International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-10 Issue-6, August 2021

Performance Comparison of Deep CNN Models for Disease Diagnosis on Apple Leaves

Kota Akshith Reddy, Sharmila Banu K, Ippagunta Sai Kanishka, Chandra Havish Siddareddi, Polsani Jahnavi

Abstract: ***The apple is one of the most cultivated fruits in the world. They are round in shape and their color varies from green to red. Apple Orchards face constant threats from a large number of insects and pathogens and the early detection of these diseases can help in mitigating these harmful effects. An apple tree takes around six to ten years to mature and produce fruit and therefore, the production costs are high and there is no room for such diseases to get a healthy fruit and a profitable yield. Delayed or incorrect diagnosis of these diseases can lead to using either inadequate or more than required chemicals or using a wrong chemical altogether to treat the plant. Historically, this problem was solved using conventional machine learning algorithms like SVMs, Decision Trees and Random Forests. However, in recent times, the approach to solve this problem has shifted to deep learning, specifically Convolutional Neural Networks. CNN's are powerful tools that can be used for image classification. We can get state-of-the-art results even by using small amounts of data and little to no data preprocessing. In this work, we are going to compare some of the state of the art CNN architectures on the task of accurately classifying a given image into different categories of diseases or as a healthy leaf. Finally, experimental results are conveyed and performance analysis of these various architectures has been done.***

Keywords: Apple Orchards,Convolutional Neural Networks , Decision Trees, Machine Learning , Multilayer Perceptron Neural Networks.

I. INTRODUCTION

Due to its rich nutritional and medicinal value, Apple has become one of the most cultivated fruits in the world.

Manuscript received on August 02, 2021.
Revised Manuscript received on August 10, 2021.
Manuscript published on August 30, 2021.
* Correspondence Author
Kota Akshith Reddy*, Department of Computer Science Major, Vellore Institute of Technology, Vellore (Tamil Nadu), India.
Sharmila Banu K, Assistant Professor, Department of Computer Science and Engineering, Vellore Institute of Technology, Vellore (Tamil Nadu), India.
Sai Kanishka Ippagunta, Pursuing, B.Tech (CSE), Department of Information Security, Vellore Institute of Technology, Vellore (Tamil Nadu), India.
Chandra Havish Siddareddi, Department of Electronics and Communications Engineering, Vellore institute of Technology, Vellore (Tamil Nadu), India.
Jahnavi Polsani, Pursuing, B.Tech, Department of Computer Science Engineering (CSE), Vellore Institute of Technology, Vellore (Tamil Nadu), India.

However, the downside to this is that the Apple tree, like many other fruit-bearing trees, is susceptible to various fungal, bacterial, and viral infections and if not treated properly in the early stages can lead to the decreased yield, a cosmetic appearance that can be unappealing, poor quality of fruit which can lead to low marketability leading to huge economic losses [10]. Early detection of these diseases and pathogens is critical for the timely deployment of pest control strategies, thereby mitigating the harmful effects of these pathogens. Therefore, it is of great importance that we develop architectures that can detect the pathogen in the early stages and such systems will have a significant impact on the agriculture sector, especially in the areas where agriculture is the primary sector.

Conventionally, the type and severity of the plant diseases are decided by trained experts by examining the plant tissue. But this process is expensive, time-consuming and error-prone. With the advancement in the field of deep learning, many Deep Neural Networks are showing promising results on the task of plant disease classification. To this end, this research work aims on developing a suitable solution for apple leaf disease prediction and furthermore compare the performance of various state of the art models. In this work, we look at the performance of Resnet [1], EfficientNet [4] and Xception [7] on the task of apple plant disease classification using the plant pathology dataset [10] which is the same dataset used in the plant pathology challenge in Kaggle. The performance of the above-mentioned models was measured using various evaluation metrics like precision, recall, f1-score and the best-suited approach for the plant disease classification has been determined. The materials and methods section explains the background and inner workings of the above-mentioned models. The section under implementation explains software and hardware requirements for the task, overview of the dataset, some basic data analysis of the dataset, and the parameter settings of each of the CNN architectures. The results and discussions section deals with the experimental findings of this research work. Finally, the conclusion section summarizes this work in brief and talks about the advantage of using such models to detect diseases on leaves.

II. MATERIALS AND METHODS

A. The ResNet Model

In theory, the training and testing error of a Convolutional Neural Network should decrease with an increase in the depth of the network. But that is not the case in practice as the training accuracy actually decreases for some point and increases as we keep on stacking more and more layers. Vanishing and exploding gradients make it very difficult to train deep neural networks.. Some of the solutions to this problem are to reduce the number of layers or perform gradient clipping. The ResNet Model resolves this problem by introducing skip connections between the layers, thereby making it possible for the gradients to flow backward from later layers to initial filters [1,2]. These skip connections add the outputs from layers to the outputs of the layers which are stacked later, thereby making it possible to train a much deeper neural network. The authors of the ResNet model tested the model with 100 and 1000 layers on the CIFAR10 dataset

Resnet's architecture takes inspiration from the VGG16 model and uses 3*3 convolutional filters. ResNet's architecture can be variable and it depends on how big the layers are and how many of them are present. The model described in [1] consists of a convolution and a pooling layer followed by 4 sets of layers with the same behavioral patterns. There are a total of 152 layers. The first of the four similar layers does 3*3 convolution with a map dimension of 64. The next 3 layers observe a comparable. The input is bypassed after every two convolutions. Identity mapping is performed by these skip connections. Two heuristics from the VGG network [3] were followed to test these skip connections: If the output feature maps have the same resolution, then the depth of the filter map remains the same or else if the resolution is halved, then the depth of the filter map is doubled.

B. The EfficientNet Model:

Scaling up ConvNets is a general practice to increase the accuracy of the model. In practice, it is common to scale any one of the following three dimensions: depth, width and image size. It is possible to scale more than one dimension arbitrarily but it is tedious and requires manual tuning. The intuition behind increasing the depth of the network is that with this increase in the number of layers, the network will be able to capture more complex and richer features than their shallow counterparts. But that is not always the case because of the vanishing and exploding gradient problem [5]. The reason for increasing the width of the network is that the wider networks are easier to train and will be able to capture more fine-grained features. However, shallow networks that are extremely wide face difficulties while capturing higher level features. Increasing the resolution of the image helps ConvNets in capturing fine-grained patterns, thereby increasing classification accuracy but these accuracy gains are plateaued with very high resolutions of images. With this increase in resolution of the images, intuitively, one has to increase the depth of the model as well so that the larger receptive fields can help capture the similar features in these high-resolution images. The vice versa is also true in order to capture the fine-grained patterns in these bigger images. So, in order to increase the accuracy of this model, one has to balance all the three dimensions rather than concentrating on any one of the dimensions.

The EfficientNet paper [4] introduces a simple yet effective method of compound scaling wherein the network depth, width and resolution of the image are scaled uniformly by a fixed set of scaling coefficients. We can simply increase the network depth , width and the resolution of the image by a constant. These constant coefficients can be found out by applying a grid search on a smaller model as the search cost becomes more expensive on larger models. The base architecture of the EfficientNet model is obtained by applying Neural Architecture Search that optimizes both accuracy and FLOPS. The base EfficientNet-B0 is based on the inverted bottleneck residual blocks (MBConv) present in the MobileNetV2 architecture in addition to the squeeze-and-excitation blocks.

C. The Xception Model:

The Xception model is inspired by the Inception model [6]. The original depth-wise separable convolution in the Inception network was performed in such a way that the channel-wise spatial convolution was implemented first followed by 1*1 convolution. But the changed depthwise separable convolution withinside the Xception network [7] plays the 1*1 convolution first that's then accompanied by means of channel-sensible spatial convolutions. Unlike the general deep CNN models which employ a channel and spatial distribution, the Xception model employs a pointwise and depth-wise convolution. Another difference from the Inception network is that this Xception network has no intermediate ReLU non-linearity. It has been shown in [7] that the Xception model was able to achieve more accuracy without any activation function when compared to models using ELU or ReLU. It has also been shown that the residual connections like that of those in the ResNet model helped with convergence in the Xception model, in terms of classification and speed. It has 71 layers and has been trained using the ImageNet dataset. It can learn rich interpretations of a wide range of images and can classify hundreds of object categories.

D. The VGG16 model:

The VGG16 model was inspired from the AlexNet architecture and it outperforms the AlexNet achieving a test accuracy of 92.7% on ImageNet [3]. Unlike AlexNet, VGG16 adopted small-size filters of size 3*3 rather than large-size filters.

The architecture was constructed using only layers of pooling and convolution with a total of 138 million parameters. It contains 13 layers of convolution, 5 layers of pooling and 3 fully connected layers. The image is sent through 2 sets of 2 layers of convolution before being sent to 1 layer of max pooling. This is followed by 3 sets of 3 convolutional and 1 max pooling layer. After the last layer of pooling, 3 fully connected layers are added with a softmax activation at the final dense, fully connected layer.

Due to the reduced size of the kernels, the model is not that computationally expensive when compared with the previous models. The convolutional layer has a small receptive field because of the small size of the filters thereby preserving the image resolution. Zero padding has been used to pad the input image and avoid rapid decrease in size of the image because of the convolutions. The max pooling is performed with a stride of 2 across a 2*2pixel window.

ReLU activation feature is used withinside the hidden layers. The main advantage of using this ReLU activation is its ability to reduce the likelihood of vanishing gradient problem and faster learning. However, the main disadvantage of VGG16 is that it is slow to train. The original model was trained for two to three weeks on Nvidia Titan GPU.

III. IMPLEMENTATION

A. Hardware Requirement

The workstation used for this work was Dell Inspiron 5579 with a 64-bit Intel(R) Core(TM) i5 8th generation octa-core processor and a CPU. The operating system was supported by touch inputs and had 8 GB RAM. The camera used for this work was inbuilt in the workstation which had a resolution of 0.92 megapixel and the video resolution was 1280 * 720 at 30 fps. However, to leverage the power of GPU computing, execution was done in a Kaggle kernel using the Tesla P100 GPU.

B. Software Requirement

The software requirements include a python environment along with several python libraries like Pandas, Numpy, Matplotlib and Sklearn. Further, the CNN's were implemented by leveraging the power of transfer learning using open-source deep learning libraries like Keras and Tensorflow. All the work is primarily done using the python programming language along with some secondary support from Microsoft Excel.

C. Dataset Description

The dataset used in this work was obtained from the Plant Pathology FGVC7 workshop of CVPR 2020 [10]. It was compiled by the Cornell Initiative for Digital Agriculture and consisted of 3642 novel images, half of which are in the training set and the other half in the testing set. All the images had the same dimensions, 1365*2048*3. The training dataset contains 1821 images along with a class associated with each and every image.

The total number of classes is four: 'Healthy', 'Multiple Diseases', 'Rust' and 'Scab'. The number of images under the class 'Multiple Diseases' are a lot less when compared with other classes. It is safe to say that the dataset is unbalanced and accurately classifying the images under 'Multiple Diseases' can be a deciding factor in selecting the best suitable CNN architecture for the task. Four sample images each from one class is shown in Fig. 1.

(a) Healthy

(b) Multiple Diseases

(c) Rust

(d) Scab

Fig. 1. Sample images from the dataset

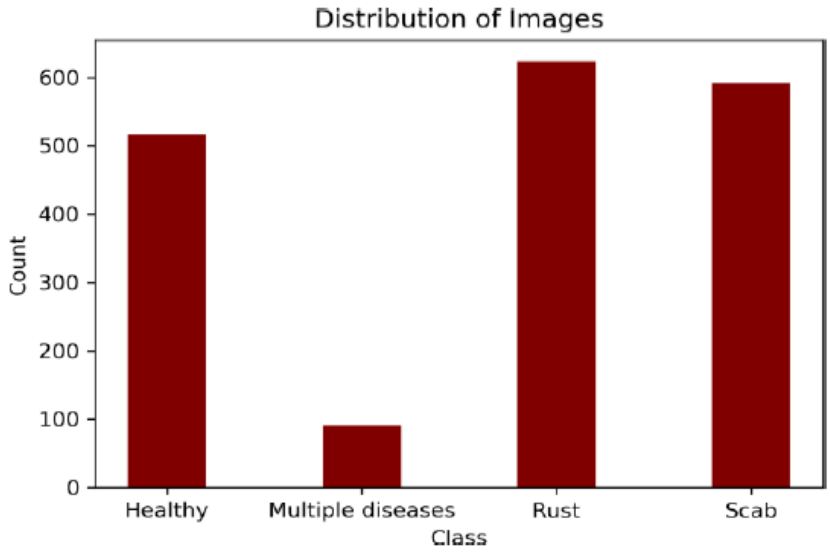

Fig. 2. Distribution of Images over the 4 classes

D. Data Augmentation:

Before training, the images are augmented using data augmentation to make learning 'harder' so that the model can learn better and more generalized representations thereby reducing the risk of overfitting the model on the training data. The following image augmentation techniques were applied in this work: rotating the image horizontally, rotating the image vertically, shifting the image horizontally by ten percent of the width, shifting the image vertically by ten percent of the height, rotating the image in the clockwise direction by 10 degrees, randomly changing the brightness of the image by a factor whose lower limit is half the brightness of the original image and the upper limit is one and a half times the brightness of the original image, randomly zooming the image by a factor between 90 percent (zoom in) and 110 percent (zoom out). Before doing all these operations, all the pixels in the image have been rescaled by a factor of 1/255.

E. Execution of Resnet Model:

The training for the Resnet model was done for a total of 15 epochs on a batch size of 32

and various performance metrics were calculated for the model. The model was fine-tuned on the 'resnet50' architecture from the Keras library using the weights of the pretrained Resnet model on the 'Imagenet' dataset. The shape of the input image was set to (384,384,3).The final output layer of the original Resnet architecture was removed and the following three layers were added: one Global Average Pooling layer, 2 dense layers with 128 and 64 units and an output dense layer with 4 units using the softmax activation. Finally, to further the adaptive learning rate, Adam optimizer was used. The model was trained for 15 epochs with 100 steps per epoch. The rate of learning was decreased by a factor of 0.3 after 3 epochs of showing no improvement on the validation accuracy. The lower boundary for the learning rate was set to 0.000001 and the shuffle parameter was set to true in order to shuffle the training data before each epoch. The architecture of this model is shown in Fig. 3.

Retrieval Number: 100.1/ijeat.F30400810621
DOI:10.35940/ijeat.F3040.0810621
Journal Website: www.ijeat.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication

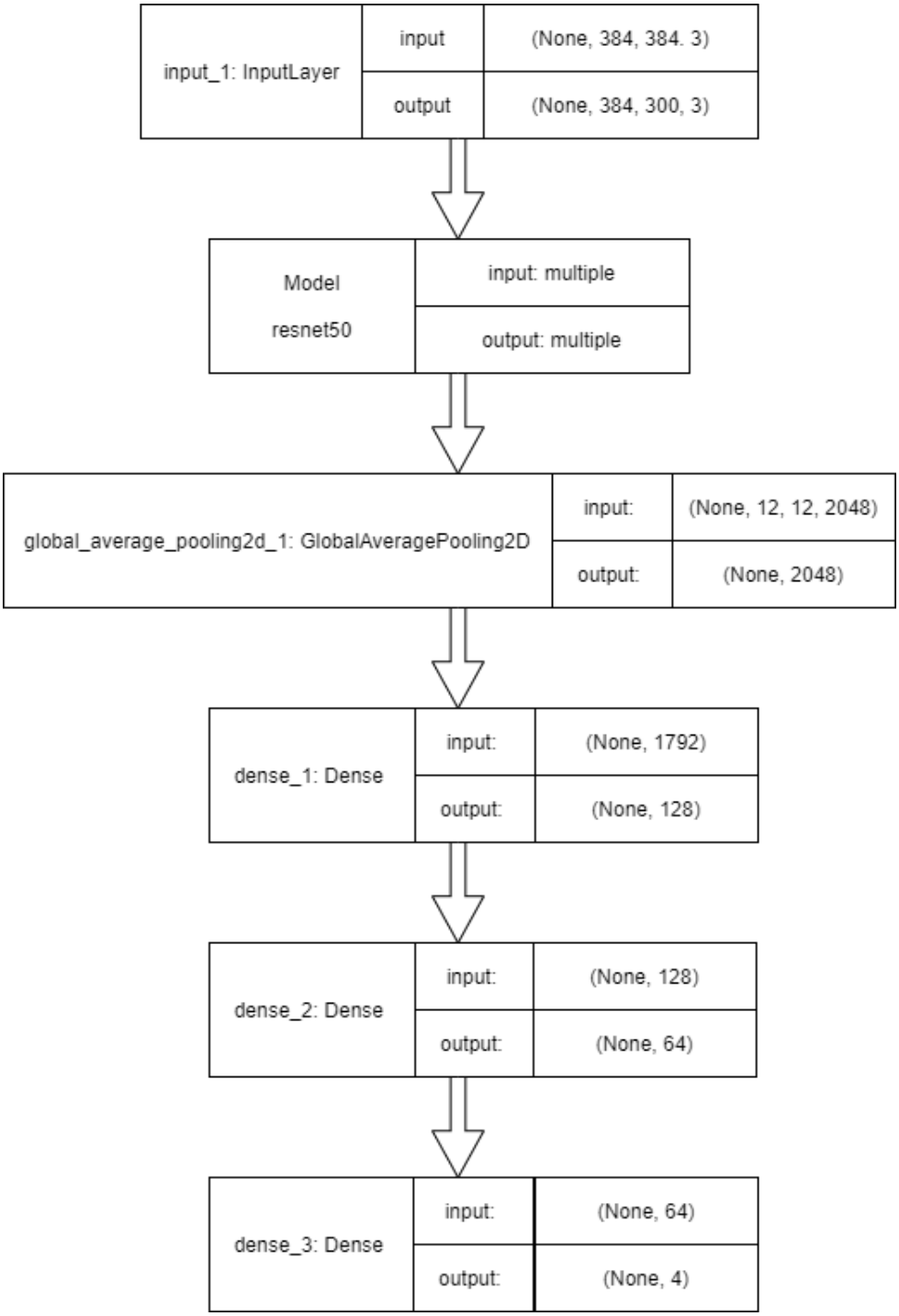

Fig. 3. The ResNet Model description

F. Execution of EfficientNet Model:

The training for the EfficientNetB4 was done for a total of 15 epochs on a batch size of 16 and was set up using transfer learning. The last layer of this architecture was removed and the following layers were added: one Global Average Pooling layer, two layers of dense connections with one twenty eight and sixty four units respectively.

Finally, to further the adaptive learning rate like in the case of the Resnet model, Adam optimizer was used. Categorical cross-entropy was set as the loss function. The input size was (300,300,3). 91 steps were performed in each epoch. The adaptive learning rate was reduced by a factor of 0.3 with patience of 3 epochs. The lower bound for the rate of learning was set to 0.000001 and the shuffle parameter was set to true. The architecture of this model is shown in Fig. 4.

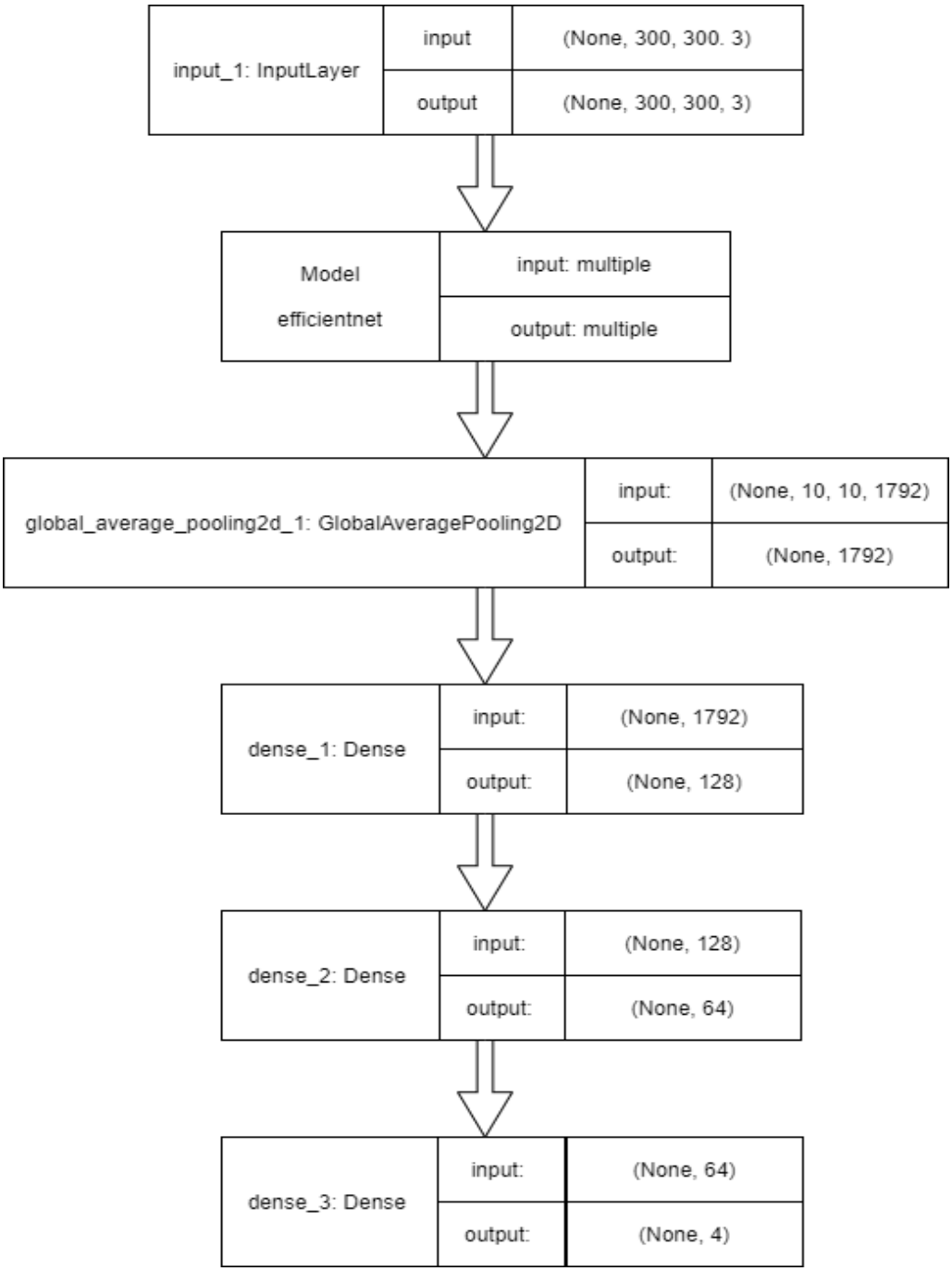

Fig. 4. The EfficientNet Model description

G. Execution of Xception Model:

The Xception model was trained for 15 epochs. The input image size was set to (299,299,3) and the batch size was set to 32. The model was set up using transfer learning. For the purpose of 4 class classification, the last dense layer containing 1000 units was removed from the architecture and the following three layers were added: one Global Average Pooling layer, two layers of dense connections with one twenty eight and sixty four units respectively using the Relu activation and an output layer of dense connections using the softmax activation. We have set up an adaptive learning rate using Adam optimizer and the rate of learning was reduced by a factor of 0.3 after patience of 3 epochs. The loss function was set to categorical cross-entropy and each epoch was run for 45 steps. The lower bound of the rate of learning was set to 0.000001. The architecture of this model is shown in Fig. 5.

Retrieval Number: 100.1/ijeat.F30400810621
DOI:10.35940/ijeat.F3040.0810621
Journal Website: www.ijeat.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication

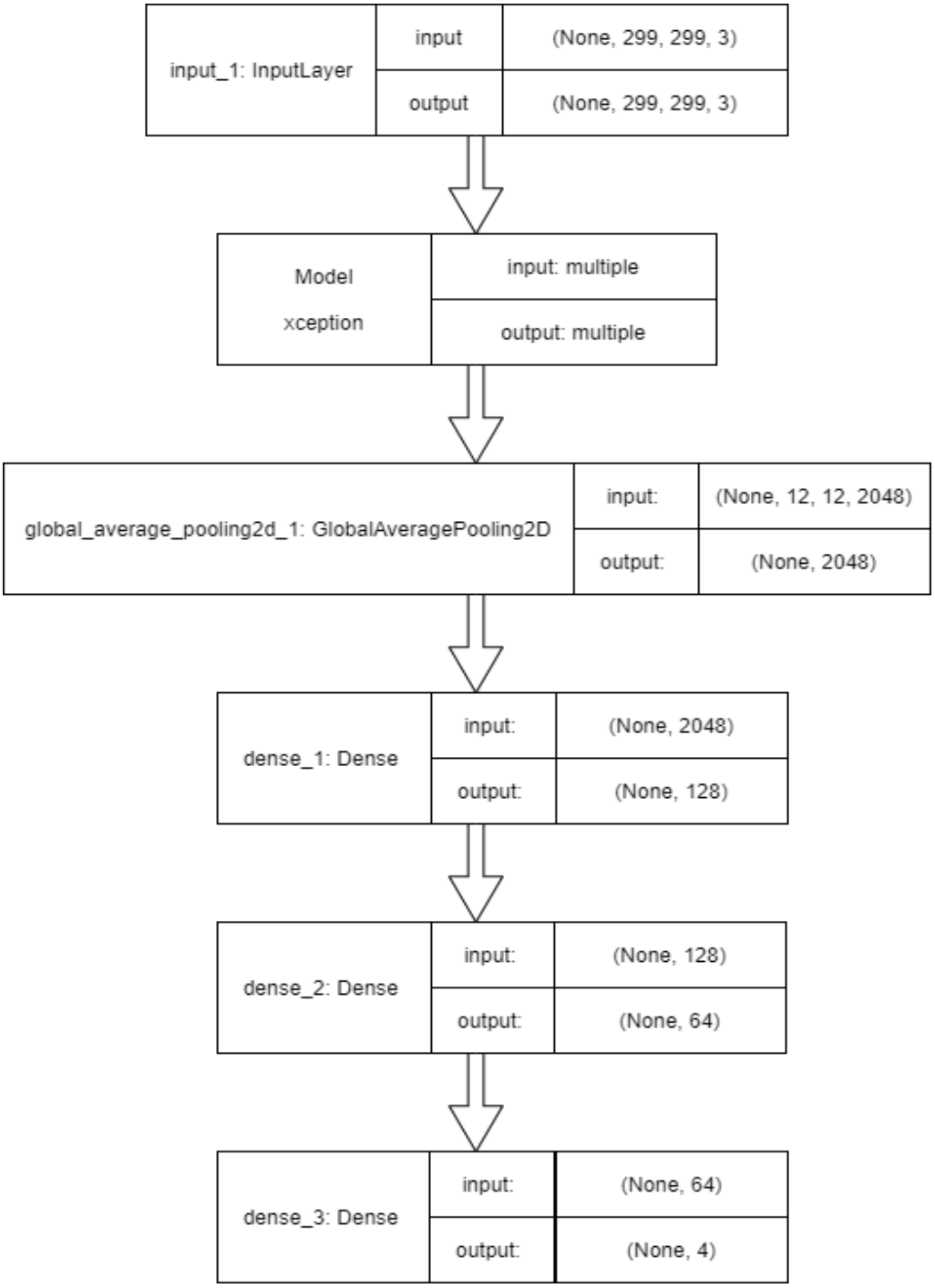

Fig. 5. The Xception Model description

H. Execution of VGG16 model:

The VGG16 model was trained for a total of 30 epochs with a batch size of 32 and input size of (224,224,3). The model was constructed using transfer learning. The original output layer was removed and 1 global average pooling along with 2 fully connected layers of 128 and 64 units and ReLU activation function was added. Finally, an output fully connected, dense layer with 4 units was added with softmax activation function for predicting the final probabilities. Categorical cross- entropy was set as the loss function. Adam optimizer was used for adaptive learning and the learning rate whose lower bound was set to 0.000001. The VGG16 model's architecture is shown in Fig. 6.

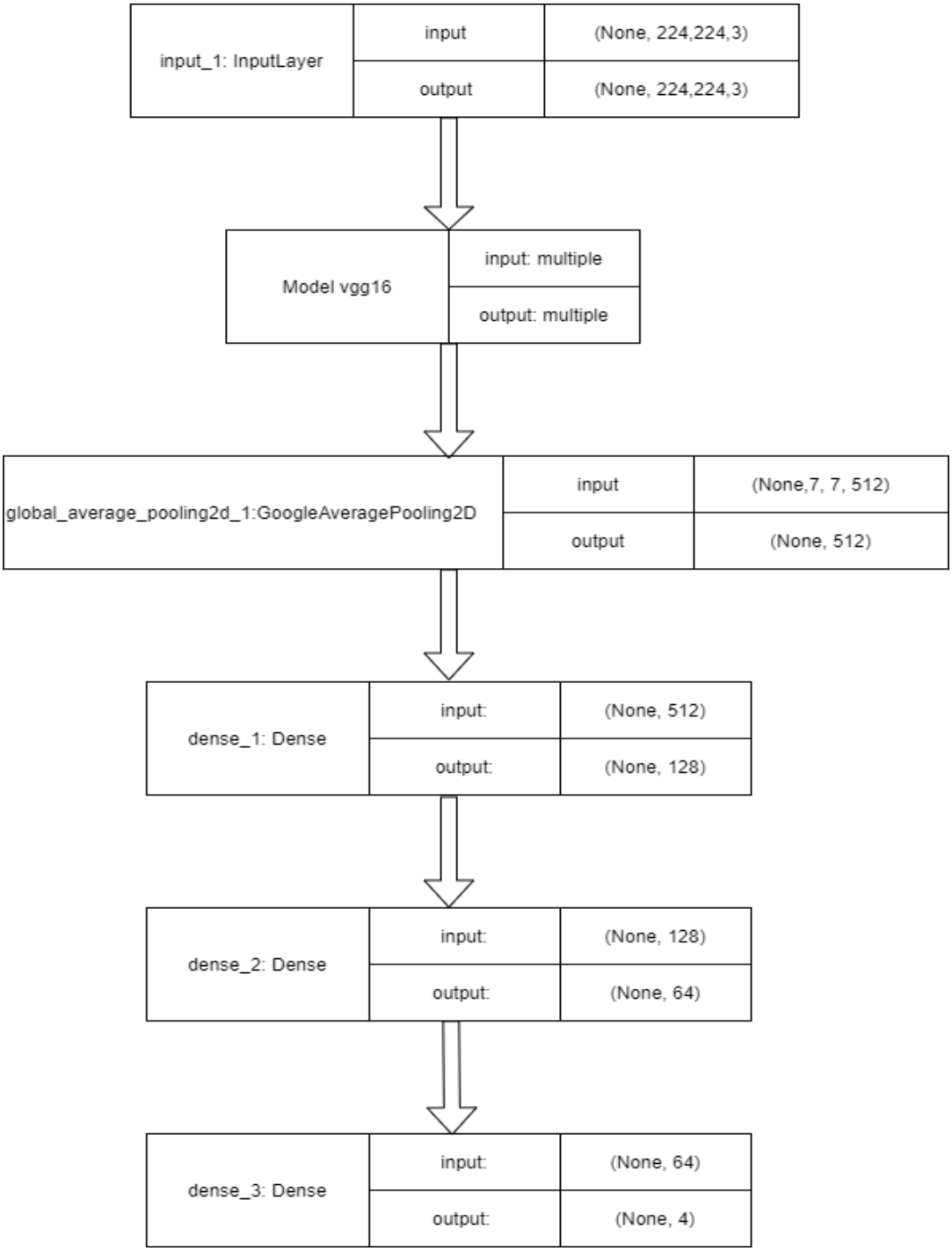

Fig.6. The VGG16 model description

IV. RESULTS AND DISCUSSIONS

Performance analysis of the models by testing the following metrics: Overall Accuracy score of the model, Precision, Recall and F1-score [8,9]. The results are visualized with the help of the matplotlib library for better understanding.

The results in Table-I represent the accuracy scores of the models. ResNet and Xception models have scored almost the same accuracy. However, the EfficientNet model has outperformed the other two by a mere two percent with an accuracy score of approximately 98 percent

The results in Table-II represent the evaluation metric scores of the Efficient model over the four classes. The highest score of precision was observed in 'rust' , highest recall and F1 score for the 'healthy' class. Least precision score, recall score and F1 score was for the 'Multiple Diseases' class.

Retrieval Number: 100.1/ijeat.F30400810621
DOI:10.35940/ijeat.F3040.0810621
Journal Website: www.ijeat.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication

Table-I: Accuracy scores of the models

Model	Accuracy score
EfficientNet	0.9753424657534246
ResNet	0.9562043795620438
Xception	0.9561643835616438

Table-II: EfficientNet model's performance in classifying the diseases on Apple Leaves

Class (Disease)	Precision score	Recall score	F1 score
Healthy	0.98113208	0.99047619	0.98578199
Multiple Diseases	0.76923077	0.76923077	0.76923077
Rust	0.98449612	0.98449612	0.98449612
Scab	0.98290598	0.97457627	0.9787234

The visualization in Fig. 7 shows the training and testing loss over the epochs and in Fig. 8**,** we can see the evaluation metrics of the EfficientNet model. Overall, the model is able to classify three out of four classes almost perfectly but is struggling with the classification of the 'Multiple Diseases' class.

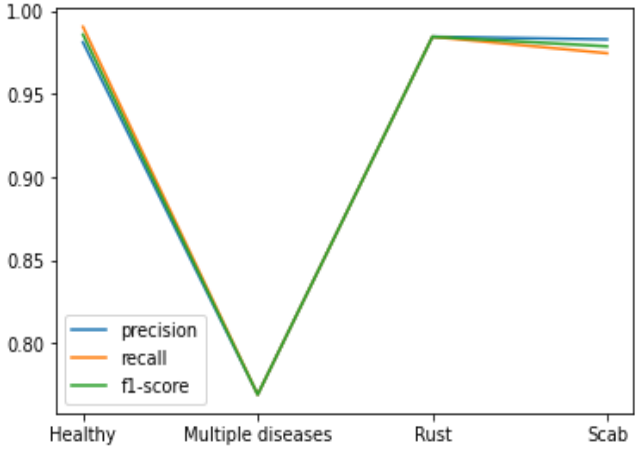

Fig. 7. EfficientNet- Evaluation metric scores

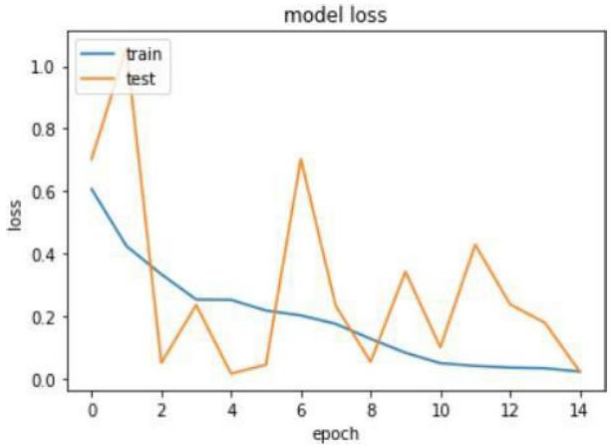

Fig. 8. Train vs Test loss over the epochs for the Efficient model

The results in Table-III represent the evaluation metrics of the ResNet model.. A perfect recall score of 1 has been obtained on the 'rust' class outperforming the EfficientNet and the Xception models. The Highest F1 score was observed for the 'rust' class as well. Looking at the table, it is safe to say that the ResNet is more accurate in classifying the 'rust' class than that of the other classes. The precision, recall and F1 scores are less than the scores obtained in the other models for almost all the classes except the 'rust' class.

Table-III: ResNet model's performance in classifying the diseases on Apple Leaves

Class (Disease)	Precision score	Recall score	F1 score
Healthy	0.94186047	0.97590361	0.95857988
Multiple Diseases	0.77777778	0.53846154	0.63636364
Rust	0.97849462	1.	0.98913043
Scab	0.96511628	0.95402299	0.95953757

The visualization in Fig. 9 shows the training and testing loss of the ResNet model over 15 epochs and Fig. 10 shows the evaluation metric scores of the ResNet model. We can observe that the model scored almost similar precision, recall and F1 scores on all the classes except the 'Multiple Diseases' class. Like EfficientNet, ResNet is also struggling with the classification of the 'Multiple Diseases'.

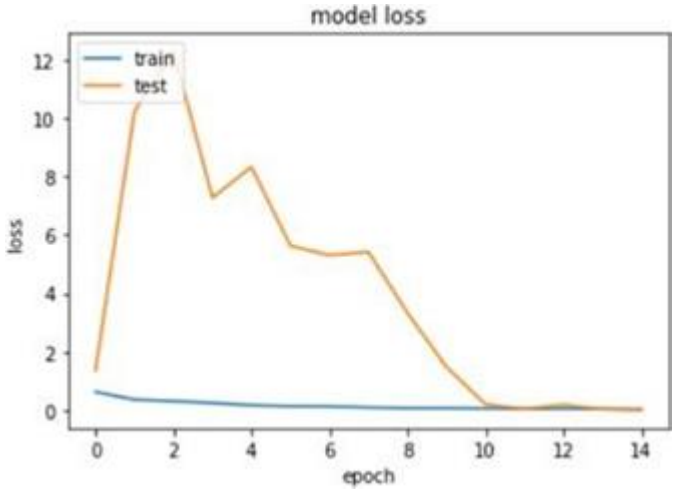

Fig. 9. Train vs Test loss over the epochs for Resnet

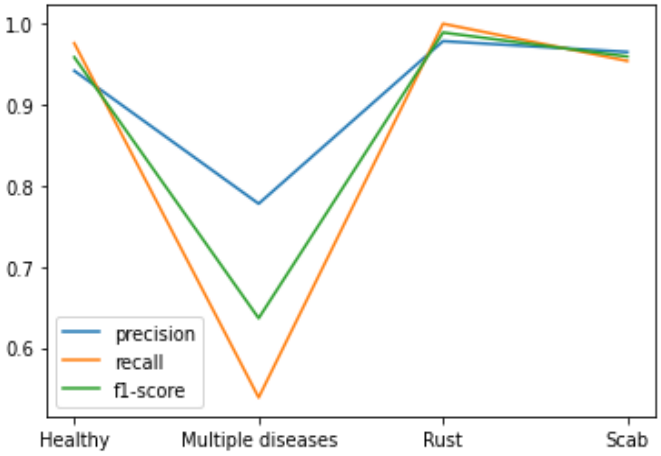

Fig. 10. ResNet model- Evaluation metric scores

Retrieval Number: 100.1/ijeat.F30400810621
DOI:10.35940/ijeat.F3040.0810621
Journal Website: www.ijeat.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication

The results in Table-IV represent the evaluation metric scores of the Xception model. Highest precision has been observed for the 'scab' class and the highest recall, F1 scores were observed for the 'rust' class. Like the EfficientNet and ResNet models, the Xception model has scored less precision, recall and F1 scores on the 'Multiple Diseases' class

Table-IV: Xception model's performance in classifying the diseases on Apple Leaves

Class	Precision	Recall/Sensitivity	F1 score
Healthy	0.98039216	0.95238095	0.96618357
Multiple Diseases	0.66666667	0.625	0.64516129
Rust	0.96825397	0.99186992	0.97991968
Scab	0.98319328	0.96694215	0.975

The visualization in Fig. 11 shows the training and testing loss of the Xception model over the course of 15 epochs and Fig. 12 shows the scores obtained by the Xception model.

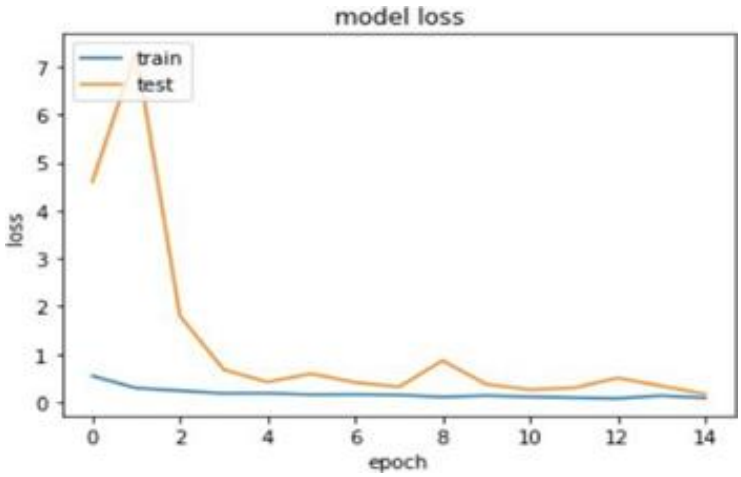

Fig. 11. Train vs Test loss over the epochs for Xception

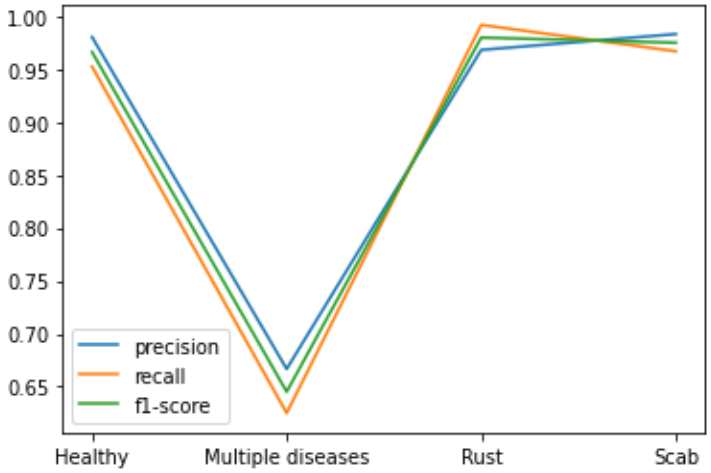

Fig. 12. Xception model- Evaluation metric scores

The results in Table-V represent the evaluation metric scores of the VGG16 model. Highest recall and F1 score are observed for the 'Healthy' and 'Rust' classes respectively but these scores are nowhere near the scores obtained through other models. Also, we can clearly see that the VGG16 is the worst among all the models in classifying the leaves with multiple diseases. The model is also not that great in classifying other three classes with almost all scores under 80%. So, it is safe to say that VGG16 is performing the worst on the task of predicting diseases on Apple leaves when compared with EfficientNet, ResNet and Xception.

Table-V: VGG16 model's performance in classifying the diseases on Apple Leaves

Class (Disease)	Precision score	Recall score	F1 score
Healthy	0.7311828	0.71578947	0.72340426
Multiple Diseases	0.	0.	0.
Rust	0.7980769	0.69166667	0.74107143
Scab	0.83	0.63358779	0.71861472

The visualization in Fig. 13 shows the training and testing loss of the VGG16 model over the course of 30 epochs and Fig. 14 shows the evaluation metric scores of the VGG16 model.

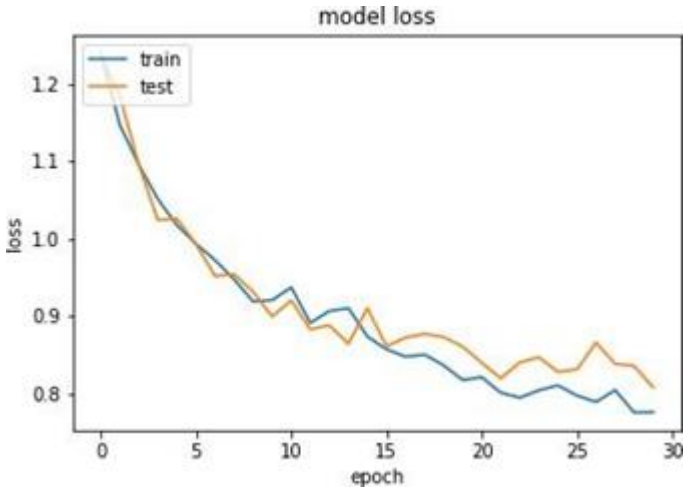

Fig. 13. Train vs Test loss over the epochs for the VGG16 model

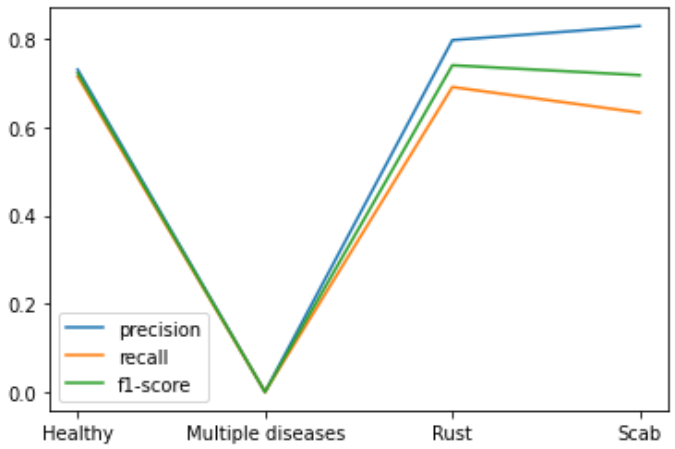

Fig. 14. VGG16 model- Evaluation metric scores

V. CONCLUSION

After implementation and analysis of the performance of the EfficientNet model, the Xception model, the Resnet model and the VGG16 model, we have come to a conclusion that the EfficientNet model is a superior model when it comes to classifying the diseases on Apple leaves. The results also substantiate this fact as the EfficientNet model has the highest classification accuracy. On top of this, the EfficientNet model is also performing well on the multiple diseases class when compared with other CNN models. The Xception model's performance on all the classes except the multiple diseases class is almost same as that of the EfficientNet model. However, it's performance on multiple diseases class is not satisfying. Train and Test losses over the course of all the epochs were also visualized to give an idea on how well the model is doing while training. The VGG16 model was not able to classify even one leaf with multiple diseases on it correctly. The reduced parameter size and the decrease in FLOPS did not affect the performance of the EfficientNet model. Furthermore, this model, with it's clever scaling at depth, width and resolution, reduces the training and inference speeds. It also reduces the battery usage and compute cost making it feasible to be used on mobile devices. If this model is successfully integrated into a mobile application, farmers and cultivators of Apples can just reach out for their phone, scan the leaves and check the condition of the leaves. In conclusion, these types of systems or applications using the EfficienNet model can help detect the diseases on Apple leaves at early stages, thereby making it possible to mitigate any type of losses.

Declaration of conflicting interests: The authors of the paper declare no conflict of interest in doing this research.

REFERENCES

1. Sun, J, Zhang, X. He, K., & Ren, S. (2016), " Deep Residual Learning for Image Recognition",Computer Vision and Pattern Recognition (pp. 770-778).
2. Shen, C.,Wu, Z., , & Van Den Hengel, A. (2019), "Wider or Deeper: Revisiting the Resnet Model for Visual Recognition", pattern recognition,119-133,90.
3. Simonyan, K., & Zisserman, A. (2014), "Very deep convolutional networks for large scale image recognition", 'arXiv preprint' arXiv:1409.1556.
4. Le, Q., & Tan, M. (2019), "Efficientnet: Rethinking model scaling for convolutional neural networks",International Conference on Machine Learning (pp. 6105-6114).
5. Zagoruyko, S., & Komodakis, N. (2016), "Wide residual networks",arXiv preprint arXiv:1605.07146.
6. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2818-2826).
7. S., Shlens, Szegedy, C., Vanhoucke, V., Ioffe,, J., & Wojna, Z. (2016), "Rethinking the Inception Architecture for Computer Vision",Conference on Computer Vision and Pattern Recognition (pp. 2818-2826).
8. Chollet, F. (2017), "Xception: Deep Learning with Depthwise Separable Convolutions",Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1251-1258).
9. Yamada, Zhao, W., W., Li, T., Digman, M., & Runge, T. (2021), "Augmenting crop detection for precision agriculture with deep visual transfer learning—A case study of bale detection",Remote Sensing, 13(1), 23.
10. Mohan, S., Thirumalai, C., & Srivastava, G. (2019), "Effective Heart Disease Prediction using Hybrid Machine Learning Techniques",IEEE access, 81542-81554, 7.
11. NBelongie, SThapa, RSnavely, & Khan, A. (2020), "The Plant Pathology 2020 challenge dataset to classify foliar disease of apples' ,arXiv preprint arXiv:2004.11958.

AUTHORS PROFILE

Kota Akshith Reddy, is a fourth year Computer Science major at Vellore Institute of Technology, Vellore, Tamil Nadu, India. His research interests lie primarily in the field of Computer Vision and Natural Language Processing. He is currently working on improving the performance of Image Captioning models through data augmentation. He has one publication in the field of Natural Language Processing.

Sharmila Banu K, has been working as Assistant Professor in the School of Computer Science and Engineering, Vellore Institute of Technology, Vellore since June 2009. She completed her M.Tech and PhD in Vellore Institute of Technology. She has a total of 12 years of Teaching Experience and 2 years of Industry Experience. Her research interests include Natural Language Processing, Deep Learning, Spatial Data Analysis and Neighborhood Rough Sets.

Sai Kanishka Ippagunta, is a keen learner and always enthusiastic toward new projects. He is pursuing his 4th year B.Tech (CSE) in information security in Vellore Institute of Technology. He has done numerous projects based on Machine Learning, Reverse engineering, and has special interest in Malware analysis and has done several ground plans related for cyber security.

Chandra Havish Siddareddi, is a passionate tech programmer who is pursuing his 4th year in Electronics and Communications engineering in Vellore institute of technology. He is currently working on digital design and Image processing. Even though he is not from a computer science background, he is currently working on Machine Learning and Business Analytics and going through different case studies to get Industrial experience.

Jahnavi Polsani, is adept at learning new things and enthusiastic to solve real-life problems using the latest technologies. She is currently pursuing B.Tech 4th year in CSE from Vellore Institute of Technology, Vellore. Her areas of interest are Problem Solving, DBMS, Data Analytics, Machine Learning, Deep Learning and Cryptography. She has a significant ability to provide valuable insights with a data-driven approach.