# Detection of Diseases in Rice Plants using Convolutional Neural Networks

Nikhil Singhal

2019IMT-067
ABV-IIITM Gwalior
Gwalior-474 015, MP, India Under supervision of: Dr. Joydip Dhar

October 27, 2022

# Presentation outline

- Introduction
- Objectives
- Working at a glance
- Image Augmentation phase
- Proposed Neural Network Architectures
- Mini-Batch Gradient Descent Activation function
- Categorical cross-entropy loss function
- Technologies used
- Results
- Analysis
- References

# Introduction

- Rice is one of the most consumed food in the world and Asia is a huge producer of it.
- A condition that reduces the yield of the crop or injures it is called crop disease. These are characterized by their symptoms. Examples of rice diseases are Brown Spot, Hispa, Leaf Blast.
- We are trying to develop a CNN based model that can predict the diseases in rice plants.

# Objectives

- To propose a model for paddy disease detection
- To classify the diseases by identifying the patterns present on leaves through CNN
- To obtain comparable accuracy for detecting rice plant diseases

# Example Images from dataset

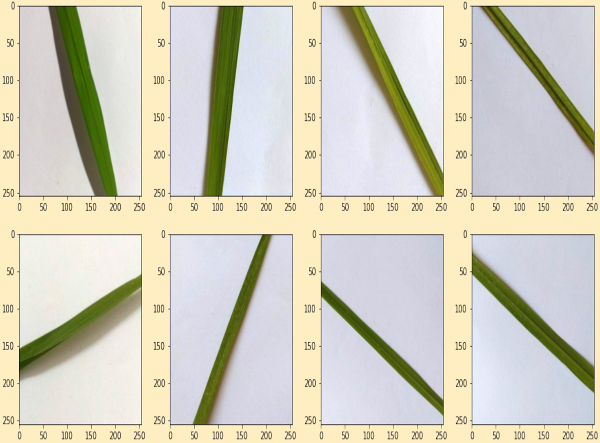


Figure: Healthy Leaf

# Contd...

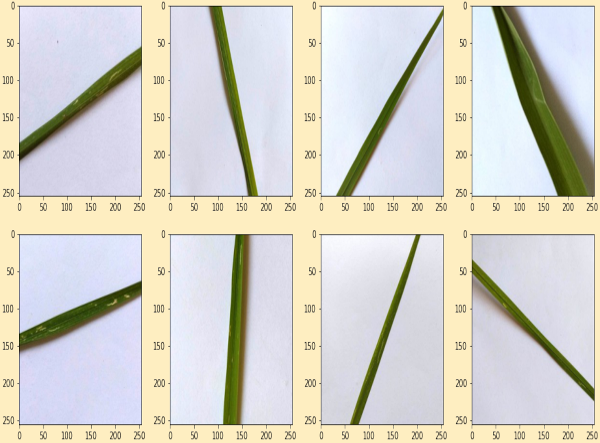


Figure: Hispa Infected Leaf

# Contd...

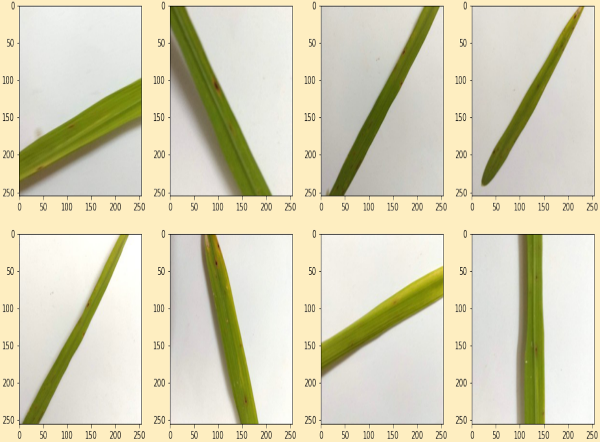


Figure: Brownspot Infected Leaf

# Contd...

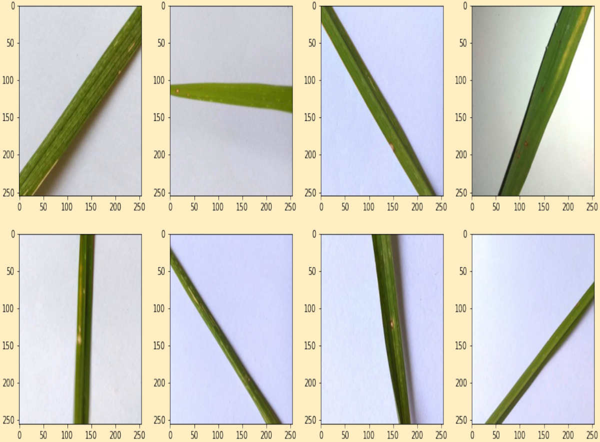


Figure: Leaf Blast Infected Leaf

# Working at a Glance

- First, the dataset is downloaded from Kaggle. Since the dataset is large(16 GB), it is first uploaded to Google drive, and then the drive is mounted to our colab notebook.
- The images are then preprocessed to reduce the dimensions of the images so that they can be fed into the neural network.
- To create diversity in the dataset, augmentation techniques are applied.
- These augmented images are then passed to the convolutional neural networks to predict the correct disease type.

# Data Augmentation

- Data augmentation[2] techniques are used to make variations in dataset.
- It applies transformations, image blurring, and contrast adjustments, rotations, to alter the images.
- This helps model to learn more effectively by creating an expanded dataset.
- Keras package by python is used to implement the augmentation techniques.

# Contd...

Figure below shows an example of the augmented images of a rice leaf plant generated by our algorithm.

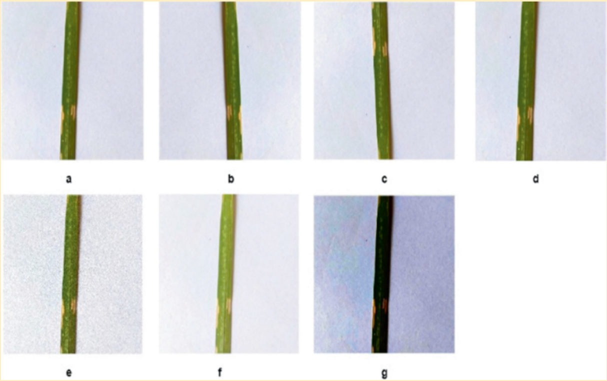


Figure: Augmented Images of a rice leaf (a)Original Image, (b)Rotation by 180 degrees, (c) Vertical flip, (d)Gaussian noise, (e)Horizontal flip, (f)High brightness, (g)Low brightness

# Transfer learning

- Transfer learning[**two**] is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task.
- The general idea is to use the knowledge a model has learned from a task with a lot of available labeled training data in a new task that doesn't have much data. Instead of starting the learning process from scratch, we start with patterns learned from solving a related task.
- Transfer learning has several benefits, but the main advantages are saving training time, better performance of neural networks, and not needing a lot of data.

# Contd...

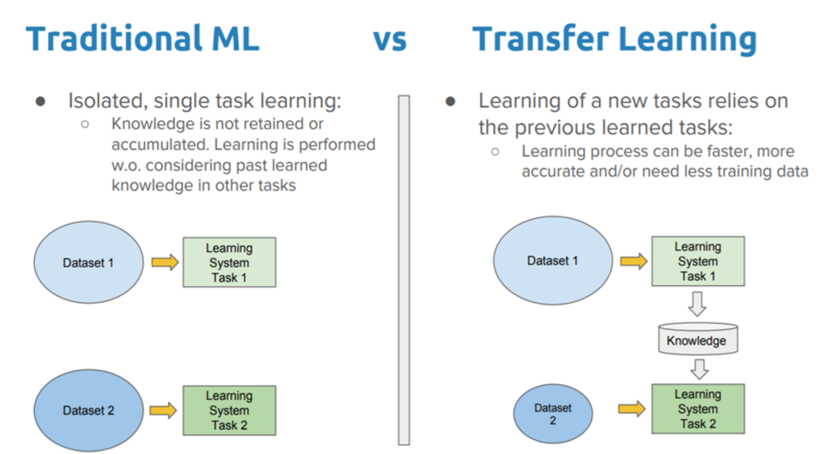


Figure: Transfer Learning[**two**]

# Models Used

In our approach, we have used 3 different models for transfer learning. We are training these three models and comparing them in terms of accuracy and efficiency. The three models are described in the further slides.

# EfficientNet v2

Primary architectural features of EfficientNet v2 are:

1. Both MBConv and the recently developed fused-MBConv are heavily employed in the initial layers of EfficientNet V2[3].
2. EfficientNetV2 advises smaller expansion ratios for MBConv as they frequently have less memory access overhead.
3. EfficientNetV2 favours smaller 3x3 kernel size. To compensate the reduced receptive field, more layers are added to it.

# Architectural Representation of EfficientNet v2

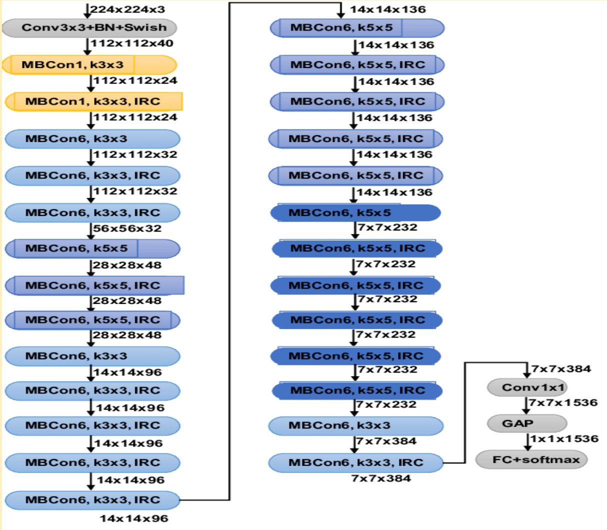


Figure: EfficientNet v2[1]

# Inception v2

1. In the Inception V2 architecture, two 3x3 convolutions are used. Due to the fact that a 5x5 convolution is 2.78 more expensive than a 3x3 convolution, this reduces calculation time and hence boosts computation speed. Therefore, using two 3x3 layers rather than 5x5 layers improves architecture performance.
2. In this architecture, nxn factorizations are transferred to 1xn and nx1 factorizations, this also reduces the computational complexity of the model.
3. In our model, we have added one flatten and two dense layers which have activation functions as relu and softmax. The loss is calculated using the categorical crossentropy function. The optimizer used is Mini-Batch Gradient Descent. We are training the model with the learning rate of 0.001 and the epoch size is 10.

# Contd...

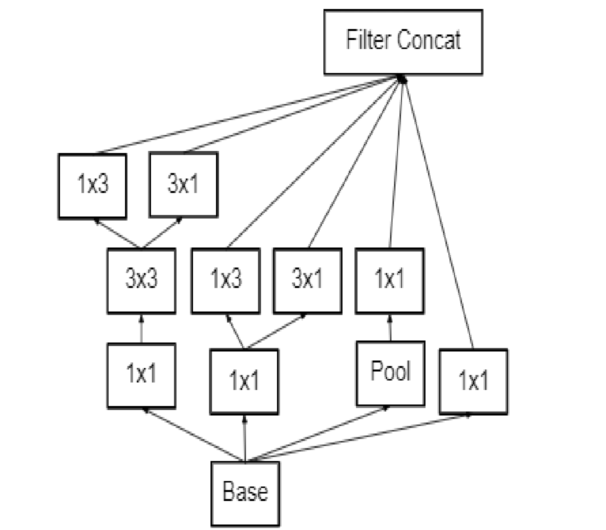


Figure: Inception v2

# Inception v3

1. On the ImageNet dataset, it has been demonstrated that the Inception v3 can achieve higher than 78.1% accuracy.
2. The model is the result of numerous concepts that have been established by various researchers over the years. Convolutions, average pooling, max pooling, concatenations, dropouts, and fully linked layers are some of the symmetric and asymmetric building components that make up the model itself.
3. It consists of a 48-layer deep neural network. To train this model, more than a million of images from more than 1000 different classes from the ImageNet Database were used.
4. The model makes considerable use of batch normalisation, which is also applied to the activation inputs. To calculate loss, Softmax is used.

# Contd...

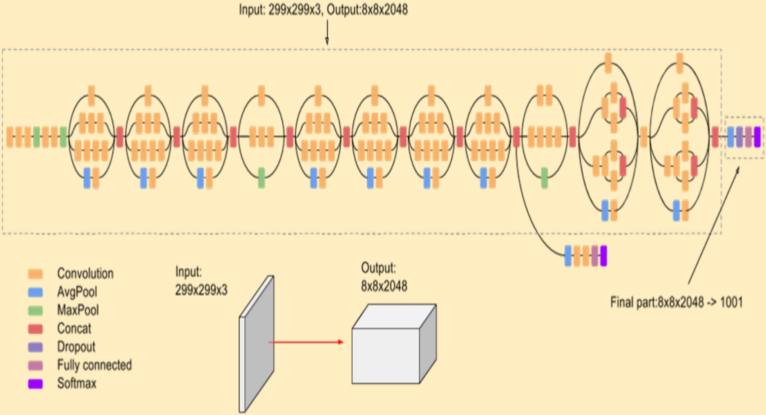


Figure: Inception v3

# Mini-Batch Gradient Descent Algorithm

Mini-batch gradient descent is an machine learning algorithm that splits the training dataset into small batches that are used to calculate model error and update model coefficients.
The formula for Mini-Batch gradient descent algorithm is,

$$\theta = \theta - \nabla C\left(\theta; X_i, Y_i\right) \tag{1}$$

Here $\theta$ is optimization parameter, C($\theta$; $X_i, Y_i$ ) is the cost, $(X_i, Y_i)$ represents the data-point from the training set.

# Categorical cross-entropy

1. Categorical crossentropy is a loss function that is used in multi-class classification tasks.
2. It is designed to quantify the difference between two-probability distributions.
3. The categorical crossentropy loss function calculates the loss of an example by computing the following sum:

$$L(y, \hat{y}) = -\sum_{j=0}^{m}\sum_{i=0}^{n}(y_{ij} * log(\hat{y_{ij}})) \tag{2}$$

   where y is the actual value and $\hat{y}$ is the predicted value, m represents the total number of datapoints and n represents the total number of classes.
4. The categorical crossentropy is well suited to classification tasks, since one example can be considered to belong to a specific category with probability 1, and to other categories with probability 0.

# Results

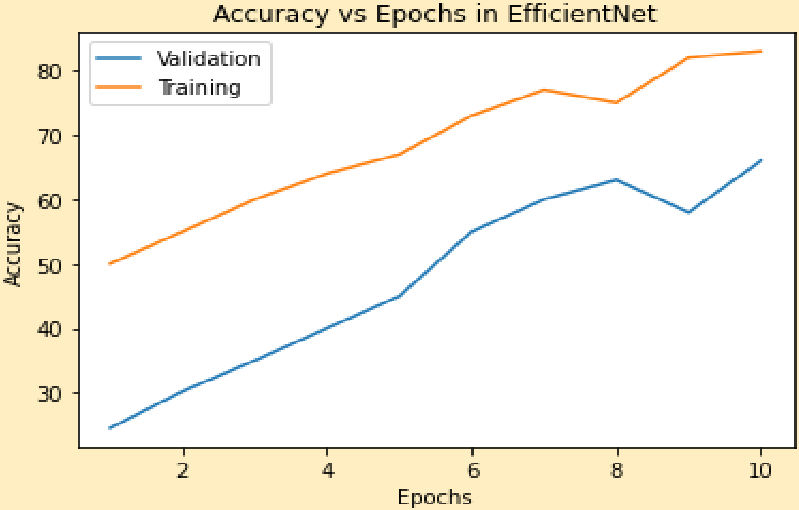


Figure: Accuracy Graph for Efficientnet

# Contd...

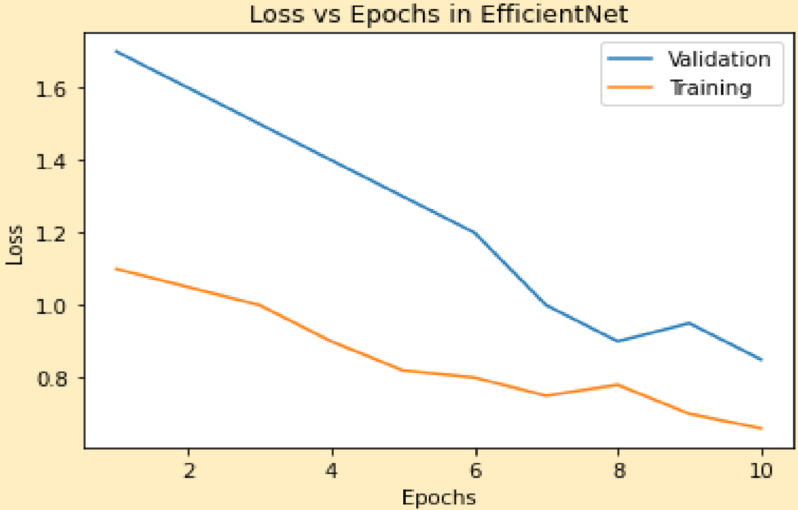


Figure: Loss Graph for Efficientnet

# Contd...

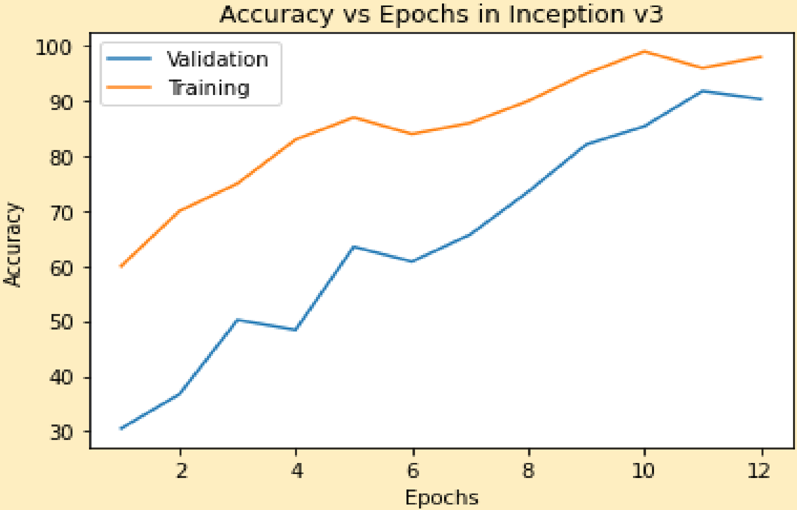


Figure: Accuracy Graph for Inception v3

# Contd...

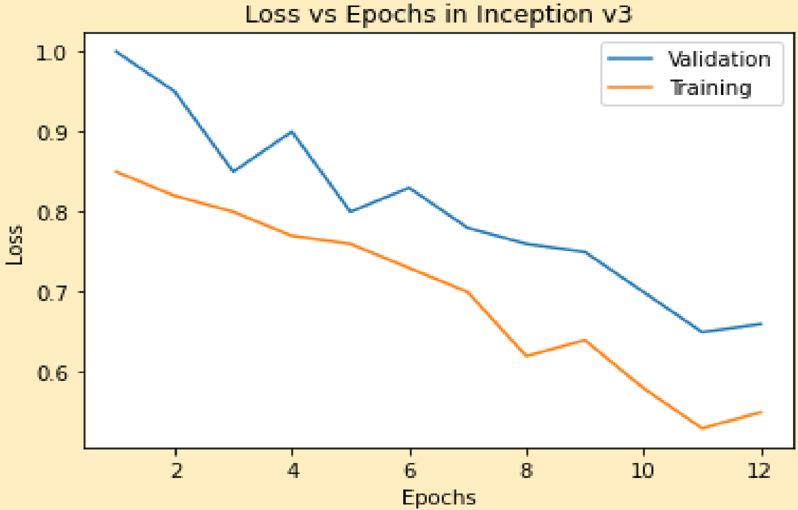


Figure: Loss Graph for Inception v3

# References


[1] N. Khatun F. T. Pinki and S. M. Islam. “âContent based paddy leaf disease recognition and remedy prediction using support vector machine”. In: *20th International Conference of Computer and Information Technology* (2019).

[2] J. P. Shah H. B. Prajapati and V. K. Dabhi. “Detection and classification of rice plant diseases”. In: *Intelligent Decision Technologies* 12 (2020).

[3] *Rice production in India*. https://en.wikipedia.org/wiki/Rice_production_in_India.

Thank You!