International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 12, No.1 (Dec-2022)

https://dx.doi.org/10.12785/ijcds/1201124

Identification of Various Rice Plant Diseases Using Optimized Convolutional Neural Network

Md. Sazzadul Islam Prottasha[1], A. B. M. Kabir Hossain[2], Md. Zihadur Rahman[3], S M Salim Reza[4], Dilshad Ara Hossain[5]

[1,4]Department of Information and Communication Technology, Bangladesh University of Professionals, Dhaka, Bangladesh
[2,3]Department of Information and Communication Engineering, Bangladesh Army University of Engineering and Technology, Bangladesh
[5]Department of Information and Communication Technology, International Islamic University, Malaysia

Received 23 Apr. 2021, Revised 12 Aug. 2022, Accepted 19 Sept. 2022, Published 31 December 2022

Abstract: Rice plant diseases has been a growing concern in recent years. Different kinds of rice plant diseases cause significant damage to the rice plants which results in reduced production of rice yield. Early detection of rice plant disease is crucial for crop protection system. However, the conventional disease detection techniques used by farmers are not highly accurate to identify the rice plant diseases in timely manner. Recent developments in Convolutional Neural Networks (CNN) has greatly improved the image classification accuracy and hence they are particularly useful in various plant disease detection. In this study, we proposed an optimized CNN architecture based on depthwise separable convolutions to identify various rice plant diseases. After collecting 12 types of disease affected rice plant images along with healthy rice plants from different rice fields of Bangladesh the images were preprocessed and augmented by which a dataset of 16770 images has been constructed. Along with the proposed CNN model, different lightweight state-of-the-art CNN architectures have been used on the dataset and results have been analyzed. The experimental analysis indicates that MobileNet v2 architecture provided the best validation accuracy of 98.7% among the all state-of-the-art CNN architectures. The proposed lightweight CNN architecture outperformed all the other state-of-the-art CNN architectures with a testing accuracy of 95.8%. Considering a small parameter size, it is evident that the proposed Convolutional Neural Network model performed significantly well in detecting the rice plant diseases accurately.

Keywords: Rice plant disease, CNN, Deep Learning, Depthwise Convolution, Agriculture.

1. Introduction

Rice is the main staple food of the half of the world population. Rice provides 21% of global human per capita energy and 15% of per capita protein intake [1]. Over the years, researchers and farmers have been trying to invent new techniques and seeds to improve the quality and productivity of rice grain. In recent years, farmers are using several types of fertilizers to increase the yield production. However, excessive use of nitrogen fertilizers has led to the further development of various diseases and pests. Several researchers have stated that it is possible to increase rice production significantly simply by reducing the impact of insects, pests, diseases and weeds. There are more than 40 types of rice plant diseases that can cause severe damage to the rice plants [2]. The overall production of rice is highly dependent on the pest management process. Early and accurate detection of pests can help the farmers in applying rapid treatment thus protecting the crop from pesticides thus increasing the overall production [2].

Traditionally, there are two types of methods used to detect any rice plant disease. The first method is by visual observation. This is the most ancient technique used by farmers. If there is any visual spot in the rice plant or any kind of pesticides, the farmers can identify the disease simply by observation. However, this technique may vary from person to person and requires great expertise. The next method is laboratory testing. This is basically done by the researchers. This technique can identify disease more precisely than the traditional visual observation method. However, laboratory testing requires time and can be costly at times. The main drawback of this technique is that laboratory testing cannot provide an immediate result. Therefore, the farmers need an easier and more convenient way of detecting different rice plant

E-mail:19541026@bup.edu.bd

diseases that is readily available. Recent developments in deep learning techniques have greatly improved the image classification accuracy and precision. Now, one of the most recent research interests is processing and analysis of images for the detection of plant diseases. The image processing method is particularly useful due to its higher accuracy and the immediate availability of the results. Hence in the agricultural context, one of the most recent research topics is the identification and classification of diseases from the images of a rice plant.

Different researchers have already presented different image processing approaches to identify various plant diseases. The work by Bhattacharya *et al.* [3] proposed a 3-layer CNN architecture for detecting 2 types of rice leaf diseases. By taking a total of 1500 images of normal and disease affected rice leaf images they proposed a model consisting 2 convolution layer and 2 max pooling layers followed by a fully connected layer. They used 9x9 and 5×5 filters in the conv1 and conv2 layer respectively. For the classification process, they split the data into 16 mini batches and ran 10 and 14 epochs respectively. For 14 epochs with 1000 images the CNN model achieved a maximum accuracy of 95.67%. While using 1500 training images, the model only achieved an accuracy of 78.44%. Considering the precision of the CNN model, we can say that changing the filter size or learning rate can lead to a better accuracy. The work by Sethy *et al.* [4] investigated different CNN architectures combined with SVM to classify rice plant diseases. They used 13 different CNN architectures along with various methods to identify rice plant diseases. By extracting the deep features of the fully connected layers of CNN architectures, they observed that the weight features of fully connected layer 6 was more significant in better classification. They combined SVM with the CNN architectures to detect rice plant diseases. The work at [5] used various Convolutional Neural Network pre-trained models to detect 6 types of common rice plant diseases. They took a total of 6330 images for the experiment. Instead of using overall classification they approached object detection method to identify the disease affected area using bounding boxes inside the image. They used RCNN, Faster RCNN, RetinaNet and YOLOv3 models on the dataset. The YOLO v3 model provided an accuracy of 79.19% in detecting the rice diseases. Considering heterogeneous backgrounds around the images, the YOLO model performed considerably well in detecting the diseases. The work at [6] proposes a deep neural network model for identifying banana leaf diseases. The training images were first resized to 60×60 pixel and converted to grayscale. LeNet architecture has been used for the feature extraction process. The dataset has been split into 80% training and 20% testing sets. For color images the model achieved an accuracy of 98.61% and for grayscale images the model achieved an accuracy of 94.44%. AlexNet model was used in [7] for identifying the olive diseases. The AlexNet model was fine tuned for the olive disease dataset. They used plant village dataset for olive diseases and trained the AlexNet model. Changing a few hyperparameters the model provided an astonishing training accuracy of 99.11%. However, there can be overfitting issues available when the model will be tested on real life images. Potato diseases identification method using CNN model has been proposed at [8]. The proposed CNN model consisting a total of five convolution layer followed by 3 fully connected layer. With hyperparameter tuning, the model achieved an accuracy of 95.85% with 90-10 train test split. Similar CNN model has been proposed for tomato [9], apple [10], rice plant [11], mango [12], cucumber [13]. The work by Kuznetsova *et al.* [14] proposed a machine vision system for identifying apples in farms. By using YOLOv3 module they achieved a 90.8% recall rate. They also measured the average time required to detect an apple by the YOLOv3 module. The work by Nguyen [15] proposes a deep learning-based technique for detection a total of 42 types of leaves. The study was conducted on a total of 5180 images. The common object detection CNN modules were used for the detection purpose.

2. METHODOLOGY

The working procedure of our research work starts with the data collection procedure followed by data preprocessing and augmentation. Once we have parameterized the input images, we used our proposed CNN model for the feature extraction process. We generated the feature maps and used them to train the model. By hyperparameter tuning we adjusted the model performance and increased the CNN model classification accuracy. The systematic approach of our working procedure is shown in the figure 1.

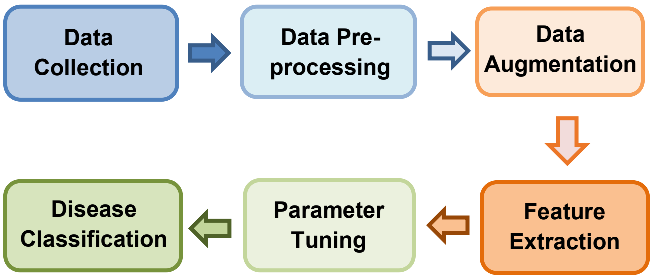


Figure 1. The working procedure of rice plant disease detection model.

A. Data Collection

We have collected a total of 1677 rice plant images from three different districts of Bangladesh namely Gazipur, Kishoreganj and Narayanganj over a period of 3 months. The images were taken under heterogeneous background so that we can consider different aspect of the images. Figure 2 shows various Stackburn sample images in different heterogeneous background under different lighting condition.

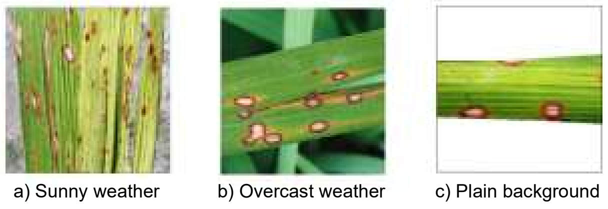


Figure 2. Different stackburn sample image variations.

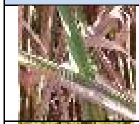
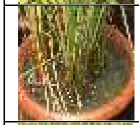
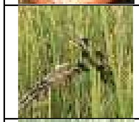
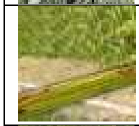
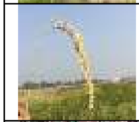
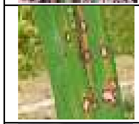
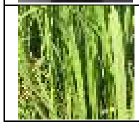
We have considered different stage of the disease occurrence thus we have achieved a comprehensive dataset considering all conditions of the disease occurrence. Table 1 shows the total number of collected images with different symptoms for each class.

TABLE I. COLLECTED IMAGES WITH DIFFERENT SYMPTOMS VARIATIONS

Name	Variations	No. of Collected images
Bacterial Blight	Early stage	120
	Final Stage	
Brown Spot	Small size spots	108
	Large size spots	
Brown Planthopper	Early stage	80
	Final stage	
False Smut	Black smut	107
	Brown smut	
Hispa	Visible spot	90
	Visible pests	
Leaf Blast	Brown spot	132
	Yellow spot	
Leaf Scald	Brown lesion	75
	Gray lesion	
Leaf Smut	Small spot	100
	Large spot	
Neck Blast	Black spot	226
	Brown spot	
Sheath Blight Rot	Black stem	220
	White stem	
Stackburn	Spot on whole leaf	80
	Spot on a part	
Stemborer	Grain	190
	Stem	
Healthy	Healthy leaf	149
	Healthy grain	

Table 2 shows the dataset description for each of the disease class with their corresponding sample image.

TABLE II. DATASET DESCRIPTION

Sample Image	Name	Description
	Bacterial Blight	It is a bacterial disease which causes wilting and browning of the leaves
	Brown Spot	It is a fungal disease which causes small, round or vertical brown spots in the leaves
	Brown Planthopper	It is caused by an insect which resides at the root of the plants and causes plant to dry out and turn brown
	False Smut	A plant pathogen transforms the individual rice grains into yellow fruiting bodies which reduces the grain quality
	Hispa	It is caused by an insect which eats the upper surface of the leaves leaving only the lower epidermis
	Leaf Blast	It is caused by a plant pathogenic fungus which damages the leaves and in severe cases kills the plant
	Leaf Scald	It is caused by a plant pathogenic fungus which infects the leaves and making it dry out quickly
	Leaf Smut	A plant pathogenic fungus causes infection which in terms create some raised angular spots on the leaves
	Neck Blast	It is caused by a fungus which causes the neck culm vulnerable and in severe cases neck calm breaks down
	Sheath Blight Rot	Caused by a fungus which creates oval or elliptical shaped brownish gray lesions on the leaves
	Stackburn	Caused by a seed borne fungus which creates circular grayish lesions with brownish red border on the leaves
	Stemborer	The pest larva drill through the upper nodes and feeds the tillers which cause the tillers to dry out
	Healthy	No visible spot or any pest. Uniform color.

B. Data Augementation

Since our dataset consists of small amount of images for each class, therefore we used several augmentation algorithm to increase the dataset size. We performed horizontal and vertical flipping, rotation of 30-degree, shear transform, random zoom, adding noise and skewing to generate 9 different versions of the original input image.

Considering a pixel inside the input image residing at coordinates of x and y, after performing flipping the coordinates will be at :

$$\text{New Coordinate } (x_{new}, y_{new}) = (width\text{-}x\text{-}1, y) \quad (1)$$

For rotation of multiple of 30 degree, the new coordinates will be :

$$x_{new} = x \times \cos(30) - y \times \sin(30) \quad (2)$$

$$y_{new} = x \times \cos(30) + y \times \sin(30) \quad (3)$$

For random zoom we increased the sample pixels by replicating the neighboring pixels. And for random noise we added the gaussian noise function with our sample images. In this way we generated different versions of our original input image. Figure 3 shows the generated augmented images for a sample image.

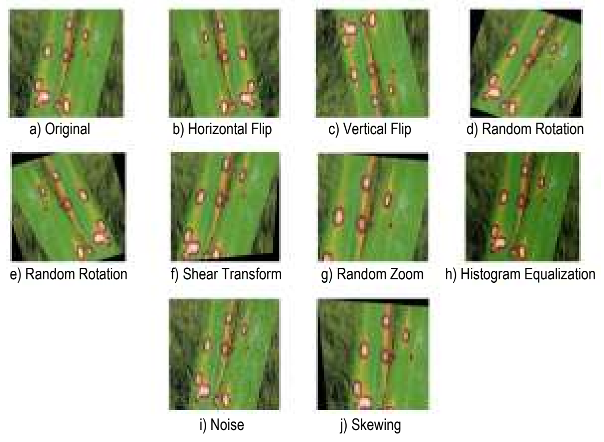


Figure 3. Generated augmented image for a sample stackburn image.

By image augmentation process we have been able to generate a total of 16770 images. We used these images for training and validation in our model. We used 1600 isolated images for the testing purpose. We used different train test data split on our experiment. Table 3 shows the various data split variations of our experiment.

TABLE III. DATASET SIZE FOR DIFFERENT DATA SPLIT

Data split	Train	Validation	Test
80-20	13416	3354	1600
70-30	11740	5030	1600

3. STATE OF THE ART CNN ARCHITECTURES

In this section we describe various state of the art CNN architectures used in our experiment. 6 different lightweight CNN architectures have been used on our dataset. We used MobileNet, MobileNet v2, Xception, DenseNet-121, NasNet Mobile and SqueezeNet.

MobileNet [16] model was focused on building a small CNN model that can be used in mobile devices and embedded application with very little processor power. MobileNet uses depthwise convolution and has 28 layers in total. The first layer of MobileNet is a normal convolution layer without any depthwise convolution. This layer takes an input of (224,224,3) size and provides an output of (112,112,32). This output is fed through the 1st depthwise convolution layer which provides an output size of (112,112,32). After that the output is passed through a pointwise 1×1 convolution that provides an output size of (112,112,64). This way the input is passed through a total of 27 convolution layers and finally provides a feature map of (7,7,1024). Finally, it is passed through softmax activation function which does the classification.

The Mobilnetv2 [17] is a modified version of MobileNet model which introduced inverted shortcut network with linear bottleneck. The Mobilnetv2 model uses ReLU6 activation function rather than ReLU. The parameter size was also reduced in MobileNetv2 model.

Xception [18] architecture introduced by google is an extreme version of inception architecture. This architecture introduced the depthwise separable convolutions. This architecture consist of 36 convolution layers with a depth of 126. These convolution layers are split into 14 modules where 12 modules having residual connections among them. This model showed the affect of hyperparameters in achieving higher accuracy with lower parameter size.

DenseNet-121 [19] model is a very deep network with fairly small parameter size. The architecture consists of dense blocks where number of filter is changed yet keeping the same feature maps. Each of the block has access of its preceding feature maps and each layer simply add new feature maps only. Any redundant feature map is discarded. This way DenseNet-121 model performs the convolutions and generates a small parameter size model.

NasNet mobile [20] architecture is focused for mobile devices. NasNet is a Neural Architecture Search model where the overall architecture is predefined but the blocks are not defined initially, instead they use reinforcement learning method to search for appropriate block. Due to it's reinforcement learning, the model provides higher accuracy yet having less parameters.

SqueezeNet [21] is a CNN architecture used for image classification that was developed in 2016 by researchers at Deep Scale, University of California, Berkeley, and

Stanford University. The main purpose of this SqueezeNet model is producing a small sized image classification model with an acceptable accuracy. SqueezeNet model stacks a bunch of fire modules and a few pooling layers. It consists of total 11 layers starting with a normal convolution layer followed by 9 fire modules and finally another convolution layer. It uses softmax activation function for the classification. With a model size of less than 1mb it provided astonishingly high accuracy in imagenet [22] dataset.

We considered all of this state-of-the-art CNN architecture due to their versatility and high accuracy. Our goal was to observe these architectures performance and make a statistical comparison with our proposed CNN model.

4. PROPOSED MODEL

We proposed a lightweight CNN architecture based on depthwise separable convolutions. The depthwise convolution is a type of convolution that is performed at each channel dimension rather than spatial dimension. Depthwise separable convolution is a combination of two different operation, depthwise convolution followed by pointwise convolution [5]. Since depthwise convolution performed at each depth hence the output is divided for each dimension. The pointwise convolution is used to combine those outputs into a new channel space. In practice this might seem extra computational cost, however depthwise separable convolutions actually reduces the computational cost and number of parameters.

Considering an input image with width W_i and height W_i and number of channel input is C. Now considering a filter size of $W_f \times W_f \times C$. Considering N filters that slides over the input multiple times then the total multiplication for normal convolution becomes :

$$Convolution\ Computation = N \times C \times {W_f}^2 \times {W_i}^2 \quad (4)$$

In depthwise convolution considering an input image with width W_i and height W_i. Since it is applied to a single channel at a time, hence the filter size will be $W_f \times W_f \times 1$. The multiplication after depthwise convolution becomes :

$$Depthwise\ Computation = N \times 1 \times {W_f}^2 \times {W_i}^2 \quad (5)$$

After the depthwise convolution a pointwise (1×1) convolution is applied to project the channel output into a single channel space. The multiplication becomes :

$$Pointwise\ Computation = N \times C \times {W_f}^2 \quad (6)$$

So, by summing the equation 5 and 6 we can calculate the total multiplication required for depthwise separable convolution.

$$Total\ Computation = N \times {W_f}^2 \times ({W_i}^2 + C) \quad (7)$$

We can compare the computational cost between normal convolution and depthwise separable convolutions using ratio R.

$$Ratio\ (R) = \frac{Computational\ cost\ of\ Depthwise\ Convolution}{Computational\ cost\ of\ normal\ Convolutions}$$

$$R = \frac{1}{N} + \frac{1}{{W_f}^2} \quad (8)$$

This ratio indicates that, the computational complexity of depthwise convolution is comparatively less than the normal convolutions.

Figure 4 shows our proposed depthwise separable CNN model. The model consists of 2 normal convolutions and 2 depthwise separable convolutions and finally two fully connected layer one with 256 neurons and another with 128 neurons with a dropout layer of 0.5.

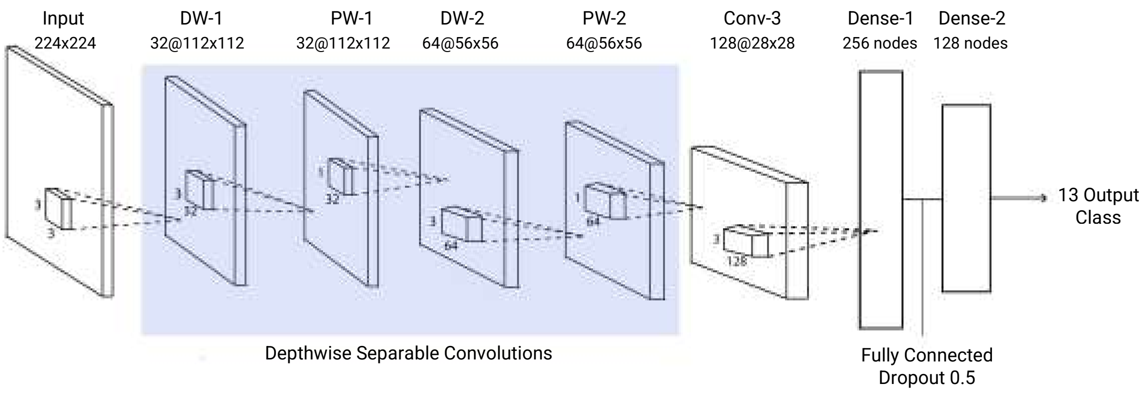


Figure 4. Proposed CNN model.

The model takes an input of 224×224 pixel size image. We used various kernel size filters on our model and observed their performance. 3×3 kernel performed best among the all different kernel dimensions. After each convolution layer a pooling and batch normalization was performed. Pooling has been used to reduce the feature map size, hence reducing the computational cost. The feature maps generated for different disease class is shown in figure 5.

To accelerate the training process, we used batch normalization [26] method which standardize the inputs for each mini batch.

In the dense layer we used Adam optimizer for updating the weights. Adam [23] is a stochastic gradient descent algorithm that computes adaptive learning rates for each parameter. Besides storing an exponentially decaying average past gradients like Rmsprop [24], Adam also stores the average past gradients exponential similar to momentum [25]. So, Adam is basically a combination of various optimizers. The decaying averages of past and past squared gradients x_t and y_t respectively can be computed as,

$$x_t = \beta_1 x_{t-1} + (1-\beta_1) g_t \quad (9)$$

$$y_t = \beta_2 y_{t-1} + (1-\beta_2) g_t^2 \quad (10)$$

Here x_t *and* y_t are the estimated mean and uncentered variance of the gradients respectively. It was observed that for the initial time steps and small decaying rates, the x_t *and* y_t are biased towards zero. Therefore, the biases have been corrected and calculated as,

$$\hat{x}_t = \frac{x_t}{1-\beta_1^t} \quad (11)$$

$$\hat{y}_t = \frac{y_t}{1-\beta_2^t} \quad (12)$$

Finally, for weight updates considering η step size, the weight W_t is calculated as,

$$w_t = w_{t-1} - \frac{\eta}{\sqrt{\hat{y}_t} + \varepsilon} \hat{x}_t \quad (13)$$

The default values for β_1, β_2 and ε are 0.9, 0.999 and 10^{-8} respectively.

5. Results

A. Experimental Setup

For experimental purpose keras framework with tensorflow backend has been used for training the models. Initially we performed the classification process on our proposed model by varying the model hyperparameters. Different size of kernels as well as variable learning rate has been used and tuned for the optimal performance. The model hyperparameters are shown as follows:

Input Size: 224×224

Fold: 10-fold

Batch size: 128

Epoch: 100

Activation function: ReLu

Optimizer: Adam

Keeping these parameters constant, we changed the other hyperparameters of our model.

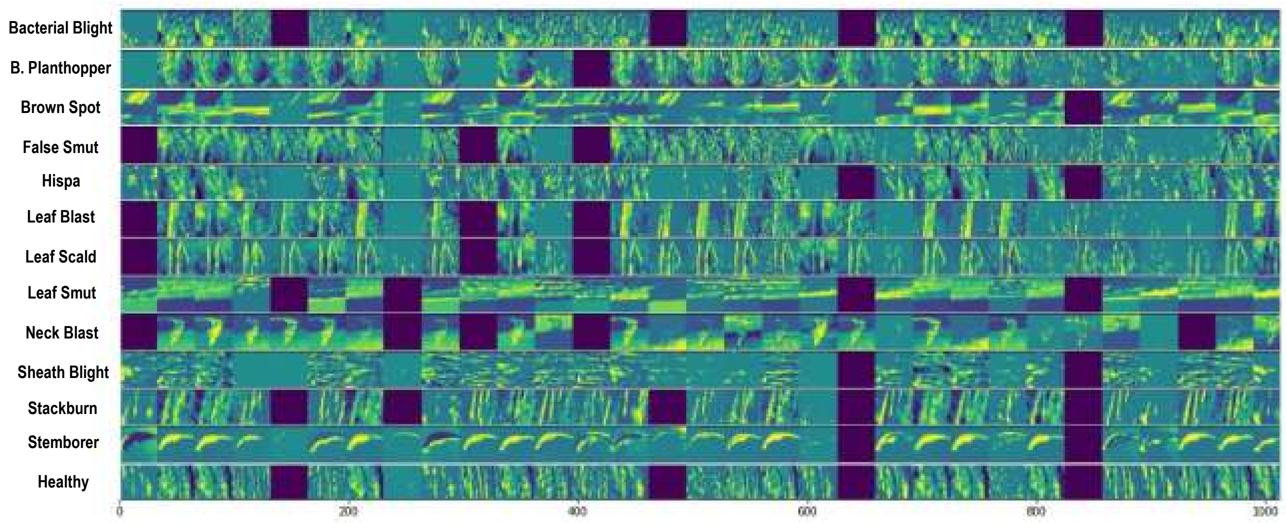


Figure 5. Generated feature maps for different disease class.

TABLE IV. PROPOSED MODEL PERFORMANCE CONSIDERING VARIOUS KERNEL DIMENSION. HERE BOLD FONT INDICATES BEST RESULT.

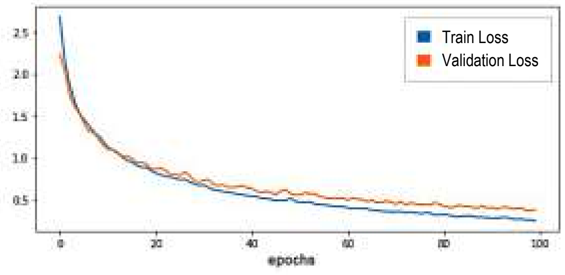
Kernel Dimension	Learning Rate	Train Accuracy	Validation Accuracy	Test Accuracy
2×2	0.01	98.3 ± 1.4%	96.4 ± 1.3%	94.8 ± 0.7%
	0.001	98.1 ± 1.3%	**97.3 ± 1.5%**	95.6 ± 1.0%
	0.0001	97.4 ± 0.6%	95.1 ± 1.0%	93.1 ± 0.9%
3×3	0.01	**98.9 ± 1.0%**	96.7 ± 1.6%	95.5 ± 0.8%
	0.001	98.7 ± 0.9%	97.1 ± 1.2%	**95.8 ± 1.0%**
	0.0001	97.8 ± 1.5%	95.9 ± 1.1%	94.6 ± 1.4%
5×5	0.01	98.1 ± 1.2%	95.7 ± 1.2%	94.6 ± 1.4%
	0.001	97.5 ± 1.0%	94.2 ± 1.5%	92.1 ± 1.9%
	0.0001	96.0 ± 0.9%	93.7 ± 0.3%	91.9 ± 1.1%

B. *Proposed CNN Performance*

The result of the proposed model performance considering different kernel dimension with variable learning rate is shown in table 4.

The result reported in table 4 clearly indicates that 3×3 kernel performs better than 2×2 and 5×5 kernel dimension. Initially data split of 80-20 and 70-30 both were considered. However, data split of 80-20 outperformed data split of 70-30. Hence we only illustrated the results of 80-20 data split where training set consists of 13416 images and validation set consist of 3354 images. The performance of 3×3 and 2×2 kernels are quite similar, however 2×2 kernel performs the classification process slowly. Hence for our experiment 3×3 kernel dimension is more appropriate. We achieved our highest mean training accuracy of 98.9% and highest mean testing accuracy of 95.8% with the 3×3 kernel. Considering variable learning rates with Adam optimizer, we observed that a learning rate of 0.001 performs better in validating and testing the data. A learning rate of 0.01 accelerates the training process but learning rate of 0.001 performs better in overall classification. The loss and accuracy curve of our proposed CNN model with 3×3 kernel with a learning rate of 0.001 is shown in figure 6.

We used max pooling in our experiment because it is used for extracting the sharpest features of the image map. Since we had heterogeneous background in our training images therefore we needed to extract the foreground images considering the edges and discard the background part. Max pooling particularly increased the testing accuracy in our dataset.



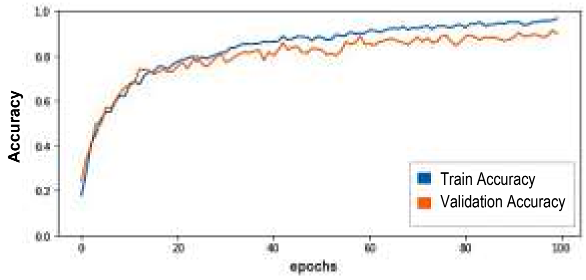


Figure 6. Loss and accuracy curve of proposed model with 3×3 kernel dimension.

To test the efficacy of our proposed CNN model on variety of conditions and different instances, we haveexperimented our depthwise separable CNN model on other rice plant disease dataset available. We ran 100 epochs with 80-20 train and validation data split on each of the dataset and the experimental results are reported in table 5.

TABLE V. PERFORMANCE ANALYSIS OF OUR PROPOSED MODEL ON DIFFERENT RICE DISEASE DATASET

Dataset	Class	Train Accuracy	Validation Accuracy
Rice Leaf Data Set [27]	3	97.2%	94.4%
Rice Seed Dataset [28]	2	98.8%	96.7%
Rice Leaf Disease [4]	4	97.1%	94.0%

The result reported in the table 5 clearly indicates that, our proposed model performed considerably well in detecting the rice plant diseases. In Rice Seed Dataset [28] our model achieved a validation accuracy of 96.7%. It is understandable that our model achieved high accuracy here because the dataset consists of only 2 class. But our proposed achieved a very high accuracy in our dataset which contains 13 disease class indicating the efficacy of our proposed model.

C. State of the art lightweight CNN Performance

Along with our proposed CNN, we also used 5 different lightweight CNN architectures on our dataset.

We used MobileNet v2, Xception, DenseNet-121, NasNet mobile and SqueezeNet v1.1 model. The main criteria for choosing these architectures are small parameter size with high classification accuracy and most of them can be turned into a lightweight application that can be used in real life.

We used fine tuning and baseline training method for training and testing the dataset. During fine tuning method we used the CNN models with pretrained imagenet weights. We freeze all the layer except the final dense layer where we initialize weights using Adam optimizer. Later the classification is performed on our dataset.

For baseline training we used the CNN models without any pretrained weights and then using the Adam optimizer we initialized random weights. The models learned the weights quickly and later adjusted the weights for optimal performance. The baseline training is slow compared to fine tuning method but provides better accuracy.

Table 6 shows the performance of various lightweight CNN models on our rice plant disease dataset along with our proposed model. For our proposed model we considered the accuracy achieved by using 3×3 kernel dimension with a learning rate of 0.001.

TABLE VI. PERFORMANCE ANALYSIS OF VARIOUS STATE OF THE ART CNN ARCHITECTURES ON RICE PLANT DISEASES. HERE BOLD FONT INDICATES BEST RESULT.

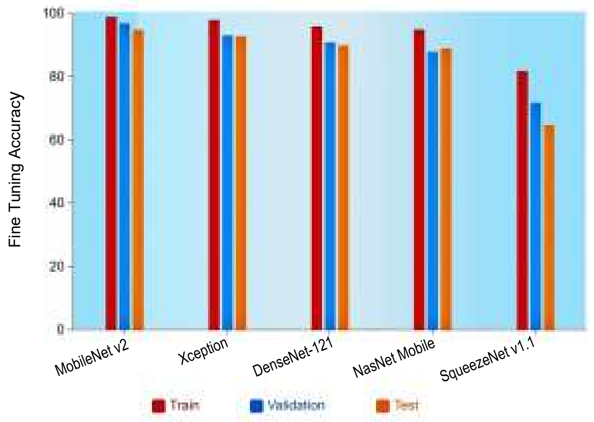
Architecture	Method	Train Accuracy	Validation Accuracy	Test Accuracy	Parameter Size
MobileNet v2	Fine Tuning	99.0 ± 0.9%	97.1 ± 0.5 %	95.1 ± 0.5%	3.5 M
	Baseline Training	**99.3 ± 0.7%**	**98.7 ± 0.3%**	**96.3 ± 0.8%**	
Xception	Fine Tuning	98.1 ± 0.9%	93.2 ± 0.8%	93.0 ± 0.5%	20 M

	Baseline Training	98.7 ± 1.0%	97.6 ± 0.4%	95.5 ± 1.5%	
DenseNet-121	Fine Tuning	96.5 ± 1.5%	91.0 ± 1.0%	90.3 ± 07%	8 M
	Baseline Training	97.3 ± 1.1%	92.7 ± 1.3%	91.8 ± 0.7%	
NasNet Mobile	Fine Tuning	95.4 ± 0.6%	88.4 ± 0.6%	89.5 ± 0.5%	4.3 M
	Baseline Training	96.2 ± 0.9%	89.2 ± 1.8%	88.1 ± 1.0%	
SqueezeNet v1.1	Fine Tuning	82.1 ± 1.9%	72.2 ± 1.8%	65.0 ± 1.0%	**0.7 M**
	Baseline Training	85.5 ± 0.5%	74.1 ± 0.9%	67.2 ± 0.8%	
Proposed CNN	Baseline Training	98.7 ± 0.9%	97.1 ± 1.2%	95.8 ± 1.0%	2.4 M

The result reported in table 6 indicates that baseline training provides better accuracy than fine tuning in state-of-the-art CNN architectures. Since we trained the models with random weights, the model learned the weights accurately in baseline training. However, the training process takes longer in baseline training.

Among the all CNN architectures, MobileNet v2 performed best in training and validating the data. During the baseline training, the Mobilenet v2 architecture achieved the highest mean validation accuracy of 98.7%. The Xception model demonstrated similar performance to the MobileNet v2 model. DenseNet-121 and NasNet model achieved satisfactory amount of classification accuracy while having less number of parameters. The SqueezeNet model could not achieve very high accuracy. However, it has the lowest number of parameters and the model size is less than 500kb. Our proposed CNN model performed considerably well while providing a testing accuracy of 95.8%. Figure 7 shows the fine tuning and baseline training accuracy chart for all the CNN architectures we have used.

Now considering the parameter size of various CNN architectures, SqueezeNet v1.1 has the lowest number of parameters. After that our proposed CNN model has the least number of parameters with a parameter size of 2.4 million parameters. MoblieNet v2 and NasNet Mobile architecture has substantially smaller parameters with a



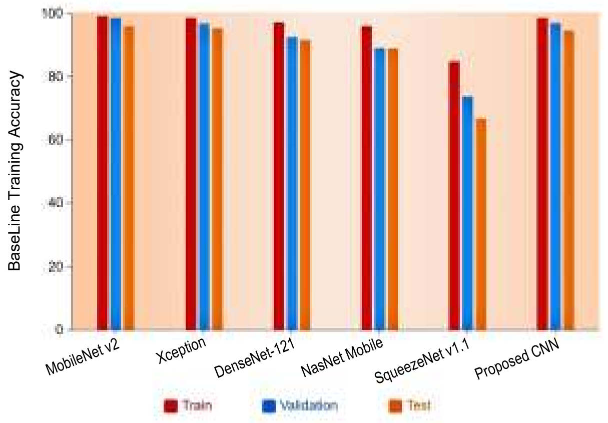


Figure 7. Accuracy graph of various state of the art CNN architectures used in rice plant disease detection.

parameter size of 3.5 millions and 4.3 millions respectively. The DenseNet-121 architecture has 8 million of parameters. Xception model has 20 million of parameters and it is also the most deep CNN architecture

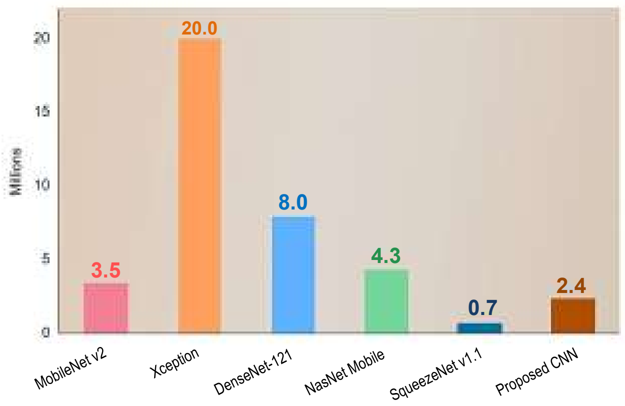


Figure 8. Parameter size comparison of various CNN architectures.

that we have used. Figure 8 shows the parameters size comparison of various CNN architectures that we have used.

Our proposed CNN model achieved a testing accuracy of 95.8% exceeding all lightweight CNN architectures except the MobileNet v2 model. The normalized confusion matrix for the testing data is shown in figure 9. By analyzing the confusion matrix, we can observe that, except brown spot disease all the other diseases have been classified with high accuracy. Some of the brown spot testing images has been misclassified as either hispa or neck blast. Since, brown spot has very similar characteristics to hispa, therefore some of the images has been misclassified. The same case happened for leaf scald where some of the testing images has been misclassified as leaf blast. Some of the leaf smut and leaf blast images have been misclassified as leaf scald images. Since the rice plant disease images are mostly similar in shape and have similar characteristics, therefore it is a great challenge to identify the diseases with high precision. All

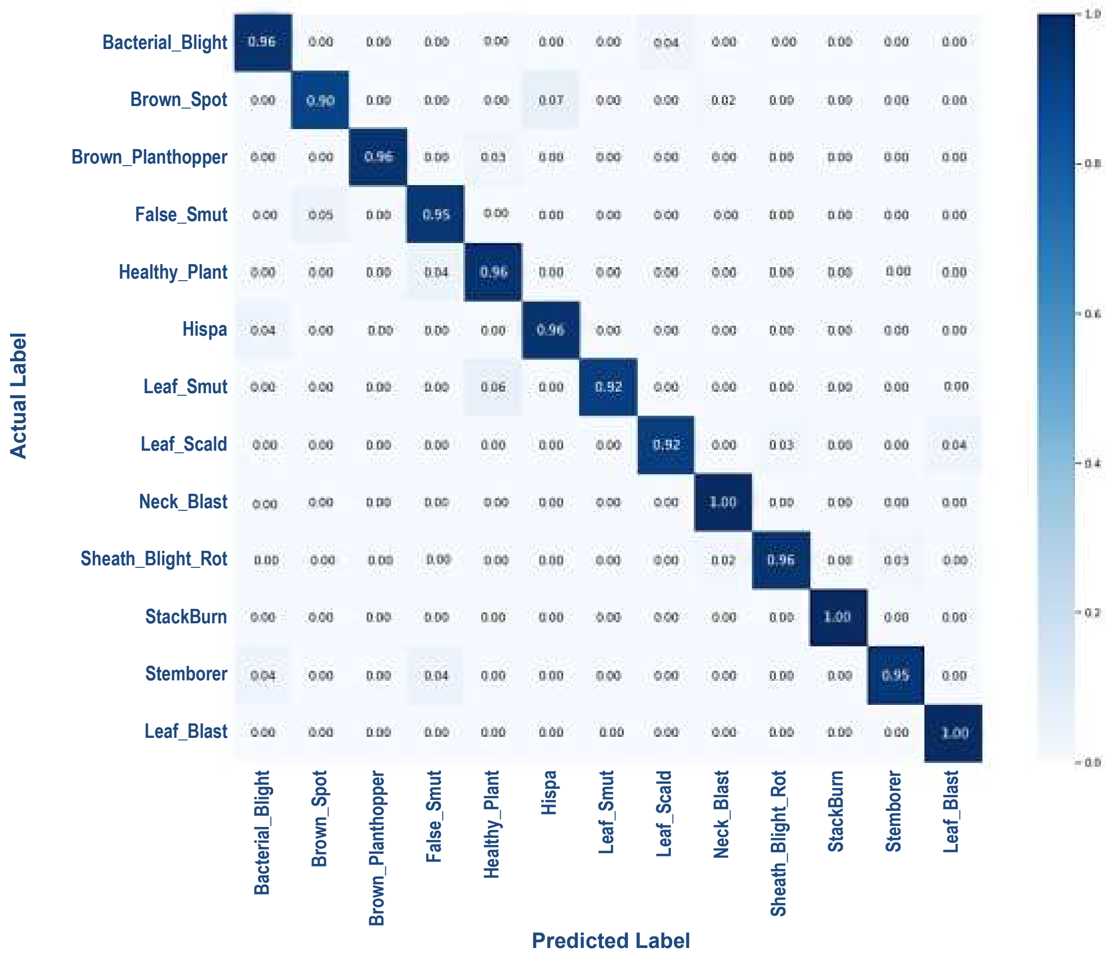


Figure 9. Confusion matrix of our proposed model on test dataset.

of the neck blast, stackburn and leaf blast testing images were identified correctly without any misclassification. Thus, our model achieved a higher testing accuracy.

D. *Findins*

Considering a parameter size of 2.4 millions only, our proposed CNN model performed well in identifying the rice plant diseases. We have acquired some interesting findings in our research. The findings are stated below.

- Baseline training provides better accuracy than fine tuning in most of the CNN architectures.
- Some of the rice plant diseases such as brown spot, leaf smut, leaf scald has been misclassified because of the fact that the characteristics of these diseases are quite similar.
- We observed biasness of our proposed model towards some of the disease classes containing high number of training images than the other classes. To avoid this, each of the disease class should contain equal number of images.
- By changing some hyperparameters, our proposed model performed significantly well with contrast to other state of the art CNN architectures while having a substantially smaller parameter size.
- Baseline training possess more computational complexity than fine tuning method while performing the classification task thus making the classification process slower.
- In our proposed model, a kernel dimension of 3×3 performed better in training data while a kernel dimension of 2×2 performed better in validating the dataset.
- Among the all CNN architectures we applied, MobileNet v2 performed better than the other state of the art CNN architectures in classifying various rice plant diseases.

6. APPLICATION AND FUTURE WORKS

A. *Rice Disease Snap*

Based on our experimental work, we developed Rice Disease Snap, an Android application that can recognize 12 different forms of rice plant diseases, as well as healthy plants. As discussed earlier, diagnosis of rice plant disease is a very delicate task and require great expertise to properly identify the diseases. Therefore, we intended to provide a feasible solution for farmers and agricultural personnel by which they can diagnose the rice plant diseases quickly and apply proper treatments. The working procedure of the Rice Disease Snap application is shown in figure 10.

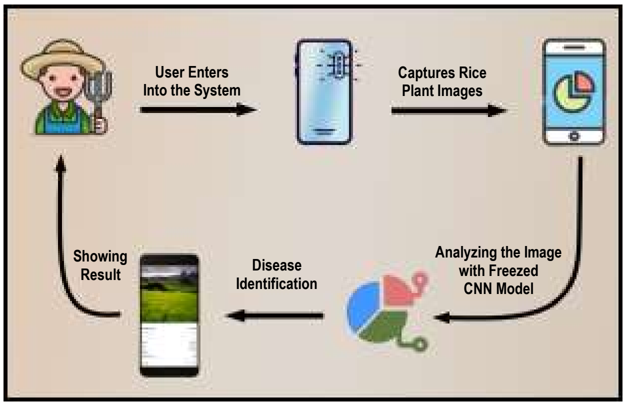

User Enters Into the System
Captures Rice Plant Images
Analyzing the Image with Freezed CNN Model
Disease Identification
Showing Result

Figure 10. Working procedure of rice disease snap application.

The android application was developed by freezing the CNN model after updating the weights of our proposed architecture. Later we optimized the freezed model and integrated it into the android application for the disease classification task. When a user first opens the app, they are presented with an interface that allows them to scan photographs of rice plants. After scanning the images, the app checks the scanned image with the freezed CNN model to find similarity and provides the result. The developed “Rice Disease Snap” android application can be found in Google Play Store [29].

B. *Future Works*

In the future, we plan to work on different aspects of our proposed model in order to improve its performance. We will try to collect different other rice disease images and include them into our dataset. We only used Adam optimizer on our proposed model. Different other weight updating optimizers can be used and their performance should be evaluated. There are different other optimizers that shows promising result in disease diagnosis. Also, we are planning to use different other CNN architectures on our dataset. Also, we are working on developing a IOS app based on rice disease diagnosis.

7. Conclusion

In this work we collected a total of 1677 rice plant images from different rice fields of Bangladesh. Later we constructed a dataset of 16770 rice plant images by the means of various image augmentation algorithms. We used 5 different lightweight state-of-the-art CNN architectures for classifying the rice plant diseases. Among the all state-of-the-art CNN architectures, MobileNet v2 architecture achieved highest training and validation accuracy of 99.3% and 98.7% respectively. We proposed a CNN architecture based on depthwise separable convolution which is an optimized version of normal convolutions. While having a substantially smaller parameter size, our proposed model achieved a validation and testing accuracy of 97.1% and 95.8% respectively. From the result analysis we can conclude that our proposed CNN model performed remarkably in identifying various rice plant diseases.

Acknowledgment

The Authors would like to thank the Information and Communication Technology (ICT) division of Bangladesh for funding this research work.

References

[1] The importance of rice, Accessed: Jan 10, 2021. [Online]. Available:http://www.knowledgebank.irri.org/ericeproduction/Importance_of_Rice.htm

[2] T. Bera, A. Das, J. Sil, and A. K. Das, “A survey on rice plant disease identification using image processing and data mining techniques,” in Emerging Technologies in Data Mining and Information Security, Springer, Singapore, pp. 365-376, 2019.

[3] S. Bhattacharya, A. Mukherjee, and S. Phadikar, "A deep learning approach for the classification of rice leaf diseases," Intelligence enabled research, pp. 61-69, Springer, Singapore, 2020.

[4] P.K. Sethy, N. K. Barpanda, A. K. Rath, and S. K. Behera, "Deep feature based rice leaf disease identification using support vector machine." Computers and Electronics in Agriculture, vol. 175, pp. 105527, 2020.

[5] K. Kiratiratanapruk, P. Temniranrat, A. Kitvimonrat, W. Sinthupinyo, and S. Patarapuwadol, "Using deep learning techniques to detect rice diseases from images of rice fields," In International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, pp. 225-237, Springer, Cham, 2020.

[6] J. Amara, B. Bouaziz, and A. Algergawy, "A deep learning-based approach for banana leaf diseases classification," Datenbanksysteme für Business, Technologie und Web (BTW 2017)-Workshopband (2017), pp. 79-88, 2017.

[7] M. Alruwaili, S. Alanazi, S. A. El-Ghany, and A. Shehab, "An efficient deep learning model for olive diseases detection," International Journal of Advanced Computer Science and Applications, vol. 10, no. 8, pp. 486-492, 2019.

[8] D. Oppenheim, and G. Shani, "Potato disease classification using convolution neural networks," Advances in Animal Biosciences, vol. 8, no. 2, pp. 244, July 2017.

[9] M. Agarwal, S. K. Gupta, and K. K. Biswas, "Development of Efficient CNN model for Tomato crop disease identification," Sustainable Computing: Informatics and Systems, vol. 28, p.100407, 2020.

[10] C. Bi, J. Wang, Y. Duan, B. Fu, J.R. Kang and Y. Shi, “MobileNet based apple leaf diseases identification,” Mobile Networks and Applications, pp.1-9, 2020.

[11] M. S. I. Prottasha, Z. Tasnim, S. M. S. Reza and D. A. Hossain, "A Lightweight CNN Architecture to Identify Various Rice Plant Diseases in Bangladesh," 2021 International Conference on Information and Communication Technology for Sustainable Development (ICICT4SD), pp. 316-320, 2021. doi: 10.1109/ICICT4SD50815.2021.9396927.

[12] A.S.A. Mettleq, I.M. Dheir, A.A. Elsharif and S.S. Abu-Naser, “Mango Classification Using Deep Learning,” International Journal of Academic Engineering Research (IJAER), vol. 3, no. 12, 2020.

[13] J. Ma, K. Du, F. Zheng, L. Zhang, Z. Gong, and Z. Sun, “A recognition method for cucumber diseases using leaf symptom images based on deep convolutional neural network,” Computers and electronics in agriculture, vol. 154, pp.18-24, 2018.

[14] A. Kuznetsova, T. Maleva, and V. Soloviev, "Detecting apples in orchards using YOLOv3," International Conference on Computational Science and Its Applications, pp. 923-934, Springer, Cham, 2020.

[15] V. T. Nguyen, T. Q. Duong, T. D. Le, and A. T. D. Nguyen, "Deep Learning-Based Methods for Plant Disease," In International Conference on Future Data and Security Engineering, pp. 166-177. Springer, Singapore, 2020.

[16] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint arXiv:1704.04861, 2017.

[17] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4510-4520, 2018.

[18] F. Chollet, “Xception: Deep Learning with Depthwise Separable Convolutions,” 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, pp 1800-1807, 2017.

[19] G. Huang, Z. Liu, L. V. D. Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4700-4708, 2017.

[20] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 8697-8710, 2018.

[21] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size," arXiv preprint arXiv:1602.07360, 2016.

[22] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," In 2009 IEEE conference on computer vision and pattern recognition, pp. 248-255, Ieee, 2009.

[23] D. P. Kingma, and B. Jimmy, "Adam: A method for stochastic optimization." 3rd International Conference for Learning Representations, San Diego, pp 1-13, 2015.

[24] G. Hinton, Lecture 6.5 — RMSProp, COURSERA: Neural Networks for Machine Learning. Technical report, 2012.

[25] N. Qian, "On the momentum term in gradient descent learning algorithms," Neural networks, vol. 12, no. 1, pp. 145-151, 1999.

[26] A. Neelakantan, L. Vilnis, Q. V. Le, I. Sutskever, L. Kaiser, K. Kurach, and J. Martens, "Adding gradient noise improves learning for very deep networks." arXiv preprint arXiv:1511.06807, 2015.

[27] H. B. Prajapati, J. P. Shah and V. K. Dabhi, "Detection and classification of rice plant diseases," Intelligent Decision Technologies, vol. 11, no. 3, pp. 357-373, 2017.

[28] Kaggle, Rice Seed Dataset, Accessed on: Feb 10, 2021. [Online]. Available:https://www.kaggle.com/rajkumar898/rice-plant-dataset

[29] Rice Disease Snap in Google Play Store (Version 1.1), [Mobile application software], Accessed on: Jul 30, 2021, Retrieved from https://play.google.com/store/apps/details?id=org.tensorflow.lite.examples.Rice_disease_snap

Md. Sazzadul Islam Prottasha obtained his bachelor's and master's degree in Information and Communication Engineering (ICE) from Bangladesh University of Professionals (BUP). Currently he is working on multiple projects related to image processing and computer vision. His research interests include Machine Learning, Computer Vision, Image Processing, Big Data Analytics, Artificial Intelligence, and Internet of Things.

A. B. M. Kabir Hossain is working as a lecturer at the Department of Information and Communication Engineering in Bangladesh Army University of Engineering and Technology (BAUET), Qadirabad Cantonment, Natore. He received his bachelor's degree in ICE from department of Information and Communication Technology from Bangladesh University of Professionals (BUP), Mirpur, Bangladesh, in 2018. His researches are in fields of Machine Learning, Signal Processing, Deep learning and Cloud Computing.

Md. Zihadur Rahman is working as a lecturer at the Department of Information and Communication Engineering in Bangladesh Army University of Engineering and Technology (BAUET), Qadirabad Cantonment, Natore. He received his B.Sc. degree in department of Information and Communication Technology from Bangladesh University of Professionals (BUP), Mirpur, Bangladesh, in 2018. Currently he is pursuing M.Sc. degree in Computer Software Engineering from Rajshahi University of Engineering and Technology (RUET), Bangladesh.

S M Salim Reza is working as an Assistant Professor of the Department of ICT, Bangladesh University of Professionals (BUP), Dhaka, Bangladesh. He is a Professional Member of IEEE and many other professional organizations. He has more than 50 research papers in ISI/ Scopus Indexed journal and many reputed International conferences. His current research interests include Information Security, Computer Vision, Telecommunications, Computer Networks, Internet of Things, Signal Processing etc.

Dilshad Ara Hossain has graduated in Computer Science and Engineering from Dhaka, Bangladesh. Currently, she has been studying in her final stage of Masters (by research) in Department of Information and Communication Technology (ICT), in International Islamic University Malaysia, Kuala Lumpur, Malaysia. She has many Indexed Journal paper and renowned conference proceedings. Her research interest includes Cyber Security, IoT, Artificial Intelligence, Machine Learning, VANET and recent computer applications.