Journal of Harbin Engineering University Vol 44 No. 11
ISSN: 1006-7043 November 2023

Computer Vision Based Automatic Leaf Disease Prediction using the Trained Convolution Base Focusing on Transfer Learning

[1]Dr.M.Kavitha, [2]Dr.M.Sivakumar, [3]Dr. Jayasudha Subburaj, [4]Ms.P.Keerthana
[134] Sri Krishna College of Engineering and Technology, Coimbatore, Tamil Nadu
[2] SIMATS School of Engineering, Saveetha Institute of Medical And Technical Sciences [Deemed University],Chennai, Tamil Nadu

Abstract— Plant are considered significant because they are the origin of humanity's supply of energy. The leaf may be infected by plant diseases at any time between sowing and harvesting, leading to a huge loss in crop production and economic market value. In monitoring large fields of crops, the identification of plant diseases has now received growing attention. When moving from one disease control strategy to another, farmers face great difficulties. Experts naked eye observation is the standard approach to the detection and identification of plant diseases adopted in nature which is still in practice by many farmers. This article presents a Transfer learning techniques which is used for automatic detection and classification of plant leaf diseases. Using a public dataset of 1821 training images out of which 516 images are healthy and 1305 images are infected images. A deep convolution neural network with trained convolution base with ImageNet using ResNet, DenseNet and EfficientNet model. The selected model yield accuracy of 97% in ResNet and 98% in DenseNet. The qualified model the EfficientNet on a held-out test range achieves an accuracy of 99.5%, showing the viability of this method. Overall, a simple path toward crop disease detection using transfer learning and compared the metrics like Accuracy, Precision, Recall and F1 score for the model ResNet, DenseNet and EfficientNet*.*

Index Terms— ResNet, DenseNet, EfficientNet, Transfer Learning.

Introduction

In the last few decades, the use of neural networks in science has seen a major change. In numerous research experiments, neural networks have been implemented and demonstrated their high performance at minimal cost. It is mainly used in various tasks including classification and identification, such as biometrics and control systems. They are also recommended in the area of plant identification. Human beings have natural ways to incorporate information among tasks. That really is, as we encounter new activities, we accept and adapt relevant data from prior learning experience. Investigating the impact (stacking hundreds of layers) is necessary to extract essential characteristics from training data / information interesting patterns in order to alleviate complex computer vision concerns using deep learning. But because of the gradients, it can be practically infeasible and troublesome to incorporate neural layers. Unexpected results can be delivered by a multi-layer deep neural network. Under such instances, as the layers grew, the training accuracy rate dropped, precisely referred to as vanishing gradients. The vanishing gradient effects on network performance with respect to features in the initial layer appear incredibly small when training the dataset. When it approaches the minimum, the gradient becomes smaller again. Mostly as result, initial layer weights are adjusted quite poorly or stay constant, owing to an increase in loss or error. Transfer learning is an approach which reduces the number of parameters by taking a segment of the trained model on some familiar task and using it again in the new model. Transfer learning can be referred to as a machine learning method which uses a neural network which has already been trained.

In order to simplify and improve the decision-making process[4], precision farming uses artificial intelligence. In

real-time and distinct machine learning (ML) algorithms, a massive quantity of data is collected to provide optimum decisions that have contributed to cost reduction. Furthermore the

area remains open to improvements especially in decision-making support systems which help turn huge amounts of data into useful recommendations. Countless techniques and methods have been discovered that can be used for purposes such as linear regression, logistic regression, random forest, Gaussian models, clustering, Naïve Bayes (NB), Decision Trees (DT)Support Vector Machines (SVM) and K-nearest neighbours (KNN). In the field of agriculture, a latest explosion in deep learning(DL) [11] methods has as well grown. succession will lead to better solutions for computer vision and artificial intelligence. These techniques give forecasts that are more precise than conventional approaches, which allow improved decision making. Due to advancements in IOT based technology, Deep learning algorithms are at this moment used in a very little time to resolve multifarious problems. The conclusion of the study in this area is not negligible. Raw data comes in many forms, analyzing these raw data in order to make conclusions and understanding the behavior of the data. The techniques and processes of data analytics using different techniques like Machine Learning, Deep learning, etc have been automated which make r raw data for human consumption, which helps in business optimization. Fig 1 details the steps to follow while processing raw data to make perfect decision.

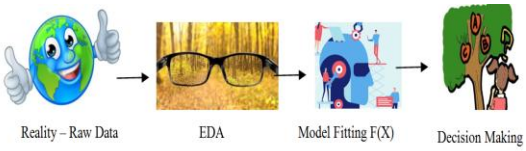


Fig 1: Steps in Raw Data analysis

By training a large model on a large dataset, we can then take advantage of these trained feature maps without needing to start afresh.

Two ways to customize a pre-trained model:

1. Feature Extraction: To derive relevant features from new instances, use the interpretations acquired by a previous network. On top of the pre-trained model, must actually introduce a new classifier which will be trained from scratch, so that you could always refurbish the previously understood feature maps for the corpus. It is not important for you to (re)train the entire model. There are indeed features in the core convolutional network which are widely applicable for computer vision image classification.

2. Fine-TuningRestore some of the upper layers of a frozen base model and train both the additional new layers of the classifier and that the last layers of the base model together. This empowers one to "fine-tune" the interpretations of relatively high features in the base model in addition to making it truly important to the specific mission.

In order to obtain interesting properties from new data, feature extraction consists of using the representations acquired from a previous network. Via a new classifier, which is trained from scratch, these features are then run. As mentioned previously, there are two components of convnets used for image classification: they begin with a sequence of layers of pooling and convolution, and they terminate with a densely connected classifier. The very first element is called the model's convolutional basis. Mostly in case of convnets, the extraction of features consists of taking the convolutionary base of a network that was previously educated, pushing the new data through it and training a classifier on top of the output.

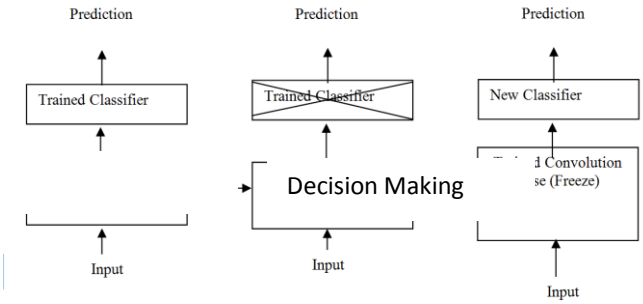


Fig 2: changing classifier while freezing the convolution base

Literature Review

The goal of this paper is to perform automatic analysis of plant disease utilizing multiple intelligent artificial techniques. The primary focus in this paper is on grape leaf disease. When the system is qualified, it can detect the illness of the plant leaf from the early part without doing its inspection over and over again[9]. Algorithms or techniques used are SVM, Back propagation etc., The images examined by these method are

426x568 pixels. There were 497 samples of scab disease, 489 samples of rust disease and 492 samples of non-disease used for training the SVMs. The findings suggest that desirable efficiency is given by the method. But the issue with the suggested approach would be that the neural network did not permit the pixels of the grape leaf disease to be efficiently segmented [1] . Detection of biotic stress in precision crop protection using advanced methods of machine learning. The problem is that the stress impact of weeds and nitrogen in maize in the Neural Network gives up to 69 percent to 58 percent accuracy and the Non-Linear Support Vector Machine gives up to 85 percent accuracy and the Vector Machine and Non-Linear Support offers up to 85 percent accuracy and is superior to a linear SVM (Support Vector Machine), but the algorithms used are very precise and prediction accuracy rate assumptions are not justified[2].

A good way to approach Computer vision problems or to help Agri-Bots is:

- Execute an Exploratory Data Analysis (EDA).
- Build a baseline model or use existing model and change the classifier
- Iterate for different layer.
- Search out a model that performs healthier.

For land cover classification tasks, Deep Learning is already a state-of-the-art strategy, which could also be beneficial for several other tasks. Notable results were achieved by a variety of deep neural networks (DNNs) algorithm usage in hyperspectral analysis [5]. In crop classification[12,13,16], fruit counting, yield forecast [12], illness discovery[8], and computer vision[7], convolutionary neural networks (CNNs) perform fine. In these experiments, the AlexNet [6] and GoogLeNet [15]architectures displayed high-tech performance[3]. So, when we analyze our original dataset with the new model, we are basically using the pre-used extracted features and training on model with our dataset. The advantage is that you can train the model with minimum resources, datasets and time.

The definitions of a domain and a mission are involved in transfer learning. A domain D consists of a space of features F and a distribution of marginal probability P(F) over the space of features, where F=f1..fn, fn €F. It can take several weeks for deep convolutional neural network models to train on very huge files. The re-use of model weights from pre-trained models developed for standard computer vision benchmark datasets, such as ImageNet image classification tasks, is a method to narrow this process. For your own problems in computer vision or to train the agri-Robot, highest ranking prototypes or models can be downloaded and then used immediately, or embedded into a newer one[12]. Transfer learning has the advantage of minimizing a neural network model's preparation time and can lead to reduced classification inaccuracies or error. In addition to the latest challenge, the weights in re-used layers is being used as the preliminary step for the training process and modified further. This use treats transfer learning as a type of activation strategy for weight. A pre-trained model or template is a preserved network, almost always on a large-scale image classification problem, that was successfully trained on a big corpus. To tailor this model to a particular mission, you either use the pre-trained model as it is or use transfer learning. The concept behind transfer learning for image classification is that this model can effectively act as a generalized conceptual world model if a model is trained on a broad and reasonably general dataset.

Methodology

A. ResNet

A residual network, or ResNet for quick, is an artificial intelligence network that uses jump links or routes to hop over certain layers to help create deeper neural networks. e will drill deeply on how skipping helps to create deeper layers of the network without dropping into the vanishing gradients problem. Different ResNet versions are available, including ResNet-18, ResNet-34,ResNet-50, etc., The numbers, while the architecture is the same, denote layers.

As seen in the fig 3 below, apply a switch to the main path in the simple neural network to create a residual block.

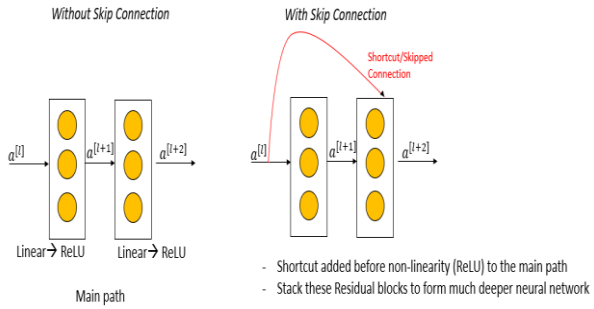


Fig 3: ResNet Architecture

AndrewNg equation:

Layer 1: $z^{[l+1]} = W^{[l+1]}.a^{[l]} + b^{[l+1]}$ (1)

ReLu operation on layer 1 : $a^{[l+1]} = g(z^{[l+1]})$ (2)

Layer 2: $z^{[l+2]} = W^{[l+2]}.a^{[l]} + b^{[l+2]}$

ReLu operation on layer 2 : $a^{[l+2]} = g(z^{[l+2]})$ (3)

- ***Skip connection introduced will be discarded***

So it is re-written as

ReLu operation on layer 2 : $a^{[l+2]} = g(z^{[l+2]} + a^{[l]})$ (4)

Solving

$a^{[l+2]} = g(z^{[l+2]})$

$a^{[l+2]} = g(W^{[l+2]}.a^{[l]} + b^{[l+2]} + a^{[l]})$ (5)

when L2 regularization is used weight delay $W^{[l+2]}=0$ and $b^{[l+2]}=0$ ***so we are left with***

$a^{[l+2]} = g(a^{[l]}) = a^{[l]}$ (6)

as ReLu is used as activation function it will give a non-negative value so we get back $a^{[l]}$

From the above, we can conclude:

- *It's better to learn the identity function of the Residual Network*
- *It is easier to bypass 1, 2 and 3 layers. The Identity function can map well with the output function without damaging the NN results. It means that higher layers perform as well as lower layers.*

B. DenseNet:

DenseNet is one of the latest discoveries for visual object identification in neural networks. With some basic variations, DenseNet is very equivalent to ResNet. ResNet uses an additive approach (+) that combine the previous layer (identity) with the subsequent layer, while the output of the preceding layer with the subsequent layer is concatenated by DenseNet (.) Problems emerge with CNNs as they go deeper. It is because the transition from the input layer to the output layer for information (and the gradient in the opposite direction) becomes so wide that it will disappear before touching the other side.

Traditional feed-forward neural networks connect the layer output to the next layer after a composite of operations is added (convolution operation or pooling layers, a batch normalization and an activation function)

The equation would be:

$x_1 = H1(x_1\text{-}1)$ (7)

ResNets extended this performance together with the skip connection, reformulating this equation into:

$x_1 = H1(x_1\text{-}1) + x_1\text{-}1$ (8)

DenseNet falls in the category of classic networks. Fig 4 shows a 5-layer dense block with a *growth rate* of k = 4 and the standard ResNet structure.

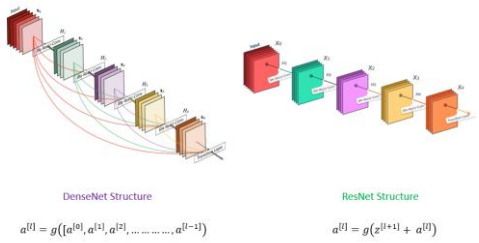


Fig 4 DenseNet Structure – Sources: G. Huang, Z. Liu and L. van der Maaten, "Densely Connected Convolutional Networks," 2018

By using the composite function operation, an output from the preceding layer serves as an input from the second layer. The convolution layer, pooling layer, batch normalization, and non-linear activation layer constitute this composite process. Such links mean that there are L(L+1)/2 direct connections on the network. In architecture, L is the number of layers.

The DenseNet has different version, like DenseNet-121, DenseNet-160, DenseNet-201, etc. The numbers indicate the integer of layers in the neural network. The digit 121 is computed as follows:

DenseNet 121:-
5+(6+12+24+16)*2 = 121
5 – convolution and Pooling Layer
3- Transition Layer (6,12,24)
1-classification Layer (16)
2- DenseBlock(1X1 and 3X3 Conv)

The grouping of layers by the above equation by adding or integrating is only possible if the function map parameters are the same. What if parameters vary? DenseNet is split into DenseBlocks in which there are different numbers of filters, but the dimensions are the same inside the block. Transition Layer uses downsampling to implement patch standardization. It is an important step for CNN. Let's look into DenseBlock and transition layer as shown in fig 5

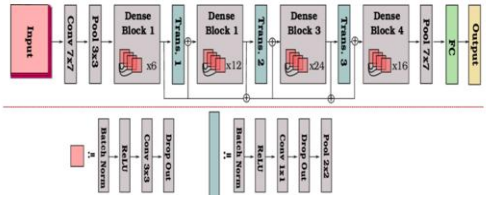


Fig 5: Source: G. Huang, Z. Liu and L. van der Maaten, "Densely Connected Convolutional Networks," 2018. EfficientNet model was proposed by Mingxing Tan and Quoc V. Le of Google Research, Brain team in their research paper 'EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks'.

C. EfficientNet

EfficientNet is another trendy Convolution neural nets based ImageNet model which achieved the SOTA on several image-based tasks in 2019. In a revolutionary way, EfficientNet conducts model scaling to reach excellent precision with considerably fewer parameters. With a far shallower architecture, it achieves the same if not greater precision than ResNet and DenseNet. Now let us train EfficientNet on leaf images and evaluate its performance. With a set of fixed scaling coefficients, the EfficientNet scaling technique uniformly scales network distance, depth, and resolution. For example, if we wish to use 2N times more computing resources, we can simply increase the depth of the network by αN, width by βN, and image size by γN, where alpha, beta, gamma are constant coefficients calculated on the original small model by a small grid scan. EfficientNet uses a composite coefficient φ to uniformly balance network width, depth, and resolution in a principled way [10] Before the EfficientNets came along, the most common way to scale up ConvNets was either by one of three dimensions - depth (number of layers), width (number of channels) or image resolution (image size).

.

DATASETS

The challenge is to differentiate between healthy leaves, those afflicted with apple rust, those with apple scab, and those with more than one disease, based on a picture of an apple leaf. The data for research is downloaded from Kaggle, The Plant Pathology-2020-fgvc7. https://arxiv.org/abs/2004.11958. The dataset consist of 3642 leaf images, train.xls and test.xls files. Out of which 1821 is for training and 1821 leaf image for testing. Fig 6 shows the sample images for training and testing.

Table 1: Train.xls file to create a Model

image_i	health	multiple_diseas	rus	sca
Train_0	0	0	0	1
Train_1	0	1	0	0
Train_2	1	0	0	0
Train_3	0	0	1	0
Train_4	1	0	0	0
Train_5	1	0	0	0
Train_6	0	1	0	0
Train_7	0	0	0	1
Train_8	0	0	0	1
Train_9	1	0	0	0
Train_10	0	0	1	0
Train_11	0	0	0	1

Fig 6 Sample images files for training and testing

Implementation And Experiment Setup

A. ResNet:

```
model= tf.keras.applications.ResNet50(
```

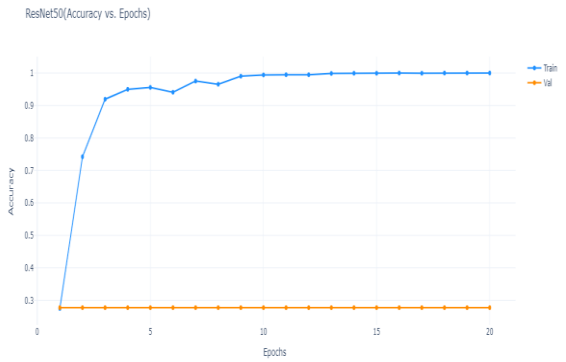
include_top=True, weights='imagenet', input_tensor=None, input_shape=None, pooling=None, classes=1000)

```
model.compile(optimizer='adam',
              loss = 'categorical_crossentropy',
              metrics=['categorical_accuracy'])
model.summary()
```

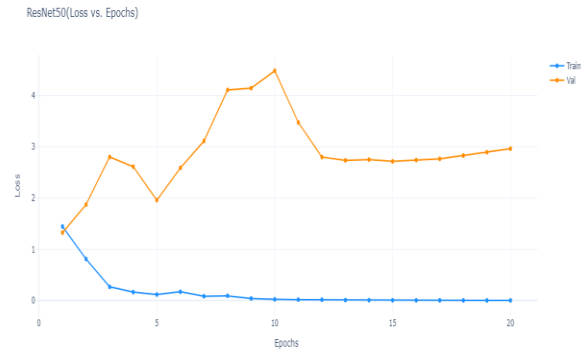
```
conv5_block3_3_bn (BatchNormali (None, 7, 7, 2048)  8192        conv5_block3_3_conv[0][0]
conv5_block3_add (Add)          (None, 7, 7, 2048)  0           conv5_block2_out[0][0]
                                                                conv5_block3_3_bn[0][0]
conv5_block3_out (Activation)   (None, 7, 7, 2048)  0           conv5_block3_add[0][0]
avg_pool (GlobalAveragePooling2 (None, 2048)        0           conv5_block3_out[0][0]
predictions (Dense)             (None, 1000)        2049000     avg_pool[0][0]
Total params: 25,636,712
Trainable params: 25,583,592
Non-trainable params: 53,120
```

a)

b) ResNet Model summary



c) ResNet - Accuracy vs Epochs



d) ResNet - Loss vs Epochs

Fig 14 : ResNet

Fig 14 a) show the summary of ResNet model fitting with 25,636,712 total parameters and 25,583,592 trainable parameters. Fig 15b) and c) shows the Accuracy and Loss with respect to Epochs.

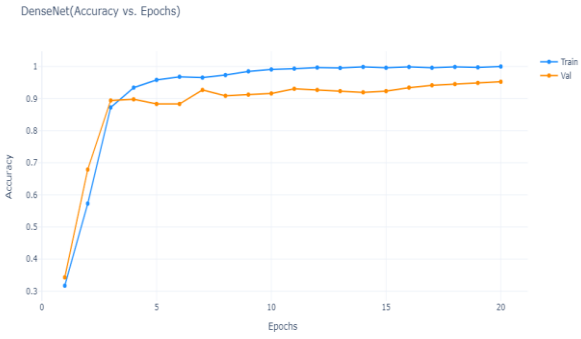
B. DenseNet:

```
model = tf.keras.Sequential([DenseNet121(input_shape=(512, 512, 3),

weights='imagenet',include_top=False),
                    L.GlobalAveragePooling2D(),

L.Dense(train_labels.shape[1],activation='softmax')])
model.compile(optimizer='adam',loss='categorical_crossentropy', etrics=['categorical_accuracy'])
model.summary();
```

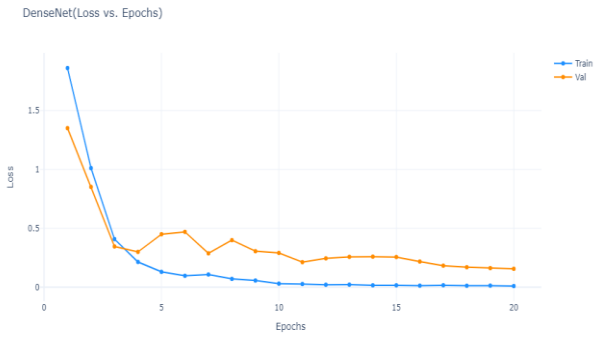
```
model.compile(optimizer='adam',
              loss = 'categorical_crossentropy',
              metrics=['categorical_accuracy'])
model.summary()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/densenet/densenet121_weights_tf_dim_ordering_tf_kernels_notop.h5
29089792/29084464 [==============================] - 1s 0us/step
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
densenet121 (Model)          (None, 16, 16, 1024)      7037504
_________________________________________________________________
global_average_pooling2d (Gl (None, 1024)              0
_________________________________________________________________
dense (Dense)                (None, 4)                 4100
=================================================================
Total params: 7,041,604
Trainable params: 6,957,956
Non-trainable params: 83,648
_________________________________________________________________
```

a) DenseNet Model Summary



b) DenseNet – Accuracy vs Epochs



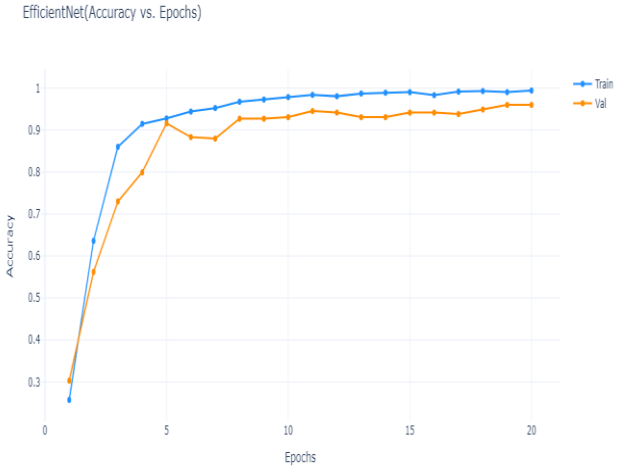
c) DenseNet – Loss vs Epochs

Fig 15: DenseNet

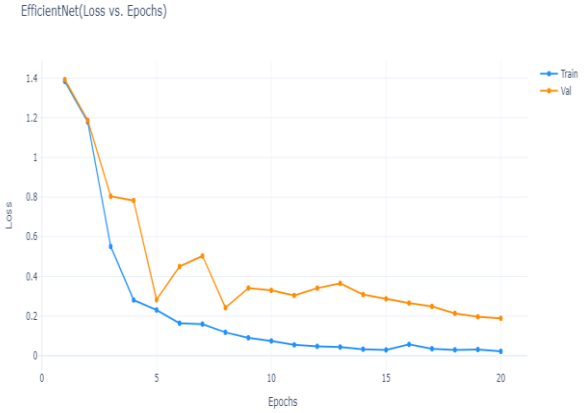
Fig 15 shows that the losses decrease and accuracies increase quite consistently. The training metrics settle down very fast (after 1 or 2 epochs), whereas the validation metrics much greater volatility and start to settle down only after 7-8 epochs. This is expected because validation data is unseen and more diffcult to make predictions on than training data.

C. *EfficientNet:*

```
model=
tf.keras.Sequential([efn.EfficientNetB7(input_
shape=(512, 512, 3),
        weights='imagenet',
include_top=False),
L.GlobalAveragePooling2D(),
        L.Dense(train_labels.shape[1],
activation='softmax')])
```



a) Efficient Net – Accuracy vs Epochs



c)EfficientNet – Loss vs Epochs

Fig 16 EfficientNet Model

Prediction made:

```
model.compile(optimizer='adam',
              loss = 'categorical_crossentropy',
              metrics=['categorical_accuracy'])
model.summary()
```

```
Downloading data from https://github.com/Callidior/keras-applications/releases/download/efficientnet/efficientnet-b7_weights_tf_dim_ordering_tf_kernels_autoaugment_notop.h5
258441216/258434480 [==============================] - 9s 0us/step
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
efficientnet-b7 (Model)      (None, 16, 16, 2560)      64097680
_________________________________________________________________
global_average_pooling2d_1 ( (None, 2560)              0
_________________________________________________________________
dense_1 (Dense)              (None, 4)                 10244
=================================================================
Total params: 64,107,924
Trainable params: 63,797,204
Non-trainable params: 310,720
_________________________________________________________________
```

a) EfficientNet – Model Summary

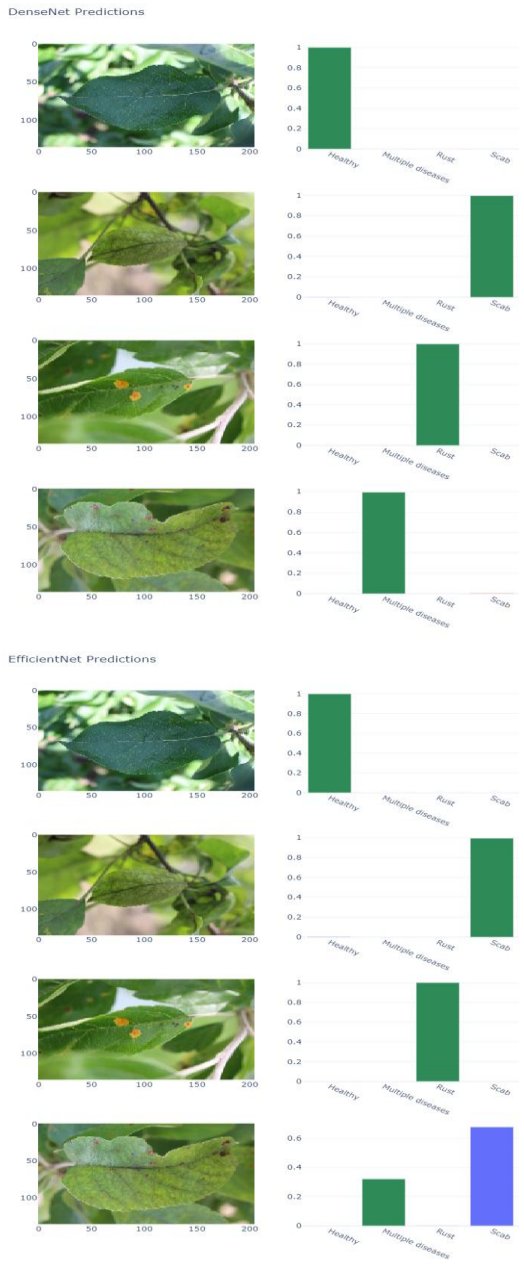


Fig 17 a)Leaf Prediction Made for DenseNet b) Leaf Prediction Made using EfficientNet

V. METRICS

Table 2: Confusion Matrix: ResNet:

		Actual class							
n=1821		Healthy	Multiple	Rust	scab	TP	FP	TN	FN
	Healthy	512	1	2	1	512	4	1305	0
	Multiple	0	84	4	3	84	7	1708	22
Predicted	Rust	0	12	607	3	607	15	1187	12
class	Scab	0	9	6	577	577	15	1222	7

Table 3: Confusion Matrix – DenseNet:

		Actual class							
n=1821		He	Mu	R	s	T	F	T	F
Pre	He	51	0	2	0	5	2	1	0
dict	Mu	0	86	2	3	8	5	1	1
ed	Rus	0	6	6	3	6	9	1	9
clas	Sca	0	4	5	5	5	9	1	6

Table 4: Confusion Matrix – EfficientNet:

		Actual class							
n=1821		He	Mu	R	s	T	F	T	F
Pre	He	51	0	1	0	5	1	1	0
dict	Mu	0	89	1	1	8	2	1	4
ed	Rus	0	1	6	2	6	3	1	2
clas	Sca	0	3	0	5	5	3	1	3

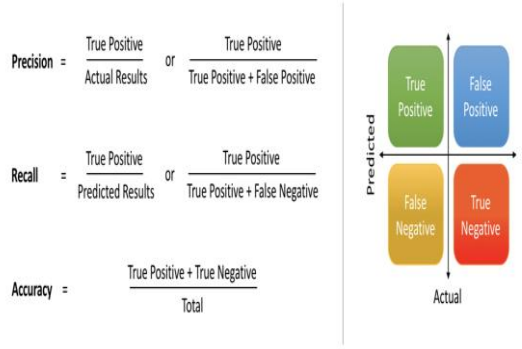


Fig 18: Metrics for evaluating the model

Two highly critical model evaluation metrics are precision and recall, fig 18. Although precision refers to the relevant percentage of your data, recall refers to the correctly identified percentage of total relevant results by your algorithm. Optimizing both of these metrics at the same time is not feasible, since one comes at the expense of another. Here is a simplified metric that takes both precision and recall into consideration, and should try to maximize this amount to make the model better. This metric is referred to as the F1-score, which is essentially the harmonic mean of precision and recall.

F1 Score = 2* Precision *Recall/(precision+Recall)

Table 5: Comparison of pretrained weights on results

Evaluation Metrics	ResNet	DenseNet	EfficientNet
Accuracy	0.977	0.986	0.995
Precision	0.966	0.977	0.991
Recall	0.940	0.967	0.987

F1	0.953	0.972	0.989
Training Time	3:20hrs	2:15 hrs	1:10 hrs

CONCLUSION

In this article, we discussed some of the significant problems and limitations of research that used CNNs to classify crop diseases automatically. In order to optimize the potential of CNNs implemented in real-world implementations, we have presented a new classifier with trained convolution base. Many CNN-based solutions already released are currently not operational for field use, primarily due to a lack of compliance with several significant computer vision principles. For unknown data samples and/or imaging environments, this lack of conformity can lead to poor generalization capabilities, which decreases the realistic use of the qualified models. Nonetheless, the works examined three Models ResNet,DenseNet and EfficientNet using transfer learning and illustrate the promise of deep learning approaches for the detection of crop diseases. our results are undoubtedly encouraging for the creation of modern agricultural computer vision instruments that could lead to making food production more productive and stable.

REFERENCES

[1] Arnal Barbedo, J.G. (2019) Plant disease identification from individual lesions and spots using deep learning. Biosystems. Engineering. 180, 96–107. https://doi.org/10.1016/j.biosystemseng.2019.02.002

[2] Behmann Jan ,A.K.Mahlein ,T.Rumpf , C. Romer ,L.Plumer, (2015) A review of advanced machine learning methods for the detection of biotic stress in precision crop protection, Springer Media New York, Precision Agriculture, 16(3), 239– 260. https://doi.org/10.1007/s11119-014-9372-7

[3] Ferentinos Konstantinos P. (2018) Deep learning models for plant disease detection and diagnosis. Computers and Electronics in Agriculture. (145), 311–318. https://doi.org/10.1016/j.compag.2018.01.009

[4] Gebbers, R., Adamchuk, V.I. (2010) Precision agriculture and food security. Science 2010(327), 828–831. DOI: 10.1126/science.1183899

[5] Gewali, U.B.; Monteiro, S.T.; Saberli. (2018) Machine learning based hyperspectral image analysis: A survey. arXiv:1802.08701, 1–42. https://arxiv.org/abs/1802.08701

[6] Krizhevsky A. Sutskever, I., Hinton, G.E.(2017) Imagenet classification with deep convolutional neural networks. Communications of the ACM 2017, (60), 84–90. https://doi.org/10.1145/3065386

[7] Lee, H.; Kwon, H.(2017) Going deeper with contextual CNN for hyperspectral image classification. IEEE Transaction on Image Processing. 26, 4843–4855. DOI: 10.1109/TIP.2017.2725580

[8] Liu, B. Zhang, Y. He, D. Li, Y. (2018) Identification of apple leaf diseases based on deep convolutional neural networks. Symmetry 2018, 10(1), 11; https://doi.org/10.3390/sym10010011

[9] Meunkaewjinda A, Kumsawat P and K. Attakitmongcol, (2008) Grape leaf disease detection from color imagery using hybrid intelligent system, 5th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 1, 513 – 516. DOI: 10.1109/ECTICON.2008.4600483

[10] Mingxing Tan and Quoc V. (2019) EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks'. International Conference on Machine Learning, 2019.

[11] Mohanty, S.P.; Hughes, D.P, Salathé, M. (2016) Using deep learning for image-based plant disease detection. Front. Plant Sci.2016,7, 1419. https://doi.org/10.3389/fpls.2016.01419

[12] Patricio, D.I.; Rieder, R. (2018) Computer vision and artificial intelligence in precision agriculture for grain crops: Asystematic review. Computer and Electronics in Agriculture.2018,153, 69–81. https://doi.org/10.1016/j.compag.2018.08.001

[13] Rahnemoonfar, M.; Sheppard, C. (2017) Deep count: Fruit counting based on deep simulated learning.Sensors 2017,17(4), 905. https://doi.org/10.3390/s17040905

[14] Steen, K.A; Christiansen, P.; Karstoft, H.; Jørgensen, R. (2016) Using deep learning to challenge safety standard forhighly autonomous machines in agriculture. Journal of Imaging. 2(1) 6. https://doi.org/10.3390/jimaging2010006

[15] Szegedy, C.; Liu, W. Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A.(2015) Going Deeper with Convolutions. In Proceedings of the IEEE Conference on Computer Vision and PatternRecognition (CVPR), Boston, MA, USA, 1–9. DOI: 10.1109/CVPR.2015.7298594

[16] Yao, C.; Zhang, Y.; Zhang, Y.; Liu, H.(2017) Application of convolutional neural network in classification of high resolution agricultural remote sensing images. International Archives of the Photogrammetry Remote Sensing and Spatial Information Science XLII-2/W7, 989–992.h ttps://doi.org/10.5194/isprs-archives-XLII-2-W7-989-2017