Date of publication JUN-30, 2024, date of current version NOV-20, 2023.
www.computingonline.net / computing@computingonline.net

*Print ISSN 1727-6209*
*Online ISSN 2312-5381*
*DOI 10.47839/ijc.23.2.3542*

# Classification of Plant Disease using a State-of-the Art Deep Learning Algorithm on a Tesla GPU

**MANJIT JAISWAL[1], KAPIL KUMAR NAGWANSHI[1], ABHISHEK JAIN[2], RISHIKESH KUMAR[1], SHREYASH GAURAV[1], YUKTA WATTI[1], ANULAL MAHTO[3], SATYENDRA SINGH THAKUR[1]**

[1]Department of Computer Science and Engineering, [2]Department of Information Technology, [3]Department of Industrial Production,
Guru Ghasidas Vishwavidyalaya (A Central University), Bilaspur-49500, India

Corresponding author: Manjit Jaiswal (e-mail: manjit.jaiswal222@gmail.com)

**ABSTRACT** This paper proposes a study conducted on various techniques that can be employed for the early detection of plant diseases. With exponential growth in the global population, there is a dire need for the detection and prevention of various types of plant diseases such as Mosaic virus in Solanum Lycopersicon (tomato), bacterial spot in Fragaria Ananassa (strawberry), late and early blight in Solanum Tuberosum (potato), huanglongbing in Citrus sinensis (orange), and Isariopsis leaf spot in Vitis vinifera (grapes). These diseases generally lead to lower yields and hence less profit. In the last two decades, there has been rapid development in the fields of image processing and deep learning. Various models of deep learning can be used for plant disease detection. The main objective is that as soon as plant leaf disease appears, there should be one device to monitor the symptoms and detect them over a large field with as much accuracy as possible. This study compares the deep learning models Resnet, MobileNet, and inceptionV3 that are implemented on a large dataset taken from the Kaggle repository. We implemented the models using Google Colaboratory tools, which provide us with Python's Jupyter notebook that runs on the Google cloud server. The GPU "Tesla T4" and CPU "Intel Xenon" were used during training, validation, and testing respectively. The training and validation accuracy of the InceptionV3 model was 98.78% and 93.94%, respectively. MobileNet classified various plant diseases with training and validation accuracies of 99.57% and 97.31. Similarly, for ResNet, the training accuracy was found to be around 99.62% and the validation accuracy was 97.16%. We hope that this work will provide a helpful resource for other researchers working in the field of agriculture to detect various types of crop diseases. Future work and some challenges still faced are also discussed in this study.

**KEYWORDS** Deep Learning; CNN; ResNet; inceptionV3; MobileNet; Plant Diseases; GPU; Tesla T4; Intel Xenon

## I. INTRODUCTION

A large portion of the world's population is dependent on agriculture. Nearly 45% of the world's population and 70% of the Indian populace do some kind of agriculture for a living [1]. In particular, in India, more than 54.6% of the total workforce is dependent on agriculture [2]. Diseases in plants are one of the major problems that are faced in the field of agriculture as they have an unfavorable effect on the yield and quality of crops. As it is shown in [3], images are represented by mathematical models on the computer, and in image feature extraction, there are many mathematical phases in which images are converted into numbers and eventually carried out the image features. The agriculture industry can greatly benefit from the proper management of diseases to produce healthy crops. Humankind has made some advancement in the detection and recognition of various plant diseases. Most of these techniques included naked eye observation, which is very ineffective, cumbersome, and slow. However, with recent developments in image processing, disease classification and identification has become easy and fast. The combination of technology and agriculture can result in beneficial profit and yield. In [4], classification of rice plant disease was performed by Adam and SGDM optimizers along with CNN model. In [5], corn leaf disease detection at an early stage was performed by enhancing the K-nearest neighbor (EKNN) algorithm. In [6], many comparisons were made using different methods such as MobileNetV2, DenseNet169, ResNetV2, and InceptionResNetV2 to detect the disease of leaf on Robusta coffee plant. This study not only compares all the state-of-the-art techniques based on their accuracy, loss, and other measures of comparison but also focuses on the accuracy as much as possible and attempts to outperform the other earlier models.

## II. LITERATURE REVIEW

Many researchers have already performed plant disease detection and classification using various machine learning and deep learning models. Some of the research work included multiple convolutional neural network (CNN) architectures such as ResNet, AlexNet, VGG, InceptionV3, and GoogleNet for detecting and classifying various plant diseases. In [7], five CNN architectures were used. These architectures included VGG, Alex Net, GoogLeNet, Overfit, and Alex Net. The dataset used here consisted of 87,848 images distributed in an 80–20 ratio for training and testing purposes. Among all these models, VGG had the highest accuracy rate of 99.48%, followed by AlexNetOWTBn and AlexNet with 99.44% and 99.06% accuracy rates, respectively.

In [8], the VGG, ResNet, and Inception models were used for classification using 144000 training images and 2982 testing images. VGG, Inception and ResNet achieved the accuracy of 87.9%, 92% and 92.9%, respectively.

In [9], CNN models were trained on a dataset containing 4,032 images of rose leaves. After image augmentation, the number of images was 40,320 and was split into 70%, 15%, and 15% for training, validation, and testing. The highest accuracy was achieved by the Early Fusion-based model, followed by the Late Fusion-based and VGG models.

In [10], a study was conducted on the detection and prevention of diseases in apple leaves using various CNN models. Of these, GoogLeNet (CNN) achieved the highest accuracy of 98.5%.

Similar study was performed in [11] on 1212 images of tomato leaves. This study focused on mobile deployment and achieved the lowest detection confidence of 70%.

Similarly, in [12], the night-CNN model for nightshade crop leaf disease detection was used. In [12], 50000 healthy and infected plant leaf images were used for training and testing purposes. In [12], the Night-CNN training and testing accuracy ranged from 93% to 95%. We are classifying the leaves of plants using different state-of-the-art algorithms and comparing the accuracy, precision, recall, AUC, and time taken by the model to train on each epoch. The models that we are working on are ResNet [13], Inception [14], and Mobile Net [15].

### *A. INCEPTION V3*

Starting with the first Inception version (Inception V1, also called GoogLeNet [16]), this architecture is based mainly on the Inception Module. It has a deeper network with high computational efficiency. This network comprises 22 layers. There is a maximum pooling layer available but no fully connected (FC) layer at the end of the architecture.

Fig. 1 shows the architecture of the naïve implementation of the inception module.

The naïve version of the inception module has some disadvantages. It is very expensive to compute, and the pooling layer preserves feature depth, which means that the total depth after concatenation can only grow at every layer.

To solve this problem, an inception module with dimension reduction was introduced. The new implementation of inception uses “bottleneck” layers that use 1 × 1 convolutions to reduce the feature depth.

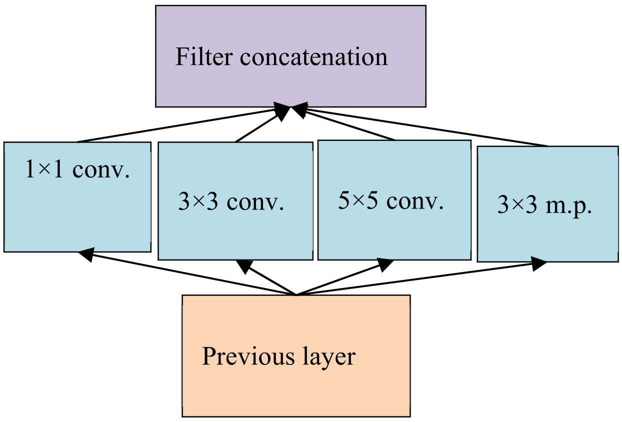

Figure 1. Inception Module, Naïve Version

Fig. 2 shows the architecture of the Inception module with dimension reduction.

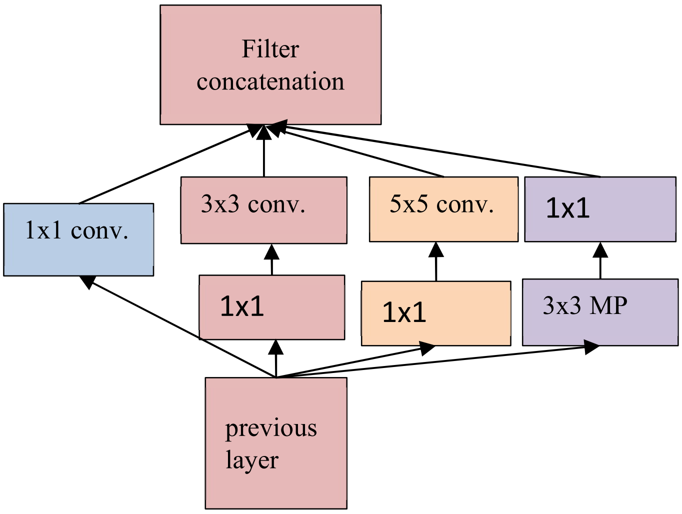

Figure 2. Inception Module with Dimension Reduction

Here conv. means convolution and MP means the Max Pooling (MP) layer.

Inception V3 uses a bottleneck inception module, which performs better than the normal inception module. In the inception V3, the kernel sizes are small because they are more efficient. This is achieved by factorizing the convolutions. Factorizing kernels into smaller factors reduces the overall complexity of the architecture.

### *B. MOBILENET*

The MobileNet architecture is a small and fast architecture that was introduced for mobile and related vision applications.

The MobileNet model is based on depth-wise separable convolutions. Depth-wise separable convolutions are depth-wise convolutions that are followed by point-wise convolutions. 1×1 convolution is called a point-wise convolution. Due to the depth-wise separable convolutions, the computation cost of the MobileNet architecture is reduced significantly. The standard convolutional layer is parameterized by a convolution kernel K of size

$$D_K \times D_K \times M \times N,$$

where $D_K$ is the spatial dimension of the kernel assumed to be square, M is the number of input channels, and N is the number of output channels, as defined previously.

Equation (1) is used to calculate the depth- wise separable convolution cost of the mobile net architecture.

$$\cos t = \frac{1}{N} + \frac{1}{D_K^2} . \quad (1)$$

Fig. 3 explains a part of MobileNet architecture with depth-wise convolution used along with batch normalization, Rectified Linear Unit (ReLU) and 1 × 1 convolution layer.

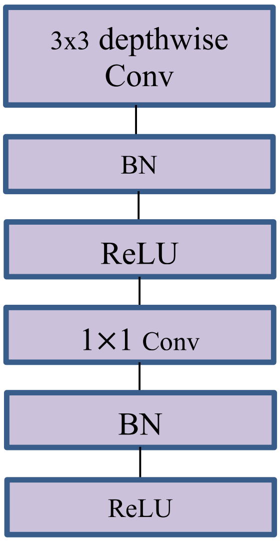

Figure 3. Depth wise Separable convolutions with Depth wise and Pointwise layers followed by batch norm and ReLU

### *C. RESNET50*

Resnet was first introduced in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2015 classification challenge. It solved the problem of deep neural networks becoming less accurate after a certain depth [17]. In this architecture, the concept of the residual block was first introduced. The residual block skips the connection block. Individual layers in this architecture give output to the next layer and to the next residual block. Residual blocks ensure that the accuracy of the architecture remains at least that of the previous block, which reduces the chances of feature loss in deeper networks. Fig. 4 explains how residual learning uses skip connections.

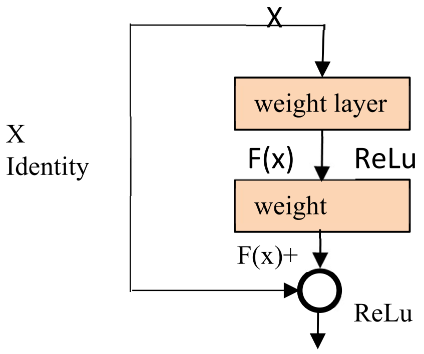

Figure 4. Residual Learning Building block

## III. RESEARCH METHODOLOGY

### *A. Dataset*

We used the New Plant Disease Dataset (augmented) [18], which can be found in the Kaggle repository. This dataset [18] was recreated using offline augmentation from the original dataset. The dataset should be representative of real-world scenarios. The original dataset can be found in [19] GitHub repository. This dataset consists of approximately 87.9 thousand RGB images and 1.43 GB in size of healthy and diseased crop leaves, which are categorized into 38 different classes. Image augmentation [20] is a process in which we use the already existing images to create new sample images. This process involves rotating, cropping, and flipping the original images. This increases the number of sample training images and makes our model more accurate for different possibilities of the images. This dataset consists of approximately 87.9 K RGB images of healthy and diseased crop leaves. These leaves were obtained from 14 different plants and 26 different diseases, which were categorized into 38 different classes based on plant and disease combination. The total dataset is divided into an 80/20 ratio of the training and validation sets, preserving the directory structure. Of these, 70295 images are for training and 17572 images are for validation. A new directory containing 33 test images is created later for prediction. All images are of size 256 × 256.

**Table 1. Distribution of images in the dataset for various species and diseases**

| S. NO | Species and diseases | Training | Validation | Total |
|---|---|---|---|---|
| 1. | Tomato Late Blight | 1851 | 463 | 2314 |
| 2. | Tomato Healthy | 1926 | 481 | 2407 |
| 3. | Grape Healthy | 1692 | 423 | 2115 |
| 4. | Orange Huanglongbing | 2010 | 503 | 2513 |
| 5. | Soyabean Healthy | 2022 | 505 | 2527 |
| 6. | Squash Powdery mildew | 1736 | 434 | 2170 |
| 7. | Potato Healthy | 1824 | 456 | 2280 |
| 8. | Corn Leaf Blight | 1908 | 477 | 2385 |
| 9. | Tomato Early Blight | 1920 | 480 | 2400 |
| 10 | Tomato Septoria Leafspot | 1745 | 436 | 2181 |
| 11. | Corm Cercospora leaf spot gray leaf spot | 1642 | 410 | 2052 |
| 12. | Strawberry Leaf Scorch | 1774 | 444 | 2218 |
| 13. | Peach Healthy | 1728 | 432 | 2160 |
| 14. | Apple Scab | 2016 | 504 | 2520 |
| 15. | Tomato yellow Leaf Curl Virus | 1961 | 490 | 2451 |
| 16. | Tomato Bacterial Spot | 1702 | 425 | 2127 |
| 17. | Apple Black Rot | 1987 | 497 | 2484 |
| 18. | Blue berry Healthy | 1816 | 454 | 2270 |
| 19. | Cherry Powdery Mildew | 1683 | 421 | 2104 |
| 20. | Peach Bacterial Spot | 1838 | 459 | 2297 |
| 21. | Apple Rust | 1760 | 440 | 2200 |
| 22. | Tomato Target Spot | 1827 | 457 | 2284 |
| 23. | Pepper Bell Healthy | 1988 | 497 | 2485 |
| 24. | Grape Leaf blight (Isariopsis Leafspot) | 1722 | 430 | 2152 |
| 25. | Potato Late Blight | 1939 | 485 | 2424 |
| 26. | Tomato Mosaic Virus | 1790 | 448 | 2238 |

| | | | | |
|---|---|---|---|---|
| 27. | Strawberry Healthy | 1824 | 456 | 2280 |
| 28. | Apple Healthy | 2008 | 502 | 2510 |
| 29. | Grape Black Rot | 1888 | 472 | 2360 |
| 30. | Potato Early Blight | 1939 | 485 | 2424 |
| 31. | Cherry Healthy | 1826 | 456 | 2282 |
| 32. | Corn Common Rust | 1907 | 477 | 2384 |
| 33. | Grape Esca (Black Measles) | 1920 | 480 | 2400 |
| 34. | Raspberry Healthy | 1781 | 445 | 2226 |
| 35. | Tomato Leaf Mold | 1882 | 470 | 2352 |
| 36. | Tomato Spider mites | 1741 | 435 | 2176 |
| 37. | Pepper Bell Bacterial Spot | 1913 | 478 | 2391 |
| 38. | Corn Healthy | 1859 | 465 | 2324 |

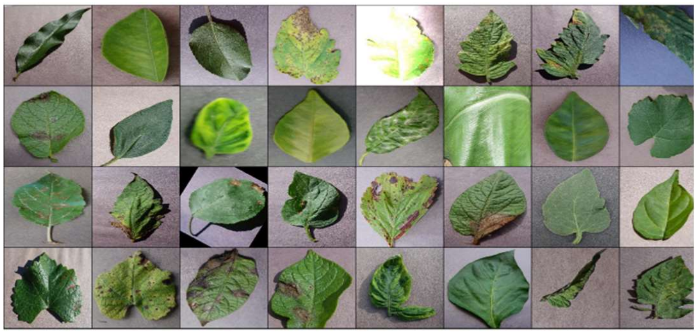

Figure 5. Sample image from the dataset

### B. Training

We used the transfer learning method on pre-trained models of Keras applications. We trained our model on a Tesla T4 GPU provided by Google Colaboratory. We used batch sizes of 128 and 50 epochs for training our dataset. We first downloaded the dataset into the Google Colaboratory from Kaggle's repository. Then, we loaded the data into the GPU using the Data Loader module provided in Keras. Since the original image was 256 ×256 so we had to convert our images into 224×224 for training purposes, as the models were designed for the ILSVRC dataset where images were of size 224 × 224. We loaded our pre-trained model weight from Keras and added more layers to use the weights and obtain the classification result. The layers that were added are Global Average Pooling, Dropout (0.1), ReLu, and SoftMax layers. The Adam optimizer [21] was used along with the categorical cross entropy [22] loss function.

The following formula is used to calculate the categorical cross-entropy in equation (2):

$$L_q(f(x),e_j)=\frac{(1-f_j(x)^q)}{q}, \tag{2}$$

where, $L_q$ is the loss function, $f(x)$ is the function for which loss is calculated, $f_j$ denotes the $j^{th}$ element in $f_j$, $q \in (0,1]$ and $e_{y_i} \in \{0,1\}^c$.

The function calculated accuracy, area under the ROC curve (AUC) score, precision, and recall.

*1) Global Average Pooling [19]*

The idea is to generate one feature map for each corresponding category of the classification task in the last convolution layer. Instead of adding fully connected layers on top of the feature maps, we take the average of each feature map, and the resulting vector is fed directly into the SoftMax layer. One advantage of global average pooling over fully connected layers is that it is more native to the convolution structure by enforcing correspondences between feature maps and categories. Thus, the feature maps can be easily interpreted as category confidence maps. Another advantage is that there is no parameter to optimize in the global average pooling; thus, overfitting is avoided at this layer. Furthermore, global average pooling sums out the spatial information; thus, it is more robust to spatial translations of the input.

*2) Dropout [24]*

The key idea is to randomly drop units (along with their connections) from the neural network during training. This prevents the units from co-adapting too much. During training, dropout samples were obtained from an exponential number of different "thinned" networks. At test time, it is easy to approximate the effect of averaging the predictions of all these thinned networks by simply using a single unthinned network with smaller weights. This significantly reduces overfitting and provides major improvements over other regularization methods. We show that dropout improves the performance of neural networks on supervised learning tasks in vision, speech recognition, document classification, and computational biology, obtaining state-of-the-art results on many benchmark datasets.

The following formula gives the output image after using the dropout layer:

$$y = f_W(x,z), \tag{3}$$

where y is the output image, x is the input image, and z is a random mask.

*3) ReLU [25]*

The ReLU of the Rectified Linear Unit is used as an activation function.

Mathematically, the ReLU function is denoted by the following formula:

$$F(x)=\max(0,x), \tag{4}$$

where x is the input image pixel.

## IV. IMPLEMENTATION AND RESULTS

Google Colaboratory (Colab) is a product of Google Research. Colab provides a Jupyter notebook environment to run python codes. Colab also provides free resources such as CPU, GPU and RAM. The specifications of the hardware and software that we have used are mentioned in Table 2.

**Table 2. The Hardware and Software Specifications**

| S. No. | Specification Type | Description |
|---|---|---|
| 1. | GPU | Tesla T4 |
| 2. | CPU | Intel Xenon (64 bi) |
| 3. | CPU RAM | 12.7 GB |
| 4. | GPU RAM | 15.0 GB |
| 5. | Disc | 78.2 GB |
| 6. | IDE | Google Colaboratory |
| 7. | Programming Language | Python |
| 8. | Operating System | Linux |
| 9. | CuDNN version | 8302 |

After comparing the three models, i.e., InceptionV3, MobileNet, and ResNet, we found that Resnet gives the highest accuracy, followed by MobileNet and InceptionV3. The comparison of models based on various criteria such as accuracy, loss, precision, recall, and AUC is given in Table 3, Tables 4, 5, and 6, respectively. The time taken by the models is also shown in Table 7.

Formulas for calculating accuracy, precision, recall, and AUC are given in equations (5) - (10).

$$precision = \frac{TP}{TP + FP}, \tag{5}$$

$$recall = \frac{TP}{TP + FN}, \tag{6}$$

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{7}$$

$$FPR = \frac{FP}{FP + TN}, \tag{8}$$

$$TPR = \frac{TP}{TP + FN}, \tag{9}$$

$$AUC = \int TPR \cdot d(FPR), \tag{10}$$

where *TP* – True Positive, *FP* – False Positive, *TN* – True Negative, *FN* – False Negative, *TPR* – True Positive Rate and *FPR* – False Positive Rate.

**Table 3. Results of training and validation accuracy for Resnet, MobileNet and InceptionV3**

| Accuracy (%) | InceptionV3 | MobileNet | ResNet |
|---|---|---|---|
| Training | 98.78 | 99.57 | 99.62 |
| Validation | 93.94 | 97.31 | 97.16 |

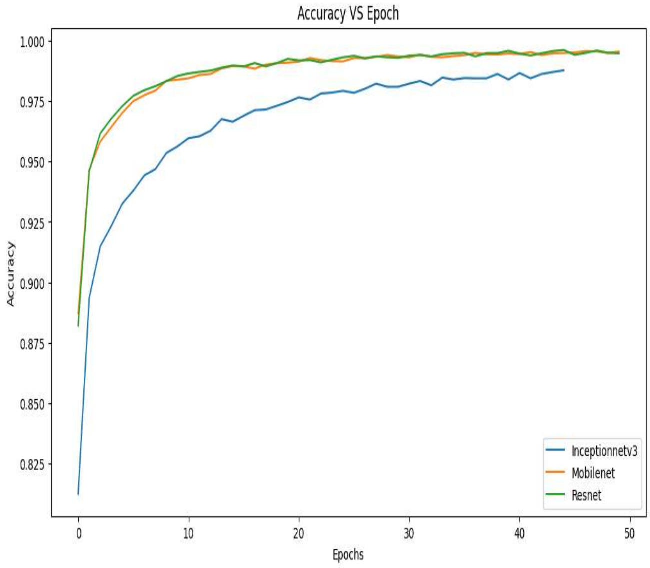

Figure 6. Graph showing the accuracy comparison between ResNet, MobileNet and InceptionV3 as shown in Table 3.

**Table 4. Results of training and validation loss for InceptionV3, MobileNet and Resnet**

| Loss (%) | InceptionV3 | MobileNet | ResNet |
|---|---|---|---|
| Training | 3.79 | 1.27 | 1.19 |
| Validation | 21.56 | 10.16 | 11.16 |

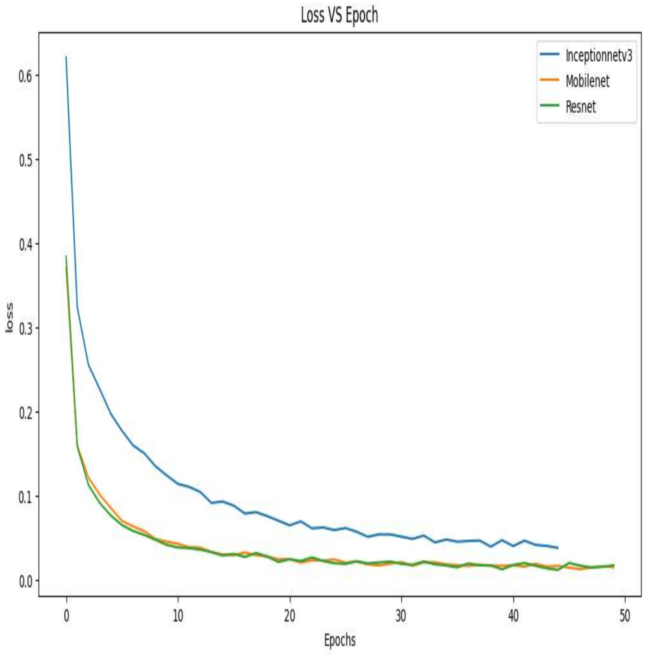

Figure 7. Graph showing loss comparison between ResNet, MobileNet and InceptionV3 as shown in Table 4.

**Table 5. Results of training and validation of Precision for InceptionV3, MobileNet and ResNet**

| Precision (%) | InceptionV3 | MobileNet | ResNet |
|---|---|---|---|
| Training | 97.17 | 98.84 | 98.89 |
| Validation | 93.36 | 96.88 | 96.64 |

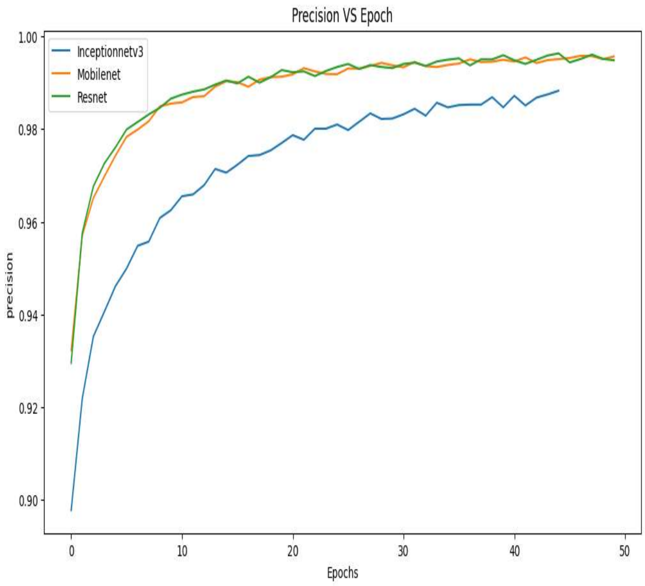

Figure 8. Graph showing the precision comparison between ResNet, MobileNet and InceptionV3 as shown in Table 5.

**Table 6. Comparison table of various models according to recall, precision, F1-score and AUC during training time**

| Models | Precision | Recall | F1-score | AUC |
|---|---|---|---|---|
| Inception V3 | 0.9717 | 0.9874 | 0.9794 | 0.9994 |
| MobileNet | 0.9884 | 0.9955 | 0.9913 | 0.9997 |
| ResNet | 0.9889 | 0.9948 | 0.9918 | 0.9995 |

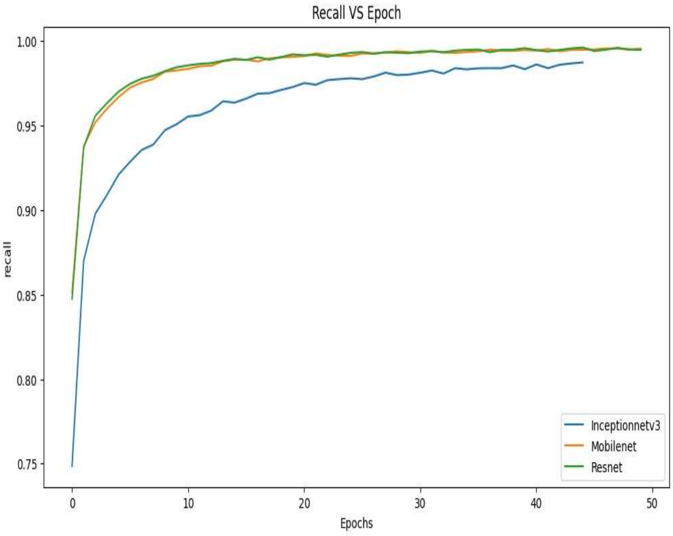

Figure 9. Graph showing recall comparison between ResNet, MobileNet and InceptionV3 as shown in Table 6.

**Table 7. Comparison table of various models according to their recall, precision, F1-score and AUC during validation time**

| Models | Precision | Recall | F1-score | AUC |
|---|---|---|---|---|
| Inception V3 | 0.9317 | 0.9285 | 0.93 | 0.9882 |
| MobileNet | 0.9736 | 0.9728 | 0.9731 | 0.9947 |
| ResNet | 0.9700 | 0.9693 | 0.9696 | 0.9935 |

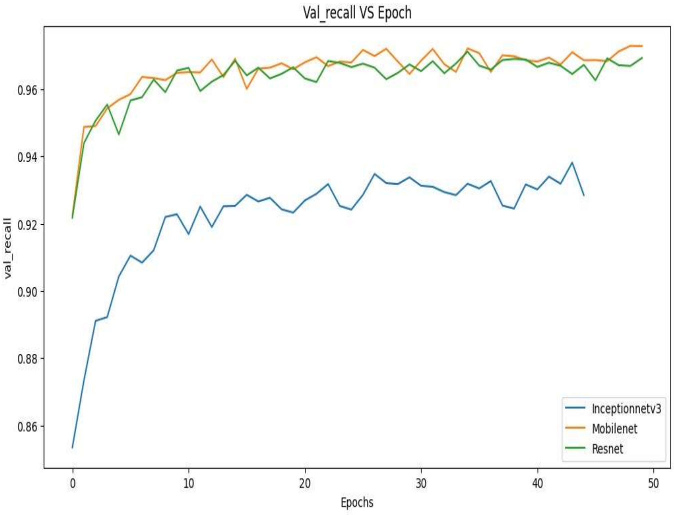

Figure 10. Graph showing recall comparison between ResNet, MobileNet and InceptionV3 as shown in Table 7.

**Table 8. Inference Time taken by various models**

| Models | Inference time (msec. per image) |
|---|---|
| Inception V3 | ~503 |
| MobileNet | ~300 |
| ResNet | ~407 |

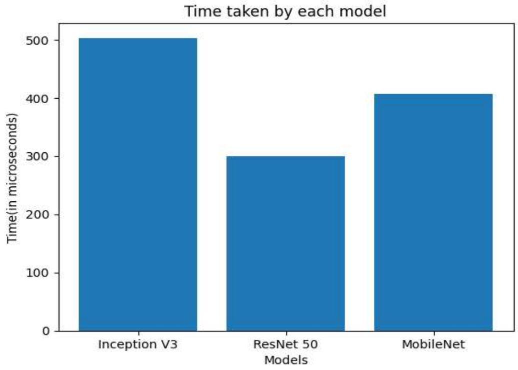

Figure 11. Bar graph showing the comparison between training time taken by InceptionV3, MobileNet and ResNet as shown in Table 8.

## V. CONCLUSION

In this paper, we have compared the different state-of-the-art models InceptionV3, MobileNet, and ResNet50 to classify various plant diseases using a new plant disease dataset that contains approximately 87900 images of various plant leaves. In this comparison, we found that ResNet50 with an accuracy of 99.62% performs better than MobileNet and InceptionV3, with the accuracy of 99.57% and 98.78%, respectively. We have also concluded that, in terms of loss, ResNet50 performs better than MobileNet and InceptionV3. ResNet50 shows a loss of 1.19%, followed by MobileNet and InceptionV3 with the loss of 1.27% and 3.79%, respectively. The inference times for MobileNet, ResNet50 and InceptionV3 is ~300msec, ~407msec and ~503msec respectively.

In the future, these models can be extended for real time on much larger datasets and can also be integrated with mobile applications for easier use by the common population. We can also improve the models to make them more lightweight, less time-consuming to train, and more accurate in real time.

## References

[1] *World food and agriculture - Statistical pocketbook*, United Nation, Rome, 2018, pp. 254.

[2] *Annual Report 2020-21*, Department of Agriculture, Cooperation & Farmers' Welfare, Ministry of Agriculture & Farmers' Welfare, Government of India, Krishi Bhawan, New Delhi, [Online]. Available at: https://agriwelfare.gov.in/Documents/annual-report-2020-21.pdf.

[3] M. Badiger, V. Kumara, S. C. N. Shetty, S. Poojary, "Leaf and skin disease detection using image processing," *Global Transitions Proceedings*, vol. 3, issue 1, pp. 272-278, 2022. https://doi.org/10.1016/j.gltp.2022.03.010.

[4] S. P. Singha, K. Pritamdasa, K. J. Devia, S. D. Devi, "Custom convolutional neural network for detection and classification of rice plant diseases," *Procedia Computer Science*, vol. 218, pp. 2026–2040, 2023. https://doi.org/10.1016/j.procs.2023.01.179.

[5] D. A. Noola, D. R. Basavaraju, "Corn leaf image classification based on machine learning techniques for accurate leaf disease detection," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 12, no. 3, pp. 2509-2516, 2022. https://doi.org/10.11591/ijece.v12i3.pp2509-2516.

[6] Y. Aufar, T. P. Kaloka, "Robusta coffee leaf diseases detection based on MobileNetV2 model," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 12, no. 6, pp. 6675-6683, 2022. https://doi.org/10.11591/ijece.v12i6.pp6675-6683.

[7] K. P. Ferentinos, "Deep learning models for plant disease detection and diagnosis," *Computers and Electronics in Agriculture*, vol. 145, pp. 311-318, 2018. https://doi.org/10.1016/j.compag.2018.01.009.

[8] F. A. Guth, S. Ward and K. McDonnell, "From lab to field: An empirical study on the generalization of convolutional neural network towards crop disease detection," *European Journal of Engineering and Technology Research*, vol. 8, issue 2, pp. 32-40, 2023. https://doi.org/10.24018/ejeng.2023.8.2.2773.

[9] S. Nuanmeesri, "A hybrid deep learning and optimized machine learning approach for rose leaf disease classification," *Engineering, Technology & Applied Science Research*, vol. 11, no. 5, pp. 7678-7683, 2021. https://doi.org/10.48084/etasr.4455.

[10] S. Alqethami, B. Almtanni, W. Alzhrani and M. Alghamdi, "Disease detection in apple leaves using image processing techniques,"

*Engineering, Technology & Applied Science Research*, vol. 12, no. 2, pp. 8335-8341, 2022. https://doi.org/10.48084/etasr.4721.

[11] L. Loyani and D. Machuve, "A deep learning-based mobile application for segmenting Tuta Absoluta's damage on tomato plants," *Engineering, Technology & Applied Science Research*, vol. 11, no. 5, pp. 7730-7737, 2021. https://doi.org/10.48084/etasr.4355.

[12] B. M. Joshi and Dr. H. Bhavsar, "Deep learning technology based Night-CNN for nightshade crop leaf disease detection," *International Journal of Intelligent System and Application in Engineering*, vol. 11, no. 1, pp. 215-227, 2023.

[13] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," CoRR, vol. abs/1512.03385, 2015. https://doi.org/10.1109/CVPR.2016.90.

[14] C. Szegedy, V. Vanhoucke, S. Ioffe and J. Shlens, "Rethinking the inception architecture for computer vision," CoRR, vol. abs/1512.00567, 2015. https://doi.org/10.1109/CVPR.2016.308.

[15] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto and H. Adam, "MobileNet: Efficient convolution neural network for mobile vision application," CoRR, vol. abs/1704.04861, 2017.

[16] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke and A. Rabinovich, "Going deeper with convolutions," CoRR, vol. abs/1409.4842, 2014. https://doi.org/10.1109/CVPR.2015.7298594.

[17] T. Poggio, A. Banburski and Q. Liao, "Theoretical issues in deep learning," *Proceedings of the National Academy of Science*, vol. 117, no. 48, pp. 30039-30045, 2020. https://doi.org/10.1073/pnas.1907369117.

[18] New Plant Diseases Dataset. [Online]. Available at: https://www.kaggle.com/datasets/vipoooool/new-plant-diseases-dataset

[19] PlantVillage-Dataset. [Online]. Available at: https://github.com/spMohanty/PlantVillage-Dataset

[20] J. Wang and L. Perez, "The effectiveness of data augmentation in image classification using deep learning," arXiv, vol. abs/1712.04621, 2017.

[21] D. P. Kingma and J. Ba, "Adam: A Method for stochastic optimization," arXiv, ICLR-2015. https://doi.org/10.48550/arXiv.1412.6980.

[22] Z. Zhang and M. R. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," vol. 13, no. 9, arXiv, 2014.

[23] M. Lin, Q. Chen and S. Yan, "Network in network," arXiv, 2014.

[24] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, "Dropout: A simple way to prevent neural network from overfitting," *A Journal of Machine Learning Research*, vol. 15, pp. 1929-1958, 2014.

[25] K. Hara, D. Saito and H. Shouno, "Analysis of function of rectified linear unit used in deep learning," arXiv, 2015. https://doi.org/10.1109/IJCNN.2015.7280578.

***MANJIT JAISWAL** is currently working as an Assistant professor in Computer Science and Engineering Department, School of Studies in Engineering and Technology, Guru Ghasidas Vishwavidyalaya, A Central University Bilaspur, Chhattisgarh, India. He received Master of Technology(M.Tech.) degree in 2012 from Maulana Azad National Institute of Technology, Bhopal, Madhya Pradesh, India. He is persuing Ph.D. in Computer Science and Engineering Department, School of Studies in Engineering and Technology, Guru Ghasidas Vishwavidyalaya, A Central University Bilaspur, Chhattisgarh, India. He has published more than 13 papers in reputed journals and conference like IEEE, Scopus indexed approved etc. His research interest fields are algorithm, Machine Learning and Deep Learning.He has more than 10 years of teaching experience. He is a member of IAENG.*

***DR KAPIL KUMAR NAGWANSHI** has received his PhD from the Chhatisgarh Swami Vivekanand Technical University Bhilai, India. He is currently working as an Associate Professor at SoS E&T Guru Ghasidas Vishwavidyalay (A Central University), Bilaspur, India. His primary domain of teaching and research includes the internet of things, digital image processing, cyber forensics, data science and engineering, AI, and computer networking. He has guided 15 MTech scholars and currently supervising six PhD scholars. He is a senior member of IEEE, YHAI, and a life member of CSI, IETE, and members of IAENG, IACSIT, and some other professional bodies. He is a reviewer of reputed journals such as IEEE Access, Imaging Science Journal, Journal of Real-Time Image Processing, and International Journal of Computer and Electrical Engineering.*

***ABHISHEK JAIN** as an Assistant Professor in Information Technology Department, Central University, Bilaspur, India and having 14 years of industrial, teaching and research experience. He has completed B.E. from GEC Raipur and MTech from NIT Allahabad currently pursuing PhD from Central University Bilaspur C.G. He is also worked as senior software engineer in TCS. He has authored more than 10 scientific publications in peer reviewed journals. He has worked in Geographical Information Systems (GIS) project Thames Water London UK and RAPDRP Gujarat. His industrial working experience include object-oriented analysis modelling and design, database administrator, Oracle, SQL, XML, XSLT, DTD, Java etc. His research focused on remote sensing, image processing, signal processing biomedical engineering, EEG signal, ECG Signal, IOT, blockchain technology.*

***RISHIKESH KUMAR** is currently pursuing Bachelor of Technology in Computer Science and Engineering from School of Studies in Engineering and Technology, Guru Ghasidas Vishwavidyalaya, Bilaspur, India. He is currently studying the final year. Her research interest fields are Machine Learning, Computer Vision and algorithm.*

***SHREYASH GAURAV** is currently pursuing Bachelor of Technology in Computer Science and Engineering from School of Studies in Engineering and Technology, Guru Ghasidas Vishwavidyalaya, Bilaspur, India. He is currently studying the final year. Her research interest fields are Machine Learning, Computer Vision, and Algorithm.*

***YUKTA WATTI** is currently pursuing Bachelor of Technology in Computer Science and Engineering from School of Studies in Engineering and Technology, Guru Ghasidas Vishwavidyalaya, Bilaspur, India. She is currently studying the final year. Her research interest fields are Machine Learning, Computer Vision.*

***DR. ANULAL MAHTO** is currently working in Guru Ghasidas Vishwavidyalaya (A Central University), Bilaspur, India. He did his Ph.D. from Birla Institute of Technology, Meshra, Ranchi, India. His research Areas are Ergonomics, Artificial Intelligence, Industrial labour psychological analysis and Manufacturing*

***MR. SATYENDRA SINGH THAKUR** is currently pursuing Ph.D. from Computer Science and Engineering, SoS(E&T), Guru Ghasidas Vishwavidyalaya (A Central University), Bilaspur, India. He did M.Sc. in Computer Science from Rajasthan Vidyapeeth University, Udaipur, Rajasthan, India. His research Areas are Artificial Intelligence, Geographical Information System and Image Processing.*