*Current Chinese Computer Science*, **2021**, *1*, 61-76

RESEARCH ARTICLE

# Leaf Image Classification with the Aid of Transfer Learning: A Deep Learning Approach

Srinivasa Rao Dammavalam[1,*], Ramesh Babu Challagundla[2], Vangipuram Sravan Kiran[1], Rajasekhar Nuvvusetty[3], Lalith Bharadwaj Baru[1], Rohit Boddeda[4] and Sai Vardhan Kanumolu[1]

*[1]Department of Information Technology, VNR Vignana Jyothi Institute of Engineering and Technology, Hyderabad, India; [2]Department of CSE, Geethanjali College of Engineering and Technology, Hyderabad, India; [3]Department of Information Technology, Gokaraju Rangaraju Institute of Engineering and Technology, Hyderabad, India; [4]Department of CSE, VNR Vignana Jyothi Institute of Engineering and Technology, Hyderabad, India*

ARTICLE HISTORY

Received: April 14, 2020
Revised: June 02, 2020
Accepted: June 18, 2020

*DOI:*
*10.2174/2665997201999200811150433*

**Abstract:** ***Background***: Crop diseases are a primary hazard to nutrient safety, which proves to be a serious problem in many places in the world due to the unavailability of essential aid. Typically agriculturalists or specialists perceive the plants with a naked eye for detection and identification of an illness. Machine vision models, in specific Convolutional Neural Networks (CNNs) have directed an impact in feature extraction to a greater extent. Since 2015, numerous solicitations for the automatic classification and recognition of crop illnesses have been established.

***Methods***: In this paper, we proposed, analyzed, and assessed various state-of-the-art models proposed over a decade. These models are pre-trained with the finest parameters where we modeled a design-oriented method with numerous leaf-images and classified them into infection and healthy class for each type of leaf independently.

***Results***: Through our examination, we concluded that VGG models stand-alone with many cited prototypes and give on par results. As declared, these VGG models (VGG16 and VGG19) are utilized for feature extraction, and further, we augmented a set of dense layers and train them consequently for classification. The performances of various machine vision prototypes were pictorially perceived and their sophisticated architecture is not only capable of extracting detailed features but also repressed many loop-holes. The performance is assessed and computed for several types of leaf images and the accuracy scores attained were more than 97.5% for VGG16 and 96.72% for VGG19.

***Conclusion***: AUC-ROC curves were portrayed to illustrate its inspiration in defining an accurate classification where VGG16 and VGG19 have at least 96.6% and 95% area under the curve (AUC) which resembles their robustness.

**Keywords:** Leaf classification, deep learning, transfer learning, automated plant diagnosis, CNNs.

## 1. INTRODUCTION

The trick of competent plant disease fortification is carefully connected to the difficulties of supportable cultivation and weather variation. Investigation outcomes designate that weather variation can change phases and amounts of pathogen improvement; it can also change host confrontation, which leads to physiological variations of host-pathogen connections. The condition is more difficult by the fact that today; diseases are increasing globally. New diseases can transpire in places where they were formerly unknown, where there is no native ability to find proper medication.

Cultivation has a huge impact on the production of food, especially with the increasing population. The plant diseases are intimidating the yield of the crop. Plant diseases can have a major impact on decreasing crop production in farming and forestry. Initial discovery and identification of plant diseases oblige to take suitable actions.

There are numerous methods to identify plant pathologies. Some diseases do not have any noticeable indications related, or appear only when it is too late to act. In these circumstances, it is essential to accomplish refined examination, typically by resources of influential microscopes. In some circumstances, the marks can only be perceived in portions of the electromagnetic band that are not obvious to the naked eye.

*Address correspondence to this author at the Department of Information Technology, VNR Vignana Jyothi Institute of Engineering and Technology, Hyderabad, India; Tel: 0423042761; E-mail: dammavalam2@gmail.com

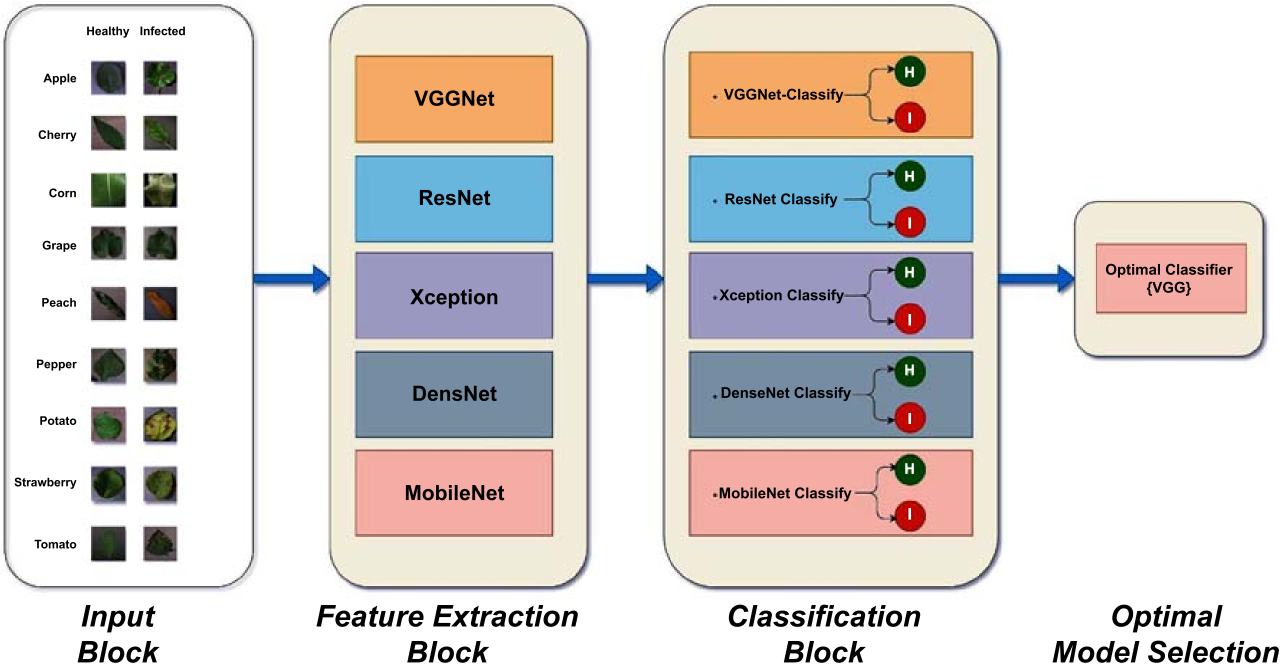

**Fig. (1).** In the above figure, we consider all the input leaf images of various plants as described in Table (**1**). In the First Block (Input) we send different sets of images (healthy and infected) of various leaf-kinds. In feature extraction block with the help of various state-of-the-vision models as mentioned in Table (**2**) and generate a feature vector. Next, these feature vectors are fed to a classifier, a Fully-connected-Neural-Network (FCN), and classify the input images as healthy(H) or Infected(I) with individual probabilities. Now we select an Optimal model which generalizes for all sets of leaf-kinds. During this experimentation, it was found that VGG models tend to perform extravagantly by overcoming various pitfalls. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

In this varying environment, suitable and well-timed disease identification comprising initial prevention is not significant. There are numerous techniques to identify plant pathologies. Some diseases do not have any perceptible indications and in those circumstances, a refined examination is compulsory. Conversely, maximum diseases produce some kind of appearance in the perceptible spectrum. In this condition, they can be discovered with the naked eye inspection of a skilled expert. To accomplish precise plant disease diagnostics, a plant pathologist should have respectable observational abilities to recognize distinctive indications.

Overcoming such problems can either be done by a professional plant pathologist or a person who belongs to that field of expertise. However, with the help of AI, we can detect and diagnose by developing models with expert precision which can be useful to common people. Introducing AI to detect and classify various plant diseases has been utilized for the past decades. In this research, an efficient solution is provided by classifying and detecting if the plant is either infected or healthy by considering leaf image as input. The flow diagram is depicted in Fig (**1**).

## 2. PREVIOUS WORKS

### 2.1. Before Evolution of ConvNets

Researches before the evolution of ConvNets applied machine learning handcrafted methods for automated diagnosis of plant diseases that had high pre-processing mechanisms. Camargo *et al.*, (2009) [1] considered cotton crops with few samples and analyzed properties of cotton crop images such as shape, texture, fractal-dimensions, *etc.* They considered such properties as features (45 features) and attained the highest accuracy of 93.1%. Rumpf *et al.*, (2010) [2] performed 3 variant tasks a) classifying healthy and infected sugar-beet leaves (binary classification). b) Spotting various diseases in Cercospora infecting beet-sugar leaves (multi-class classification). And c) Finally, identifying the disease even before the visual occurrence of symptoms for performing the classification tasks. They used SVM which attained a high accuracy of 97% for binary classification and 86% for multi-class classification. Al Bashish *et al.*, (2010) [3] developed a diagnostic model for detection and classification of the plant leaf stem, where they designed four-phase learning for the model to leverage its performance. In the first phase, they captured a significant colored part of the leaf by thresholding to capture mostly the greener pixels of the image. In the next step, they mask this greener pixel. Subsequently, the boundaries containing infected clustered objects are removed. Now, this fully pre-processed data is sent to a pre-trained neural network which classifies 6 classes (5 are infected and 1 is healthy) with 92.7% accuracy. Inspiring from the previous study, Al-Hiary *et al.*, (2011) [4] developed a software solution for automated diagnosis of plant leaf in which classification is done with a precision lying between 83-94%. They additionally achieved a 20% speed-up compared to that of the previous approach [5]. Arivazhagan *et al.*, (2013) [6] considered a few samples of various leaf images (<1k) and classified them with SVM to attain an accuracy of 87.66%.

We observed that most of the researchers performed heavy-tailed pre-processing on leaf images to build an accurate model but, these handcrafted former methods are considered to be hyperparameters which are to be tuned carefully and this decreases model versatility. This problem of handcrafted pre-processing can be overcome by the latest advancements in computer vision using deep learning techniques.

### 2.2. After Evolution of ConvNets

There was much advancement in classification and recognition of plant leaf images after the revival of ConvNets. Many researchers developed deep models either by training input data or by sharing weights of pre-trained models to reduce computational complexity and share the efficacy of model performance. This led to insightful results both in terms of performance and efficiency. Sladojevic *et al.*, (2016) [7] developed a deep model for recognition of 13 different kinds of plant leaf images and attained a precision between 91-98% and on an average, the precision was 96.3% and additionally, they stated augmentation improved performance of the model gradually per epoch. Mohanty *et al.*, (2016) [8] evaluated the model with trained and pre-trained AlexNet [9] and GoogleNet [10] models by identifying 14 distinct crops with 26 varying diseases. For a greater generalization, they varied test samples from 20-80% and they attained the highest accuracy score of 99.35% for a pre-trained model. Singh *et al.*, (2017) [11] used an algorithm for image segmentation and further automated detection and classification were done using ConvNets. They were able to classify 5 leaf disease classes with an average accuracy of 97.6%. Amara *et al.*, (2017) [12] designed fully connected ConvNets for feature extraction and fully connected dense layers to classify banana colored and grey-scaled images. They have inspired from [13] and performed various test sets for a good generalization. Test sets were varied from 20-80% and attained the highest accuracy for 99.72% colored banana leaves with 50% test samples and 97.57 for grey-scaled images with 40% test samples. Petrellis *et al.*, (2017) [14] developed a mobile app for the diagnosis of plant diseases based on attributes of color, shape, previous weather data, *etc.* Classification and detection of disease constricted with few samples with an accuracy score of 90% (individual accuracy scores were above 85%). Brahimi *et al.*, (2017) [15] classified nine different infected classes of tomato leaves using ConvNets (inspired architecture from AlexNet and GoogleNet) and attained an accuracy score of 99.18%. They also compared machine learning models (Random Forest and SVM) with Convnets to show the efficiency of deep pre-trained ConvNets.

Oppenheim *et al.*, (2017) [16] classified 4 diseased and 1 healthy class of potato leaves with very little sample (approx. 1k) and generalization was carried out with train-test methodology on various test sample sets, *i.e.* varied from 0.1 to 0.9, during experimentation they attained high accuracy of 95.85% for 0.1 test size and least accuracy of 83.21 for 0.9 test size. Fuentes *et al.*, (2017) [17] developed a disease and pest recognition model for tomato plant leaves by collecting data using a phone camera. This model is an ensemble of various state-of-the-art detection models, i.e Faster-RCNN, R-FCN, and SSD for deep meta-ensemble learning. In this model, they used ResNet and VGG as feature extractors which eventually led to greater performance. Al Bashish *et al.*,(2018) [18] evaluated the performance of deep neural networks in plant pathology by considering 21 distinct plants and 171 diseases of which plants are affected. They additionally investigated the challenges and factors influencing the use of deep-vision models in plant pathology either to classify or detect diseases. Prajwala *et al.*, (2018) [19] developed a unified model inspired by the architecture of LeNet which classifies tomato leaves and its infection kind with middling accuracy of 94-95%. Wang-Su.J *et al.*,(2018) [20] inspired by the architecture of GoogleNet proposed a model that classifies and detects infection from healthy leaves [21]. This model attains classification accuracy between 99.6-99.8% and the recognition rate was more than 94% even if a large part of the leaf (30%) got damaged. Serawork. W *et al.*, [22] used LeNet architecture for soybean-plant disease classification with an adequate sample size (13k) for 4 infection classes. They have shown that their accuracy score was improved with the use of regularization techniques (Dropout and Data Augmentation) to a good extent and attaining the highest accuracy score of 99.32% by overcoming the drawback of data imbalance.

It is observed that most of the researchers obtained greater performance for deep-vision models due to the fact of shared weight mechanisms in deeper networks and using deep vision models overcame the major problem of handcrafted pre-processing techniques. Most of the research proposed has either done classification or detection for i) A single plant-leaf ii) Multiple leaves all combined iii) or classifying only disease classes of the leaf. In this research, we propose a strategy of classifying infected and healthy plant leaf by developing a model that fits that specific plant leaf kind. Such models are tuned carefully to attain higher efficiency and versatility.

## 3. METHODOLOGY

### 3.1. Models Description

The success rate of deep learning is consistently growing in various domains like computer vision(CV) [23-27], complex networks [28-30] and Natural Language Processing (NLP) [31, 32]. In computer vision, the visual recognition challenge ILSVRC [33] shed a light on deep learning frameworks. The success rate in classification has outrageously increased with the evolution of Convolution Neural Networks. The first victorious attempt in developing deep convolutional neural networks was AlexNet. Thereafter researchers proposed state-of-the-art methods over a decade. Researchers evaluated their models progressively by intensifying the depth of the network and redesigning networks to overhaul existing models (Fig **2**). Various models were proposed to improve the ability to recognize patterns and grasp them substantially. In Table (**2**), we observed that with a periodic increase in depth of the network the parameters increase heavily. These parameters boost the computational cost which in turn increases training time. To overcome such a cost increment in models, we imply the method of transfer learning [34]. Transfer learning has been implemented in recent advancements [35-38]. In transfer learning, the primary focus is on preserving the weights of profound layers of

**Table 1. Various crop images taken for experimentation [source https://www.kaggle.com/vipoooool/new-plant-diseases-dataset].**

| Leafs Kinds | Classes | Infected Sub-Classes | No of images | No of Images in Healthy & Infection Class |
|---|---|---|---|---|
| Apple | Infection | scab | 630 | 1526 (48.12%) |
| | | black rot | 621 | |
| | | cedar apple | 275 | |
| | Healthy | * | 1645 | 1645 (51.86%) |
| Cherry | Infection | Powdery mildew | 1052 | 1052 (55.2%) |
| | Healthy | * | 854 | 854 (44.8%) |
| Corn | Infection | Cercospora leaf spot | 513 | 2690(69.8%) |
| | | Common rust | 1192 | |
| | | Northern Leaf Blight | 985 | |
| | Healthy | * | 1162 | 1162(30.2%) |
| Grape | Infection | Black rot | 1180 | 3639(89.5%) |
| | | Leaf blight | 1076 | |
| | | Esca | 1383 | |
| | Healthy | * | 423 | 423(10.5%) |
| Peach | Infection | Bacterial spot | 2297 | 2297(86.4%) |
| | Healthy | * | 360 | 360(13.6%) |
| Pepper | Infection | Bacterial spot | 997 | 997(40.2%) |
| | Healthy | * | 1478 | 1478(59.8%) |
| Potato | Infection | Early Blight | 1000 | 2000(92.9%) |
| | | Late Blight | 1000 | |
| | Healthy | * | 152 | 152(7.1%) |
| Strawberry | Infection | Leaf Scorch | 1109 | 1109(70.9%) |
| | Healthy | * | 456 | 456(29.1%) |
| Tomato | Infection | Bacterial Spot | 2127 | 16569(91.2%) |
| | | Early blight | 1000 | |
| | | Late Blight | 1909 | |
| | | Leaf Mold | 952 | |
| | | Septoria leaf spot | 1771 | |
| | | Target spot | 1404 | |
| | | Mosiac Virus | 373 | |
| | | Yellow leaf curl virus | 5357 | |
| | | Spider mites | 1676 | |
| | Healthy | * | 1591 | 1591(8.8%) |
| Blueberry** | Heathy | * | 1502 | 1502(100%) |
| Orange** | Healthy | * | 5507 | 5507(100%) |
| Raspberry** | Healthy | * | 371 | 371(100%) |
| Soybeans** | Healthy | * | 5090 | 5090(100%) |
| Squash** | Infection | Powdery Mildew | 1835 | 1835(100%) |

[* - Represents the Healthy class without any Infection in that particular Leaf kind]; [**- Represents the Absence of either a Healthy or Infection class in that particular Leaf Kind].

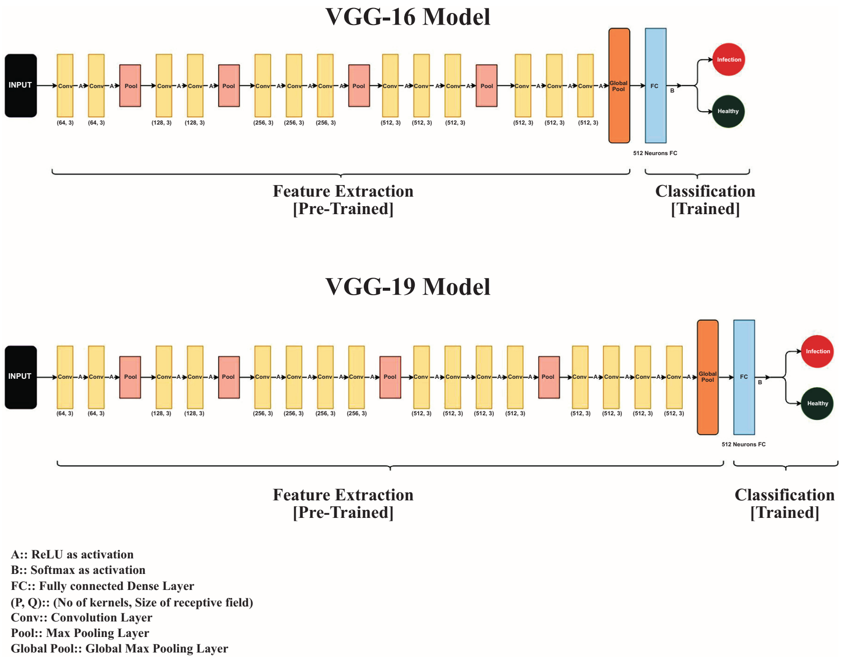

**Fig. (2).** Methodology for classification through VGG-16 and VGG-19 models. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

the humongous network which helps in modeling an analogous problem statement. The mentioned models in Table (**2**) are high performance deep neural networks with intense pattern recognition capability in the domain of computer vision. These models are trained on images posed on ILSVRC. The primacy of transfer learning is they provide high throughput in performance by reducing the effort of fine-tuning with elegant feature extraction capability, in turn, reducing the performance cost *i.e.* training time. By the means of transfer learning methodology, these pre-trained neural networks are implied to analyze the performance of these methods on the Plant-Village database [39] and classify each crop for their healthy and infected classes and complete data information is tabulated (refer Table (**1**)). Understanding each model precisely over various crops helps us to determine the stellar model that can be used at a generic circumstance.

AlexNet gave a good intuition for understanding ConvNets as it has high parameters for a lower depth and did not give effective learning compared to existing methods. After these various benchmark networks, some other methods [40-42] were proposed with increasing depth and contrast in network architecture. This evolution is due to the change in depth of network usage of different receptive fields for convolution and varying design in architectures [43, 44]. Therefore, the advancement from AlexNet to GoogleNet is an elegant construct of network architecture. Next in VGG, we have small receptive fields (3x3, 5x5) for capturing low-level features; Larger receptive fields (7x7, 9x9) are used in AlexNet which are obliged to capture heavy-tailed features. Henceforth, we started with emerging and potential neural networks VGG Nets [45], ResNets [46, 47], DenseNet [48], MobileNet [49, 50] and Xception [51], which have been state-of-the-art models over the subsequent years in ILSVRC. As in VGG, the researchers have used a small receptive field with variation in size (5x5, 3x3). In some cases, they have used a minute receptive field which helps in the exact reconstruction of image passing on to successive layers which are inspired by NiN [52]. This improved classification performance helped in deducing error rate to that of GoogleNet. Next ResNet team came up with a novel methodology of weight sharing through sharable receptive fields in deep networks to overcome the hurdle of vanishing gradient. They have used an identity mapping mechanism for shortcut connection a.k.a skip connections for greater depth of layers which further improved classification performance by a drop in error rate to a good extent. DenseNets, Xception, MobileNets came into picture soon after the research for ResNets was done. The Xception network explicitly collates the performance and shows the pitfalls of GoogleNet, enhances the network by redesigning the Inception module which elevates performance by lessening computational cost. DenseNets presented various forms of deeper architectures that were densely connected with vast shared layers among them. The DenseNets architecture was divided into 2 major blocks named Dense Block and Transition Block, where Dense Blocks helped in understanding the intricate features iteratively. For a proper assessment of the model, we evaluated every metric in a detailed fashion. Now using these state-of-the-art models we extract precise patterns by discussing in detail the pros and stumbling blocks of every selective model. Further, these models have high efficacy towards computational balance and error moderation simultaneously. Deep learning enhancement at various levels has been understood and an unaltered intuition is explained [53-55].

Table (**2**) describes the models which we used even with their updated versions. The conv + pooling layers describe the models that are pre-trained and used as feature extractors. Dense layers help to construct a classifier, and we trained them. Individual model performance is captured and evaluated with the metrics mentioned in Table (**3**). The first dense layer consists of 512 Neurons and subsequently, this feed is forwarded to the next layer with ReLU [56] as activation.

**Table 2. Models taken for experimentation.**

| Models | Conv+pooling Layers | Dense Layers | Total Layers | Non-Trainable Parameters | Trainable Parameters | Total Parameters |
|---|---|---|---|---|---|---|
| VGG16 | 16+1* | 2 | 19 | 17.7M | 0.2M | 17.9M |
| VGG19 | 19+1* | 2 | 22 | 20M | 0.2M | 20.2 |
| ResNet50 | 50+1* | 2 | 53 | 23M | 1M | 24M |
| DenseNet-121 | 121+1* | 2 | 124 | 7M | 0.5M | 7.5M |
| Xception | 126+1* | 2 | 129 | 20.8M | 1M | 21.8M |
| MobileNetV2 | 88+1* | 2 | 92 | 2.2M | 0.6M | 2.8M |
| MobileNet | 88+1* | 2 | 92 | 3.2M | 0.5M | 3.7M |
| ResNet101 | 101+1* | 2 | 103 | 42M | 1M | 43M |
| ResNet50V2 | 50+1* | 2 | 52 | 23.5M | 1M | 24.5M |
| ResNet101V2 | 101+1* | 2 | 103 | 42.6M | 1M | 43.6M |
| DenseNet169 | 169+1* | 2 | 171 | 18.3M | 0.9M | 19.2M |
| DenseNet201 | 201+1* | 2 | 203 | 12.4M | 0.8M | 13.4M |

[*- Represents we have done Global max pooling at the terminal layer of the network and fed feed to subsequent dense layers].

Next, 2 Neurons are assigned to return the feed as probabilities of the healthy and infection class with softmax [57] activation. In this process of feature extraction from methods, we have not used any computation over the stacked deep layers(convolution and pooling) as they are pre-trained models assigned with balanced weights upon them. Now Global Max Pooling is performed [58, 59] after the feature extraction process which unravels to a linear layer. Now the attached two Dense Layers are trained and parameters that are generated are not more than 1M which eventually reduces the effort of the training.

## 3.2. Train-test & Metrics

### *i) Train-Test*

As it is computationally expensive to test through K-Fold, in deep learning train-and-test strategy is used. Before splitting data into train and test segments, we pre-processed it by resizing isotopically to 224x224x3. No further preprocessing was done. Next, we conventionally divided the complete data into training and testing sets randomly (with a fixed mixture). Training consists of 80% and testing consists of 20%. During training, we sent samples of a batch of 16. The use of appropriate batch size would lead to faster convergence for smaller learning rates (1E-2) for the specified optimizer [60, 61]. As we do not conduct large training, we train our dense networks with the TensorFlow [62] platform where weights are updated through an optimized version of backpropagation [63-65]. The models are trained for 10 epochs and used Adam [66] as an optimizer by assigning learning as 0.001. We did not use any regularization such as batch normalization [67] and dropout [68] as we are constrained with pre-trained weights. Categorical cross-entropy is used as our loss function. As sigmoid results in probability for a binary classification problem, we want to evaluate model performance deliberately with all possible metrics for individual classes and this is done by choosing softmax as non-linear activation.

### *ii) Metrics*

A collection of metrics is chosen to evaluate the performance and to avail of the underlying purpose of using them. These are some of the classification metrics widely used and have a significant role in proper judgment analysis for the models. Most of the classification metrics are derived from the confusion matrix. Generally, the confusion matrix generates a complete description of exactly and incorrectly classified instances for a given label descriptor.

**Accuracy:** It is a generic metric which gives proper intuition of how well the instances are classified, *i.e.* correctness of instances as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

**Precision:** This tells about the correctly classified class labels to the aggregation of correctly and Incorrectly classified instances.

$$Precision = \frac{TP}{TP + FP}$$

**Recall(Sensitivity):** This describes the correctly classified class label to that of all the classified labels belongs to that class.

$$Recall = \frac{TP}{TP + FN}$$

**F1-Score:** This describes the weighted aggregate of both precision and recall.

$$F1\ Score = \frac{2x(Precision\ x\ Recall)}{Precision + Recall}$$

**Table 3. Comparison of various models utilized in the proposed methodology for various crops.**

| Apple | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Base Metrics | | | Precision | | Recall | | F1-Score | |
| **Models** | **Accuracy*** | **MSE*** | **Loss*** | **Healthy** | **Infection** | **Healthy** | **Infection** | **Healthy** | **Infection** |
| Vgg-16 | 0.9748 | 2.31E-07 | 0.3928 | 0.99 | 0.96 | 0.96 | 0.99 | 0.98 | 0.97 |
| Vgg-19 | 0.9764 | 0.0218 | 0.2185 | 0.98 | 0.97 | 0.98 | 0.98 | 0.98 | 0.98 |
| ResNet-50 | 0.9795 | 0.0167 | 0.0695 | 0.96 | 1 | 1 | 0.96 | 0.98 | 0.98 |
| DenseNet-121 | 0.7858 | 0.1743 | 0.8612 | 0.85 | 0.74 | 0.72 | 0.86 | 0.78 | 0.79 |
| Xception | 0.5921 | 0.3886 | 6.6731 | 0.58 | 0.61 | 0.75 | 0.42 | 0.66 | 0.5 |
| MobileNet-224 | 0.674 | 0.2335 | 0.7508 | 0.62 | 0.93 | 0.98 | 0.35 | 0.76 | 0.51 |
| MobineNet-96 | 0.4252 | 0.36649 | 1.0343 | 0.46 | 0.35 | 0.61 | 0.23 | 0.52 | 0.28 |
| ResNet-101 | 0.9433 | 0.0504 | 0.2632 | 0.9 | 1 | 1 | 0.89 | 0.95 | 0.94 |
| ResNet-50[V2] | 0.4803 | 0.5197 | 112.94 | 0 | 0.48 | 0 | 1 | 0 | 0.65 |
| ResNet-101[V2] | 0.5197 | 0.4803 | 163.39 | 0.52 | 0 | 1 | 0 | 0.68 | 0 |
| DenseNet-169 | 0.7638 | 0.1884 | 0.7744 | 0.71 | 0.86 | 0.91 | 0.61 | 0.8 | 0.71 |
| DenseNet-201 | 0.5717 | 0.3977 | 3.1769 | 0.63 | 0.54 | 0.43 | 0.72 | 0.51 | 0.62 |
| | | | | | | | | | |
| **Cherry** | | | | | | | | | |
| | **Base Metrics** | | | **Precision** | | **Recall** | | **F1-Score** | |
| **Models** | **Accuracy*** | **MSE*** | **Loss*** | **Healthy** | **Infection** | **Healthy** | **Infection** | **Healthy** | **Infection** |
| Vgg-16 | 0.9895 | 0.0105 | 0.1517 | 1 | 0.98 | 0.98 | 1 | 0.99 | 0.99 |
| Vgg-19 | 0.9764 | 1.43E-03 | 0.2185 | 0.98 | 0.97 | 0.98 | 0.98 | 0.98 | 0.98 |
| ResNet-50 | 0.5707 | 0.371027 | 1.64 | 0.52 | 1 | 1 | 0.19 | 0.69 | 0.32 |
| DenseNet-121 | 0.5602 | 0.4185 | 3.6087 | 0.82 | 0.55 | 0.08 | 0.99 | 0.14 | 0.7 |
| Xception | 0.4581 | 0.5394 | 38.3264 | 0.46 | 0 | 0.98 | 0 | 0.63 | 0 |
| MobileNet-224 | 0.5026 | 0.4297 | 1.6376 | 0.49 | 1 | 1 | 0.06 | 0.65 | 0.12 |
| MobineNet-96 | 0.5314 | 0.449681 | 1.8839 | 0 | 0.53 | 0 | 1 | 0 | 0.69 |
| ResNet-101 | 0.8953 | 0.083 | 0.4181 | 0.82 | 1 | 1 | 0.8 | 0.9 | 0.89 |
| ResNet-50[V2] | 0.5314 | 0.4686 | 87.0492 | 0 | 0.53 | 0 | 1 | 0 | 0.69 |
| ResNet-101[V2] | 0.4686 | 0.5314 | 392.7 | 0.47 | 0 | 1 | 0 | 0.64 | 0 |
| DenseNet-169 | 0.555 | 0.4082 | 2.5325 | 0.62 | 0.55 | 0.13 | 0.93 | 0.22 | 0.69 |
| DenseNet-201 | 0.8429 | 0.1403 | 1.3135 | 0.98 | 0.78 | 0.68 | 0.99 | 0.8 | 0.87 |
| | | | | | | | | | |

**(Table 3) contd…**

| Corn | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Base Metrics | | | Precision | | Recall | | F1-Score | | |
| Models | Accuracy* | MSE* | Loss* | Healthy | Infection | Healthy | Infection | Healthy | Infec-tion | |
| Vgg-16 | 0.9961 | 3.93E-09 | 0.1007 | 1 | 1 | 0.99 | 1 | 0.99 | 1 | |
| Vgg-19 | 0.9974 | 2.62E-11 | 0.0594 | 1 | 1 | 1 | 1 | 1 | 1 | |
| ResNet-50 | 0.9987 | 9.90E-04 | 0.004 | 1 | 1 | 1 | 1 | 1 | 1 | |
| DenseNet-121 | 0.6706 | 0.019 | 11.7568 | 1 | 0.67 | 0 | 1 | 0.01 | 0.8 | |
| Xception | 0.5694 | 0.421 | 12.1214 | 0.41 | 0.76 | 0.67 | 0.52 | 0.51 | 0.62 | |
| MobileNet-224 | 0.6887 | 0.243 | 0.8206 | 0.52 | 1 | 1 | 0.53 | 0.68 | 0.7 | |
| MobineNet-96 | 0.8638 | 0.0937 | 0.2907 | 0.72 | 0.99 | 0.98 | 0.81 | 0.83 | 0.89 | |
| ResNet-101 | 0.9507 | 0.0407 | 0.2076 | 0.87 | 1 | 1 | 0.93 | 0.93 | 0.96 | |
| ResNet-50[V2] | 0.6693 | 0.0263 | 78.554 | 0 | 0.67 | 0 | 1 | 0 | 0.8 | |
| ResNet-101[V2] | 0.6693 | 0.3307 | 158.76 | 0 | 0.67 | 0 | 1 | 0 | 0.8 | |
| DenseNet-169 | 0.6706 | 0.3271 | 5.5298 | 1 | 0.67 | 0 | 1 | 0.01 | 0.8 | |
| DenseNet-201 | 0.7471 | 0.2313 | 2.2658 | 0.69 | 0.76 | 0.44 | 0.9 | 0.53 | 0.83 | |
| | | | | | | | | | | |
| Grape | | | | | | | | | | |
| | Base Metrics | | | Precision | | Recall | | F1-Score | | |
| Models | Accuracy* | MSE* | Loss* | Healthy | Infection | Healthy | Infection | Healthy | Infec-tion | |
| Vgg-16 | 0.9975 | 2.56E-10 | 0.0399 | 0.99 | 1 | 0.99 | 1 | 0.99 | 1 | |
| Vgg-19 | 0.9988 | 0.0013 | 0.0234 | 1 | 1 | 0.99 | 1 | 0.99 | 1 | |
| ResNet-50 | 0.9779 | 1.70E-06 | 0.0538 | 0.83 | 1 | 1 | 0.98 | 0.9 | 0.99 | |
| DenseNet-121 | 0.8991 | 0.0975 | 1.2671 | 0.71 | 0.9 | 0.06 | 1 | 0.11 | 0.95 | |
| Xception | 0.8954 | 8.02E-04 | 22.0678 | 0 | 0.9 | 0 | 1 | 0 | 0.94 | |
| MobileNet-224 | 0.4157 | 0.43422 | 1.401 | 0.15 | 0.99 | 0.96 | 0.35 | 0.26 | 0.52 | |
| MobineNet-96 | 0.8954 | 0.0085 | 0.681 | 0 | 0.9 | 0 | 1 | 0 | 0.94 | |
| ResNet-101 | 0.722 | 2.24E-06 | 0.9316 | 0.27 | 1 | 1 | 0.69 | 0.43 | 0.82 | |
| ResNet-50[V2] | 0.8954 | 0.1046 | 14.6937 | 0 | 0.9 | 0 | 1 | 0 | 0.94 | |
| ResNet-101[V2] | 0.187 | 0.0044 | 40.9428 | 0.11 | 0.99 | 0.99 | 0.09 | 0.2 | 0.17 | |
| DenseNet-169 | 0.8954 | 0.1045 | 1.9226 | 0 | 0.9 | 0 | 1 | 0 | 0.94 | |
| DenseNet-201 | 0.8954 | 1.00E-04 | 4.5094 | 0 | 0.9 | 0... | 1 | 0 | 0.94 | |
| | | | | | | | | | | |

**(Table 3) contd…**

| Peach | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Base Metrics | | | Precision | | Recall | | F1-Score | |
| Models | Accuracy* | MSE* | Loss* | Healthy | Infection | Healthy | Infection | Healthy | Infec-tion |
| Vgg-16 | 0.9906 | 9.74E-09 | 0.1262 | 0.99 | 0.99 | 0.94 | 1 | 0.96 | 0.99 |
| Vgg-19 | 0.9868 | 0.0131 | 0.1124 | 1 | 0.99 | 0.9 | 1 | 0.95 | 0.99 |
| ResNet-50 | 0.9868 | 0.0086 | 0.0285 | 0.91 | 1 | 1 | 0.98 | 0.95 | 0.99 |
| DenseNet-121 | 0.8365 | 0.1422 | 1.1554 | 0.45 | 1 | 1 | 0.81 | 0.62 | 0.9 |
| Xception | 0.8327 | 0.167 | 10.2604 | 0 | 0.86 | 0 | 0.96 | 0 | 0.91 |
| MobileNet-224 | 0.7444 | 0.1793 | 0.5464 | 0.1 | 0.86 | 0.11 | 0.84 | 0.11 | 0.85 |
| MobineNet-96 | 0.8684 | 0.1308 | 0.7623 | 0 | 0.87 | 0 | 1 | 0 | 0.93 |
| ResNet-101 | 0.9737 | 0.0189 | 0.0584 | 1 | 0.97 | 0.8 | 1 | 0.89 | 0.99 |
| ResNet-50[V2] | 0.8684 | 0.1316 | 30.9314 | 0 | 0.87 | 0 | 1 | 0 | 0.93 |
| ResNet-101[V2] | 0.1316 | 0.8684 | 177.018 | 0.13 | 0 | 1 | 0 | 0.23 | 0 |
| DenseNet-169 | 0.8853 | 0.098 | 0.4764 | 0.91 | 0.88 | 0.14 | 1 | 0.25 | 0.94 |
| DenseNet-201 | 0.8684 | 0.131 | 3.881 | 0 | 0.87 | 0 | 1 | 0 | 0.93 |
| | | | | | | | | | |
| Pepper | | | | | | | | | |
| | Base Metrics | | | Precision | | Recall | | F1-Score | |
| Models | Accuracy* | MSE* | Loss* | Healthy | Infection | Healthy | Infection | Healthy | Infec-tion |
| Vgg-16 | 0.9758 | 0.0212 | 0.1095 | 0.98 | 0.97 | 0.98 | 0.98 | 0.98 | 0.97 |
| Vgg-19 | 0.9657 | 0.0289 | 0.1784 | 0.98 | 0.95 | 0.96 | 0.97 | 0.97 | 0.96 |
| ResNet-50 | 0.8222 | 0.1374 | 0.5132 | 0.77 | 1 | 1 | 0.56 | 0.87 | 0.72 |
| DenseNet-121 | 0.8242 | 0.1461 | 0.6859 | 0.8 | 0.87 | 0.93 | 0.67 | 0.86 | 0.76 |
| Xception | 0.3939 | 0.5856 | 12.3771 | 0.48 | 0.34 | 0.32 | 0.5 | 0.38 | 0.4 |
| MobileNet-224 | 0.6081 | 0.2885 | 0.8742 | 0.6 | 1 | 0.61 | 0.52 | 0.61 | 0.41 |
| MobineNet-96 | 0.4727 | 0.3202 | 0.8696 | 0.7 | 0.43 | 0.19 | 0.88 | 0.3 | 0.58 |
| ResNet-101 | 0.8222 | 0.1512 | 0.6528 | 0.77 | 1 | 1 | 0.56 | 0.87 | 0.72 |
| ResNet-50[V2] | 0.5838 | 0.4167 | 26.7995 | 0.59 | 0.3 | 0.98 | 0.01 | 0.74 | 0.03 |
| ResNet-101[V2] | 0.5939 | 0.4061 | 34.8977 | 0.59 | 1 | 1 | 0 | 0.74 | 0.01 |
| DenseNet-169 | 0.7475 | 0.2036 | 0.997 | 0.74 | 0.76 | 0.88 | 0.55 | 0.8 | 0.64 |
| DenseNet-201 | 0.6263 | 0.3059 | 1.671 | 0.74 | 0.53 | 0.56 | 0.72 | 0.64 | 0.61 |
| | | | | | | | | | |

**(Table 3) contd…**

| Potato | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Base Metrics | | | Precision | | Recall | | F1-Score | | |
| Models | Accuracy* | MSE* | Loss* | Healthy | Infection | Healthy | Infection | Healthy | Infec-tion |
| Vgg-16 | 0.9884 | 1.21E-06 | 0.0647 | 0.91 | 0.99 | 0.94 | 0.99 | 0.92 | 0.99 |
| Vgg-19 | 0.9907 | 7.40E-07 | 0.0414 | 0.94 | 0.99 | 0.94 | 0.99 | 0.94 | 0.99 |
| ResNet-50 | 0.8863 | 0.0878 | 0.3105 | 0.4 | 1 | 1 | 0.88 | 0.57 | 0.93 |
| DenseNet-121 | 0.9258 | 0.0742 | 1.0387 | 0 | 0.93 | 0 | 1 | 0 | 0.96 |
| Xception | 0.9258 | 0.0742 | 9.6483 | 0 | 0.93 | 0 | 1 | 0 | 0.96 |
| MobileNet-224 | 0.9258 | 0.0558 | 0.1972 | 0.5 | 0.94 | 0.28 | 0.98 | 0.36 | 0.96 |
| MobineNet-96 | 0.9258 | 0.0737 | 0.378 | 0 | 0.93 | 0 | 1 | 0 | 0.96 |
| ResNet-101 | 0.9258 | 5.95E-06 | 0.2388 | 0.5 | 1 | 0.97 | 0.92 | 0.66 | 0.96 |
| ResNet-50[V2] | 0.9258 | 0.0742 | 15.4327 | 0 | 0.93 | 0 | 1 | 0 | 0.96 |
| ResNet-101[V2] | 0.2622 | 0.7297 | 28.8968 | 0.09 | 0.99 | 0.97 | 0.21 | 0.16 | 0.34 |
| DenseNet-169 | 0.9258 | 0.0742 | 0.9729 | 0 | 0.93 | 0 | 1 | 0 | 0.96 |
| DenseNet-201 | 0.9258 | 0.0742 | 2.8591 | 0 | 0.93 | 0 | 1 | 0 | 0.96 |
| | | | | | | | | | |

| Strawberry | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Base Metrics | | | Precision | | Recall | | F1-Score | |
| Models | Accuracy* | MSE* | Loss* | Healthy | Infection | Healthy | Infection | Healthy | Infec-tion |
| Vgg-16 | 1 | 7.23E-07 | 4.88E-05 | 1 | 1 | 1 | 1 | 1 | 1 |
| Vgg-19 | 1 | 7.18E-17 | 3.81E-10 | 1 | 1 | 1 | 1 | 1 | 1 |
| ResNet-50 | 1 | 1.99E-04 | 0.0022 | 1 | 1 | 1 | 1 | 1 | 1 |
| DenseNet-121 | 0.7125 | 0.2812 | 3.4411 | 1 | 0.71 | 0.07 | 1 | 0.13 | 0.83 |
| Xception | 0.6901 | 0.3099 | 40.7192 | 0 | 0.69 | 0 | 1 | 0 | 0.82 |
| MobileNet-224 | 0.3514 | 0.4453 | 1.3857 | 0.3 | 0.63 | 0.8 | 0.15 | 0.43 | 0.24 |
| MobineNet-96 | 0.7891 | 0.1538 | 0.4771 | 0.94 | 0.77 | 0.34 | 0.99 | 0.5 | 0.87 |
| ResNet-101 | 1 | 9.58E-04 | 0.0086 | 1 | 1 | 1 | 1 | 1 | 1 |
| ResNet-50[V2] | 0.6901 | 0.3094 | 32.0733 | 0 | 0.69 | 0 | 1 | 0 | 0.82 |
| ResNet-101[V2] | 0.5431 | 0.4464 | 10.5017 | 0.38 | 0.8 | 0.75 | 0.45 | 0.51 | 0.58 |
| DenseNet-169 | 0.7348 | 0.2474 | 2.3549 | 0.94 | 0.72 | 0.15 | 1 | 0.27 | 0.84 |
| DenseNet-201 | 0.6869 | 0.3128 | 10.8117 | 0 | 0.69 | 0 | 1 | 0 | 0.81 |
| | | | | | | | | | |

**(Table 3) contd…**

| Tomato | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | **Base Metrics** | | | **Precision** | | **Recall** | | **F1-Score** | |
| **Models** | **Accuracy*** | **MSE*** | **Loss*** | **Healthy** | **Infection** | **Healthy** | **Infection** | **Healthy** | **Infection** |
| Vgg-16 | 0.9939 | 0.0049 | 0.029 | 0.99 | 0.99 | 0.99 | 0.97 | 0.99 | 0.98 |
| Vgg-19 | 0.9672 | 0.0275 | 0.1569 | 0.73 | 1 | 0.99 | 0.97 | 0.84 | 0.98 |
| ResNet-50 | 0.9736 | 0.0216 | 0.0751 | 0.77 | 1 | 0.98 | 0.97 | 0.86 | 0.99 |
| DenseNet-121 | 0.9144 | 0.0856 | 2.5672 | 0 | 0.91 | 0 | 1 | 0 | 0.96 |
| Xception | 0.9146 | 0.0854 | 7.9042 | 1 | 0.91 | 0 | 1 | 0.01 | 0.96 |
| MobileNet-224 | 0.2585 | 0.5943 | 2.286 | 0.09 | 0.94 | 0.86 | 0.2 | 0.17 | 0.33 |
| MobineNet-96 | 0.9135 | 0.0828 | 0.384 | 0 | 0.91 | 0 | 1 | 0 | 0.95 |
| ResNet-101 | 0.9584 | 0.0308 | 0.1103 | 0.68 | 1 | 0.97 | 0.96 | 0.8 | 0.98 |
| ResNet-50[V2] | 0.914 | 0.0856 | 38.2829 | 0 | 0.91 | 0 | 1 | 0 | 0.96 |
| ResNet-101[V2] | 0.9144 | 0.0856 | 31.296 | 0 | 0.91 | 0 | 1 | 0 | 0.96 |
| DenseNet-169 | 0.913 | 0.0846 | 0.9528 | 0.44 | 0.92 | 0.05 | 0.99 | 0.1 | 0.95 |
| DenseNet-201 | 0.8945 | 0.0956 | 0.6806 | 0.44 | 0.99 | 0.86 | 0.9 | 0.58 | 0.94 |

[*-Represents Base metrics where they're generated combining Infection and Healthy classes and for evaluation of all metrics above mentioned, we used the Keras library where TensorFlow runs at the backend].

**AUC-ROC Curve:** Receiver Operating characteristics (ROC) is a metric generally used for a classification task with varying thresholds.

So, for this, TPR (true-positive-rate) and FPR (false-positive-rate) are calculated and the area under the curve (AUC) is promoted by laying FPR on the x-axis and TPR on the other. This curve helps in determining the fragility of the model.

$$TPR = \frac{TP}{TP + FN}$$

$$FPR = \frac{FP}{TN + FP}$$

## 4. RESULTS AND DISCUSSION

As mentioned in Table (**3**), the above-mentioned metrics for models are generated and tabulated in Table (**2**). Models with base metrics (mentioned with *) are evaluated as they are prominent and have dominance to that of others. Precision, recall, and F1-score are evaluated to determine estimates of the models over various classes (Healthy & Infected). It can be observed that potato is the imbalanced class with a dominance of infection class (above 90% samples). In such scenarios, we have to consider the metric evaluated on an individual class to provide an optimal estimate for our constructed model. In potato, it is observed that various models are having null values for precision, recall, and f1-score on a single class (*i.e.*, healthy class). This says that the models do not have to sustain a plethora in imbalance and learning only through the samples containing higher class distribution(*i.e.*, infection class). In this case, considering base metrics can lead to inappreciable estimations. Therefore, to alleviate such problems additional metrics are generated with class-wise performance. These metrics are to be carefully noticed which shows variations as outlined above (Fig **3**).

While considering base metrics, accuracy and loss determine a model performance, and additionally, MSE (mean squared error) is added which shows the deviation from ground truth to that to the estimated one. As mentioned, we have admitted categorical cross-entropy as our loss function.

$$Categorical\ crossentropy = -\frac{1}{S}\sum_{i=1}^{S}\sum_{j=1}^{L} y_{ij}.\log\left(P_{ij}\right)$$

$$MSE = \frac{1}{S}\sum_{i=1}^{S}\sum_{j=1}^{L}\left(y_{ij} - P_{ij}\right)^2$$

Where $S$ represents no. of samples or instances for a class. $L$ represents no of class labels.

$y_{ij}$ represents true class label for specified class and $P_{ij}$is the estimated class label.

## 5. NOTICED PITFALLS AND DRAWBACKS

### 5.1. Depth of Network

We presented the below visualizations by considering the image (Apple black-rot infection class) shown in Fig. (**1**) which is of size 224 x 224. This image is passed as input into various models provided in the above section and generated visualizations of the activation maps at three different phases of the network. In the first phase, we have considered a succession to the input layer; next, the second phase consists of the central (middle) layer, and the third phase consists of rearmost layer visualizations.

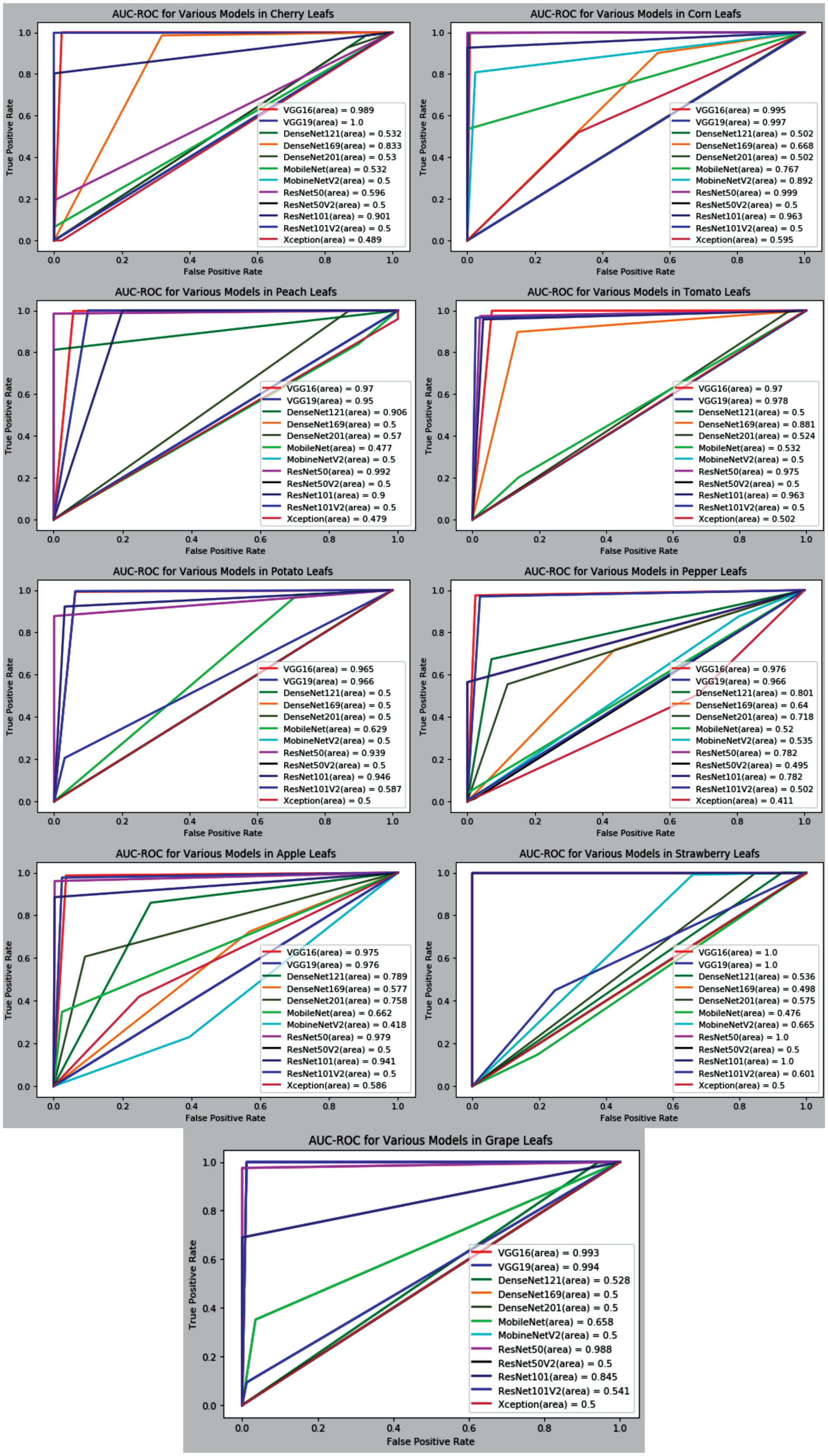

**Fig. (3).** Illustrations of results obtained from various models. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

**Fig. (4).** Input Image for Experimentation. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

**Vgg16:**

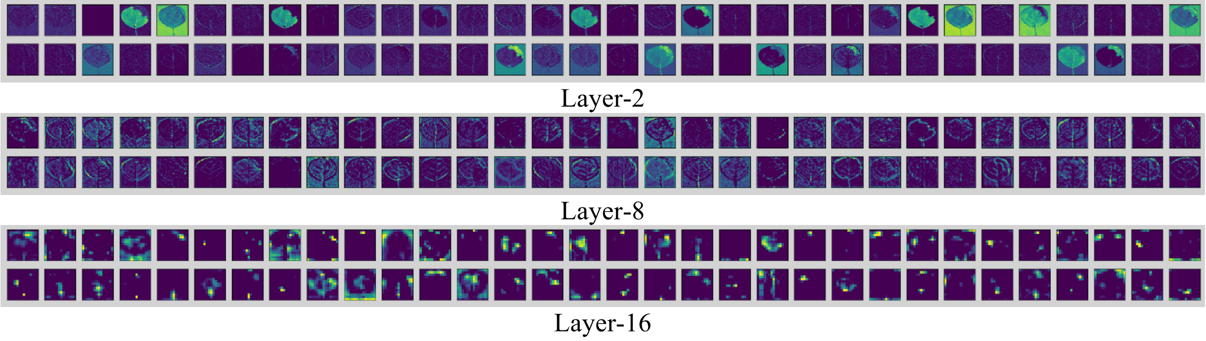

**Fig. (5).** VGG16 illustrations of various layers for Apple. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

**Vgg19:**

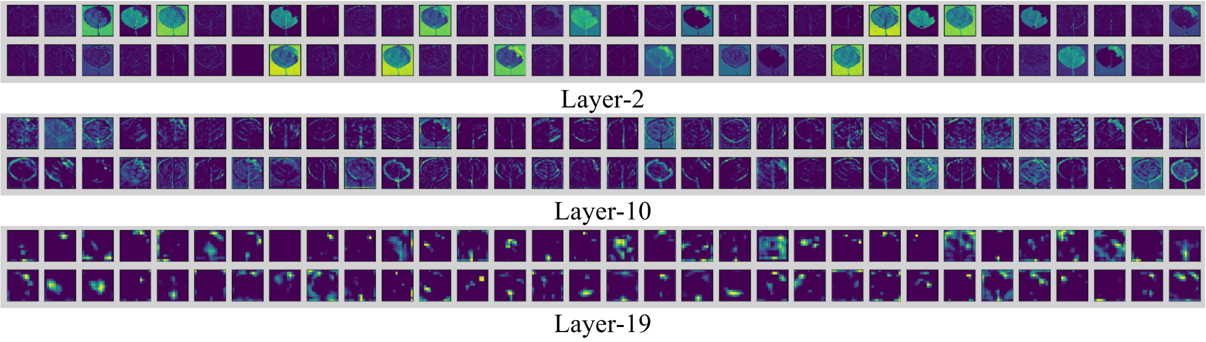

**Fig. (6).** VGG19 illustrations of various layers for Apple. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

The above representation describes the feature extraction mechanism in convolution neural networks. Each image in a specified layer describes the information shared by each kernel (also, filter) by convolving over the image. We depicted the internal mechanism of each convolutional layer feature extraction for respective models. We observed that the VGG model has the efficacy to extract insightful features due to the small receptive fields in the model. Thus, feature extraction is done precisely. In the rearmost layer, visualizations of the infected portion in the leaf, the image is identified by both the VGG models (VGG16 & VGG19). Therefore, the performance of the VGG model was on par with that of the remaining models (Figs **4**-**6**).

As intensifying the depth of layers, learning of the network becomes sophisticated. ResNet and DenseNet models consist of too many skip connections; these features learned in the previous layers will be the same as in the next layers. Therefore, the rearmost layer feature extraction is almost the same as the input layer (as seen in Figs. **7** and **8**). As the infected portion was not identified properly, these models did not yield greater performance as compared to that of the VGG model.

**Resnet50**:

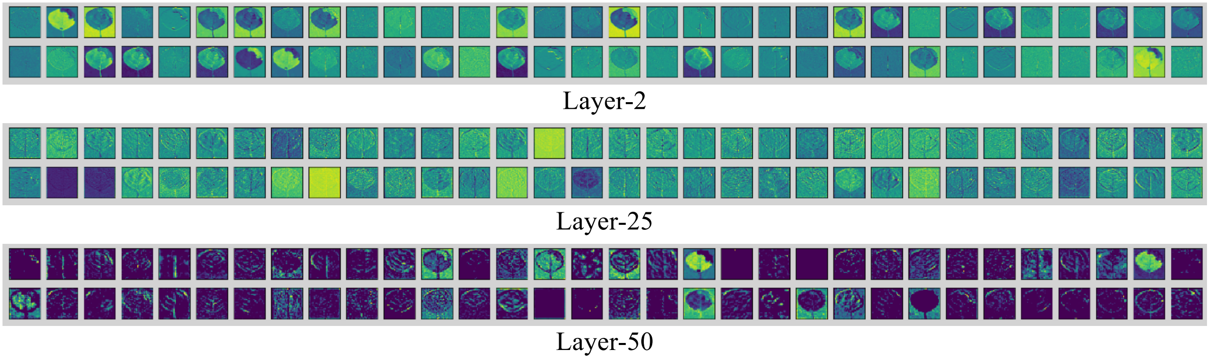

**Fig. (7).** Resnet50 illustrations of various layers for Apple. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

**Densenet121**:

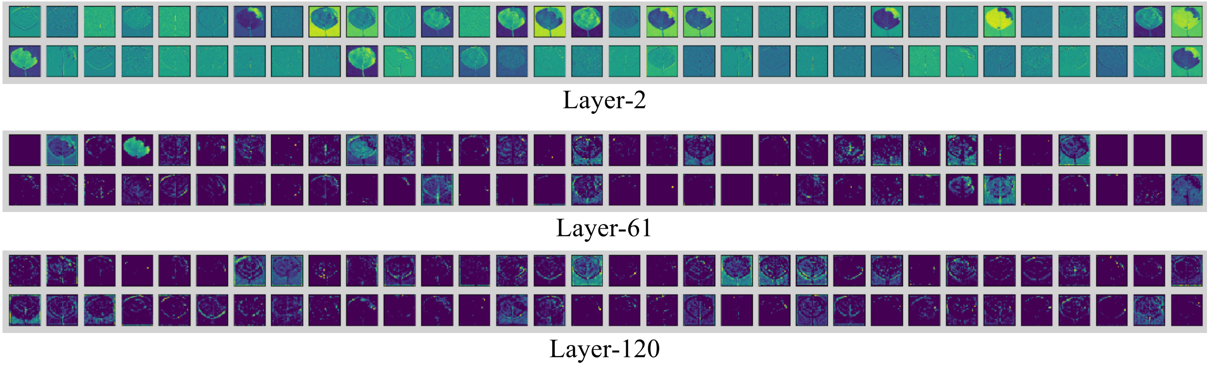

**Fig. (8).** Densenet121 illustrations of various layers for Apple. (*A higher resolution / colour version of this figure is available in the electronic copy of the article*).

### 5.2. Low Sample Size

With a low sample space, most of the deep learning models cannot make inferences as the pattern recognition capability for them is not nurtured during training. Therefore, with a few samples, we cannot decide with convincing results.

### 5.3. Data Imbalance

Generally, in many of the cases in deep learning, we observe class imbalance in data which means one or the other class is having a majority amount of data and others do not. This inappropriate distribution of data amongst the classes in a model makes a model learn better for the class having a higher number of samples which may cause overfitting in data. However, we observed that VGG Models were able to give efficient and insightful results even for an imbalanced class. During the prosecution, it was observed that Tomato and Potato appeared to have a high imbalance of 8.8% and 7.1% in the healthy class, respectively. The results generated seemed to have high precision and F1-score only for Infection class as they contained large data for some models. However, VGG models were elegant and overcame this issue with that of other mentioned models.

## CONCLUSION

In this paper, we analyzed and assessed (Table **2**) the state-of-the-art models proposed for the complex machine vision problems. As the models are pre-trained with optimum weights, a design-oriented approach is developed with various leaf-images and classified them into infection and healthy class for each type of leaf individually. With this analysis, it is concluded that VGG models outperform with various mentioned models (Table **1**) and give on par outcomes. As mentioned, these VGG models (VGG16 & VGG19) are used for feature extraction mechanisms; furthermore, a set of dense layers is added to train them accordingly. We have visually seen the performance of VGG models and their elegant architecture that is not only able to extract precise features but also overcome various pitfalls.

The performance is evaluated for various kinds of leaf images. The accuracy scores are above 97.5% for VGG16 and 96.72% for VGG19. AUC-ROC [69, 70] curves are depicted to illustrate its influence in determining an accurate classification whereas VGG16 and VGG19 have at least 96.6% and 95% AUC, respectively. These VGG Models were resilient to a lower sample size and did not show drastically varying performance with data imbalance. In the future, classifying each typical infection in each crop (leaf) would be taken as a challenge to design a reliable model.

## CONSENT FOR PUBLICATION

Not applicable.

## FUNDING

This work was partially supported by the TEQIP-III from Jawaharlal Nehru Technological University Hyderabad, under Collaborative Research Scheme.

## CONFLICT OF INTEREST

The author declares no conflict of interest, financial or otherwise.

## ACKNOWLEDGEMENTS

Declared none.

## REFERENCES

[1] A. Camargo, and J. S. Smith, "Image pattern classification for the identification of disease causing agents in plants", *Comput. Electron. Agric.,* vol. 66, no. 2, pp. 121-125, 2009. http://dx.doi.org/10.1016/j.compag.2009.01.003

[2] T. Rumpf, "Early detection and classification of plant diseases with support vector machines based on hyperspectral reflectance", *Comput. Electron. Agric.,* vol. 74, no. 1, pp. 91-99, 2010. http://dx.doi.org/10.1016/j.compag.2010.06.009

[3] A. Bashish, *Dheeb, Malik Braik, and Sulieman Bani-Ahmad. "A framework for detection and classification of plant leaf and stem diseases." 2010 international conference on signal and image processing..* IEEE, 2010.

[4] H. Al-Hiary, "Fast and accurate detection and classification of plant diseases", *Int. J. Comput. Appl.,* vol. 17, no. 1, pp. 31-38, 2011.

[5] D. Al Bashish, M. Braik, and S. Bani-Ahmad, "Detection and classification of leaf diseases using K-means-based segmentation and", *Information Technology Journal,* vol. 10, no. 2, pp. 267-275, 2011. http://dx.doi.org/10.3923/itj.2011.267.275

[6] S. Arivazhagan, "Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features", *Agric. Eng. Int. CIGR J.,* vol. 15, no. 1, pp. 211-217, 2013.

[7] S. Sladojevic, M. Arsenovic, A. Anderla, D. Culibrk, and D. Stefanovic, "Deep neural networks based recognition of plant diseases by leaf image classification", *Comput. Intell. Neurosci.,* vol. 2016, 2016.3289801 http://dx.doi.org/10.1155/2016/3289801 PMID: 27418923

[8] S.P. Mohanty, D.P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection", *Front. Plant Sci.,* vol. 7, p. 1419, 2016. http://dx.doi.org/10.3389/fpls.2016.01419 PMID: 27713752

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, Imagenet classification with deep convolutional neural networks.Advances in neural information processing systems. 2012, pp. 1097-1105

[10] C. Szegedy, "Going deeper with convolutions", *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 2015pp. 1-9 Boston, MA, USA.

[11] V. Singh, and A. K. Misra, "Detection of plant leaf diseases using image segmentation and soft computing techniques", *Inf. Process. Agric.,* vol. 4, no. 1, pp. 41-49, 2017. http://dx.doi.org/10.1016/j.inpa.2016.10.005

[12] Jihen Amara, Bassem Bouaziz, and Alsayed Algergawy, "A deep learning-based approach for banana leaf diseases classification", *Datenbanksysteme für Business, Technologie und Web (BTW 2017)-Workshopband,* 2017.

[13] M. H. Saleem, J. Potgieter, and K. Mahmood Arif, "plant disease detection and classification by deep learning", *Plants (Basel),* vol. 8, no. 11, E468, 2019. http://dx.doi.org/10.3390/plants8110468 PMID: 31683734

[14] N. Petrellis, "Mobile application for plant disease classification based on symptom signatures", *Proceedings of the 21st Pan-Hellenic Conference on Informatics,* 2017 http://dx.doi.org/10.1145/3139367.3139368

[15] M. Brahimi, K. Boukhalfa, and A. Moussaoui, "Deep learning for tomato diseases: classification and symptoms visualization", *Appl. Artif. Intell.,* vol. 31, no. 4, pp. 299-315, 2017. http://dx.doi.org/10.1080/08839514.2017.1315516

[16] D. Oppenheim, and G. Shani, "Potato disease classification using convolution neural networks", *Adv. Anim. Biosci.,* vol. 8, no. 2, pp. 244-249, 2017. http://dx.doi.org/10.1017/S2040470017001376

[17] Alvaro Fuentes, "A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition", *Sensors 17.9,* 2017. http://dx.doi.org/10.3390/s17092022

[18] A. Camargo, and J. S. Smith, "An image-processing based algorithm to automatically identify plant disease visual symptoms." *Biosystems engineering* 102.1 (2009): 9-21.

[19] T. M. Prajwala, A. Pranathi, K. S. Ashritha, Nagaratna B. Chittaragi, S. Koolagudi, "Tomato leaf disease detection using convolutional neural networks", *IEEE Xplore, Eleventh International Conference on Contemporary Computing (IC3),* 2018

[20] W-S. Jeon, and S-Y. Rhee, "Plant Leaf Recognition Using a Convolution Neural Network", *Int. J. Fuzzy Logic Intelligent Syst.,* vol. 17, no. 1, pp. 26-34, 2017. http://dx.doi.org/10.5391/IJFIS.2017.17.1.26

[21] P. M. Mainkar, S. Ghorpade, and M. Adawadkar, "Plant leaf disease detection and classification using image processing techniques", *Int. J. Innovative Emerging Res. Engineering,* vol. 2, no. 4, pp. 139-144, 2015.

[22] S. Wallelign, M. Polceanu, and C. Buche, "soybean plant disease identification using convolutional neural network", *Artificial Intelligence Research Society Conference (FLAIRS-31),* 2018pp. 146-151

[23] I. Goodfellow, H. Lee, Q.V. Le, A. Saxe, and A.Y. Ng, "Measuring invariances in deep networks", *Adv. Neural Inform. Processing syst.,* pp. 646-654, 2009.

[24] G. Larsson, M. Maire, and G. Shakhnarovich, "Fractalnet: Ultra-deep neural networks without residuals", *ArXiv Preprint ArXiv:1605.07648,* 2016.

[25] C. Cortes, X. Gonzalvo, V. Kuznetsov, M. Mohri, and S. Yang, "Adanet: Adaptive structural learning of artificial neural networks", *Proceedings of the 34th Int. Conference Machine Learning JMLR,* vol. 70, 2017pp. 874-883

[26] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks", *Proceedings of the IEEE conference on computer vision and pattern recognition,* 2017pp. 1492-1500 http://dx.doi.org/10.1109/CVPR.2017.634

[27] M. Tan, and Q.V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks", *ArXiv Preprint ArXiv:1905.11946,* 2019.

[28] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: a review of methods and applications", *arXiv preprint arXiv:1812.08434,* 1812.

[29] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, "Gated graph sequence neural networks", *ArXiv Preprint ArXiv:1511.05493,* 1511.

[30] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks", *ArXiv Preprint ArXiv:1710.10903,* 1710.

[31] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch", *J. Mach. Learn. Res.,* vol. 12, no. Aug, pp. 2493-2537, 2011.

[32] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, "Gradient-based learning applied to document recognition", *Proceedings IEEE,* vol. 86, no. 11, pp. 2278-324, 1998.

[33] J. Deng, W. Dong, R. Socher, L.-J. Li, and K. Li, "Imagenet: A large-scale hierarchical image database", *CVPR,* 2009.

[34] S. J. Pan, and Q. Yang, "A survey on transfer learning", *IEEE Trans. Knowl. Data Eng.,* vol. 22, no. 10, pp. 1345-1359, 2010. http://dx.doi.org/10.1109/TKDE.2009.191

[35] K. Weiss, T.M. Khoshgoftaar, and D. Wang, "A survey of transfer learning", *J. Big Data,* vol. 3, p. 9, 2016. http://dx.doi.org/10.1186/s40537-016-0043-6

[36] Y. Bengio, "Deep learning of representations for unsupervised and transfer learning", JMLR: *Workshop Conference Proceedings* vol. 27, pp. 17-37, 2012.

[37] G. Mesnil, Y. Dauphin, X. Glorot, S. Rifai, Y. Bengio, I. Goodfellow, E. Lavoie, X. Muller, G. Desjardins, D. Warde-Farley, and P. Vincent, "Unsupervised and transfer learning challenge: a deep learning approach", JMLR: Workshop and Conference Proceedings, vol. 27. pp. 97-111, 2012.

[38] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M.S. Lew, "Deep learning for visual understanding: A review", *Neurocomputing,* vol. 187, pp. 27-48, 2016.
http://dx.doi.org/10.1016/j.neucom.2015.09.116

[39] F.N. Iandola, S. Han, M.W. Moskewicz, K. Ashraf, W.J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and < 0.5 MB model size", *ArXiv Preprint ArXiv:1602.07360,* 1602.

[40] S. L. Smith, P. J. Kindermans, C. Ying, and Q. V. Le, "Don't decay the learning rate, increase the batch size", *ArXiv Preprint ArXiv:1711.00489,* 1711.

[41] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, Rethinking the inception architecture for computer vision. *CVPR.* 2016.
http://dx.doi.org/10.1109/CVPR.2016.308

[42] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway networks", *ArXiv Preprint ArXiv:1505.00387,* 1505.

[43] J Moody, and C Darken, "Learning with localized receptive fields",

[44] W. Luo, Y. Li, R. Urtasun, and R. Zemel, “Understanding the effective receptive field in deep convolutional neural networks”, *Adv. Neural Information Processing Syst.,* 2016, pp. 4898-4906.

[45] S. Karen, Z. andrew, “Very deep convolutional networks for large-scale image recognition", *ArXiv 1409.1556.,* 2014.

[46] K. He, X. Zhang, S. Ren, J. Sun, “Deep residual learning for Image recognition", Computer Vision and Pattern Recognition, Dec. 2015 .

[47] K. He, and X. Zhang, "Identity mappings in deep residual networks", *European conference on computer vision,* , 2016pp. 630-645.

[48] G. Huang, Z. Liu, L. Van Der Maaten, and K.Q. Weinberger, "Densely connected convolutional networks", *Proceedings of the IEEE conference on computer vision and pattern recognition,* 2017pp. 4700-4708.
http://dx.doi.org/10.1109/CVPR.2017.243

[49] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications", *ArXiv,* 2017.

[50] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks", *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition,* 2018pp. 4510-4520 Salt Lake City, UT
http://dx.doi.org/10.1109/CVPR.2018.00474

[51] F. Chollet, "Xception: deep learning with depthwise separable convolutions", *Conference on Computer Vision and Pattern Recognition*, 7, Oct. 2016.

[52] M. Lin, Q. Chen, and S. Yan, *Network in Network, CoRR,* 2013.

[53] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning", *Nature***,** Vol. 521, pp. 436-444, 2015.

[54] I. Goodfellow, Y. Bengio, and A. Courville, "Deep learning",

[55] C. Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu, "Deeply-supervised nets", *Artificial Intelligence Statistics,* pp. 562-570, 2015.

[56] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines", *In Proceedings of the 27th International Conference on International Conference on Machine Learning (ICML’10). Omnipress, Madison, WI, USA,* pp. 807-814, 2010.

[57] W. Liu, Y. Wen, Z. Yu, and M. Yang, Large-margin softmax loss for convolutional neural networks. *ICML,* Vol. 2, pp. 7, 2016.

[58] Y. L. Boureau, F. Bach, Y. LeCun, and J. Ponce, "Learning mid-level features for recognition", *IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* 2010, pp. 2559-2566 *IEEE*, 2010.
http://dx.doi.org/10.1109/CVPR.2010.5539963

[59] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition", *IEEE Trans. Pattern Anal. Mach. Intell.,* Vol. 37, 2015, pp. 1904-1916.
http://dx.doi.org/10.1109/TPAMI.2015.2389824 PMID: 26353135

[60] S. L. Smith, P. J. Kindermans, C. Ying, and Q. V. Le, "Don't decay the learning rate, increase the batch size", *ArXiv Preprint ArXiv:1711.00489,* 1711.

[61] E. Hoffer, I. Hubara, and D. Soudry, Train longer, generalize better: closing the generalization gap in large batch training of neural networks.*Advances in Neural Information Processing Syst.* 2017, pp. 1731-1741.

[62] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, and M. Kudlur, "Tensorflow: A system for large-scale machine learning", *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16),* 2016pp. 265-283

[63] Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, and L.D. Jackel, "Backpropagation applied to handwritten zip code recognition", *Neural Comput.,* 1989.
http://dx.doi.org/10.1162/neco.1989.1.4.541

[64] R. Hecht-Nielsen, Theory of the backpropagation neural network.*Neural networks for perception..* Academic Press, 1992, pp. 65-93.
http://dx.doi.org/10.1016/B978-0-12-741252-8.50010-8

[65] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors", Nature**,** Vol. 323, no. 6088, pp. 533-536, 1986.

[66] D. Kingma, and J. Ba, "Adam: A Method for Stochastic Optimization", *International Conference on Learning Representations,* 2014

[67] S. Ioffe, and C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift*ICML,* 2015.

[68] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, *Dropout: A simple way to prevent neural networks from overfitting,* JMLR, 2014.

[69] J. Davis, and M. Goadrich, "The relationship between Precision-Recall and ROC curves", *Proceedings of the 23rd international conference on Machine learning,* 2006, pp. 233-240.
http://dx.doi.org/10.1145/1143844.1143874

[70] K. Hajian-Tilaki, "Receiver operating characteristic (ROC) curve analysis for medical diagnostic test evaluation", *Caspian J. Intern. Med.,* vol. 4, pp. 627-635, 2013.
PMID: 24009950