Leveraging Deep Learning for Advanced Plant Identification and Health Assessment: An Exploration of Techniques, Datasets, and Practical Applications.

This dissertation evaluates the effectiveness of state-of-the-art Convolutional Neural Network (CNN) models for plant identification and health assessment using real-world datasets. The study encompasses a comprehensive training pipeline, a review of pertinent literature, and a comparative analysis of performance metrics applied to the Plant Village dataset. The training pipeline delineates the implementation of cutting-edge CNN models for real-life plant identification and classification challenges. Performance metrics are utilized to gauge the efficacy of these models, providing empirical data on their performance. Additionally, the comparative analysis assesses the architectural merits of each model and its efficacy in mitigating issues such as the vanishing gradient problem. Through this research, we aim to demonstrate the potential of CNN models in automating agricultural processes, thereby contributing to increased efficiency and productivity in the agricultural sector.

I certify that all material in this dissertation which is not my own work has been identified.

Christopher Man

April 2024

Contents

1 Introduction

In recent years, convolutional neural networks (CNNs) have experienced an explosive growth in research and literature, underscoring their remarkable versatility and effectiveness across diverse domains. Among these, the agricultural sector emerges as an especially promising arena for their application. Employing cutting-edge deep learning techniques for plant health and identification stands as a crucial aspect of technological advancement in agriculture. In countries like Ethiopia, Mali, Comoros, and Liberia, where agriculture comprises an average GDP share of 36.66% [1], leveraging CNN technology in vertical farming holds immense promise. It can elevate food production, conserve land and water resources, create employment opportunities, enhance access to fresh produce, bolster climate resilience and promote sustainability. These efforts collectively contribute to both economic progress and environmental stewardship. Furthermore, vertical farming is an innovative approach to agriculture where crops are cultivated in stacked layers within controlled environments, such as indoor facilities or greenhouses [2]. The application of CNN's in this context can revolutionise the way crops are grown and managed. It is then identified that the application of a high quality CNN within a vertical form for crop production and management, can lead to major positive impacts for countries that have a large proportion of their GDP reliant on agriculture.

The application of a high quality CNN's in this context can increase automation and plant disease detection which would increase the overall efficiency of harvesting and monitoring. There would be major global positive benefits to crop health and success if we automated plant disease detection systems early on in the agriculture process, mainly due to the fact that a large amount of crops fail due to a lack of early health monitoring and intervention. Kitran and authors identified something similar [3], suggesting that that as the world population increases at a rapid rate the direct need for food increases. The traditional methods farmers used are not sufficient in serving the increased demand hence they hamper soil health via the use of dangerous pesticides. "*This affects the agricultural practice a lot and in the end the land remains barren with no fertility*" [3]. However they propose that through the use of automation we can direct combat this issue whilst also increasing overall food production without harmful impacts on the environment. Their work supporting the argument that automation of agriculture practices including artificial intelligence may have a large positive effect to entire agricultural sector. [3].

Therefore the need for a machine learning based plant identification research paper arises "*Leveraging Deep Learning for Advanced Plant Identification and Health Assessment: An Exploration of Techniques, Datasets, and Practical Applications*". My research question not only adds to the current literature on deep learning models, but provides conclusive results on how the cutting edge CNN models perform on the "*Plant Village*" dataset [4]. The performance metrics produced by transfer learning aid in the comparative analysis of each model. This analysis illuminating whether or not CNN models can yield high enough accuracy to be used practically within the automation of agricultural tasks; But also suggesting how the model design and architecture of each model impact model performance and computational resource usage.

2 Materials and Methodology

2.1 Dataset

When examining deep learning models such as convolutional neural networks (CNNs) employed in computer vision applications, the quality of the dataset used is of high importance. While the design and complexity of the model are significant factors, a high-quality dataset is indispensable for achieving optimal accuracy in model performance. Abhinav Jain and authors [5] identified 6 factors of a structured dataset which contribute to having high quality datasets: ''*Label Noise, Class Imbalance, Data Valuation, Data Homogeneity, Data Transformation, and Data Cleaning*". To evaluate the top performing CNN models backed by contemporary literature and research, we will analyse the "*Plant Village*" dataset through some of these factors.

Created by David P. Hughes et al, uploaded to Tensorflow, the "*Plant Village*" dataset consists of 54,303 healthy and unhealthy leaf images divided into 38 categories based on species and disease

Figure 1: Example data from Plant Village [4]

[4]. This dataset was selected not only for its high-quality features, but also for its comprehensive 38 classifications. Each of the 38 different classifications include plants in their diseased and non diseased counter parts, each classification containing a large volume of images. For example the "*Apple Scab Leafed Plant*" has around 1000 images of that classification. The images are from a multitude of angles with different lightning conditions *see figure 1.*

The authors of the "*Plant Village*" dataset were able to achieve a large set of images via data augmentation. A method of artificially transforming each image to inflate the total amount of images. This technique of using data augmentation was analysed in the literature "*Data augmentation for improving deep learning in image classification problem*" in which the Authors had seen a profound increase in model accuracy for image classification when using data augmentation [6].

Machine learning researchers employ diverse strategies of data augmentation to rapidly expand image datasets. Some of the methods that the authors of the "*Plant Village*" dataset employed methods such as rotation, flipping, scaling, cropping, translation, noise addition, and color adjustments. By varying these parameters, they were able to generate a multitude of new images, enriching the dataset's diversity and enhancing the model's ability to generalize effectively. Hence at a cursory glance of the "*Plant Village*" dataset we are able to identify that it is a high quality dataset which can be used in training a deep machine learning model.

2.2 Transfer Learning

Transfer learning is a method of using a model with pre trained weights without having to train a model from scratch identified in the paper "*Exploring Benefits of Transfer Learning in Neural Machine Translation*". Tom Kocmi identified that using transfer learning can greatly decrease total training time and computational resources needed. This approach of using transfer learning is another dimension of scaling a model that many researchers are now using when training CNN models [7].

Another method of using transfer learning is to freeze 80-90% of the models layers leaving some of the final layers to be changed by training cycle. However Tom Kocmi found that freezing too many layers would lead the network to "*struggle to change its behaviour from the parent when most of the network was frozen*" [7]. Suggesting that model freezing in some applications would greatly hinder the model sensitivity and ability to adapt to diverse dataset. However, greatly speeding up the training time of the model as it would only need to update the weights of the final layers.

Due to the benefits of transfer learning such as a reduction in model training time and a higher

starting accuracy within early training cycles, this method of model utilization will be used for the comparative analysis conducted. However, layer freezing within transfer learning will not be utilized. The justification for this is that the images within "*Plant Village*" have a large diversity because of the different type of plant species. Meaning that the model my struggle to adapt and be sensitive to the dataset. In using transfer learning on pre existing models without layer freezing during training, would still see a large benefit as it is unlikely that the majority of the layer weights in each model would change as they are already pre optimised for computer vision tasks. Therefore transfer learning without layer freezing will be used for the comparative analysis of VGG11 [8], GoogleNet [9], ResNet [10], DenseNet [11], MobileNet [12] and EfficientNet [13] models on the "*Plant Village*" dataset.

2.3 State of the Art CNN Models

Convolutional Neural Networks (CNN's) are an effective method of processing images through a multi-layered network to classify and identify features in an image. With the current literature suggesting that CNN models provide the highest accuracy in any image classification and object detection task [7]. Traditional CNN models consist of a section of hidden layers and a section of classification layers. The hidden layers derive from convolutional, activation and max pooling layers. Alongside the classification layer, matching the layers to an output node and normalised using a soft-max function generally [14].

With the rise of machine learning in research and real world applications, researchers have developed their own models with permutations based upon the traditional CNN model. Generally researchers will approach increasing a model's scale by either increasing the depth or width of the model, and in some cases taking a unique approach [13].

The CNN models that will be used for a comparative analysis on the "*Plant Village*" dataset will be the models VGG11 [8], GoogleNet [9], ResNet [10], DenseNet [11], MobileNet [12] and EfficientNet [13]. An analysis on the models performance metrics on the "*Plant Village*" dataset and the model architecture, will be used to justify if the specific model can be leveraged in real world plant identification and classification tasks.

2.3.1 VGG Model

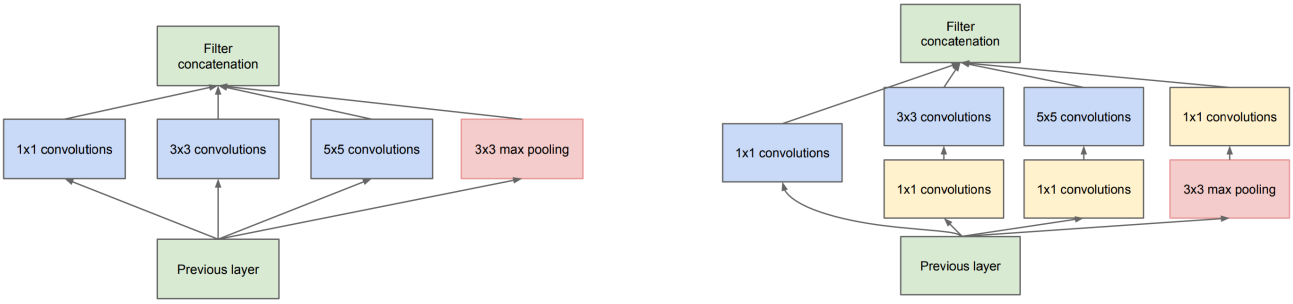
In 2015 Karen Simonyan and Andrew Zisserman presented the VGG11 model in "*Very Deep Convolutional Neural Networks For Large-Scale Image Recognition*" [8]. The VGG11 model during its time in 2015 was one of the first models which took a new direction in improving model performance. Modifying the layer size of each convolution to be set of 3x3 convolutional filters that could parse very deeply in terms of model dimension.

Being one of the earliest models that focused on model depth, it was able to achieve extremely high accuracy on datasets that were used to compare machine learning models. Winning the "*ImageNet Challenge in 2014*", the authors were able to achieve first and second place in regards to classification and localisation tasks [8].

The VGG11 architecture is a CNN model that is now used as a control group for testing newer machine learning models due to it being one of the first CNN models to successfully to create a very deep set of layers in order to achieve higher model performance. The model increasing the dimension of depth, via optimization. Opposed to increasing model width and or layer by layer architecture. This new type of model architecture was one of the early markers which suggested the potential for extremely high accuracy CNN models; But also for a model that could be used in a practical real world setting with much utility. Henceforth, the VGG11 model will be used in testing as the lower boundary benchmark compared to the other newer CNN models; Granting perspective on the leaps and bounds in innovation that the newer models provide.

2.3.2 GoogleNet Model

Authors Christian Szegedy from Google et al in the paper "*Going deeper with convolutions*" cited a new approach to achieving high accuracy deep learning models, highlighting the need for an accurate model which focuses on computational resource optimization. Aiming to not only increase the depth of the model but also to increase the width of the network [9]. The group of researchers based their approach



(a) Inception Module, naive version [13]

(b) Inception Module, dimension reductions [13]

Figure 2: GoogleNet Architecture

on the "*Hebbian Principle*". The principle coming form the work of Donald Hebb stating "*When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased*"' [15]. From that principle they formed the Inception Model, a model that forms clusters of units together in the next layer, these are then connected to the units from the previous layer. Grouping the units together into generalised filter banks, formed into groups based upon units with high correlation assuming each unit from previous layers refer to some part of the input image *see figure 2b*.

This theory deriving the Inception model which is now known as the GoogleNet model. After they conducted a comparative performance evaluation analysis they concluded that "*Our results seem to yield solid evidence that approximating the expected optimal sparse structure by readily available dense building blocks is a viable method for improving neural networks for computer vision*". Concluding that expanding each layer horizontally while increasing density yields a valuable impact on performance metrics of a model [9]. In practical application during the "*ILSVRC 2014*" classification challenge GoogleNet came 1st place in 2014 with the lowest top-5 error percent.

The difference that this model has compared to other CNN models is that it focuses on creating a more compact wider model via having optimized a cluster of layers together. A rather different approach to other models which focus on increasing model depth. This unique approach to model scaling may largely be beneficial for in yielding a high accuracy model in regards to plant classification tasks.

2.3.3 ResNet

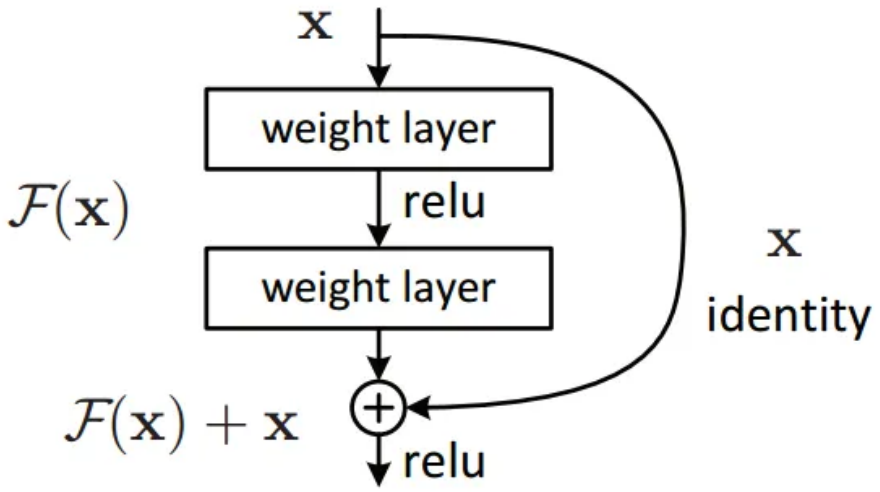
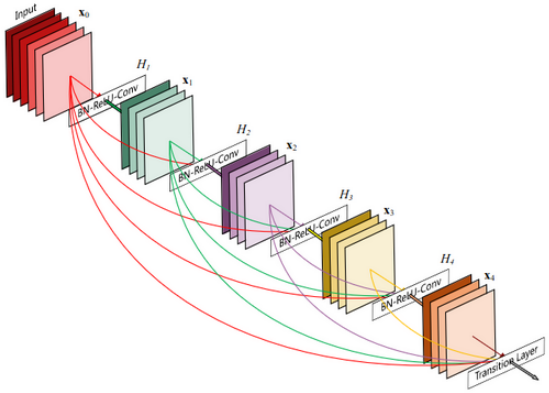


Figure 3: Residual Block Design [16]

In the research "*Deep Residual Learning for Image Recognition*" [10] a ResNet model was proposed. Authors Kaiming He et al were able to achieve a model that not only won first place in the "*ILSVRC 2015*" classification competition but in 6 others being the "*ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation in ILSVRC and COCO 2015 competitions*" [10] . This newly proposed ResNet had a complex modification to simple traditional CNN's which had a regular series of layers. The Resnet model instead uses a series of residual blocks *see figure 3*. This residual block method of utilising layers meant that some layers may be skipped and not used. The researchers defined this as a "*Skip Connection*", directly combating the issues of the vanishing gradient problem. Where as a model becomes deeper and more layers are added alongside activation functions, the gradient of the loss function approaches zero. This means that the model becomes harder to train on a dataset.

Kaiming He and authors were able to yeild a large margin of decreased error with their model compared to other CNN models. Training on the "*ILSVRC 15*" dataset model they were able to achieve a top-5 localization error of only 8.9 %. Whereas other models such as VGG11 achieved a 26.9%. The "*Skip Connection*" feature of the ResNet model allowed it to have dominant performance compared to previous CNN model architecture. The direction that the authors took to greatly increase model depth whilst addressing over fitting and the Vanishing Gradient Problem was a unique approach that greatly changed the research field of convolutional neural networks. Other models that were proposed afterward such as DenseNet and EfficientNet later adopted similar features present in ResNet. The unique "*Skip Connection*" feature of ResNet alongside its high performance metrics suggest that it can have applicable use in plant identification and classification during the comparative analysis.

2.3.4 DenseNet Model



(a) DenseNet Architecture Example [11]

Method	Depth	Params	C10	C10+	C100	C100+	SVHN
Network in Network [22]	-	-	10.41	8.81	35.68	-	2.35
All-CNN [32]	-	-	9.08	7.25	-	33.71	-
Deeply Supervised Net [20]	-	-	9.69	7.97	-	34.57	1.92
Highway Network [34]	-	-	-	7.72	-	32.39	-
FractalNet [17]	21	38.6M	10.18	5.22	35.34	23.30	2.01
with Dropout/Drop-path	21	38.6M	7.33	4.60	28.20	23.73	1.87
ResNet [11]	110	1.7M	-	6.61	-	-	-
ResNet (reported by [13])	110	1.7M	13.63	6.41	44.74	27.22	2.01
ResNet with Stochastic Depth [13]	110	1.7M	11.66	5.23	37.80	24.58	1.75
	1202	10.2M	-	4.91	-	-	-
Wide ResNet [42]	16	11.0M	-	4.81	-	22.07	-
	28	36.5M	-	4.17	-	20.50	-
with Dropout	16	2.7M	-	-	-	-	1.64
ResNet (pre-activation) [12]	164	1.7M	11.26*	5.46	35.58*	24.33	-
	1001	10.2M	10.56*	4.62	33.47*	22.71	-
DenseNet ($k = 12$)	40	1.0M	**7.00**	5.24	**27.55**	24.42	1.79
DenseNet ($k = 12$)	100	7.0M	**5.77**	**4.10**	**23.79**	**20.20**	1.67
DenseNet ($k = 24$)	100	27.2M	**5.83**	**3.74**	**23.42**	**19.25**	**1.59**
DenseNet-BC ($k = 12$)	100	0.8M	**5.92**	4.51	**24.15**	22.27	1.76
DenseNet-BC ($k = 24$)	250	15.3M	**5.19**	**3.62**	**19.64**	**17.60**	1.74
DenseNet-BC ($k = 40$)	190	25.6M	-	**3.46**	-	**17.18**	-

(b) DenseNet Performance Comparison [11]

Figure 4: DenseNet Architecture and Performance Comparison

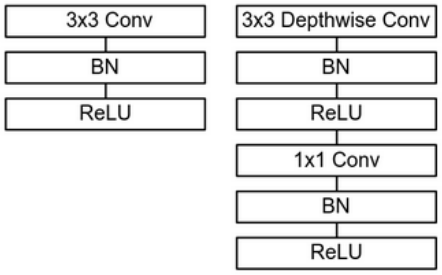
In the paper "*Densely Connected Convolutional Networks*", the authors proposed a new type of model which combines features of layer by concatenation similar to how ResNet model uses residual blocks. The researchers proposed the DenseNet model, claiming extremely deep model depth, whilst retaining high accuracy and efficiency when training. [11]. To optimise the flow of information across network layers, DenseNet directly links layers that have corresponding feature-map dimensions. Maintaining the feed-forward structure, each layer receives extra inputs from preceding layers and transmits its feature-maps to subsequent layers . The published work refers that the model has a "*counter-intuitive effect of this dense connectivity pattern*" [11].

Surmising that the DenseNet models ability to not have to learn redundant feature maps, allows the model complexity as a whole to be significantly reduced. Seeing a large reduction in total model parameters. In training the model, the researchers were able to achieve high performance metrics with DenseNet in comparison to other models at the time *see figure 4b*.

The models feature of reducing total parameters via removing redundant feature maps not only reduces total complexity but also combats the vanishing gradient problem. The communication of inter

networked layers, allows the model to develop extreme model depth whilst model sensitivity. This is seen as a large benefit to plant identification tasks such as identified in the "*Analysis of the Effect of Data Scaling on the Performance of the Machine Learning Algorithm for Plant Identification*" [6]. The authors of DenseNet proposed that the dense architecture of the model has a regularising effect, reducing over fitting on tasks with smaller training set sizes. [11]. In the "*Plant Village*" dataset some of the plant species have a varying amount of images, suggesting that DenseNet may be able to achieve high precision and recall for those plant species with limited data.

2.3.5 MobileNet Model



(a) Left: Standard Convolutional layer with batchnrom and ReLU. Right: Depthwise Separable convolutions with depthwise and Pointwise layers followed by batchnorm and ReLU [12]

Model	Top-1 Accuracy	Million Mult-Adds	Million Parameters
Inception V3 [18]	84%	5000	23.2
1.0 MobileNet-224	83.3%	569	3.3
0.75 MobileNet-224	81.9%	325	1.9
1.0 MobileNet-192	81.9%	418	3.3
0.75 MobileNet-192	80.5%	239	1.9

(b) MobileNet for Stanford Dogs [17]

Figure 5: MobileNet Construction and Stanford Dogs Performance

In 2017 Andre G Howard et al proposed the MobileNet model in the research paper "*MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*" [12]. The authors identifying the need for a low resource intensive deep neural network, proposed a model that utilises depth wise separable convolutions.

The MobileNet model was then presented achieving the balance between low latency and high accuracy, alongide a tune-able hyper parameterised architecture. [12]. The depth wise separable convolution used in MobileNet is a type of factorised convolutional technique. Breaking down a traditional convolution into two distinct components; by first applying a unique filter to each input channel by performing a depth-wise convolution then merging the results with a 1 x 1 convolutional layer. Unlike a conventional convolutions that simultaneously filters and integrate inputs into a fresh output set, depth-wise separable convolutions divides this process into two separate stages: one for filtering and another for integration *see figure 5a.* This decomposition technique significantly minimises computational requirements and reduces the model's overall size.

The researchers using the MobileNet model on the Stanford Dogs Dataset [17], found that compared to other CNN models it had a significant reduction in total model parameters. Finding that the "*Inception V3*" also known as GoogleNet, had a top-1 accuracy of 84% with a total of 23.2 million parameters, whereas the "*1.0 MobileNet-244*" had a top-1 accuracy of 83.3% and a total of 3.3 million parameters. Seeing a reduction in the total complexity of the model by roughly 19.9 million parameters whilst only seeing a reduction in accuracy by 0.7% *see figure 5b.* When looking at increasing model performance researchers tend to only approach increasing model width or depth. However the authors of MobileNet took a different approach, they optimised a model so that it uses significantly less computational resources whilst retaining high accuracy. This is an important factor that makes MobileNet stand out compared to other models. As in research or real world scenarios computational resources are always limited. So this MobileNet model may be very practical in training when attempting to be leverage in a plant classification and identification tasks. By not only achieving high accuracy but also taking into account of the models environmental limitations.

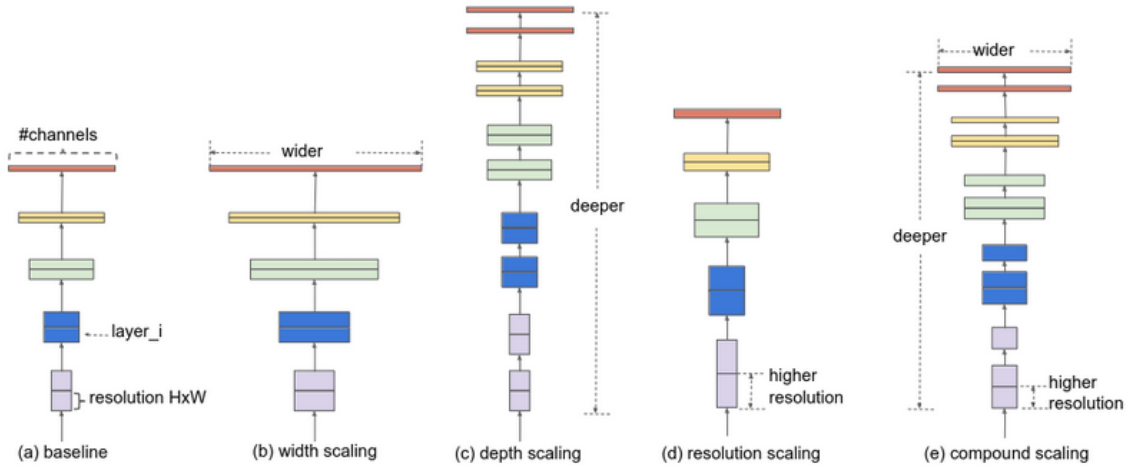


Figure 6: EfficientNet Cited Design [13]

$$\mathcal{N} = \bigodot_{i=1\ldots s} \mathcal{F}_i^{L_i}\left(X_{\langle H_i, W_i, C_i\rangle}\right)$$

Figure 7: EfficientNet Scaling Coefficient [13]

2.3.6 EfficientNet

EfficientNet a CNN model from Mingxing Tan and Quoc Le took a unique approach to scaling the dimensions of a CNN model. Instead of increasing one dimension of a model such as ResNet and DenseNet, the authors created a compound scaling coefficient; Scaling baseline, width, depth, resolution and compound scaling *see figure 6*. In order to present this multi level directional scaling the authors created a scaling coefficient formula *see figure 7*. Using this compound scaling coefficient on models such as MobileNet, ResNet and ImageNet the authors were able to largely increase the accuracy of these models from their baseline architecture; Seeing positive increases in all performance metrics. In one experiment they decreased the inference latency comparison measured with batch size 1 on a single core of Intel Xeon CPU E5-2690, by 5.7x compared to ResNet and EfficientNet , alongside increasing Top-1 Accuracy across the board from 3-4% [13].

The fundamental approach of scaling a model dimension in all directions as described by Mingxing Tan et al, largely changed and impacted the landscape of CNN's. Especially in regards to machine learning tasks used for object classification by not only reducing the total parameters of a model but also by increasing its true accuracy [13] . Hence, this proposed multi directional scaling coefficient, suggests the ability to leverage EfficientNet in providing high quality predictions in a plant health identification and classification tasks. The models ability to scale effectively depending on the computational resource and the dataset may indicate this model being highly performant when used on the "*Plant Village*" dataset.

3 Experiment Design

3.1 Training Setup

3.1.1 Hardware

Graphics processing units (GPU's) allow machine learning models to train significantly faster, due to GPU architecture being greatly beneficial for parallel matrix calculations. GPU architecture can also be easily mapped to the kernals within a machine learning model. Seeing a wide use of GPUs within model training and evaluation within machine learning research [18].

All CNN models trained within the comparative analysis will be complied with GPU support. All

code built within GoogleCollab [19] using PyTorch [20], a open source deep machine learning library written in the Python Language. The code running on a machine using Debian latest stable version running on eight "*Intel(R) Xeon(R) CPU @ 2.30GHz*" in parallel and a "*Tesla T4 GPU*" with 51 GB of high speed RAM.

3.1.2 Training Pipeline

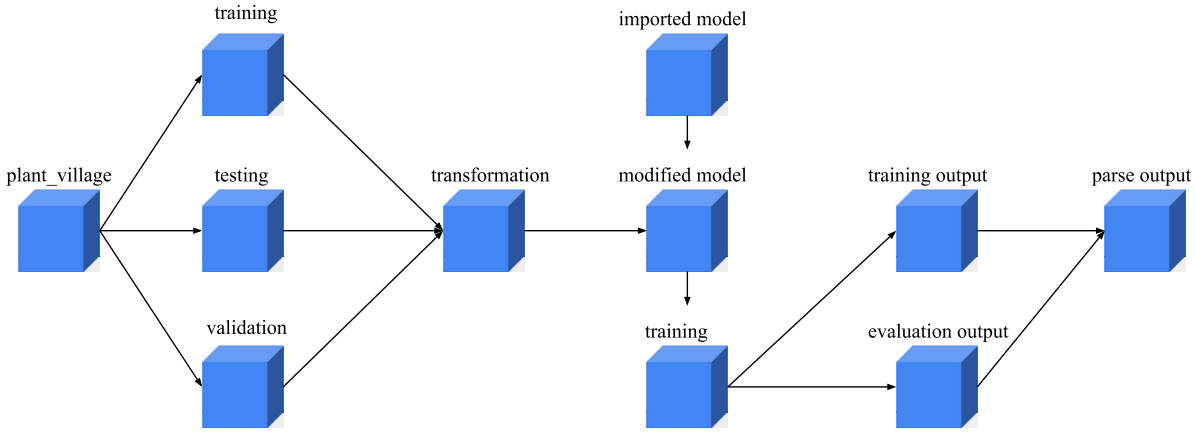


Figure 8: Training Pipeline

The "*Plant Village*" dataset was downloaded and ported to the GoogleCollab [19] coding environment. A custom data loader written in Python [20], dividing the data in "*Plant Village*" by training, testing and validation datasets. Proportionally divided up by a percentage of 80% training, 10% testing and 10% validation datasets. The data was partitioned randomly, seeded by each training of the models. This ensured experiment diversity by allowing each model to have a unique spread of plant images when being trained.

The PyTorch TorchVision library [20] was then used to transform the datasets for prepossessing and augmentation. Resizing all the images to 244 x 244, normalised with a mean = [0.485, 0.456, 0.406] and std [0.229, 0.224, 0.225]. The batch size for image dataset was set to 32. The pre trained model was then modified to have the final layer match the 38 classifications of the "*Plant Village*" plant species. Optimizer "*Adam*" was chosen with a loss function cross entropy was used with a fixed learning rate of 0.001. Each model being trained for 5 epochs due to computational resource limits. Choosing to keep the transformations of all models identical to ensure the experiment environment was uniform under all testing and training conditions. Once the model was trained and weights had updated the model was evaluated and processed *see figure 8.*

3.1.3 Performance Metrics

When a model is trained a set of performance metrics are produced, these are then used and evaluated against each other in order to compare model performance overall. Model accuracy and loss are calculated by the PyTorch [20] library on the training and testing dataset. Similarly the validation accuracy and loss are calculated using the validation dataset. The models accuracy, loss, validation accuracy, precision, recall, and F1_score will be calculated using the formulas in *figure 9* and *figure 1.* Furthermore, for the top 2 performing models a confusion matrix will be complied. Assisting in identifying the models sensitivity of each plant species within "*Plant Village*". Supporting visual graphs will also be constructed to be reference in the result discussion. Supporting figures such as graphing the accuracy and loss of the models alongside their training times.

Metrics	Formula	Description
	Classification Model [95,99,100]	
Accuracy	$\frac{True\ Positive+True\ Negative}{All\ Cases}$	Overall ability of a model to make the correct classification
Precision	$\frac{True\ Positive}{True\ Positive+False\ Positive}$	Ability of a classification model to make correct predictions within the positive class
Sensitivity (Recall)	$\frac{True\ Positive}{True\ Positive+False\ Negative}$	Ability to correctly identity positive labels
Specificity	$\frac{True\ Negative}{True\ Negative+False\ Positive}$	Ability to correctly identity negative labels
F-score	$\frac{2\times Precision\times Sensitivity}{Precision+Sensitivity}$	Harmonic mean of sensitivity and precision
Area Under the Curve (AUC) of the Receiver Operating Characteristic Curve	$\frac{1}{2}\ (Sensitivity + Specificity)$	Ability of a model to avoid misclassification

Figure 9: Key Performance Metrics [21]

	Positive	Negative
Positive	True Positive (TP)	False Negative (FN)
Negative	False Positive (FP)	True Negative (TN)

Table 1: Model Confusion Matrix

4 Model Result and Discussion

4.1 Model Result

The purpose of the discussion is to compare and analyse the model performance metrics on the "*Plant Village*" dataset. Using high performing CNN models backed by literature via transfer learning without layer freezing. The models VGG11 [8], GoogleNet [9], ResNet [10], DenseNet [11], MobileNet [12] and EfficientNet [13] were used in a model training pipeline which produced the performance metrics outline in *table 2*.

4.2 Training Accuracy and Loss

In terms of model accuracy the top performing models were EfficientNet with 99.01% accuracy, MobileNet with 98.52% and DenseNet with 97.93%. All of the models achieved an accuracy of 90% and above, with the lowest accuracy being VGG11 with 90.06% accuracy. The 5 models ordered by accuracy in *table 2* had accuracy's which deviated by 1.32% on average excluding VGG11. The last model being VGG11 had a 9.01% accuracy difference when compared to EfficientNet and a 7.69% difference when compared to GoogleNet which was the next highest performing model. The validation accuracy matched similarly to the training accuracy.

In regards to model loss, all of the models performed well with very low loss values. with the lowest

	EfficientNet	MobileNet	DenseNet	ResNet	GoogleNet	VGG11
Accuracy %	0.9901	0.9852	0.9793	0.9777	0.9769	0.9006
Loss	0.0355	0.0448	0.0454	0.0617	0.0619	0.3722
Validation Accuracy %	0.9901	0.9852	0.9793	0.9777	0.9769	0.9006
Validation Loss	0.0334	0.0495	0.0709	0.0816	0.0734	0.3477
Precision %	0.9887	0.9836	0.9767	0.9765	0.9720	0.8912
Recall %	0.9868	0.9830	0.9771	0.9725	0.9680	0.8791
F1_Score %	0.9871	0.9831	0.9763	0.9735	0.9689	0.8812
Average time per Epoch (s)	638.3	181.8	599.7	249.5	284.6	479.5
Total Training Time (s)	3,191.7	908.8	2,998.3	1,247.6	1,422.8	2,397.7

Table 2: Model Evaluation Results

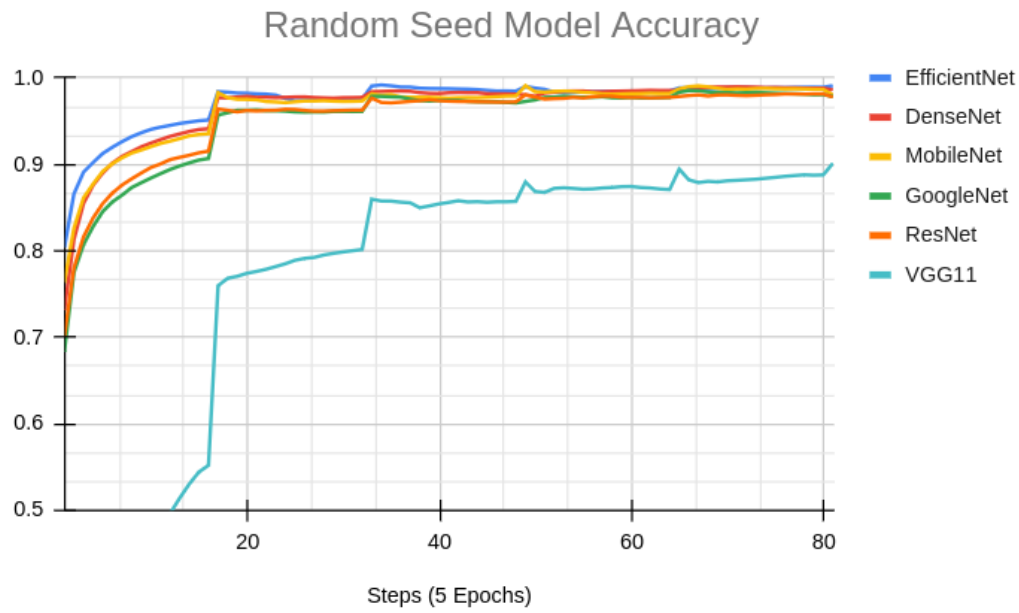


Figure 10: Model Accuracy over time

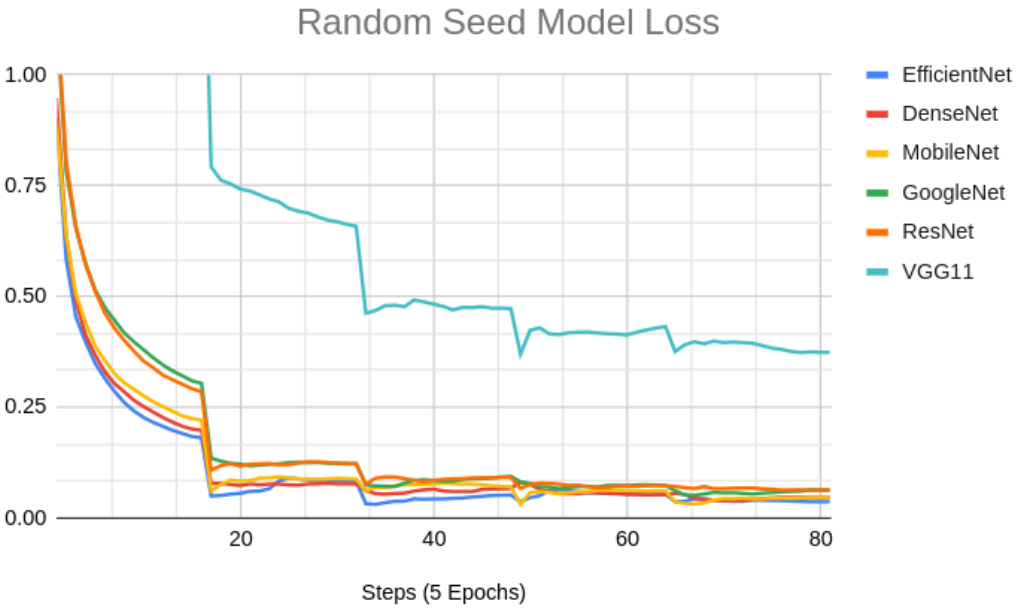


Figure 11: Model Loss over time

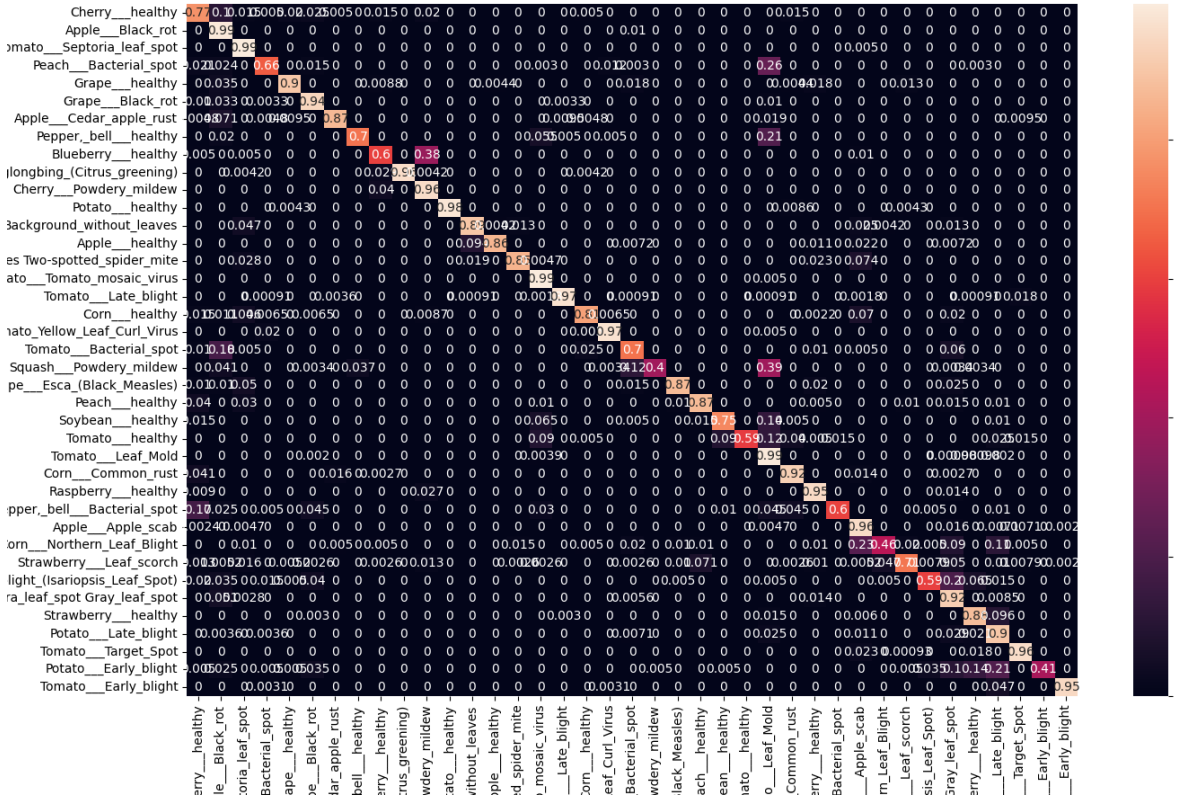


Figure 12: EfficientNet Seed1 Confusion Matrix

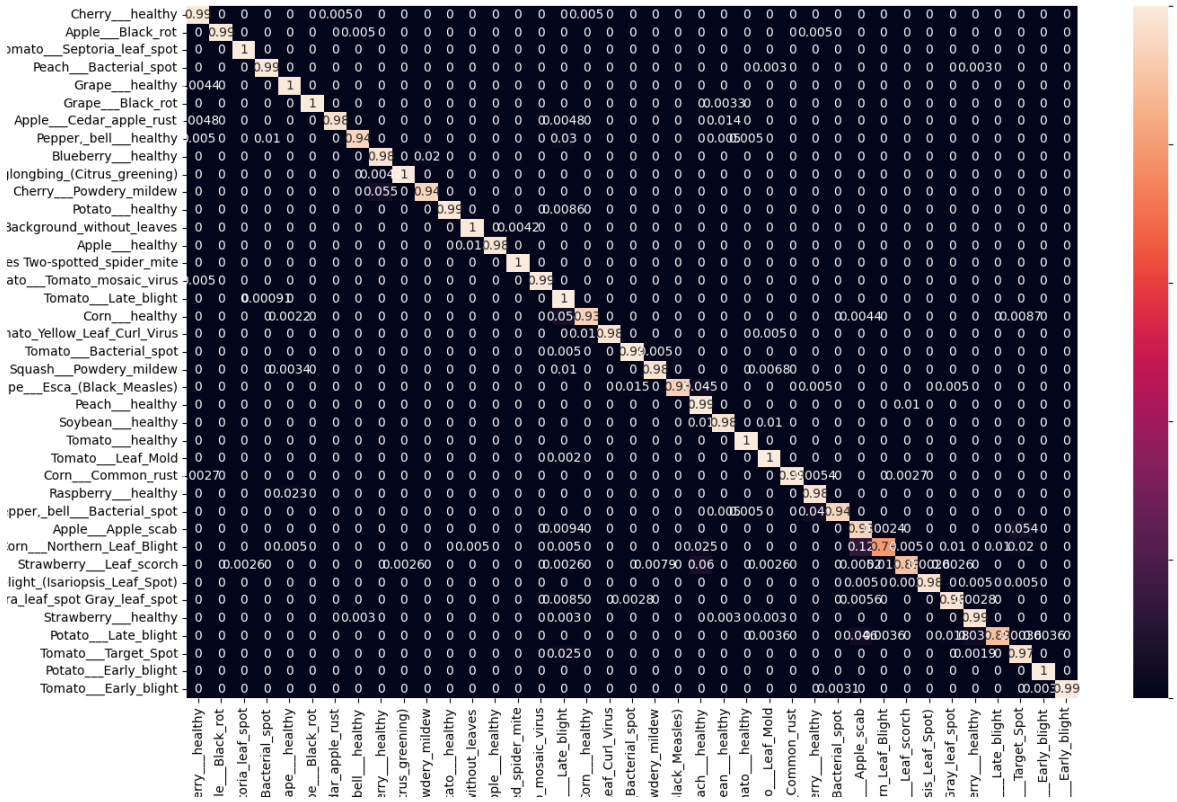


Figure 13: MobileNet Seed1 Confusion Matrix

loss value being 0.0355 and 0.3722 produced by EfficientNet and MobileNet. VGG11 performed the weakest similarly to its accuracy value, produced a loss value of 0.3722. The loss values of VGG11 compared to EfficientNet had a difference of 0.3367. VGG11 in both accuracy and loss performed the worst which was expected as the model was used as a baseline controlled model; This decision was made as the VGG11 model compare to the other models is a very old model architecture that still perform well, hence used as the bottom control boundary for testing. An average loss value disregarding VGG11 across the 5 models had a average loss value of 0.04986.

The models accuracy and loss values over each step for 5 epochs can be seen in *figure 10* and in *figure* 11.

Observing the accuracy and loss graphs produced, one can hypothesis accurately that given more training epochs the models would perform better having higher accuracy and lower loss values as the training time increased. However due to the issue of over fitting there would be a rate of diminishing return and the values would plateau with over fitting increasing.

The high accuracy of all the models and low loss metrics presented in *table 2* suggests that in our training pipeline and experiment environment we were able to practically leverage deep learning models for plant identification and classification. However there is still further analysis that was conducted to confirm this statement. A discussion of transfer learning, model validity and model architecture was explored further.

4.2.1 Training Discussion: Accuracy and Loss

Transfer learning granted the ability to train models on datasets that already had pre-determined weights. This technique presented by Tom Kocmi [7] was successful in contributing to the performance metrics of the models seen in *table 2*. This had direct implications on the model's training time and high accuracy as seen in *table 2* , with only having a total of 5 epochs per model an accuracy of 90.06% and above was achieved with EfficientNet having a 99.01%. This remarkable accuracy was achieved with a small amount of epochs, whilst enabling a speed up of total training times across all models. Models such as MobileNet were able to have a total training time of only 908.8 seconds and ResNet with a 1247.6 seconds.

The direct effect this has in real world applications is that it has sped up model training and evaluation pipelines across all research into deep learning models. If a model has to be re-tuned, dataset has to be increased and or hyper parameters changed this can all be done marginally faster then compared to models that don't utilise transfer learning in training. The reduction in total training time for a single model also leads to a direct reduction in computational resources needed.

Another contributing factor to the high accuracy and low loss performance metrics of the models was the high quality "*Plant Village*" dataset. The dataset included 54,303 healthy and unhealthy leaf images. In addition to the dataset using data augmentation to artificially inflate the total dataset size via, color sheering, angle transformation, rgb transformations, zoom transformations etc. The more data that a machine learning model has to train on, there is a direct impact on the increased accuracy in the feature map weightings. This sentiment was also echoed by researchers Jayme Barbedo in the in the "*Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification*" [22], where the author similarly concluded that the quality and diversity of the dataset has a direct impact on model performance metrics. Having a large high quality dataset allows your model's architecture to be utilised to its fullest extent, not being bottle necked by the data being passed through.

4.3 Model Validity

4.3.1 Validation Performance Metrics

When evaluating the accuracy and loss values produced by a machine learning model, a multitude of other performance metrics must be analysed to ensure model validity.

A validation dataset was used to compare and validate the performance of the models on images that the training dataset did not have. This produced the validation accuracy and loss performance metrics in *table 2*. The validation accuracy across all models followed identically to the training

accuracy. However the validation loss values across all models were higher. An average loss value disregarding VGG11 for all models was 0.06176. Showing the average validation dataset loss being higher then the training dataset loss, an average increase of 0.0119. The leading cause for this is that the validation dataset is only 10% of the total "*Plant Village*" dataset whereas the training dataset was comprised of 80% of the total images. So a slight decrease in validation accuracy is to be expected. The validation dataset is a useful metric to see if the model is hyper sensitive to the data and does not adapt well, in the case of the models in *table 2* we can see that this is not the case and the models tested have a balance of dataset sensitivity and adaptability.

The precision and recall of all models were above 0.8791. The high precision indicating that all models have a high positive prediction ratios. Following the high recall values indicating that the model is correctly identifying true positives from all positive samples within the entire datasets. These two evaluation metrics indicate that all the models trained on the "*Plant Village*" dataset were able to not only identify positive predictions, but also true positives among the large dataset.

The models with the highest precision being EfficientNet and MobileNet with precision scores of 0.9887 and 0.9836 reflect the confusion matrix's which can be seen in in *figure* 12 and 13. The precision and recall of the models suggest that they not only correct classification labels, but that the predict the true positive labels in the dataset. As the models are pre trained this means that the models can easily adapt to any dataset which is useful when leveraging the models for other plant datasets.

4.3.2 EfficientNet and MobileNet Confusion Matrix

Within the confusion matrix of MobileNet being the second highest accuracy model *see figure 13* the category with the least true positives was the "*Corn Northern Leaf Blight*" with a value of 0.7. However the inaccuracies are very sparse and generalised meaning that MobileNet did not have bias when struggling a single particular plant identification. However in comparison to EfficientNet the best performing model in terms of accuracy and loss, the confusion matrix yields different results compared to MobileNet. The confusion matrix from EfficientNet *see figure 12* shows the model predicting a higher amount of false predictions in *"squash powdery mildew"*, *"tomato healthy"*, *"corn northern leaf blight"* and *"potato early blight"*. This was expected due to EfficientNet being a significantly more complex model when compared to MobileNet. With further testing, granted more training epochs were given to both models the EfficientNet model may eventually produce a more generalised true predicted accuracy when compared to MobileNet.

4.3.3 Dataset Seed Randomization

A technique that aided in model and training validity which is later reflected in the results of the models in *table 2* was the randomization of the dataset. During the training phase of every model the "*Plant Village*" dataset was randomized. The percentage split in training, testing and validation datasets stayed the same however the images in each classification were reshuffled on each model. This was done in an attempt to ensure that each model had a unique dataset image seed. This is important as it did not allow the grouping of similar images in one image batch which would have a bias on model accuracy, as the model would have a higher chance of spotting similar features within a single image batch. All of the model inputs were made the same with a input size of 224x224 with a learning rate of 0.001 for testing environment uniformity.

4.4 Discussion on Performance Metrics and Model Architecture

Leveraging deep learning models on the "*Plant Village*" dataset, we were able to output a multitude of performance metrics including validation metrics which was used show their accuracy. However there is a discussion as to why some models performed better then others on the same dataset, this is due to the model architectural design.

4.4.1 VGG11 and GoogleNet Models

The VGG11 model performed the worst in almost all performance metrics when compared to the other CNN models, however still achieving an accuracy of 0.9006% in both training and validation datasets; Achieving an average loss in both training and validation datasets of 0.35995. Even though the VGG11 model had the worst performance metrics out of the 5 other CNN models in all metrics the data would still deem this model leverage-able in a practical object classification and identification task.

In *figure 10* and in *figure 11* suggest that given more time to train for further epochs the model may perform significantly better. The 14 year old CNN architecture still performing relatively mediocre in all performance metrics compared to newer models suggest the fundamental approach to CNN's that Karen Simonyan et al took, was a step in the right direction in terms of approaching deep learning models. The research conducted by Karen Simonyn et al, and in *table 2* proposes that increasing model depth has a strong correlating factor to model accuracy; Many of the other modern CNN models citing VGG11 as inspiration for their architecture design. [8]

The VGG11 CNN model proposed the vector of creating deeper models as a way of achieving higher model performance. On contrary the GoogleNet model proposed a method of not only increasing depth but also increasing the width of the network *see figure 2b*. Its clustering of layers model architecture shows a clear advantage over VGG11 which just optimized for model depth. In bench-marking, GoogleNet was able to achieve a 7.63% increase in accuracy and a 0.3103 value reduction in loss. Alongside significant improvements in precision recall and "*F1 Scores*" *see figure table 2*. However the largest margin of change was roughly a 51% decrease in total training time. Following the other 6 models used in testing, GoogleNet achieved the second shortest training time for 5 epochs of 1422.8 seconds. Clearly illustrating that the unique architecture design of GoogleNet not only decreased training time but increased accuracy, which is beneficial for plant leveraging machine learning models on plant based classification tasks.

4.4.2 DenseNet and ResNet Models

In testing the model results of DenseNet, Resnet and GoogleNet performed very similarly, however DenseNet and ResNet outperformed Googlenet by slight margins *see table 2*. DenseNet and ResNet have similar architectures, very deep convoluting layers that have a method of interacting with previous input layers. ResNet's implementation of residual blocks and DenseNet implementation of a feed forward structure. With a comparison of these two models to VGG11, this clearly shows that the feed forward style of the architectures yields dominant performance in terms of model accuracy.

However a factor that is not reflected in the results presented in *table 2* is the ability for DenseNet and ResNet to train for long epochs achieving higher accuracy without over fitting. The latter which causes a model to memorize the training data and not generalise. The total training epochs was limited due to computational restrictions, however given future training if these training epochs was increased from 5 to 10 cycles we may see results that yield DenseNet and ResNet to be the most dominant models in regards to performance. The increased training cycles may cause the other models to over-fit where DenseNet and ResNet would not; And continue to generalise whilst accumulating higher accuracy. However, further training and validation need to be conducted in order to validate this theory.

4.4.3 MobileNet and EfficientNet Models

The MobileNet and EfficientNet models are the top 2 performing models in terms of accuracy and loss *see table 2*. With EfficientNet being the top performing model. However the two models had significantly different total training times with MobileNet being the fastest with 908.8 seconds and EfficientNet with the longest 3191.7 seconds. The MobileNet architecture using depth wise separable convolutions presents a streamline model architecture for low resource but intensive deep neural networks, reflected in the evaluation metrics seen in *see table 2* and its confusion matrix *see figure 13*. The unique architecture of MobileNet reflects in its ability to be highly accurate and have very low loss values within early epochs which is a benefit to model training *see figure 10* and *see figure 10*. This allows a model that can be trained extremely quickly and accurately with limited epochs.

When researchers proposed the EfficientNet model they clearly identified that the model took a unique approach at model scaling. They proposed a compound scaling coefficient that scaled in all dimension that a model could *see figure 6*. This is reflected in the models results out performing all of the other models in accuracy and loss values alongside the model evaluation metrics. However this also reflects an extremely long total epoch training time of 3,191.7 seconds being the longest out of all the models *see table 2*.

In one area where EfficientNet performs worse then MobileNet is within its ability to predict correct true positive labels *see figure 13* and *figure 12*. A comparison of the EfficientNet and MobileNet confusion matrices shows that within 5 epoch cycle, the MobileNet model has better generalization over all plant classifications when compared to EfficientNet. However this may be a limit due to the limited training epoch of 5 which was used to train all of the models. Hypothesising that with further training epochs EfficientNet may generalize better then MobileNet due to Efficient-Net's complex architecture and higher validation accuracy. However further tests and training would need to be conducted again to prove this hypothesis.

4.4.4 Vanishing Gradient Problem

A important factor when discussion the performance of machine learning models is to identify their performance in combating the vanishing gradient problem. The vanishing gradient problem is caused by the value product of derivatives decreasing so far until the point of the partial derivative of the loss function used approaches zero. This results in the next partial derivatives to vanish [23].

DenseNet and ResNet may see slower weight vanishing during back-propagation compared to the other models due to their feed forward architectural design. Suggesting that if all models had increased training epochs DenseNet and ResNet may be the only models that can train for longer while achieving higher accuracy without succumbing to the vanishing gradient problem. The limited training epochs due to computational restrictions lead to a inaccurate illustration of DenseNet and ResNet model's full capacity. Alluding to these two models not being as effective in smaller training constraints when compared to MobileNet and Efficient. However further training and validation need to be conducted in order to validate this theory.

5 Conclusion

Through the experiment conducted we were able to "*Leverage Deep Learning for Advanced Plant Identification and Health Assessment: An Exploration of Techniques, Datasets, and Practical Applications*". The technique of data augmentation allowed for a very diverse and high quality dataset via image transformations. Transfer Learning enabled the ability to train pre trained models on a existing dataset, only modifying the final layers for fine tuning without layer freezing. The training pipeline described enabled a experiment that concluded a multitude of performance metrics that described model performance *see table 2* on the "*Plant Village*" Dataset. And lastly the performance metrics were evaluated and explained.

5.1 Future Contributions

This research has outline that there are useful techniques that can be employed when training and evaluating deep learning models that can make it practically useful for real world applications. Data augmentation allowed for a limited dataset to be artificially inflated via transformations of the pre existing images. Without this technique being utilised it can be argued that the models would not have performed as they did, as it can be observed that if the models have more training data to work with they would be able achieve higher accuracy. However for future contributions this should be tested as some models may out perform other models if the dataset is limited, a rigorous training pipeline should be developed to train all of the 6 models presented with a gradual increase in total images. This is an important vector that should be looked at as in a real life scenario even with data augmentation the dataset may still be limited, henceforth skewing the best performing model in that scenario.

Another technique that was employed during the experiment was the use of transfer learning. Practically transfer learning has a direct impact in reducing total resource constraints as the models are pre trained with the layer weights already pre configured. Within transfer learning a model can have its weights "*frozen*" so that only the final set of layers can have their weights altered. In our experiment conducted this was not chosen due to justification being that the model would be able to be more sensitive to the dataset. However these claims were not backed by empirical evidence. Hence, given future works one should test this theory by conducting a training pipeline where there is a comparison of models trained with transfer learning layer freezing, and one without.

During the training pipeline conducted for the experiment the dataset was randomised for the training, testing and validation datasets. However the models were only trained once for 5 epochs and then the performance metrics were then evaluated. Given further contributions to the literature it would be argued that training a single model with random seeds multiple times, then averaging those results would provide better experiment results due to the chance of lucky dataset ordering and seeding would have less of an impact.

Comparing all of the 6 models used in the training pipeline for the "*Plant Village*" dataset the *table 2* figure was produced. Due to the limited computational resources used for the experiment only 10 epochs were allocated per model being trained. If future direction was provided to this research an increase of epochs may yield different results for the final performance evaluation for each of the models. Alongside this the performance metrics would be increased to evaluate a "*ROC Curve*" alongside a "*F-Score*" Which was not evaluated during the experiment.

5.2 Model Performance on Plant Village

Given the model results presented in *table 2* it can be argued that the CNN models performed extremely well in a practical plant based classification scenario. The extremely high accuracy values across the board alongside low loss values in comparison to the VGG11 model suggest that the current machine learning literature is progressing at an exponential rate. The highest accuracy models seen in *table 2* being MobileNet and EfficientNet applied in a real world classification scenario would perform well according to the experiment conducted. The EfficientNet and MobileNet models could greatly aid the automation process within vertical farms. However the MobileNet model may be more desirable in this a practical scenario due to the MobileNet model being able to very efficiently trained as can be seen in *table 2*. The benefits of having fast model training time is that if new data needs to be added into the model and or the hyper-parameters need to be trained it can be done very efficiently.

5.3 Importance of Model Architecture

The 6 models used on the "*Plant Village*" dataset had identical training following the training pipeline identified in *see figure 8* however, they all produced different performance metrics. This may be due to each model's architecture taking a unique approach in scaling accuracy and performance. ResNet and DenseNet putting emphasis on model depth, GoogleNet redefining its architecture with its inception module dimension reduction and MobileNet using depth wise separable convolutions.

All of these are linear when compared to the highest performing model EfficientNet which scales in all directions depending on the scaling coefficient. The ability for EfficientNet to adapt in all dimensions and of the dataset has produced the highest performing model when used as seen in *table 2*. Given further contributions to this research, if more training epochs were given to all models it would be hard to assume if EfficientNet would compare better in terms of combating the vanishing gradient problem.

Hence this research suggesting a need for further work, the need to derive a model that not only scales effectively in all directions using the scaling coefficient, but also a model that can scale extremely deeply while combating the vanishing gradient problem via a residual block or skip connection that is found within DenseNet and ResNet.

5.4 Closing Thoughts

Due to the high model performance of EfficientNet and MobileNet as illustrated in the experiment and supporting discussion. It is evident that cutting edge convolutional neural networks can be used to accurately and effectively identify different plant species of both healthy and diseased varieties. Its implementation into the agricultural industry could have revolutionary change in increasing the automation within the sector. Allowing for significant improvements in health identification and monitoring systems. Given further contributions to this research, a new suggested model architecture has been proposed, one that takes architectural designs from both EfficientNet and DenseNet models. This new model architecture may produce the highest functioning CNN computer vision model given the right research protocols and training pipelines. Not only combating the vanishing gradient problem, but also exponentially increasing the accuracy and lowering loss values of the model as a whole. Changing the notions of how to scale a convolutional neural network in all machine learning literature.

References

[1] "Gdp share of agriculture by country, around the world," 2022.

[2] A. M. Beacham, L. H. Vickers, and J. M. Monaghan, "Vertical farming: A summary of approaches to growing skywards," *J. Hort. Sci. Biotechnol.*, vol. 94, no. 3, pp. 277–283, 2019.

[3] K. Jha, A. Doshi, P. Patel, and M. Shah, "A comprehensive review on automation in agriculture using artificial intelligence," *Artificial Intelligence in Agriculture*, vol. 2, pp. 1–12, 06 2019.

[4] D. P. Hughes and M. Salathe, "An open access repository of images on plant health to enable the development of mobile disease diagnostics," 2016.

[5] A. Jain, H. Patel, L. Nagalapatti, N. Gupta, S. Mehta, S. Guttula, S. Mujumdar, S. Afzal, R. Sharma Mittal, and V. Munigala, "Overview and importance of data quality for machine learning tasks," *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, 08 2020.

[6] A. Mikołajczyk and M. Grochowski, "Data augmentation for improving deep learning in image classification problem," in *2018 International Interdisciplinary PhD Workshop (IIPhDW)*, pp. 117–122, 2018.

[7] R. Tombe and S. Viriri, "Effective processing of convolutional neural networks for computer vision: A tutorial and survey," *IETE Technical Review*, vol. 39, pp. 49–62, 09 2020.

[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.

[9] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," 2014.

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[11] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," 2018.

[12] 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," 2017.

[13] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," 05 2019.

[14] M. Mishra, "Convolutional neural networks, explained," 08 2020.

[15] C. Keysers and V. Gazzola, "Hebbian learning and predictive mirror neurons for actions, sensations and emotions," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 369, p. 20130175, 06 2014.

[16] S. Sahoo, "Residual blocks — building blocks of resnet," 11 2018.

[17] A. Khosla, N. Jayadevaprakash, B. Yao, and L. Fei-Fei, "Novel dataset for fine-grained image categorization," in *First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*, (Colorado Springs, CO), June 2011.

[18] D. Schlegel, "Deep machine learning on gpus," 2015.

[19] Google, "Google colaboratory," 2019.

[20] "Pytorch," 2023.

[21] A. Lee Kf, W.-S. Gan, and G. Christopoulos, "Biomarker-informed machine learning model of cognitive fatigue from a heart rate response perspective," *Sensors*, vol. 21, p. 3843, 06 2021.

[22] J. G. A. Barbedo, "Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification," *Computers and Electronics in Agriculture*, vol. 153, pp. 46–53, 2018.

[23] H. H. Tan and K. H. Lim, "Vanishing gradient mitigation with deep learning neural network optimization," in *2019 7th International Conference on Smart Computing Communications (ICSCC)*, pp. 1–4, 2019.