A PROJECT REPORT ON

RICE LEAF DISEASE DETECTION WITH CNN

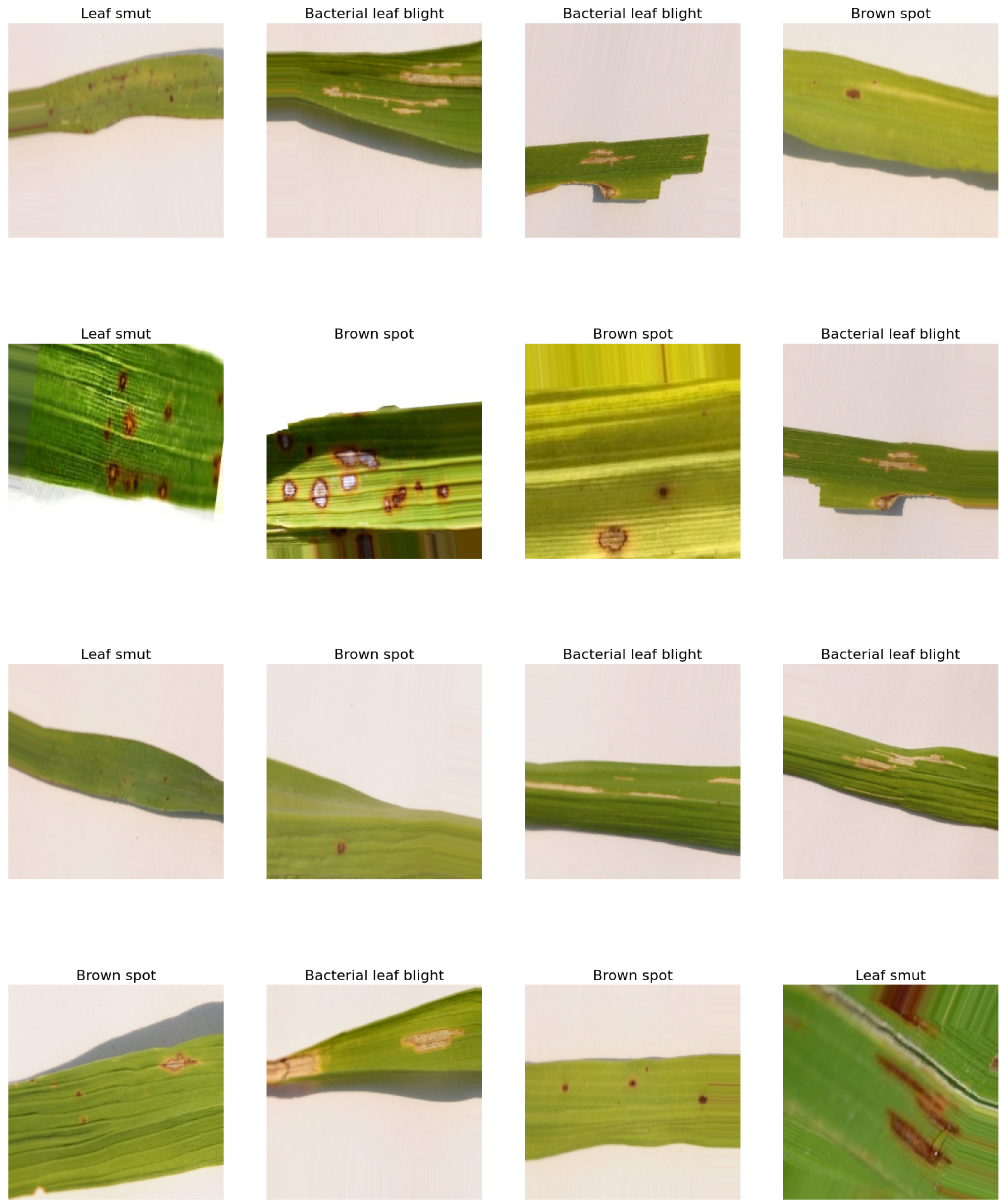
Name: Bhushan Ingale
Date: 07/10/2023
Team ID: PTID-CDS-SEP-23-1652
Project ID: PRCP-1001-RiceLeaf

Objectives

- Task 1: Prepared a complete Data Analysis report on the given data.
- Task 2: Created a model which can classify the three major attacking diseases of Rice plants like Leaf blast, Bacterial blight and Brown spot.
- Task 3: Analysed various techniques like Data Augmentation, Transfer Learning and created a report on that.

Dataset

- The Dataset consisted of 120 jpg images of disease-infected Rice leaves.
- The images were grouped into 3 classes based on the type of disease. There were 40 images in each class.
- Classes
  - Leaf smut
  - Brown spot
  - Bacterial leaf blight



Data Augmentation

- Image Data Augmentation is a technique that can be used to artificially expand the size of a training dataset by creating modified versions of images in the dataset
- Transformations include a range of operations from the field of image manipulation, such as shifts, flips, zooms, and much more.
- Image Data Augmentation is typically only applied to the training dataset, and not to the validation or test dataset. This is different from data preparation such as image resizing and pixel scaling; they must be performed consistently across all datasets that interact with the model.

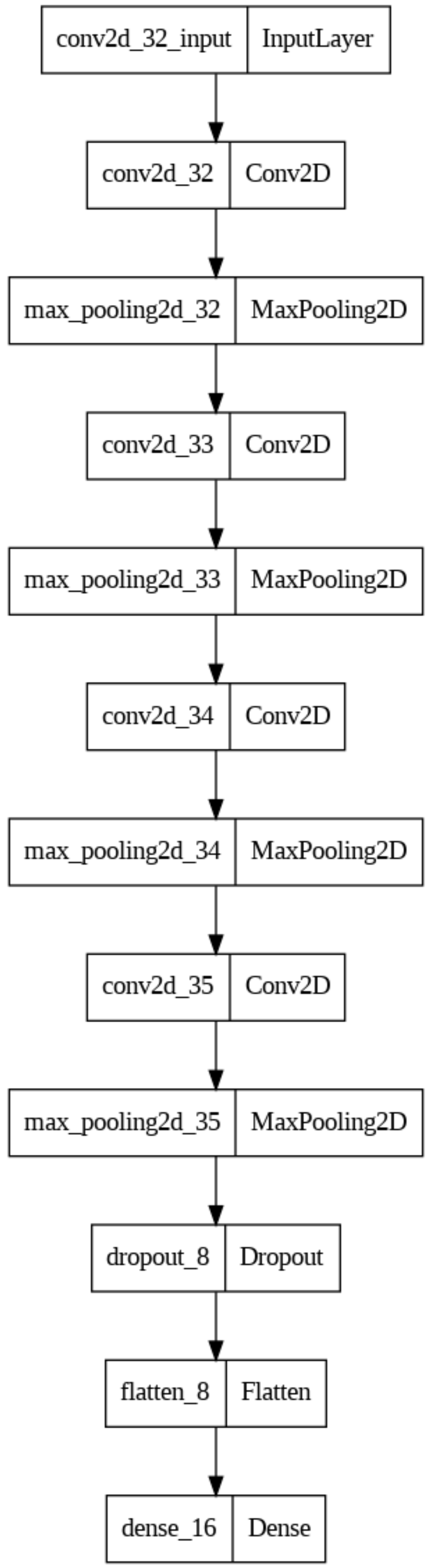
ImageDataGenerator

- This Keras' deep learning library provides the ability to use Data Augmentation automatically when training a model.
- This is achieved by using the ImageDataGenerator class.
- First, the class may be instantiated and the configuration for the types of Data Augmentation are specified by arguments to the class constructor.
- A range of techniques are supported, as well as pixel scaling methods. We will focus on five main types of Data Augmentation techniques for image data; specifically-

    1. Image shifts via the width_shift_range and height_shift_range arguments.
    2. Image flips via the horizontal_flip and vertical_flip arguments.
    3. Image rotations via the rotation_range argument
    4. Image brightness via the brightness_range argument.
    5. Image zoom via the zoom_range argument.

Baseline CNN | Sequential

- We have used a Sequential Model for implementing Convolutional Neural Networks.
- We did not perform any Batch Normalisation.
- The model consisted 11 Layers
- Input Size = (256, 256, 3)
- Kernel Size = (3, 3)
- Pool Size = (2, 2)
- Epochs = 35

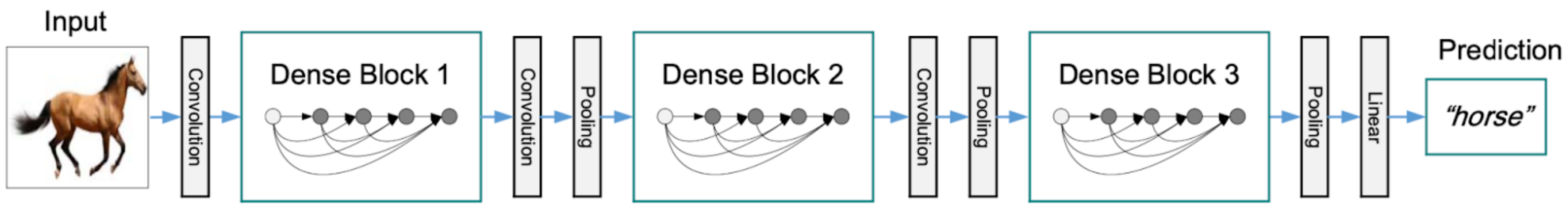

conv2d_32_input | InputLayer
conv2d_32 | Conv2D
max_pooling2d_32 | MaxPooling2D
conv2d_33 | Conv2D
max_pooling2d_33 | MaxPooling2D
conv2d_34 | Conv2D
max_pooling2d_34 | MaxPooling2D
conv2d_35 | Conv2D
max_pooling2d_35 | MaxPooling2D
dropout_8 | Dropout
flatten_8 | Flatten
dense_16 | Dense

Transfer Learning | DenseNet201

- **Dense Convolutional Network** (DenseNet) connects each layer to every other layer in a feed-forward fashion. They alleviate the vanishing-gradient problem, strengthen feature propagation, encourage feature reuse, and substantially reduce the number of parameters.

- DenseNet works on the idea that convolutional networks can be substantially deeper, more accurate, and efficient to train if they have shorter connections between layers close to the input and those close to the output.

- We have used Keras' DenseNet201 as it is a relatively small model yet very effective.

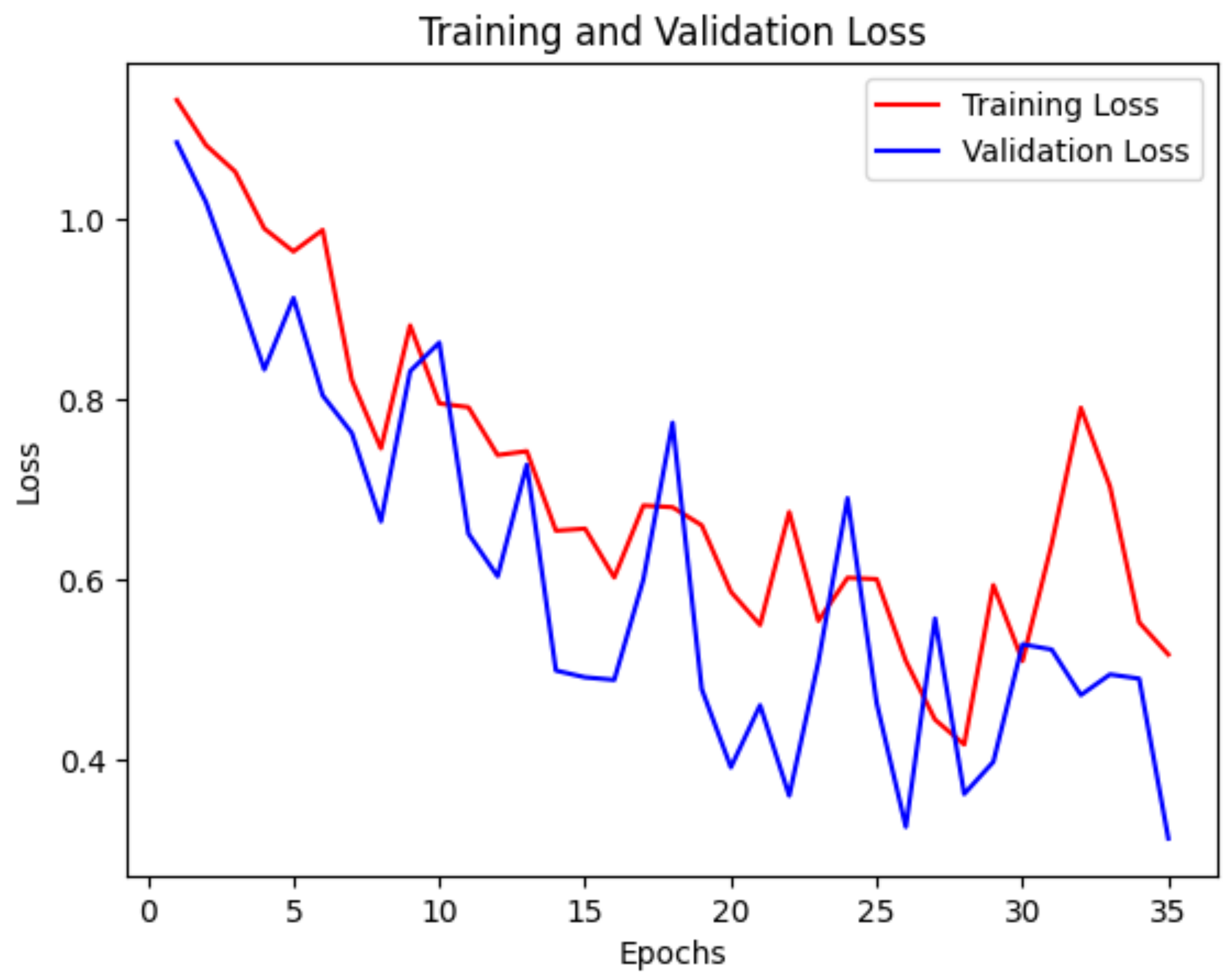


'Densely Connected Convolutional Networks' by Gao Huang, Zhuang Liu & Laurens van der Maaten
Source: https://browse.arxiv.org/pdf/1608.06993.pdf

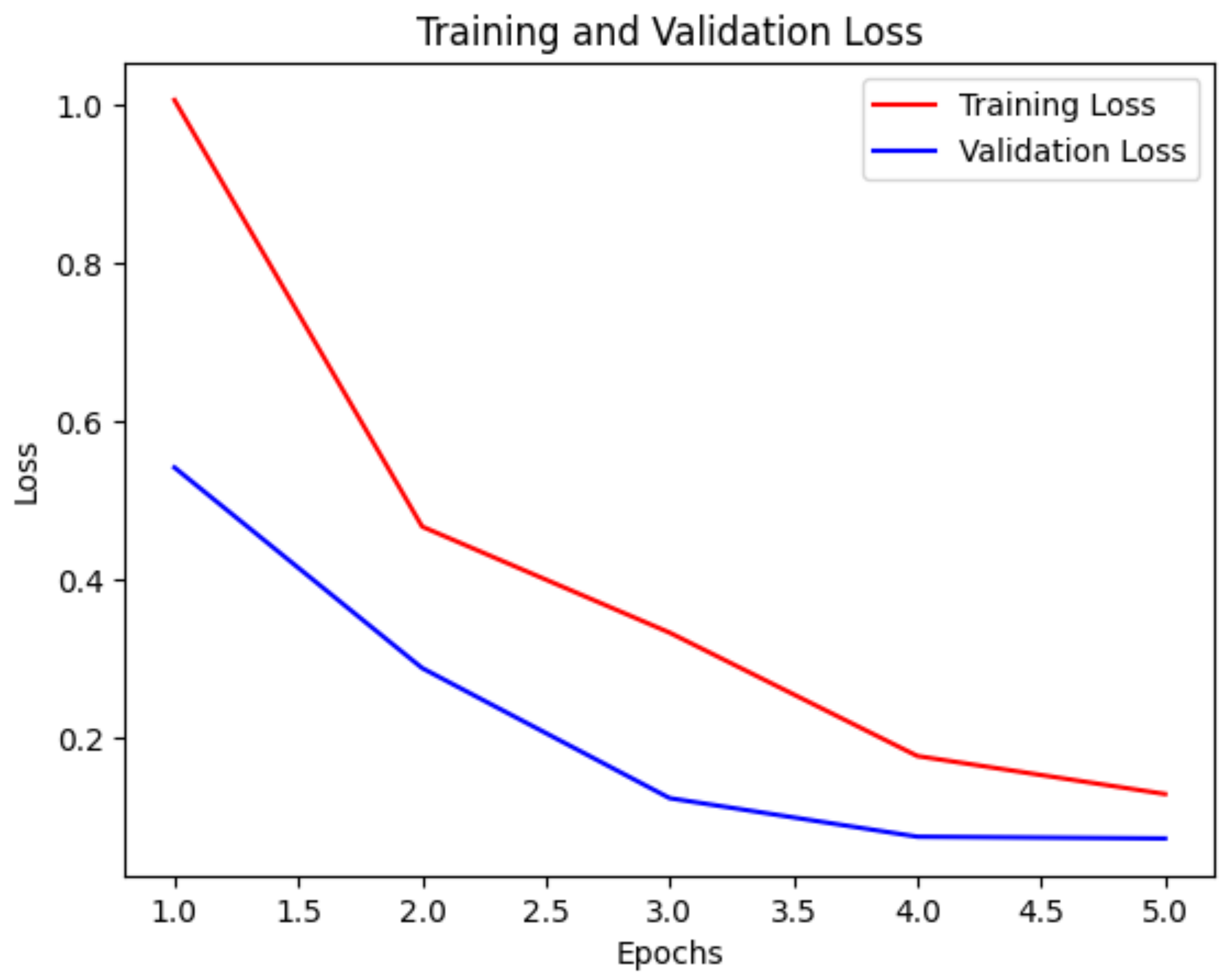
Challenges

- In order to improve accuracy, the Images had to be Augmented.
- Deciding the Number of Convolution Layers and Epochs for the Baseline Model was challenging.
- Small training dataset.
- Studying the concepts of Data Augmentation, Transfer Learning and Splitting the Image Data in Train-Val-Test-Split.

Model Comparison | Loss



Sequential CNN



CNN with DenseNet201

Model Comparison | Accuracy

	Train Accuracy	Validation Accuracy	Test Accuracy	Epochs
CNN with Sequential	76.52	89.47	80.77	35
CNN with DenseNet201	97.39	100	96.15	5

Conclusion

We built a model that was able to identify Rice Leaf Disease Classes with 96% Accuracy. We selected the model that was based on the DenseNet201 since it provided a near perfect Accuracy and had a relatively lower time of execution.