol Food Nutr*, **44(6)** (2005) 423–435, https://doi.org/10.1080/ 0367024050 0348771.
23 Akila M & Deepan P, Detection and classification of plant leaf diseases by using deep learning algorithm, *Int J Eng Res Technol*, **6** (2018) 2–7.
24 Ren S, He K, Girshick R & Sun J, Faster R-CNN: Towards real-time object detection with region proposal networks, *NIPS'15: Proc 28th Int Conf Neural Inf Process Syst*, 2015, 91–99.
25 Dai J, Li Y, He K & Sun J, R-FCN: Object detection via region-based fully convolutional networks, *NIPS'16: Proc 30th Int Conf Neural Inf Process Syst*, 2016, 379–387.
26 Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu C Y & Berg A C, SSD: Single shot multibox detector, in *Computer Vision – ECCV 2016, ECCV 2016, Lecture Notes in Computer Science*, edited by B Leibe, J Matas, Sebe N & M Welling, **9905** (Springer, Cham) 2016, 21–37, https://doi.org/10.1007/978-3-319-46448-0_2.
27 Adhikari S, Saban K K, Balkumari L, Shrestha B & Baiju B, Tomato plant diseases detection system using image processing, *Proc 1st KEC Conf Eng Technol* (Lalitpur) **1** (2018) 81–86.
28 Redmon J, Divvala S, Girshick R & Farhadi A, You only look once: Unified, real-time object detection, *Proc IEEE Conf Comput Vis Pattern Recognit*, (2016) 779–788.
29 Karthik R, Hariharan M, Anand S, Mathikshara P, Johnson A & Menaka R, Attention embedded residual CNN for disease detection in tomato leaves, *Appl Soft Comput*, **86** (2020) 105933.

30 Agarwal M, Singh A, Arjaria S, Sinha A & Gupta S, Toled: Tomato leaf disease detection using convolution neural network, *Procedia Comput Sci*, **167** (2020) 293–301.

31 Elhassouny A & Smarandache F, Smart mobile application to recognize tomato leaf diseases using convolutional neural networks, *2019 Int Conf Comput Sci Renew Energies (ICCSRE)* IEEE (Agadir, Morocco) 2019, 1–4, doi: 10.1109/ICCSRE.2019.8807737.

32 Howard A G, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M & Adam H, Mobilenets: Efficient convolutional neural networks for mobile vision applications, arXiv preprint arXiv:1704.04861 (2017).

33 Widiyanto S, Fitrianto R & Wardani D T, Implementation of convolutional neural network method for classification of diseases in tomato leaves, in *2019 Fourth Int Conf Inf Comput (ICIC)* IEEE (Semarang, Indonesia) 2019, 1–5, doi: 10.1109/ICIC47613.2019.8985909.

34 Aduwo J R, Mwebaze E & Quinn J A, Automated vision-based diagnosis of Cassava mosaic disease, *Ind Conf Data Mining- Workshops* (New York, NY) (2010) 114–122.

35 Abdullakasim W, Powbunthorn K, Unartngam J & Takigama T, An images analysis technique for recognition of brown leaf spot disease in cassava, *Tarım Makinaları Bilimi Dergisi*, **7** (2011) 165–169.

36 Mwebaze E & Owomugisha G, Machine learning for plant disease incidence and severity measurements from leaf images, in *Mach Learn Appl* (ICMLA), *15th IEEE Int Conf*, IEEE (Anaheim, CA) 2016.

37 Janakiramaiah B, Kalyani G, Prasad L V, Karuna A & Krishna M, Intelligent system for leaf disease detection using capsule networks for horticulture', *J Intell Fuzzy Syst*, (2021) 6697 – 6713.

38 Fuentes A, Yoon S, Kim S C & Park D S, A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition, *Sensors*, **17(9)** (2017) 2022, doi: 10.3390/s17092022.

39 Ashqar B A M & Abu-Naser S S, Image-based tomato leaves diseases detection using deep learning, *Int J Acad Eng Res*, 2(12) (2018) 10–16.

40 Zhang K, Wu Q, Liu A & Meng X, Can deep learning identify tomato leaf disease? *Advances in Multimedia*, **2018** ID 6710865, (2018), https://doi.org/10.1155/2018/6710865.

41 Durmuş H, Güneş E O & Kırcı M, Disease detection on the leaves of the tomato plants by using deep learning, *Proc 2017 6th Int Conf Agro- Geoinf* IEEE (2017) 1–5.

42 Ferentinos K P, Deep learning models for plant disease detection and diagnosis, *Comput Electron Agric*, **45** (2018) 311–318.

43 Türkoğlu M & Hanbay D, Plant disease and pest detection using deep learning-based features, *Turk J Electr Eng Comput Sci*, **27** (2019) 1636–1651.

44 Amara J, Bouaziz B & Algergawy A, A Deep learning-based approach for banana leaf diseases classification, *Proc BTW (Workshops)* (Stuttgart, Germany) (2017) 79–88.

45 Kaggle competition, https://www.kaggle.com/c/cassava-disease/overview, (2019).

46 Tan M & Le Q, Efficientnet: Rethinking model scaling for convolutional neural networks, *Int Conf Mach Learn PMLR*, (2019).

47 Reza & Ali M, Realization of the contrast limited adaptive histogram equalization (CLAHE) for real-time image enhancement, *J Signal Process Syst*, **38(1)** (2004) 35–44, https://doi.org/10.1023/B:VLSI.0000028532.53893.82