

Crop disease classification using Deep Learning

by

Deepali Yadav

In partial fulfilment of the
requirements for the degree of
MSc
in
Artificial Intelligence and Applications



Department of
Computer and Information Sciences

12 August 2024

Abstract

This paper aims to present extensive approach to crop disease classification using deep learning techniques. Correct and timely identification of crop disease is required for ensuring agricultural productivity. Traditional methods, manual inspection are time consuming and sensitive to errors. In this study, the performance of three state of the art convolutional neural network architectures InceptionV3, ResNet, and Xception was evaluated on a dataset comprising images of crops affected by various diseases. The models were trained and tested to classify 17 different crop-disease combinations. Evaluation metrics used are accuracy, precision, recall, and F1-score are used to gauge the performance of each model. Additionally, employed early stopping and dropout regularization techniques to mitigate overfitting and enhance generalization.

The results indicate that the Xception model achieved the higher overall accuracy of 97.05%, outperforming both InceptionV3 and ResNet50. Early stopping was particularly beneficial for the InceptionV3 and Xception models, helping them avoid overfitting and improve generalization. However, early stopping slightly reduced the accuracy of the ResNet50 model, suggesting potential underfitting. Dropout regularization proved effective across all models, maintaining high training accuracy and stabilizing validation performance.

The study also highlights challenges such as class imbalance and specific class misclassification, which were more pronounced in certain classes like "Potato Healthy" and "Rice Brown Spot." Future work will focus on addressing these challenges through advanced techniques such as learning rate optimization, class balancing methods, and further hyperparameter tuning.

This comprehensive evaluation underscores the robustness of the Xception architecture in agricultural image classification tasks, while also identifying areas for improvement and further research.

Declaration

This dissertation is submitted in part fulfilment of the requirements for the degree of MSc of the University of Strathclyde.

I declare that this dissertation embodies the results of my own work and that it has been composed by myself. Following normal academic conventions, I have made due acknowledgement to the work of others.

I declare that I have sought, and received, ethics approval via the Departmental Ethics Committee as appropriate to my research.

I give permission to the University of Strathclyde, Department of Computer and Information Sciences, to provide copies of the dissertation, at cost, to those who may in the future request a copy of the dissertation for private study or research.

I give permission to the University of Strathclyde, Department of Computer and Information Sciences, to place a copy of the dissertation in a publicly available archive.

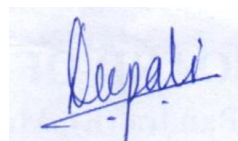
(please tick) Yes ☒ No ☐

I declare that the word count for this dissertation (excluding title page, declaration, abstract, acknowledgements, table of contents, list of illustrations, references and appendices is 9139 words.

I confirm that I wish this to be assessed as a Type 1 2 3 4 5

Dissertation (please circle)

Signature:



Date: 12 Aug. 24

Acknowledgements

I would thank my supervisor Joseph El-Gemayel for his invaluable guidance, constructive feedback, and continuous support throughout the course of this study.

I'm also grateful to my colleagues and peers for their insightful discussions, suggestions, and encouragement. Their collaboration and shared knowledge significantly enhanced the quality of this research.

Lastly, I would like to acknowledge the dedication and support of my family, who provided me the emotional and moral support needed during this course.

Contents

1	Introduction	1
2	Background & Literature Review	3
2.1	Residual Neural Network (ResNet) for crop disease classification	3
2.2	Hybrid Convolutional Encoder Network	3
2.3	CNN models for crop Tomato disease classification	4
2.4	Impact of Dataset Size and variety on CNN performance	4
2.5	Public Dataset and CNN architecture comparison	5
2.6	Deep learning for Grapevine Yellow disease Detection	6
2.7	Transfer learning with Inception and Xception	7
2.8	Impact of image preprocessing techniques on CNN performance	7
2.9	Impact of Regularization	8
2.10	Major Findings.....	9
3	Methodology.....	10
3.1	Dataset	10
3.2	Model Selection.....	11
3.3	Transfer Learning	12
3.4	Image size.....	12
3.5	Preprocessing Function	12
3.6	Data Augmentation.....	13
3.7	Data Split	13
3.8	Model Implementation	14
3.9	Model Training	14
3.9.1	Activation Function	14
3.9.2	Batch size.....	14
3.9.3	Optimizer	15
3.9.4	Learning Rate	15
3.9.5	Regularization.....	15
3.9.6	Hardware Setup.....	15
3.10	Models	16
3.10.1	Inception	16
3.10.2	Inception V3 architecture	17
3.10.3	Xception.....	17
3.10.4	Xception architecture	19
3.10.5	ResNet.....	19
3.10.6	Resnet architecture.....	20
4	Analysis	21
4.1	Evaluation Metrics	21
4.1.1	Confusion Matrix.....	21
4.1.2	Classification Report.....	22
4.2	Inception Output Analysis	23
4.3	ResNet Output Analysis.....	25

4.4	Xception Output Analysis.....	27
4.5	Confusion Points	29
Conclusion & Recommendations		31

List of figures

Figure 1 :	Example of leaf images from the dataset.....	11
Figure 2:	Inception module, naïve version [17]	16
Figure 3:	Inception module with dimension reductions [17].....	17
Figure 4:	Depth Wise Convolution [20].....	18
Figure 5:	Point Wise Convolution [20]	18
Figure 6:	Residual learning: a building block. [21].....	20
Figure 7 :	Xception Confusion Matrix	22
Figure 8 :	Inception Training and Validation Accuracy	23
Figure 9 :	Inception Training and Validation Loss	24
Figure 10 :	ResNet Training and Validation Accuracy	25
Figure 11 :	ResNet Training and Validation Loss	26
Figure 12 :	Xception Training and Validation Accuracy	27
Figure 13 :	Xception Training and Validation Loss	28

List of Tables

Table 1:	Model Accuracy comparison.....	29
----------	--------------------------------	----

1 Introduction

One of the basic needs of Humans is food and agriculture is the primary source of production. Agriculture remains one of the main and important occupation around the world as well as significant source of employment in rural areas. Most of the food grown worldwide for consumption are grown by small scale farmers in Asia and Africa. Forty- four percent of the earths land is used for agriculture using 48 million square kilometres, which is around five times the size of USA. [1] With expand in use of chemicals to increase the production of the crops have been increased it has some drawbacks as well which affects the soil in long run, water contamination. It also effects on human health as person exposed to pesticide while sprinkling it in the field and consumption of crop having residues of pesticides. Other major threat to agriculture is crop diseases, there are various causes for crop disease one of which is by pathogens for e.g. fungi, bacteria, virus etc. and occurs due to the contaminated water and environment. Now in this century with advance technology it is possible to detect the disease effecting the crop production beforehand which helps in making informed decision, increase in crop production with correct treatment. Early intervention can help in practicing sustainable production making crop production more efficient, and environmentally friendly.

Deep learning has a considerable influence in the field of Artificial Intelligence allowing computers to do tasks such as classification, object detection etc. In Deep learning artificial neural network is inspired by function of the human brain where nerves and its deep connection used for transferring signals important for the human body to perform. Deep Neural Network model further extends Artificial Neural Network by including hidden layers between the input and output layer and learn the various feature of the data enabling the model to perform well. With further advancement more and more models and architecture have been explored and one of the famous is Convolutional Neural Network.

Recent developments in computational systems, particularly with Graphical Processing Units (GPUs), have led to significant growth in Machine Learning and the

emergence of Deep Learning. This involves using complex neural network architectures with many processing layers, which have transformed various sectors, including image and voice recognition.

Researchers started exploring Deep learning in the field of agriculture in the 2010s focusing on the production prediction, soil analysis, weather prediction and forecasting. With increase in the ability of image classification researchers began exploring its application in plant types and diseases using deep learning. Therefore, in this paper [2] using a plant disease dataset various models and architecture are trained to accurately identify and classify crop disease. This can further help in disease management as technology advances models can be integrated to mobile applications and used by the farmers to prevent crop from getting destroyed by diseases allowing for timely treatment.[2] The dataset used is already has pre classified images, the output of the analysis should be how well the model generalizes on unseen data that is kept aside to be tested after the model is trained. The method used are image classification using pre trained models like Inception, Xception and ResNet50. Both training and testing is done on Kaggle notebook, a cloud based computational environment having computer resources such as GPU required for large computations, available for 20 hours free of charge for a week.[3]

The paper is further divided into Sections: Section 2 provides details of existing work done by various researchers and what models worked for them and about the dataset used. Section 3 provides details of the dataset and the methods used for creating the model, implementation details and experiments done Section 4 . describes analysis of the result got from the experiment, confusion points Section 5 concludes the paper with the findings and provides recommendations about improvements and alternate experimentation approaches.

2 Background & Literature Review

2.1 Residual Neural Network (ResNet) for crop disease classification

Much research has shown the use of neural networks, for crop disease classification. The use of pre-trained models has proved to be a powerful technique. The authors [4] used Residual Neural Network (ResNet) architecture, and discusses the application of convolutional neural networks, in plant disease diagnosis and image recognition using algorithms for multi-disease identification in plants deployed on mobile applications making them accessible for farmers for real time disease identification. The use of mobile capture device is relevant as they are widely available. This approach allows quicker response to disease, potentially reducing the impact on crop yield. It highlights the high accuracy achieved in disease detection under real field condition, with images taken in uncontrolled environments, the algorithms deployed on mobile applications for real time disease identification.[4] The algorithm was validated on wheat images fed to a real mobile application under real field condition as well and the observed results shows an increased accuracy from 0.78 to 0.84 causing low specificities.[4] The one of the challenges faced was the variability in image quality and environmental conditions for disease classification.

2.2 Hybrid Convolutional Encoder Network

Convolutional encoder network, combination of CNN and autoencoders by [5] is a hybridized deep learning neural network uses only the encoding portion of the autoencoders to extract relevant characteristics from images. CNN is the basics of the model consisting of layers that transform input data through series of convolutions, pooling and fully connected layers. Autoencoder is used to learn efficient encodings of the input images. The encoding part generates high dimensional feature vectors, which are aggregated to enhance feature extraction process. The dataset is derived from PlantVillage and for this research the dataset was reduced to 900 images focusing on 3 crops potato, tomato and maize out of which 600 served as the training set and the remaining 300 as the test set. Different filter size was tested. The result showed varying accuracy depending on the filter size and the number of epochs

utilized achieving 97.50% accuracy for a 2x2 Convolution filter size and 100% accuracy for a filter size of 3x3 . It also mentions challenges related to misclassification as a result of the small size of the dataset, large training time due to lack of availability of GPU. [5] For further exploration it suggests involving using hyperparameters such as dropout and regularization to improve performance.

2.3 CNN models for crop Tomato disease classification

Tomato crops are significantly impacted by various diseases so, For a particular class 'Tomato' from the PlantVillage dataset use 9 different disease and a healthy class is used for classification utilizing pre-trained models deep learning based CNN models.[6] The models were trained using transfer learning where only the top-level classifiers are trained initial layers of the base CNN model are frozen leveraging the knowledge gained from the ImageNet dataset and setting the learning rate to a very small value by fine tuning, unfreezing the CNN layers for further training allowing to model to increase model's performance. A 5 fold cross validation is applied during training process to evaluate the performance across different subsets of the dataset. The application of various optimization algorithms, including SGD, RMSprop, and Adam, to enhance the training process of the CNN models. The model is assessed based on the classification accuracy, with the DenseNet model using the RMSprop algorithm achieved an accuracy of 98.63% .It concluded that deep learning-based CNN architecture specially ResNet, Xception and DenseNet are effective for classifying.[6]

2.4 Impact of Dataset Size and variety on CNN performance

Dataset size and variety affects deep learning effectiveness. In a study by [7] used a dataset having images of 12 species with different characteristics, such as number of samples, disease, and conditions. The performance of the CNNs varied depending on the classes considered, the similarity between the training and test datasets, and the characteristics and unpredictability of backgrounds of the image. In some cases, removing the image backgrounds improved classification accuracy. In other cases, background deletion had a negative impact, particularly when the symptoms of

different disease were very similar, and the background had been used by the CNN as discriminating factor. While, as the number of classes increases, the likelihood of misclassification also rises due to a higher probability of diseases with similar symptoms.[7] If the training dataset is not varying enough to capture the full range of variation and conditions found in the field, The CNN's ability to generalize and correctly classify new, unseen images is compromised. In summary, large and more varied datasets are essential for training CNNs to achieve higher accuracy, the use of restricted image datasets for training results in high error rates and lack of robustness when the models are applied to images that differ significantly from those in the training set.[7]

2.5 Public Dataset and CNN architecture comparison

The authors [8] used public dataset VillagePlant comprising of 54,306 images includes 14 different crops species and 26 diseases. The convolutional neural network was employed by standardizing hyper parameters across all experiments. Experimented with different architecture AlexNet and GoogLeNet and training mechanism such as transfer learning and training from scratch, as well as various image processing techniques such as colour, grayscale, and segmented images dataset with the train-test set distribution such as 80-20 where 80% for training and 20% for testing, 60-40, 50-50, 40-60, 20-80.[8] In all this configuration, transfer learning yields better results and GoogLeNet performs better than AlexNet overall accuracy for AlexNet is 85.53% and GoogLeNet is 99.34%.[8] The model's performance was robust across different training and testing set splits, indicating it was not overfitting to the data. All the configurations ran for a total of 30 iterations each. The training utilized Stochastic Gradient Descent and a learning rate set to 0.005. However, when tested on images obtained from online sources that were not part of the controlled dataset, the model's accuracy dropped to about 31%, suggesting model may not generalize well to real world conditions.[8]

Authors using an open database of 87,848 images consisting of laboratory conditions and real field conditions with 25 plant species in 58 classes of healthy and diseased

combination is used for the study.[9] The architecture used was AlexNet, AlexNetOWTBn, GoogLeNet, Overfeat, VGG. The training was conducted on an NVIDIA GTX1080 GPU in Linux environment.[9] VGG convolutional neural network achieved an accuracy of 99.53% on 17,543 unseen images.[9] It highlights the importance of large database and its contribution in achieving the high success rate however, it sheds light on how the testing dataset used was part of the same dataset that is used for training.[9] In the literature report by [8] reported a reduction in model performance when the testing set is not from the same dataset as training. The trained models required minimal computational power for image classification, taking 2 milliseconds on a single GPU making it feasible to integrate the models into mobile applications. [9]

2.6 Deep learning for Grapevine Yellow disease Detection

A study by [10] uses a convolutional neural network for detection of grapevine yellow in red grape vine using colour images of leaf clippings. Grape vine is a critical disease affecting grape vines productions. It utilizes 2 sets of data for training. The first dataset involved surveying, sampling, photographing and diagnosing grapevine and other is VillagePlant dataset to enhance the training process as it provides large volume of data. [10] The architecture ResNet101 overpowered all the other architecture such as AlexNet, GoogleNet, InceptionV3, ResNet50 and SqueezeNet for this application. However, ResNet50 being the second-best option the analysis doesn't show any significant difference in performance metrics hence it is also a suitable choice due to its lower complexity and training cost.[10] They conducted three experiments 1. Detection of Grapevine by computers for which in conclusion architecture ResNet101 had the highest accuracy 2. Comparison of Deep learning model to one not trained with deep learning. The deep learning model AlexNet using a pretrained CNN trained on ImageNet. The image is not fully prepared and collect the activation of the convolutional part, later the collected activation is categorized by a support vector machine for final prediction[10] 3. A test is conducted where human participants are presented with at least 100 images for each leaf to identify if the leaf has grapevine

yellow or not. In conclusion, Deep learning model has 35.97% and 22.88% predictive value (PPV) from manual inspection, than SVM and humans respectively.[10]

2.7 Transfer learning with Inception and Xception

Authors presents research paper on transfer learning-based plant disease detection using 2 pre-trained architectures, InceptionV3 and Xception model. [11] This architecture used is because of the small size of the model and less computational requirements. The dataset used was VillagePlant containing 70,285 training images 17,562 validation images and the accuracy achieved of 97.5 % by Xception outperforming the inception model. [11] The Xception architecture consisted of 36 convolutional layers organized into 14 modules with model size of 88 MB and approximately 22.9 million parameters This paper aims to explore pre trained model having highest accuracy. [11] A study by [12] discuss the performance of the Xception model from the scratch by comparing it to the pre trained version of the model. The dataset used is image of seeds consisting of 15 classes and total of 3018 images. The pre-trained model outperformed the model that is trained from scratch achieving a perfect accuracy on both the validation and test sets.[12] The pre trained model also had lower loss value during training indicating it converged quickly. The research provides comparative analysis between model trained from scratch with pre trained Xception model.[12]

2.8 Impact of image preprocessing techniques on CNN performance

Some of the deep learning model showed results in crop disease classification are DenseNet121, VGG16, InceptionV4, ResNet. Data processing enhances the accuracy of crop disease classification model. Before the model is trained it is necessary to increase the quality of the image being fed to the model by using various techniques that can significantly affect the performance .The researcher uses PlantDoc dataset for the experiments to use techniques like auto- orientation, resizing, grayscale, contrast adjustment, object isolation, conversion, static crop and tiling total of 7 image preprocessing techniques before applying to the training. [13] It is evaluated on 2 of the algorithms YOLOv5 and Faster-RCNN Image preprocessing

techniques resizing, contrast adjustments improved the accuracy and enhance mean average precision by upto 13 %. [13] The paper implies that the choice of preprocessing techniques can influence training and inference times. [13]

2.9 Impact of Regularization

One of the most important challenges often face in deep learning is overfitting, when the model over studies the training data most of the time when the dataset is not sufficient. The aim of the study done by [14] is to explore the impact of 3 popular regularization techniques i.e. data augmentation, drop out, weight decay. Exploring the combination of these methods in the models and observe the insights.[14] The dataset used for this experiment is CIFAR-10 and flower dataset. CIFAR has 60000 colour images while the flower has only 3640 images very vast difference between both the dataset. Convolutional neural network model with 3 layers and fully connected layers with dropout layers is designed to extract features. The convolutional layer is connected to ReLU activation function. Authors investigates on

1. Data Augmentation involving increasing the size of the dataset by applying conversions to the images.[14]
2. Weight decay method to penalize the larger weights.[14]
3. Dropout is a technique where randomly neurons are set to zero during training ensuring model does not rely on a particular neuron.[14]

By combining 2 3 methods to determine if the model's performance better instead of single method. Out of the 3 Dropout outperformed the weight decay and data augmentation in reducing the overfitting on both the dataset. [14] As per in combination dropout with data augmentation performs better than the compared combinations as augmentation with weight decay and weight decay with drop out. [14] Whereas when all the 3 methods are applied together than it outperforms the combination of 2 methods. [14] The study provides that the combination of all the 3 methods results in slower convergence while for weight decay it achieves convergence quicker. [14] The trade-off between achieving the desired accuracy while maintaining the speed of the convergence during training should be balanced to get an improved model performance. [14]

2.10 Major Findings

In summary, Among the various models and approaches presented in the literature review studies have highlighted the potential of pretrained models in achieving high accuracy for plant disease classification. As transfer learning allowed models to leverage relearned features from large data set like ImageNet.[7] The performance of the Convolution neural network is significantly influenced by the quality and quantity of the dataset. Hybrid models, such as convolutional encoder networks that combine CNNs with autoencoders, showed promise in enhancing feature extraction and improving classification accuracy. [5] These models were particularly effective in smaller datasets by maximizing the extraction of relevant features. Deeper architectures, such as ResNet101 and DenseNet, outperformed shallower models like AlexNet. However, the increased complexity and training cost of deeper models necessitated a trade-off between performance and computational resources. In comparative studies, deep learning models outperformed traditional machine learning approaches and human experts in terms of accuracy and predictive value, especially in complex disease identification tasks.[10] Various optimization techniques, including different learning rates, batch sizes, and optimizers (SGD, Adam, RMSprop), were explored. These parameters had an important impact on the training process and final model performance. Fine-tuning these hyperparameters was influential for achieving optimal results.

With good accuracy there was also some gaps identified in the literature presented. The reports included challenges in handling the variability in image quality and environmental conditions. Models trained on controlled datasets often struggled when tested on real-world data, leading to reduced accuracy. Some research faced limitations due to small dataset sizes, which impacted the generalization and robustness of the models. Smaller datasets often led to overfitting and less reliable performance in diverse real-world scenarios. Most studies focused on a limited number of crops and diseases, raising questions about the models' ability to generalize across a broader spectrum of agricultural scenarios.

3 Methodology

This section describes the methods employed for the classification of plant diseases using deep learning models. It describes the dataset used in detail. Three high performance convolutional neural network, Xception, InceptionV3 and ResNet50 is explained. Each model is initially trained with the same learning rate later some optimization is done to enhance the accuracy. The strategies employed to achieve high classification accuracy aiming to balance computational resource, generalization and accuracy is explained further in this section.

3.1 Dataset

The dataset has 13,324 images containing both healthy and unhealthy leaf images divided into 17 classes. [15] The images are structured into different folder for each disease category for convenient access and organization. The Potato and corn images collected from PlantVillage dataset and were taken at experimental research stations associated with universities in USA. The plucked leaf were placed on a sheet so that the images were taken under full light using a standard digital camera. [16] The identity of disease was done by expert plant pathologists. [16] Rice, Wheat and Sugarcane are collected from individual dataset available from Kaggle. The dataset has 5 crop species i.e. Potato, Rice, Wheat, Corn and Sugarcane. The folders are labelled with the name of the disease type i.e. fungal and bacterial. The images are in JPG (Joint photographic Expert Group) format and is size of 4.45 GB. The Kaggle notebook access the dataset from Kaggle, but it does not allow to split the data into sets hence a working directory is created where first the data is copied and later the copied data is used for splitting the data into train, validation and test sets.



Figure 1 : Example of each leaf images from a class of dataset.

3.2 Model Selection

The three models Xception, InceptionV3 and Resnet50 have consistently performed well in image classification challenges and competitions, proving their robustness, reliability and generalization. The models are capable of capturing intricate details. The model's ability to use the pre trained weights helps in enhancing model performance and reduces need of training from scratch. The techniques used in the model reduces computational load making its use feasible in resource constrained environments. Their strengths align well with the requirements of accurately classifying plant diseases from images, making them excellent choices based on the insights gathered from the literature review.

3.3 Transfer Learning

Transfer learning is a technique in which the model developed for a primary task can be reused for the secondary task gaining knowledge from the model developed during primary task and used for a different but related problem.[17] This helps in performing better specially in cases where the data is limited. It uses a pre-trained model, initially this model is trained on a large number of data, such as image classification on ImageNet which has 1000 classes and a total of 14 million plus images as it is trained on such vast dataset it captures numerous numbers of edges, texture and shapes. It is often said to provide great results than the model which is trained from scratch or having less amount of data. The model is pre-trained due to which it takes less time to train as it has already learned features from the previous task speeding up the training process. One of the great advantages of using pre-trained model is it reduces computational resources as the model has already immensely learned during the pretrained task.

3.4 Image size

The images fed to the model are first resized as the models require input to be of same size. Images from PlantVillage size is 256x256px and for the other category it varies. For Xception image is resized to 299x299px and for ResNet50 and InceptionV3 it is resized to 244x244px. The PIL (Python Imaging Library), integrated within Keras 'ImageDataGenerator' class is used for resizing images by specifying the target size of the image.

3.5 Preprocessing Function

The deep learning model expects the image pixel value to be scaled before it is applied. There is a particular preprocessing function for the models which is used to scale the pixel values of the images. For TensorFlow and Keras it has a preprocessing function specific to model to normalize the pixel values.

```
'tf.keras.applications.xception.preprocess_input',  
'tf.keras.applications.resnet.preprocess_input' ,  
'tf.keras.applications.inception_v3.preprocess_input'
```

3.6 Data Augmentation

Data Augmentation is a part of data preprocessing technique used to generate new data using the primary data which will help in increasing the size of data while adding more variety of data by transformations which is useful during the training. It also helps to avoid model from overfitting. In some cases, it is also useful in improving accuracy of the model. ImageDataGenerators is used for augmentation.

Data Augmentation used for the models:

- Image Rotation 20 degrees,
- Width of the image shifted up to 20% ,
- Height of the image shifted up to 20%,
- Shear range skews image along one axis,
- Image range zoomed in and out by 20%,
- The images are horizontally flipped ,
- Fill mode used to fill all the new pixels generated by transformations.

3.7 Data Split

The data split is done using the library scikit learn using function `train_test_split`. It is important that the data is split into training, validation and test sets. The allocated data for training is 70 % essential to learn and capture features of the data ensuring that the model has enough data to learn the underlying patterns and features of the data. For validation 10 % to monitor the training of the data and ensure the model is not overfitting. Finally, 20 % for test data to estimate the model's performance. Test set ensures that the model is evaluated on unseen data providing estimation on how well the model will generalize. After dividing the data, it is moved into the respective folder.

3.8 Model Implementation

TensorFlow and Keras frameworks are used to implements the Inception, ResNet and Xception model. For each of the model pre-trained weights from ImageNet are initialized.

3.9 Model Training

The main aim of the model is to ensure that it generalize well with the unseen data making accurate predictions. In model training the learning algorithm learns from the training dataset to minimize the error in its predictions to perform specific task image classification. All the models were trained for a total of 30 epochs balancing the computational efficiency and sufficient training time for the model.

3.9.1 Activation Function

Activation function is used to add nonlinearity in the neural network converting the weighted sum of the input from a node.[18] The activation function effects the efficiency and accuracy of neural network and helps the model learn intricate patterns. Activation functions used in models are ReLU and SoftMax.

ReLU is used in the dense layer allowing the model to learn nonlinear link.[18]

$$F(x) = \max(0, x) \quad (1)$$

SoftMax is commonly used for multiclass classification tasks in output layer providing probability distribution over the classes. [19]

$$SoftMax(x_i) = \exp(x_i) / \sum_j \exp(x_j) \quad (2)$$

3.9.2 Batch size.

It is the number of training samples used in single epoch. Batch size balances training speed and stability. Batch size of 32 are used which update the model's weights and

most used as it requires less memory suitable because of computational resources constraints.

3.9.3 Optimizer

The optimizer used for the models is Adaptive Moment Estimation (Adam) which is very popular and widely used in deep learning. The use of the optimizer is to reduce the time taken for training a model. Adam uses both RMSprop and Gradient descent with momentum advantages. The algorithm converges faster towards the minima when gradients are exponentially weighted average while RMSP takes longer to converges as it takes exponentially moving average of squared gradients. [20] The gradient descent rate is controlled with minimum oscillation to reach the global minimum and to avoid oscillating in the local minima.

3.9.4 Learning Rate

Learning rate is a parameter used to define the rate at which the model will learn. It is crucial to tweak the learning rate to enable the model to learn well with the number of layers in a specific training epoch. The weights of the models are updated during training with the learning rate 0.01 and 0.0001

3.9.5 Regularization

Regularization is a technique used to keep a check on the overfitting by putting some constraints to the model. Early stopping is a regularization technique where training stops when the performance does not improve on the validation set preventing overfitting conserving computational resource. All the 3 model is first trained without early stopping and then with early stopping and dropout.

Dropout works by dropping the neurons randomly during training. This keeps in check that there is no dependency of a particular neuron for a model.

3.9.6 Hardware Setup

The models were trained using Kaggle Notebooks, which provide access to powerful cloud-based GPU instances. Kaggle offers free access to high-performance

computing resources, The use of Kaggle Notebooks allowed for efficient training and testing of the models without the need for local hardware investments.

3.10 Models

3.10.1 Inception

The Inception architecture allows convolution with multiple filters i.e. 1x1, 3x3, 5x5 in the same layer allowing the model to extract features at different scales at a time. As the information in each picture varies, it is difficult to determine the optimal filter size. Employing different size filters, the network can adapt to varying sizes of features within the image. [21] The information which is the area covered by the feature of the item that is require different size filters are applied. [21] The network is wider rather than deeper. It has several parallel paths that process the input simultaneously which allows the network to learn diverse features making it effective for image recognition tasks and after that the outputs are concatenated. It also includes max pooling operations which helps in reducing the spatial. [22] As shown in the Figure 2 The 1x1 convolution helps in reducing the number of channels which cuts the computational resources cost.[22] To prevent the vanishing gradient problem, auxiliary classifiers are added in between the layers during training. These auxiliary classifiers calculate the loss in between the process and contribute to the overall loss function, improving gradient flow and aiding in the learning process. [22]

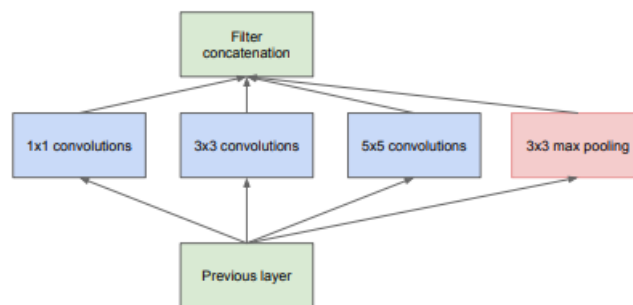


Figure 2: Inception module, naïve version [22]

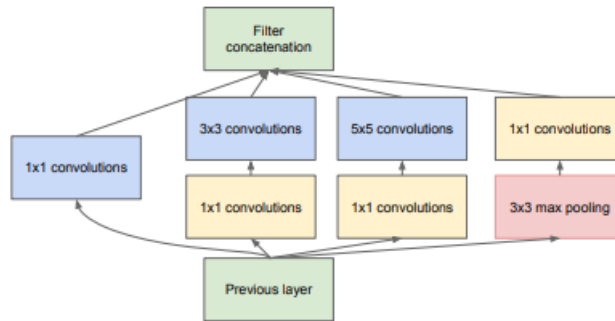


Figure 3: Inception module with dimension reductions [22]

3.10.2 Inception V3 architecture

A 22-layer deep CNN model called inception was proposed by [22] first in 2014. The model uses Factorized convolutions and auxiliary classifier. In particular, the 5x5 convolutions are replaced by 3x3 convolution reducing 28% number of parameters. In Inception V3 considering the layers of the pre trained model and base model has 94 layers with 23,918,385 total parameters and additional layers added to the base are 1 each of global average pooling layer, dense layer with 1024 units using ReLU activation, and output dense layer with 17 unit connected with the number of units matching the number of classes using SoftMax activation resulting in a total of 97 layers in the complete model. The size of the model is approximately 95 MB which reflects the storage required for the parameters and weights. The top classification layer is not included to allow specific tailored layers to the new classification problem.

3.10.3 Xception

Xception (Extreme Inception) based on depth wise separable convolutions which processes each channel independently and pointwise convolutions allowing the network to have spatial and cross channel correlations reducing the number of parameters and computational complexity. [23] Demonstrating state of the art performance on large ImageNet dataset making it suitable for classification and recognizing objects tasks. Standard convolution has 9 times a greater number of

multiplications as that of Depth wise separable convolution. [24] Inspired by the Resnet architecture it has residual connection reducing the vanishing gradient problem in the training of deep networks.

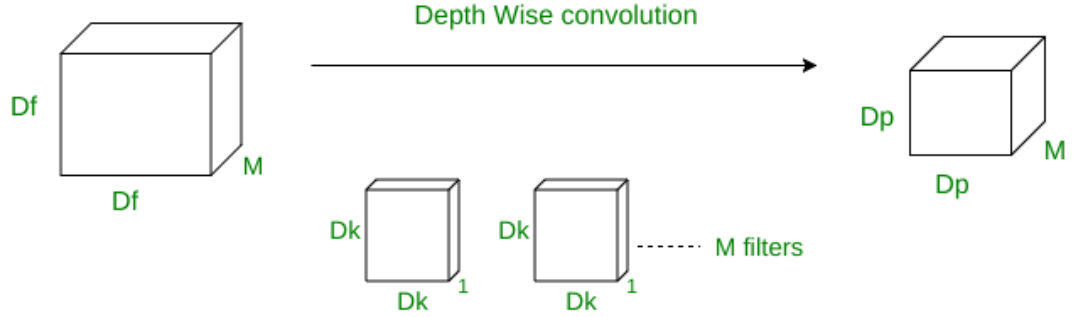


Figure 4: Depth Wise Convolution [25]

Depth wise convolution figure 4. Shows the size of a filter $D_k \times D_k \times 1$ which is applied to M number of channels $D_f \times D_f \times M$ and each filter produced an output $D_p \times D_p \times 1$ and after it is applied to all depth wise filter the output dimension is $D_p \times D_p \times M$ [23], [25]

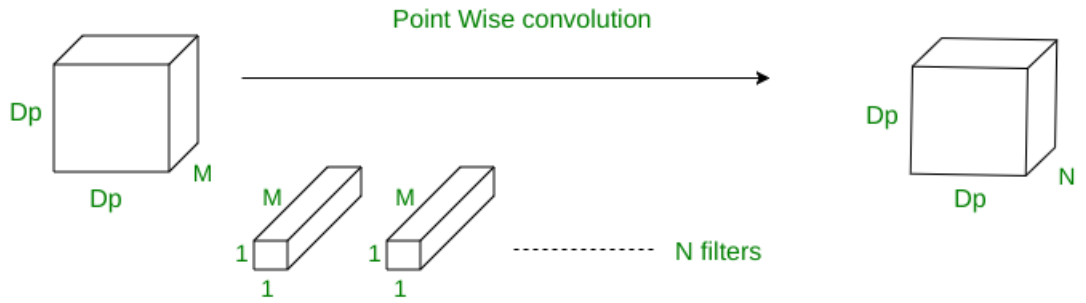


Figure 5: Point Wise Convolution [25]

Pointwise convolution Figure 5. Shows the output of the depth wise convolution $D_p \times D_p \times M$ is applied with $1 \times 1 \times M$ pointwise convolution to N number of filters which produces N output channels with final output dimensions $D_p \times D_p \times N$ [23], [25]

The depth wise convolution multiplication is $D_p^2 \times D_k^2 \times M$ and pointwise multiplication is $D_p^2 \times M \times N$ Therefore total computational cost is $M \times D_p^2 \times (D_k^2 + N)$ which compared to traditional convolution is $D_p^2 \times D_k^2 \times M \times N$ is smaller and also

reduces parameter making it efficient for environment with less parameter maintaining high performance. [23], [24], [25]

3.10.4 Xception architecture

Xception architecture uses depth wise separable convolution was proposed by [23]. Inspired by Inception and ResNet models, it has fewer parameters and aims on achieving high performance focusing on efficient convolution operations. The architecture comprises of total approximately 89 layers of convolutions adding both depth wise and pointwise convolution with total of 25,026,081 parameters including the global average pooling layer, dense layer with 1024 units using ReLU activation, and output dense layer with 17 unit connected with the number of units equal to number of classes using SoftMax activation. Xception compared to Inception has a smaller number of convolution layers, but large number of parameters count due to the model's complex nature of depth wise separable convolution leading to a significant high number of parameter count. The size of the model is approximately 100 MB storing each 32-bit floating point. The top classification layer is not included in this model as well for the same purpose. Due to its architecture model requires significant computational resources for training.

3.10.5 ResNet

ResNet outperformed all the other algorithms and won ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2015. [26] It was trained on ImageNet's data over 1.2 million training images having 1000 different classes. [26] In ResNet the input of the layer is added to the output of another layer and the original input is applied to the convolution as well creating a stack further down. In Deep Learning network due to backpropagation problem of vanishing gradient surfaces. ResNet architecture is famous for mitigating the vanishing gradient problem by using residual connection. [27] It introduces skip connection, which results in reducing vanishing gradient as small gradient values due to multiplication which are introduced during back propagation are skipped allowing training of deep networks [27]

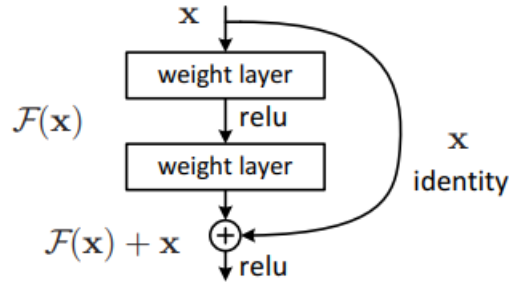


Figure 6: Residual learning: a building block. [26]

As show in the figure 5. Residual learning block has 2 unit and a shortcut connection that is adding the original input to the output. These blocks are stacked together. The network fits the residual mapping instead of layers learning the mapping thus the layer which is degrading the performance is skipped by regularization. The output of the residual block is represented by equation [26]

$$Y = F(x) + x \quad (3)$$

Where x is the input, $+ x$ is a skip connection and function of x $F(x)$ is residual mapping.

3.10.6 Resnet architecture

ResNet introduced by Kaiming [27] to overcome the vanishing gradient problem using residual connections. It has 49 convolutional layers in stack of residual blocks, having a total of 50 convolutional operation within each residual blocks. In total Approximately 25,636,712 parameters including the global average pooling layer, dense layer with 1024 units using ReLU activation, and output dense layer with 17 unit connected with the number of units equal to number of classes using SoftMax activation. The size of the model is approximately 111 MB. The ResNet has high number of filters and filter size in convolutional layers resulting in a large number of parameters despite having a less number of layers compared to Xception and Inception due to architectural and design principle.

4 Analysis

4.1 Evaluation Metrics

To measure classification performance `sklearn.metrics` [28] module is used for evaluating the output of each model. Some of the main evaluation metrics considered for classification tasks are Accuracy, Confusion Matrix, F1-score, Precision Recall etc. These metrics provide insights into the accuracy, reliability of the model.

4.1.1 Confusion Matrix

Confusion Matrix is a function which after prediction visually represents the number of right and wrong predicted classes with the help of a matrix. The x-axis is predicted class and y-axis is true class. [28] Figure 7 shows how the correctly classified labels have high values indicating better performance. If the row 1 and column 2 has a high number than that class 2 is misclassified as class 1. Hence it helps in visually identifying areas where the model has misclassified a class.

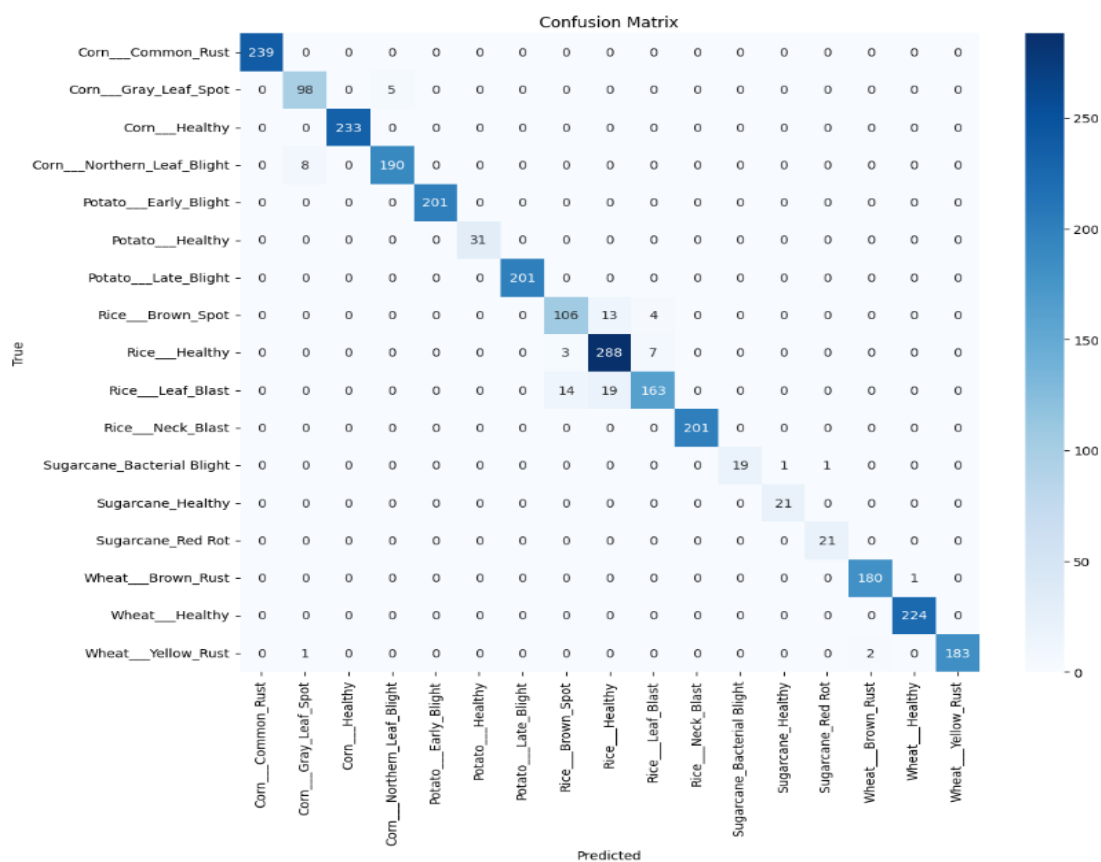


Figure 7 : Xception Confusion Matrix

4.1.2 Classification Report

1. Accuracy

Accuracy is the number of instances predicted correctly divided by the number of total instances [29]

2. Precision

Precision measures the percentage of true positive predictions out of all positive prediction specifying number of predicted instances that are correct.[29]

3. Recall

Recall measures the percentage of true positive predictions out of all actually positive instances. [29]

4. F1 score

F1 score is the harmonic mean of the precision and recall. [29]

5. Support

Support is the number of instances in a class. [29]

4.2 Inception Output Analysis

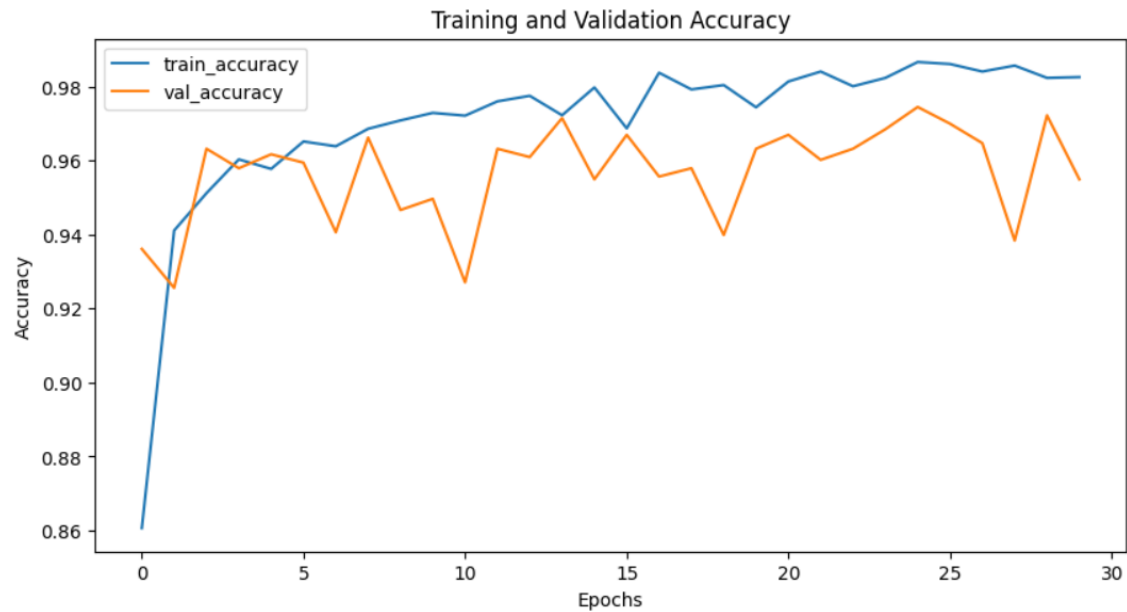


Figure 8 : Inception Training and Validation Accuracy

As shown in the graph figure 8. the training accuracy increases and stabilizes around 98% specifying that the model is learning the training data very well. The validation accuracy is fluctuating and stabilizes around 96% indicating slight overfitting. The performance results shows that InceptionV3 achieves high validation and training accuracy. Model performs well with some of the classes having F1 scores of 1.00 while classes "Potato Early Blight" and "Rice Leaf Blast" have lower F1 score could be due to less distinct features, smaller sample sizes and requires improvement. In general, the classification across most classes is correct which indicates strong generalization and the model's architecture and hyperparameters are well suited to the dataset.



Figure 9 : Inception Training and Validation Loss

Graph figure 9. showing training and validation loss over 30 epochs . The training loss keeps decreasing as the epoch increases as the model is studying training data effectively and almost starts to stabilize around epoch 25. The validation loss also decreases indicating model generalizing well on unseen data but with increase in epoch there is a sudden spike as the model starting to overfit the training data. Trying to mitigate this issue of overfitting, the model retested using Early dropout and the model resulted in stopping the process at 8 epochs achieving an accuracy of 95.48% whereas it was 94.32% without early stopping. It helped in reducing overfitting to some extent as seen in the improved performance. To address the issues to further improve the model Augmentation of the images particularly with the lower F1 scores can help the model improve generalization. Considering other regularization techniques or dropout with different value can prevent overfitting. The model's performance varied across different classes, with some achieving F1 scores of 1.00, indicating perfect precision and recall. However, the classes exhibited lower F1 scores could be due to several factors diseases may exhibit symptoms that are visually similar to healthy plants or other diseases, making them harder for the model to distinguish. Limited training samples for these classes could result in insufficient learning, leading to poorer performance.

4.3 ResNet Output Analysis

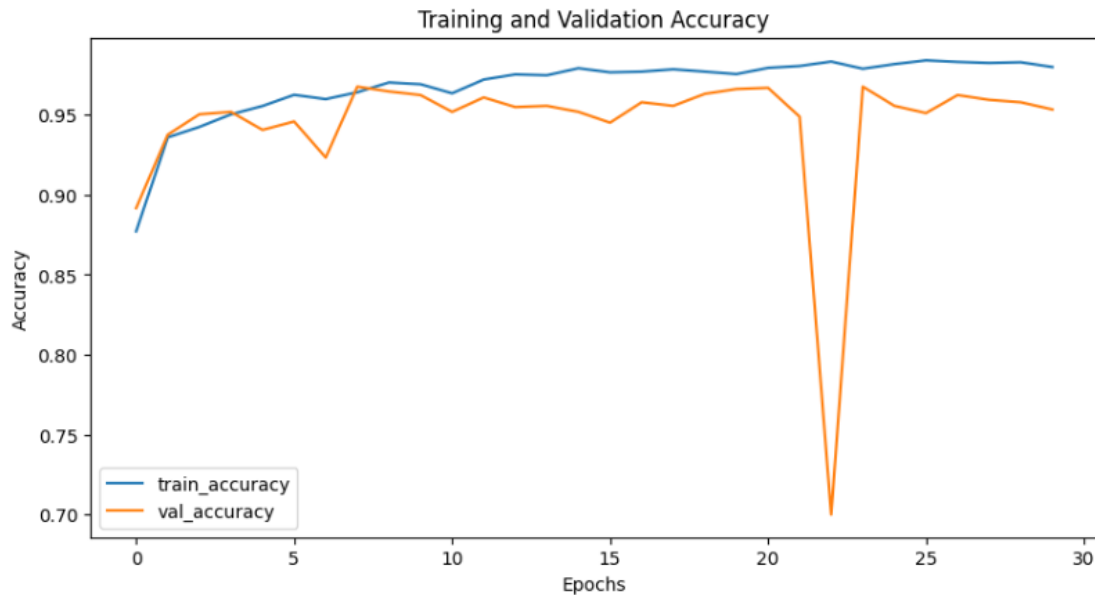


Figure 10 : ResNet Training and Validation Accuracy

The graph figure 10. Shows training and validation accuracy. The training data seem to learn quite steadily slightly fluctuating but generally maintaining the high accuracy as the model is learning effectively. The validation accuracy shows some fluctuations but follows the training trend. There is a notable drop between 20 to 25 epochs which is recovered back maintaining the high validation accuracy. There are many factors that could be considered, the drop can be instances where the model is overfitting the training data because of high learning rate. When the learning rate to train the model is decreased from 0.0001 to 0.01 the validation accuracy continues to drop and recover every 5 epochs, and the overall accuracy drops to 76.36%. To overcome this problem considering a learning rate finder to identify the optimal learning rate is required so that the behaviour of accuracy can be observed while tweaking the learning rate.

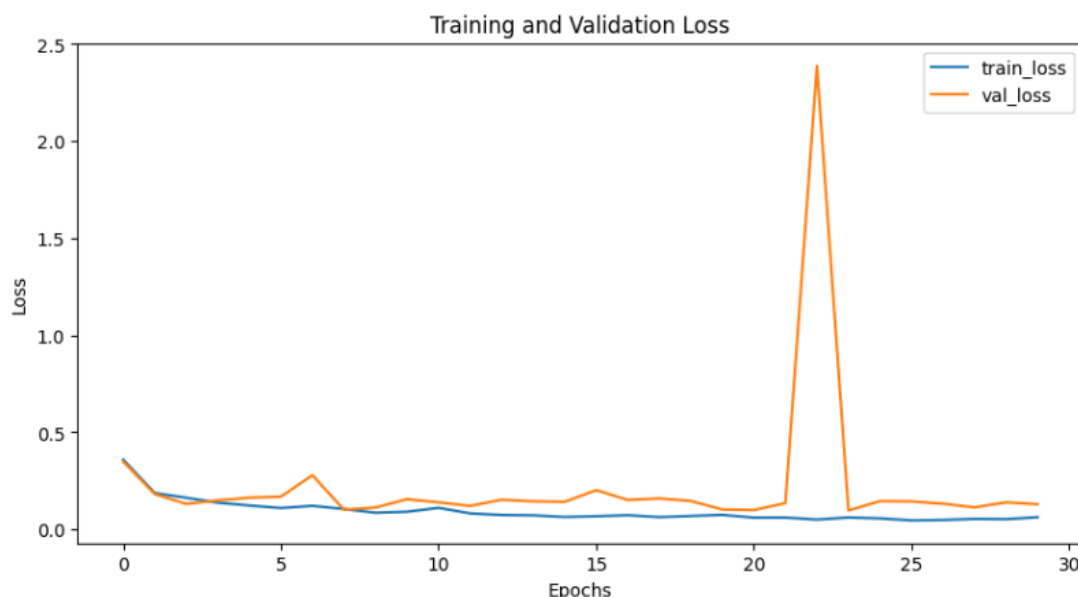


Figure 11 : ResNet Training and Validation Loss

Figure 11. Shows graph of training and validation loss of the model over 30 epochs. The training loss is consistent and decreasing as the model is learning in slower pace. For the validation loss it also continues to decrease until epoch 20 the peak between the 20 to 25 epoch maybe because of some anomaly but it continues to be stable after it. The model is performing well with no major signs of overfitting. The ResNet model achieves an accuracy of 95 % on the test data. It performs consistently across different classes. Classes Potato healthy and Rice Brown Spot has lower precision. Overall, the precision of the model is low indicating that it might be over predicting certain classes. Just to increase the robustness of the model when tested with early dropping the accuracy decreased to 94.14%. The effect of dropout maintained the generalization but requires better tuning for accuracy to increase.

Compared with Inception Both models achieve high accuracy on training and validation datasets. InceptionV3 reached a training accuracy of around 98% and a validation accuracy of approximately 96%, while ResNet achieved 95% on test data. The ResNet model showed some fluctuation in validation accuracy, particularly with learning rate adjustments, whereas InceptionV3's validation accuracy was more stable but still showed signs of slight overfitting. ResNet, is highly effective but

requires more careful tuning of hyperparameters, particularly the learning rate, to avoid fluctuations in performance.

4.4 Xception Output Analysis

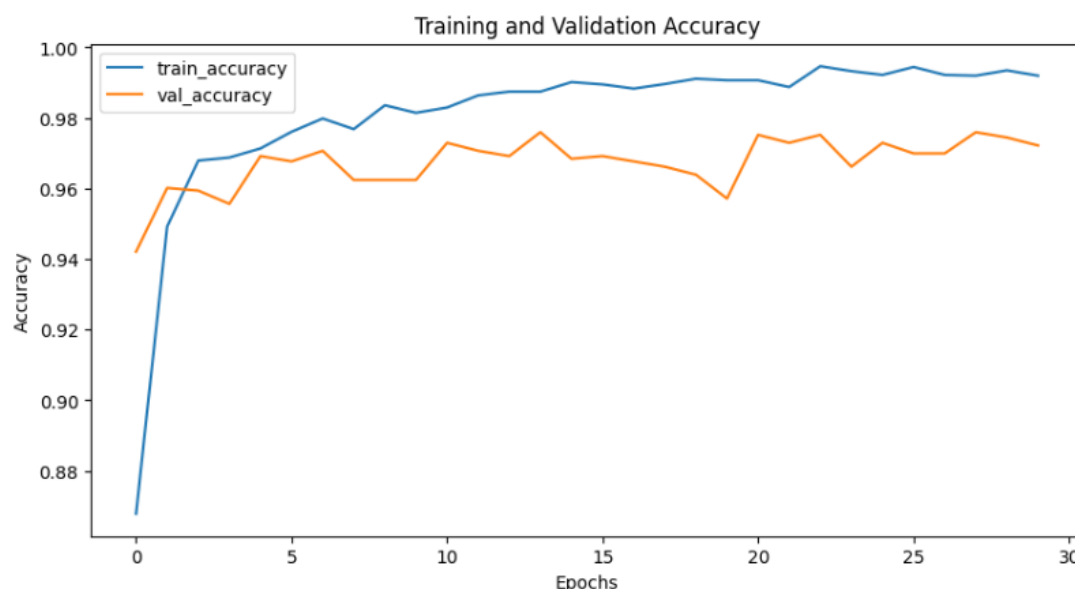


Figure 12 : Xception Training and Validation Accuracy

In Figure 12 Graphs shows Training and Validation Accuracy Xception model achieved an overall accuracy of 96.71 % on the test dataset compared to Inception and ResNet without the use of early stopping. With early stoping the accuracy is 97.05 % which is not much of a difference. The model is performing well across all classes. Some classes Rice brown spot and Rice leaf blast shows lower precision and recall indicating false negative. These classes have fewer samples compared to others, the model is not learning their features effectively.



Figure 13 : Xception Training and Validation Loss

In figure 13 shows the training and validation loss. The training loss is steadily decreasing, and the model is learning well as the epochs increases. The Validation loss is fluctuating and does not follow training loss the model is overfitting with increase in epochs. The validation loss oscillates but remains relatively low as it is generalising well. To overcome overfitting the model was trained with early stopping and it provided slight improvement benefitted from fine tuning. To address lower precision classes, class imbalance handling might help. The Xception model achieves a final accuracy of 96.71%, which is comparable to both ResNet and InceptionV3. The marginal increase to 97.05% with early stopping is slightly higher than the ResNet's final accuracy of 95% and close to InceptionV3's validation accuracy of 96%. InceptionV3 shows slightly better stability in validation performance but has some issues with overfitting similar to Xception. ResNet, despite its solid performance, displays more pronounced fluctuations in validation accuracy, especially with learning rate adjustments. Xception shows a steadily decreasing training loss, suggesting effective learning, but its fluctuating validation loss highlights ongoing overfitting concerns. This is somewhat similar to ResNet's training dynamics, where validation accuracy showed fluctuations, particularly due to learning rate adjustments. ResNet,

with its deeper architecture, is more complex and sensitive to hyperparameter settings compared to Xception and InceptionV3. In contrast, InceptionV3 provides a good balance between stability and performance, though it requires fine-tuning to avoid overfitting, as seen in Xception's need for early stopping to improve results. The Xception model demonstrates a strong ability to learn from training data and generalize to unseen data, achieving high accuracy comparable to ResNet and InceptionV3. While the slight increase in accuracy with early stopping suggests that the model was already well-optimized, the ongoing issues with overfitting and lower precision for certain classes indicate areas for improvement. Table. 1 shows comparison between the 3 models.

Table 1 : Model Accuracy comparison

Model	Accuracy of Basic model with augmentation and learning rate 0.01	Accuracy of model with Augmentation and learning rate 0.0001	Accuracy of model with Augmentation and learning rate 0.0001 with Early Stopping and Drop out
ResNet	76.36%	95.03%	94.14%
Inception	79.24%	94.32%	95.48%
Xception	91.34%	96.71%	97.05%

4.5 Confusion Points

Some of the common misclassifications was Corn gray leaf spot, rice brown spot and rice leaf blast challenging for the models to classify. Xception performs better than the ResNet and Inception having higher precision and recall. ResNet misclassifies Rice blast and Rice brown spot with the other rice classes having higher false negatives and false positives confusing with other rice diseases. Inception performs slightly better than ResNet in identifying but having higher rate of false negatives. Xception was struggling with Rice brown spot having less precision and recall. Potato

healthy class was perfectly distinguished by the Xception model with perfect precision and recall. ResNet also has perfect recall score but 0.63 precision indication other instance being misclassified as Potato healthy similarly, Inception showing strong performance but slightly more false positives as precision is 0.94. Each model has its own strength and weakness. The differences in precision and recall across the models indicate that they may be making trade-offs between correctly identifying positive instances (recall) and avoiding false positives (precision). The diseases like Rice brown spot and Rice leaf blast often present very similar visual symptoms, such as small lesions on the leaves, which can make it difficult for models to distinguish between them. Certain classes (like Rice brown spot or Rice leaf blast) might have fewer examples in the dataset, the models are not learning their features effectively. Even though sugarcane classes despite having less data but due to being distinct in visual in the whole dataset it is getting trained well leading to better performance.

Conclusion & Recommendations

Three different convolutional neural network architectures InceptionV3, ResNet, and Xception were evaluated for classifying images of crops affected by various diseases. The models were evaluated using a number of metrics, including accuracy, precision, recall, and F1-score. Early stopping and dropout regularization techniques were employed to improve model generalization and prevent overfitting.

Early stopping and dropout regularization are powerful techniques for preventing overfitting in neural networks. Early stopping helps to halt training at the optimal point, while dropout ensures that the model does not become overly reliant on any specific neurons. Regularization techniques, dropout was employed to mitigate overfitting. The effectiveness of these techniques varies depending on the model and the specific training dynamics. In the cases of the Xception and Inception models, these techniques were effective in improving generalization and achieving high accuracy, in the InceptionV3 model it contributed to improved generalization and higher validation accuracy. The observed effects of dropout varied across models, emphasizing the need for model-specific regularization strategies as for the ResNet model, the tuning of these techniques may require further adjustments to attain the best results.

The choice of optimizer also significantly impacted the training outcomes. The Adam optimizer, with its adaptive learning rate mechanism, proved effective in achieving quick convergence and high accuracy for all models. In comparison, Stochastic Gradient Descent (SGD) required careful tuning of the learning rate and momentum to achieve comparable performance. Adam's ability to adjust learning rates dynamically for each parameter made it particularly well-suited for the complex models and dataset.

The Xception model emerged as the top performer, achieving the highest accuracy of 96.71% on the test dataset. This model demonstrated excellent generalization capabilities, consistently performing well across most classes. The analysis of the training and validation loss indicates that the model learns effectively but could benefit

from regularization to prevent overfitting. The implementation of early stopping further improved the accuracy to 97.05%, although the improvement was marginal. The model when tuned with early stopping and dropout performed well, which helped prevent overfitting. It shows high precision, recall, and F1-scores across most classes, though there are a few areas with room for improvement, particularly in balancing precision and recall for certain classes like Rice Brown Spot and Rice Leaf Blast. This indicates that Xception is a robust model with strong baseline performance, capable of handling complex classification tasks effectively. The model's depth and use of depth wise separable convolutions contributed significantly to its superior performance. Overall, the Xception model demonstrates strong performance and reliability.

The ResNet model performs very well on the given dataset, with an overall accuracy of 95%. It shows high precision, recall, and F1-scores across most classes but less than Xception model, though there are a few areas with room for improvement, particularly in balancing precision and recall for certain classes like Potato Healthy and Rice Brown Spot. This analysis indicates that while the model is highly effective, further fine-tuning and possibly more data augmentation or addressing class imbalances could enhance its performance even further. The model achieved high accuracy but showed a slight decrease with early stopping, this suggests that the model might have required more epochs to fully learn the training data, and early stopping might have caused underfitting. Adjusting the patience parameter or allowing more epochs could potentially address this issue. ResNet50's residual connections played a crucial role in mitigating the vanishing gradient problem, thus enhancing its ability to learn deep representations.

InceptionV3, while achieving a respectable accuracy of 94.32%, showed room for improvement, particularly in classes like "Potato Early Blight" and "Rice Leaf Blast." It showed excellent performance with high training and validation accuracy, indicating good generalization. It achieves high precision, recall, and F1-scores for most classes, making it a reliable choice for this classification task. When compared to ResNet50 and Xception, InceptionV3 provides a balanced architecture with efficient

parameter usage and high performance. The use of early stopping improved the accuracy to 95.48%, indicating that the model benefited from regularization to prevent overfitting. InceptionV3's architecture, with its inception modules, allowed for efficient feature extraction at multiple scales, contributing to its overall good performance.

The choice of learning rates (0.0001 and 0.01) provided insights into the sensitivity of the models to this hyperparameter. Adam was the preferred optimizer for Xception, Inception, and ResNet due to its adaptive nature, efficiency in handling deep architectures, and ability to converge faster with less tuning compared to SGD. The impact of these choices was evident in the steady improvements and high accuracy achieved by all models, demonstrating the importance of selecting appropriate learning rates and optimizers. Only two learning rates 0.0001 and 0.01 were tested, observing that a lower learning rate generally resulted in more stable convergence and better final accuracy. But this may not have been optimal for all models. A more systematic approach, and the accuracy obtained such as using a learning rate finder or grid search, could help identify the optimal learning rate, improving model convergence and performance.

Certain classes, such as "Potato Healthy" and "Rice Brown Spot," had lower precision and recall scores, indicating misclassification issues. These issues could stem from class imbalance or insufficient training data for these specific classes. Techniques like oversampling, under sampling, or using class weights could help address this.

Overall, the Xception model emerged as the best performer with an accuracy of 97.05% after applying early stopping. The application of these techniques requires careful tuning to avoid underfitting, as seen in the ResNet model. Future work could involve more extensive hyperparameter tuning, addressing class imbalance, and exploring additional regularization techniques to further improve model performance.

In summary, this study underscores the effectiveness of advanced CNN architectures in plant disease classification. Each model has a unique architecture designed to handle specific aspects of deep learning tasks, with ResNet50 focusing on depth, InceptionV3 on feature diversity, and Xception on convolutional efficiency.

Each model exhibited unique strengths, with Xception leading in overall accuracy and generalization. The importance of hyperparameter tuning, optimizer selection, and regularization techniques was evident in enhancing model performance. This comprehensive evaluation provides a solid foundation for deploying deep learning models in practical agricultural applications, potentially revolutionizing plant disease diagnosis and management.

Future work could explore the integration of ensemble methods and the application of these models to a broader range of agricultural datasets to further improve accuracy and robustness. Implement systematic hyperparameter optimization techniques such as grid search, Bayesian optimization or random search to fine-tune the learning rates, batch sizes, and other critical parameters. Using learning rate finders can help identify the optimal learning rates more efficiently. Expand the use of data augmentation to increase the size and variability of the training dataset. Employ techniques such as rotation, scaling, translation, flipping, and colour jittering. This will help enhance the model's ability to generalize and manage unseen data more effectively. Utilize advanced techniques to manage class imbalance, such as over sampling minority classes, under sampling majority classes, and generating synthetic samples using SMOTE or similar methods. Incorporating class weighting in the loss function can also ensure that the model pays adequate attention to less represented classes. Further investigate and apply various regularization techniques beyond dropout and early stopping. Techniques such as L1/L2 regularization (weight decay), batch normalization, and the use of data augmentation as regularization should be considered to reduce overfitting and improve generalization. Experiment with newer or more sophisticated model architectures and hybrids, such as EfficientNet, DenseNet, or custom architectures tailored to the specific characteristics of plant disease images. This can potentially yield better feature extraction and classification performance. Validate and evaluate the models on real-world data collected directly from agricultural fields. This will ensure that the models are robust and effective in practical applications, where environmental conditions and image quality can vary significantly. Implementing these recommendations, future research can enhance the

performance, reliability, and applicability of CNN models in plant disease classification, contributing to more effective disease management and improved agricultural productivity.

The deployment of deep learning models on mobile devices and edge computing environments is a significant step toward bringing advanced agricultural technology directly into the hands of farmers. By enabling real-time disease diagnosis and crop management solutions, these models can greatly enhance decision-making processes in the field. However, the challenges of limited computational resources on mobile devices necessitate the development of lightweight yet effective models. Balancing performance with efficiency is key to ensuring that these technologies are accessible and practical for widespread use in diverse agricultural settings.

Real-world case studies demonstrate both the successes and limitations of deploying these models in practical applications. For instance, models adapted for specific regional crops and local disease prevalence have shown promise in improving crop yields and reducing losses. However, these successes are often tempered by challenges such as difference in environmental conditions, differences in crop varieties, and the need for continuous model updates to remain effective. These case studies underscore the importance of tailoring models to local contexts and the need for ongoing research and development to overcome the limitations encountered in real-world deployments.

While significant strides have been made in deploying AI-driven crop management tools on mobile and edge devices, continued focus on optimizing these models for real-world applications is essential. By addressing the challenges of computational limitations and local adaptation, these technologies can become more widely adopted, ultimately contributing to more sustainable and efficient agricultural practices worldwide.

The integration of deep learning models with IoT devices and precision agriculture technologies represents a transformative step forward in modern farming practices. By leveraging real-time data from drones, sensors, and other IoT devices, these

models can offer unprecedented accuracy in monitoring crop health and predicting disease outbreaks. This real-time integration not only enhances decision-making but also enables more targeted interventions, reducing the need for broad-spectrum treatments and promoting more sustainable farming practices.

Moreover, the development of sustainable and climate-resilient models is crucial in the face of growing environmental challenges. AI-powered tools can help anticipate disease patterns influenced by climate change, allowing farmers to adapt their strategies proactively. Research focused on these areas highlights the importance of creating adaptable models that can generalize well across various environmental conditions, ensuring their effectiveness in diverse agricultural settings.

In conclusion, the convergence of AI, IoT, and precision agriculture holds immense potential to revolutionize crop management, making agriculture more efficient, sustainable, and resilient to the challenges posed by climate change. Future research and development should continue to focus on enhancing the adaptability and scalability of these technologies to ensure their widespread adoption and success in diverse agricultural environments.

References

- [1] H. Ritchie and M. Roser, 'Half of the world's habitable land is used for agriculture', *Our World Data*, Feb. 2024, Accessed: Aug. 07, 2024. [Online]. Available: <https://ourworldindata.org/global-land-for-agriculture>
- [2] H. N. Ngugi, A. E. Ezugwu, A. A. Akinyelu, and L. Abualigah, 'Revolutionizing crop disease detection with computational deep learning: a comprehensive review', *Environ. Monit. Assess.*, vol. 196, no. 3, p. 302, Feb. 2024, doi: 10.1007/s10661-024-12454-z.
- [3] 'Notebooks Documentation'. Accessed: Aug. 07, 2024. [Online]. Available: <https://www.kaggle.com/docs/notebooks>
- [4] A. Picon, A. Alvarez-Gila, M. Seitz, A. Ortiz-Barredo, J. Echazarra, and A. Johannes, 'Deep convolutional neural networks for mobile capture device-based crop disease classification in the wild', *Comput. Electron. Agric.*, vol. 161, pp. 280–290, Jun. 2019, doi: 10.1016/j.compag.2018.04.002.
- [5] A. Khamparia, G. Saini, D. Gupta, A. Khanna, S. Tiwari, and V. H. C. Albuquerque, 'Seasonal Crops Disease Prediction and Classification Using Deep Convolutional Encoder Network', *Circuits Syst. Signal Process.*, vol. 39, Feb. 2020, doi: 10.1007/s00034-019-01041-0.
- [6] A. Krishnaswamy Rangarajan, P. Raja, and A. Ramesh, 'Tomato crop disease classification using pre-trained deep learning algorithm', *Procedia Comput. Sci.*, vol. 133, pp. 1040–1047, Jan. 2018, doi: 10.1016/j.procs.2018.07.070.
- [7] J. Barbedo, 'Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification', *Comput. Electron. Agric.*, vol. 153, pp. 46–53, Oct. 2018, doi: 10.1016/j.compag.2018.08.013.
- [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, Sep. 2016, doi: 10.3389/fpls.2016.01419.
- [9] K. P. Ferentinos, 'Deep learning models for plant disease detection and diagnosis', *Comput. Electron. Agric.*, vol. 145, pp. 311–318, Feb. 2018, doi: 10.1016/j.compag.2018.01.009.
- [10] A. Cruz *et al.*, 'Detection of grapevine yellows symptoms in *Vitis vinifera* L. with artificial intelligence', *Comput. Electron. Agric.*, vol. 157, pp. 63–76, Feb. 2019, doi: 10.1016/j.compag.2018.12.028.
- [11] M. A. Moid and M. Ajay Chaurasia, 'Transfer Learning-based Plant Disease Detection and Diagnosis System using Xception', in *2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, Nov. 2021, pp. 1–5. doi: 10.1109/I-SMAC52330.2021.9640694.
- [12] Y. Gulzar, Z. Ünal, S. Ayoub, and F. Reegu, 'Exploring Transfer Learning for Enhanced Seed Classification: Pre-trained Xception Model', 2024, pp. 137–147. doi: 10.1007/978-3-031-51579-8_14.
- [13] E. Karantoumanis, V. Balafas, M. Louta, and N. Ploskas, 'Computational comparison of image preprocessing techniques for plant diseases detection', Sep. 2022, pp. 1–5. doi: 10.1109/SEEDA-CECNSM57760.2022.9932972.
- [14] M. D. Liman, S. Osanga, E. Alu, and S. Zakariya, 'Regularization Effects in Deep Learning Architecture', *J. Niger. Soc. Phys. Sci.*, p. 1911, Mar. 2024, doi: 10.46481/jnsps.2024.1911.

-
- [15] 'Top Agriculture Crop Disease India'. Accessed: Aug. 07, 2024. [Online]. Available: <https://www.kaggle.com/datasets/kamal01/top-agriculture-crop-disease>
 - [16] D. P. Hughes and M. Salathe, 'An open access repository of images on plant health to enable the development of mobile disease diagnostics', Apr. 11, 2016, *arXiv*: arXiv:1511.08060. doi: 10.48550/arXiv.1511.08060.
 - [17] K. Team, 'Keras documentation: Transfer learning & fine-tuning'. Accessed: Aug. 09, 2024. [Online]. Available: https://keras.io/guides/transfer_learning/
 - [18] J. Brownlee, 'A Gentle Introduction to the Rectified Linear Unit (ReLU)', *MachineLearningMastery.com*. Accessed: Aug. 09, 2024. [Online]. Available: <https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/>
 - [19] 'Softmax function', *Wikipedia*. Jul. 09, 2024. Accessed: Aug. 09, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Softmax_function&oldid=1233456609
 - [20] CodeEmporium, *Optimizers - EXPLAINED!*, (Feb. 10, 2020). Accessed: Aug. 07, 2024. [Online Video]. Available: <https://www.youtube.com/watch?v=mdKjMPmcWjY>
 - [21] Code With Aarohi, *InceptionV3 | Inception Network*, (Sep. 15, 2020). Accessed: Aug. 07, 2024. [Online Video]. Available: <https://www.youtube.com/watch?v=ekixid21Q5U>
 - [22] C. Szegedy *et al.*, 'Going Deeper with Convolutions', Sep. 16, 2014, *arXiv*: arXiv:1409.4842. doi: 10.48550/arXiv.1409.4842.
 - [23] F. Chollet, 'Xception: Deep Learning with Depthwise Separable Convolutions', Apr. 04, 2017, *arXiv*: arXiv:1610.02357. doi: 10.48550/arXiv.1610.02357.
 - [24] CodeEmporium, *Depthwise Separable Convolution - A FASTER CONVOLUTION!*, (Mar. 18, 2018). Accessed: Aug. 07, 2024. [Online Video]. Available: <https://www.youtube.com/watch?v=T7o3xvJLuHk>
 - [25] 'Depth wise Separable Convolutional Neural Networks', *GeeksforGeeks*. Accessed: Aug. 07, 2024. [Online]. Available: <https://www.geeksforgeeks.org/depth-wise-separable-convolutional-neural-networks/>
 - [26] K. He, X. Zhang, S. Ren, and J. Sun, 'Deep Residual Learning for Image Recognition', Dec. 10, 2015, *arXiv*: arXiv:1512.03385. doi: 10.48550/arXiv.1512.03385.
 - [27] Code With Aarohi, *ResNet Explained Step by Step(Residual Networks)*, (Jul. 14, 2020). Accessed: Aug. 07, 2024. [Online Video]. Available: <https://www.youtube.com/watch?v=Uuc1wdqMFtQ>
 - [28] '3.4. Metrics and scoring: quantifying the quality of predictions', *scikit-learn*. Accessed: Aug. 07, 2024. [Online]. Available: https://scikit-learn/stable/modules/model_evaluation.html
 - [29] 'precision_recall_fscore_support', *scikit-learn*. Accessed: Aug. 07, 2024. [Online]. Available: https://scikit-learn/stable/modules/generated/sklearn.metrics.precision_recall_fscore_support.html
