Bachelor of Science in Computer Science & Engineering

A System for Classifying Fresh and Rotten Fruits using Convolutional Neural Network

by

Tanvir Anjum Ayman

ID: 1504101

Department of Computer Science & Engineering

Chittagong University of Engineering & Technology (CUET)

Chattogram-4349, Bangladesh.

June, 2021

A System for Classifying Fresh and Rotten Fruits using Convolutional Neural Network

Submitted in partial fulfilment of the requirements for
Degree of Bachelor of Science
in Computer Science & Engineering

by

Tanvir Anjum Ayman

ID: 1504101

Supervised by

Sharmistha Chanda Tista

Assistant Professor

Department of Computer Science & Engineering

Chittagong University of Engineering & Technology (CUET)

Chattogram-4349, Bangladesh.

The thesis titled '**A System for Classifying Fresh and Rotten Fruits using Convolutional Neural Network'** submitted by ID: 1504101, Session 2019-2020 has been accepted as satisfactory in fulfilment of the requirement for the degree of Bachelor of Science in Computer Science & Engineering to be awarded by the Chittagong University of Engineering & Technology (CUET).

Board of Examiners

Chairman

Sharmistha Chanda Tista
Assistant Professor
Department of Computer Science & Engineering
Chittagong University of Engineering & Technology (CUET)

Member (Ex-Officio)

Dr. Md. Mokammel Haque
Professor & Head
Department of Computer Science & Engineering
Chittagong University of Engineering & Technology (CUET)

Member (External)

Muhammad Kamal Hossen
Associate Professor
Department of Computer Science & Engineering
Chittagong University of Engineering & Technology (CUET)

Declaration of Originality

This is to certify that I am the sole author of this thesis and that neither any part of this thesis nor the whole of the thesis has been submitted for a degree to any other institution.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. I am also aware that if any infringement of anyone's copyright is found, whether intentional or otherwise, I may be subject to legal and disciplinary action determined by Dept. of CSE, CUET.

I hereby assign every rights in the copyright of this thesis work to Dept. of CSE, CUET, who shall be the owner of the copyright of this work and any reproduction or use in any form or by any means whatsoever is prohibited without the consent of Dept. of CSE, CUET.

Signature of the candidate

Date:

Acknowledgements

I am grateful to Almighty God, who has given me the ability to complete my project and intend to perform the completion of B.Sc. Engineering degree. I am indebted to my supervisor Sharmistha Chanda Tista, Assistant Professor, Department of Computer Science and Engineering, Chittagong University of Engineering and Technology, for her motivation, proper guidance, constructive criticisms, and endless patience progress of the project. All these things contributed to my humble development as a network analysis practitioner. She supported me by providing books, conference and journal papers, and practical bits of advice. From the very beginning, Madam always encouraged me with proper guidelines, so the project never seemed a burden. All my teachers helped me with their invaluable assistance in every phase of my learning process over the course of four years. Finally, I'd like to express my appreciation to my friends, seniors, and the department staff for their valuable suggestions and assistance in completing the project.

Abstract

The production of pickles, juices, or sauces is solely dependent on fruits. To have a better quality in these products, related industries need to sort and select fresh and non-defective fruits. Manual selection of these fruits is time-consuming, error-prone, and not efficient at all. An improved automated system is needed to overcome these problems.

In this present study, the main goal is to minimize these errors and having a better time-efficient process by developing a system that can classify fresh and rotten fruits automatically. To achieve this goal, a CNN-based fruit classifier has been developed, and also a simple Web App is made to assist the classification of fruits without any hassle.

Comparison has been made among various CNN models. Further analysis also has been done to evaluate their performances. It is shown that the model has achieved a satisfactory level of correctness and high accuracy. The web application is made by using Streamlit. A user-friendly interface has been given to the application so that this model could be used beyond factories, such as small fruit shops to ensure the quality of the fruits.

Keywords—Convolutional Neural Network, Fruits, Fresh, Rotten, Classifier, Evaluation, Prediction.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

Deep learning is an artificial intelligence (AI) function that mimics the human brain's processing of data and pattern creation in order to make decisions. Deep learning is an artificial intelligence subset of machine learning that uses neural networks to learn unsupervised from unstructured or unlabeled data. Deep neural learning or deep neural network are other terms for the same thing.

In other words, deep learning is a sub type of machine learning, employs artificial neural networks at a hierarchical level to carry out the machine learning process. Artificial neural networks are constructed in the same way as the human brain, with neuron nodes connected in a web-like pattern [1].

The term "Convolutional Neural Network" refers to the network's use of the convolutional mathematical procedure. Convolutional neural networks are a type of neural network that uses convolution rather than standard matrix multiplication in at least one layer.

In comparison to other image classification methods, CNN's require very little pre-processing. This means that the network learns to optimize the filters (or kernels) by automatic learning, as opposed to hand-engineered filters in traditional techniques. This lack of reliance on prior information or human intervention in feature extraction is a significant benefit [2].

CNN is one of the machine learning-based classification methods. CNN has considerable advantages over other methods such as SVM, decision trees, and Bayesian Networks due to their high accuracy.

Convolutional Neural Networks can be used to recognize and classify things, as previously stated. So, can we use this to tell if a fruit of the same shape, such

as an apple or an orange, is fresh or rotten? Yes, this method can be used to distinguish between fresh and rotten fruits.
Classification and hand selection of fresh fruits from a large number of fruits is difficult, time-consuming, and prone to human error. Using a computer-based method for this task can lower the rate of human error while also saving time.

1.1.1 Objectives

1. To develop a CNN model that can distinguish between fresh and rotten fruits from simple images.
2. To make a comparison among various models to maximize the accuracy rate.
3. To create a simple Web App that can predict images based on it's previous learning.

1.2 Framework/Design Overview

The framework and workflow used in this project is quite simple. As this project needs to create a CNN model, we have used TensorFlow's high level API Keras as our main framework.
And about workflow, the workflow is also simple. Starting from dataset collection, then to image pre-processing and model developing, our work ends into showing classification for fruits.

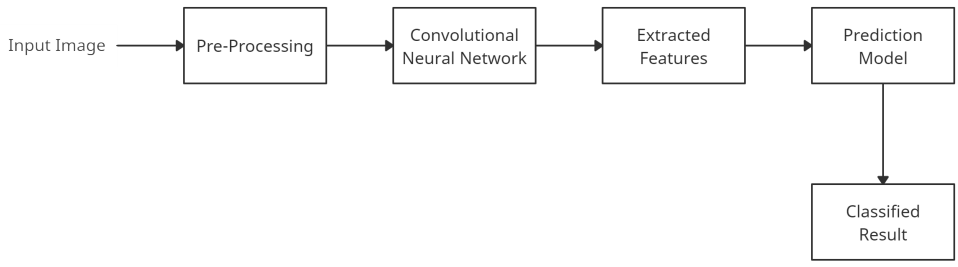


Figure 1.1: Basic Workflow.

1.3 Difficulties

Some difficulties were faced while developing the project. We tried our best to overcome all this difficulties to max out the accuracy though. The main difficulties were:

1. Dataset: First of all the dataset. It's the only organised dataset found on the internet. Although the dataset contains approximately 13600 images, but it doesn't have that much diversity.

2. Limitation of hardware: For this kind of model training, a computer with heavy graphics processing unit is needed. But the PC I am using doesn't have anything like this so I was needed to be dependent of Google Colab. Colab helped to overcome the training difficulty but it needs a stable internet connection. The training process failed several times due to internet failure.

3. Overfitting: Overfitting is a common problem in model training. Initially our model was also overfitting with the mentioned dataset.

4. Web App: We have tried to build a Web App with the most accurate model (ResNet50 in our case) but there's a limitation of free hosting. Also the storage is of just 1GB and RAM is of 500MB only. So we couldn't host our best model rather we have hosted MobileNet V2 based model in our Web App which is of comparatively lower accuracy. The ResNet50 model based Web App is hosted locally.

1.4 Applications

We may think a scenario of a fruit factory for instance. They needs a lot fruits daily. But choosing and taking non defective fruits from a huge lot is quite tiring and time consuming mostly. Besides the human error rate will be very high in this kind of cases.
This project can come handy in this type of situation. This will reduce error and

human cost for the factories and most importantly it will save the most valuable thing "Time".

1.5 Motivation

The main motivation driving this project is summarized below:

1. Developing a software based detector and classifier for the factories.
2. Reducing the error rate, human cost and saving time.
3. Available for everybody by launching a Web App.

1.6 Contribution of the thesis

The project achieves a specific set of objectives. The project's areas of contribution are summarized below:

1. Developed a CNN model using Tensorflow that can classify fruits.
2. Improving the main evaluation metrices (i.e Precision and Recall) on the same dataset.
3. Developed an easy to use web application that can classify a fruit from a given image.

1.7 Thesis Organization

The remaining parts of this report is organized as below:

1. Chapter 2 describes the background studies and some basic overview and discussion on CNN and our framework.
2. Chapter 3 discusses about the detailed methodology and workflow. This shows how the model was created and trained, the tunings and tweakings of the model and describes about other pretrained model that are used.
3. Chapter 4 shows the analysis of the results and give us a brief comparison

of the results from various models. Also accuracy of some random tests will be shown.

4. Chapter 5 provides an overview of this thesis project as well as some future recommendations.

1.8 Conclusion

This chapter provides an overview. This chapter describes the apex of the entire system, as well as the difficulties. This section also discusses the motivation for this work and contributions. The background and current state of the problem will be discussed in the following chapter.

Chapter 2

Literature Review

2.1 Introduction

The purpose of this study is to look into the challenges that were faced while creating and training various CNN models to classify the fresh and rotten fruits. This chapter discusses various applications previously done, as well as their limitations and advantages, by providing a summary of the previous study.

2.2 Related Literature Review

2.2.1 Neural Network

Neural networks are a collection of algorithms that are loosely modeled after the human brain and are designed to recognize patterns. They interpret sensory data using machine perception, labeling or clustering raw input. Neural networks assist us in clustering and classifying data. Neural Networks usually have three types of layers:

1. Input Layer: To construct the neural network, the input layer accepts large amounts of data as input. We have used a dataset containing images of three types of fruits here.

2. Hidden Layer: This layer processes data and extracts features by conducting complex computations. These layers have weights and biases that are constantly changed before the training phase is completed as part of the training. There are several weights and one bias for each neuron. The values are transferred to the output layer after they have been computed.

3. Output Layer: Using appropriate activation functions, the output layer

generates predicted output. The output can be numerical or categorical in nature [3].

2.2.1.1 Convolutional Neural Network (CNN)

CNN or Convolutional Neural Network is a form of neural network that uses a convolution layer and a pooling layer. To extract features, the convolution layer condenses an area, or a collection of elements in input data, into a smaller area. Filtering a field, which is the same as multiplying weights to an input data, is how it's done. Within a given field, the pooling layer selects the data with the highest value. These layers are responsible for extracting a key function or can be said as a feature from the input for classification [4].

2.2.2 TensorFlow

TensorFlow is a Google open-source library designed primarily for deep learning applications. Traditional machine learning is also supported. TensorFlow was originally designed for large numerical computations rather than deep learning. However, it proved to be very useful for deep learning development as well, so Google made it open source. TensorFlow accepts data in the form of multidimensional arrays with higher dimensions known as tensors. Multi-dimensional arrays come in handy when dealing with large amounts of data [5].

2.2.3 Keras

Keras is an open-source software library for artificial neural networks that offers a Python interface. Keras serves as a user interface for TensorFlow. Keras allows to define and train a deep learning model just by writing a few lines of code [6].

2.2.4 Models

Here we have used various types of CNN models including user defined and pre-trained models. A baseline model for our dataset is written using keras and trained. Also few types of pre-trained models have been used and trained on our

dataset to make a comparison among them and finding out the best classification result. The models that are used are listed below:

- Baseline Model (User defined)
- Resnet50
- MobileNet v2
- VGG16
- Inception

2.2.5 Backgrond Study

There has been enormous research being carried out to assist in the quality assessment of fruits and vegetables in agriculture or in the industry. Many applications requiring visual inspection of fruits and vegetables such as tomatoes, dates and mangoes [7] have used machine vision based systems.

Here the classification of ripe and unripe tomatoes are done by CNN but using a very small dataset [8]. But for getting a higher accuracy, we have to use a comparatively big dataset for CNN. And this is also used for only a single type of fruit, it can't be applied for various fruits of same shape.

The surface bruise of the fruits are detected here but it didn't classify the fruit [9]. It just detects the bruise on the surface but as we are trying to use our model for industry, we have to classify the state of fruits whether to select them or not.

A project that has used the same dataset from Kaggle and created a baseline model and impoved model using MobileNet v2. We have applied our own baseline model and ResNet50 and some other models on the dataset to make comparisons on the classifiers accuracy [10].

Here a bruise detection system is developed using thermal images but it is expensive to buy a thermal camera and collect some thermal image [11]. Besides it is not classifying the fruits that can be used in a industry.

In this work, surface bruise of fruits are detected using Biospeckle technique but

again, it is not classifying the fruits [12]. So it can't be used in a industry for selection the fruits.

Both SVM and CNN are used as classifier but the accuracy level of the algorithm depends on the size of dataset [13]. SVM is a popular method for classification of object using computer vision but the limitation of SVM is that it works perfectly only for a comparatively small dataset. For a huge dataset of various states of fruits or multiple fruits of same shape like apples and oranges, CNN is the best method to apply for classifying among them. A fruit recognition system is implemented using SVM to recognize the fruit but it is not the classification among fresh and rotten fruit, it just show the names of the fruits [14].

2.3 Conclusion

So, this section gives us a brief on the previous works and also the things that were used in our project. The details of the work procedure will be illustrated in the following chapter. Finally, prediction models and their results interpretation offer the work a new dimension.

2.3.1 Implementation Challenges

Some challenges were faced while creating and training the models. First of all, we have only 6 types of test and train class though this numbers are higher than the previous works. Then, as the dataset is a comparatively small dataset for CNN in spite of containing more than 12000 images, so there was a overfit problem. Thankfully all the limitations have been handled quite perfectly.

Chapter 3

Methodology

3.1 Introduction

Aim of the project was to develop a fruit classifier system using Convolutional Neural Network. This section explains the detailed work procedure. A brief on the used framework is given. Also some snapshots are added to explain the full workflow.

3.2 Overview of Framework

3.2.1 Convolutional Neural Network

A classifier assigns a class label to a data point in machine learning. An image classifier, for example, assigns a class label to the objects in a picture. A CNN, or convolutional neural network, is a form of classifier that excels at handling this challenge!
CNN that is a neural network, is a type of algorithm that recognizes patterns in data. In general, neural networks are made up of layers of neurons, each having its own set of weights and biases that can be learned. Let's dissect a CNN into its constituent parts [15] [16].

- Tensor: A tensor is analogous to an n-dimensional matrix. Except for the output layer, tensors in the CNN above will be three-dimensional.
- Neuron: A neuron is a type of function that accepts several inputs and produces a single output.
- Layer: A layer is just a group of neurons that all perform the same operation and have the same hyperparameters.

- Kernel weights and biases: While each neuron is unique, they are tweaked throughout the training phase to let the classifier to adapt to the problem and dataset.
- Differentiable score function: A CNN transmits a differentiable scoring function, which is seen as class scores in the output layer's visualization.

3.2.2 Layers of the Network

Starting with a simple image of CNN, the function of each layers will be explained.

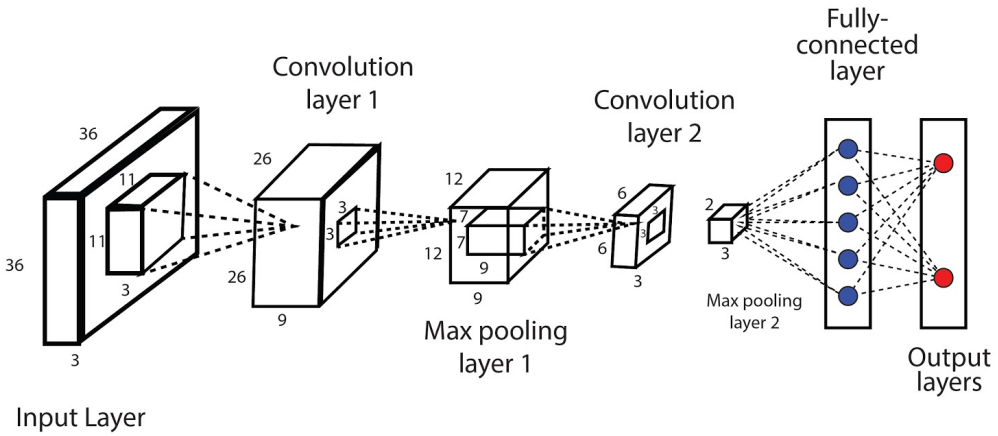


Figure 3.1: Architecture of a Convolutional Neurral Network.

3.2.2.1 Input Layer

The input image into the CNN is represented by the input layer (leftmost layer). Our input layer will have three channels as a RGB image will be used as an input. Input layers are linked to convolutional layers, which perform a variety of activities such as padding, striding, kernel operation, and other functions. This layer is considered a building block of convolutional neural networks.

3.2.2.2 Convolutional Layers

CNN is built on convolutional layers, which contain learned kernels (weights) that extract features that distinguish different images from one another. That is exactly what we want for classification! The convolutional neuron performs an elementwise dot product with a unique kernel and the output of the corresponding neuron from the previous layer. This produces as many intermediate results as there are distinct kernels. The convolutional neuron is the sum of all intermediate results combined with the learned bias.

3.2.2.3 Activation Functions

Because they are so precise, neural networks are widely used in current technologies. The best-performing CNNs now have a ridiculous number of layers that can learn more and more features.

- **ReLU (Rectified Linear unit) Activation function:** Because of their non-linearity, these ground-breaking CNNs can achieve such incredible accuracies. ReLU injects the model with much-needed non-linearity. The Rectified linear unit, or ReLU, is the most extensively used activation function today, ranging from 0 to infinity and converting all negative values to zero.

$$ReLU(x) = max(0, x)$$

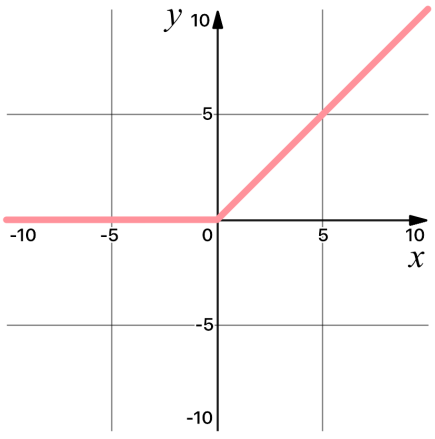


Figure 3.2: The graph of the ReLU activation function, which ignores all negative data.

- **Softmax Activation Function:** Softmax is primarily used for decision making at the last layer, i.e. the output layer. The softmax basically assigns a value to each input variable based on their weight, and the sum of these weights eventually equals 1. Both sigmoid and softmax are equally suitable for binary classification, but in the case of multi-class classification problems, we usually combine softmax and cross-entropy.

$$Softmax(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$$

3.2.2.4 Pooling Layers

Pooling layers come in a variety of shapes and sizes in different CNN architectures, but they all have the same goal: to gradually shrink the network's spatial extent, which reduces the network's parameters and overall computation. Max-Pooling is the sort of pooling used in our architecture. The Max-Pooling operation requires selecting a kernel size and a stride length during architecture design. Once selected, the operation slides the kernel with the specified stride over the input while only selecting the largest value at each kernel slice from the input to yield a value for the output. Our pooling layers are using a 2x2 kernel.

3.2.2.5 Flatten and fully connected Layers

To fit the input of a fully-connected layer for classification, this layer turns a three-dimensional layer in the network into a one-dimensional vector. A tensor of dimension 8x8x2 would, for example, be transformed into a vector of size 128. The features from the input image were retrieved by the network's previous convolutional layers, but now it's time to classify them. To classify these features, we use the softmax function, which requires a 1-dimensional input. It's also linked to the final classification model, which is referred to as a fully-connected layer.

3.3 Detailed Explanation

This section describes the methodology or the workflow that was followed at the time of implementation of the project. The main workflow is shown below-

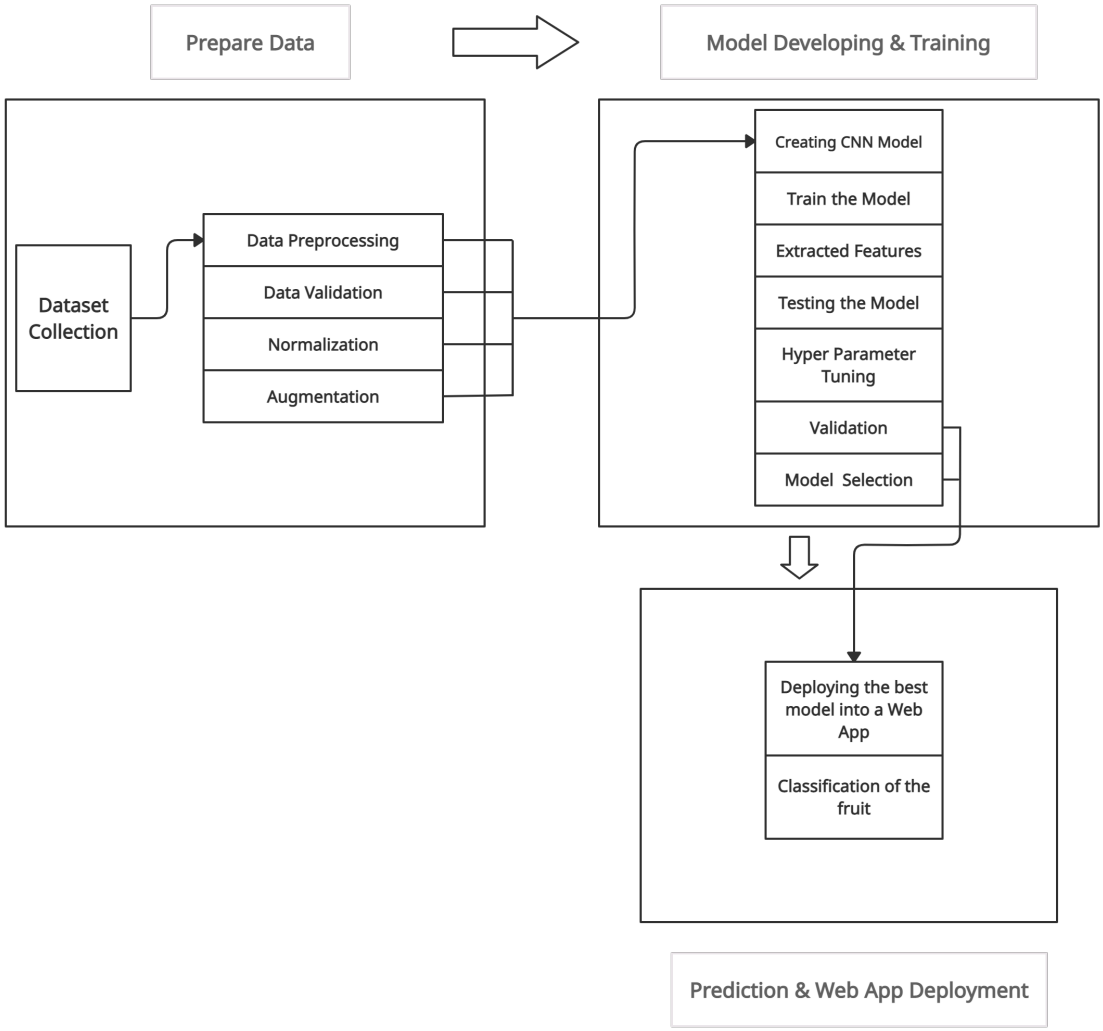


Figure 3.3: Methodology.

3.3.1 Dataset

We have collected a dataset from Kaggle [17] containing around 13600 images of fruits. The whole dataset is divided into two parts, train dataset and test dataset. Each of them contains six different types of fruits labeled as "freshapples", "freshbanana", "freshoranges", "rottenapples", "rottenbanana" and "rottenoranges". These six types will be taken as our classification classes.

3.3.2 Pre-Processing

Pre-Processing is must for developing a classifier like this because these image data need to be made suitable for a CNN model. Our dataset has also gone through a Pre-Processing step. We have used "ImageDataGenerator", a library

method included in TensorFlows's High level api "tensorflow.keras" that generates batches of tensor image data with real-time data augmentation [18].

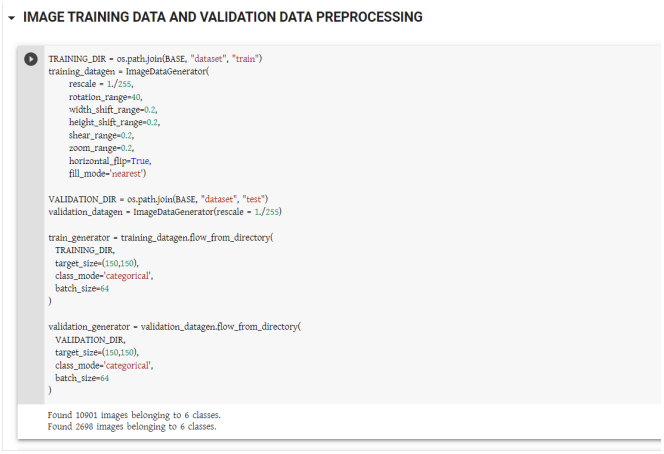


Figure 3.4: Image Training Data and Validation Data Pre-Processing.

1. **Rescale:** The ImageDataGenerator class can be used to rescale pixel values from 0-255 to the preferred range of 0-1 for neural network models. This process is called Normalization.

2. **Rotation:** The data created is randomly rotated by an angle in the range of `+rotation_range` to `-rotation_range` when the `rotation_range` parameter is specified (in degrees).

3. **Width and Height Shifting:** The `width_shift_range` is a floating point number between 0.0 and 1.0 that specifies the upper bound of the fraction of the total width by which the image will be randomly shifted to the left or right.

   The `height_shift_range` also does the same but it shifts image vertically instead of horizontaly.

4. **Shear Intensity:** The image's shape is slanted due to shear transformation. Shear transformation differs from rotation in that we fix one axis and stretch the image at a specific angle known as the shear angle. This results in a sort of 'stretch' in the image that is not visible in rotation. The angle of the slant in degrees is specified by `shear_range`.

5. **Zoom:** The `zoom_range` option generates a random zoom. A zoom of less

than 1.0 enlarges the image, while a zoom of more than 1.0 enlarges the image.

6. **Flip:** Here we have used `horizontal_flip` only. The generator will generate photos that will be horizontally flipped at random.

7. **Fill Mode:** As this augmentations are done randomly, so there's always a chance that we may have some points those don't have any value. Here, we need to fill those points with some values. We can use `fill_mode` and set the parameter to "nearest". This is the default setting, which selects the nearest pixel value and repeats the process for all empty values.

8. **Flow From Directory**: This `flow_from_directory` generates batches of augmented data based on the path of a directory.

9. **Class Mode:** The type of label arrays that are returned is determined by this parameter. Here we have used `class_mode="categorical"` will be 2D one-hot encoded labels.

3.3.3 Feature Extraction

Feature extraction is a dimensionality reduction procedure that reduces a large set of raw data into smaller groupings for processing [19]. The enormous number of variables in these enormous data sets necessitates a lot of computational resources to process. Feature extraction refers to strategies for selecting and/or combining variables into features in order to reduce the amount of data that needs to be processed while still accurately and thoroughly characterizing the original data set. We can see the features extracted by the model from an image from output of each layers. Extracted feature from an image of a banana will be something like this

3.3.4 Convolutional Neural Network and Prediction Model

After doing all the previous steps, now we have to select a suitable CNN model that gives us the best result on the used dataset. We have developed a simple CNN model by ourselves and named it as "Baseline Model". We have checked

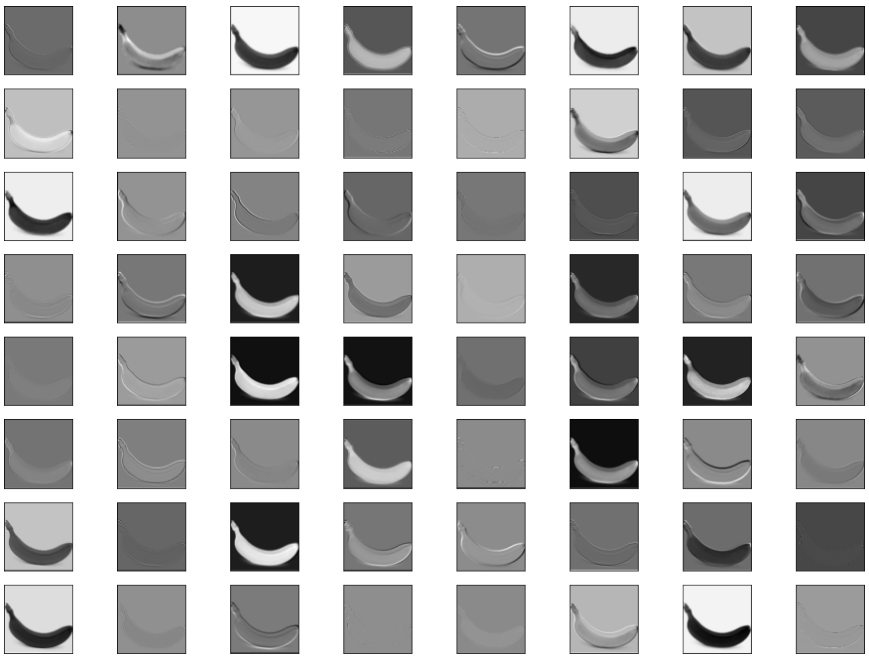


Figure 3.5: Features extracted from a sample banana image.

the accuray and loss of this model. Also we have used some other pretrained model for transfer learning mentioned before. All of the comparisons among all this models will be discussed in the results section. A brief illustration on our own Baseline Model will be shown here.

3.3.4.1 Baseline Model

▾ BASELINE MODEL

▾ CONVOLUTIONAL NEURAL NETWORK DECLARATION

```
[ ]  model = Sequential()
     model.add(Conv2D(64, (3,3), activation='relu', input_shape=(150, 150, 3)))
     model.add(MaxPooling2D(2, 2))
     # The second convolution
     model.add(Conv2D(128, (3,3), activation='relu'))
     model.add(MaxPooling2D(2,2))
     # The third convolution
     model.add(Conv2D(256, (3,3), activation='relu'))
     model.add(MaxPooling2D(2,2))
     # The fourth convolution
     model.add(Conv2D(512, (3,3), activation='relu'))
     model.add(MaxPooling2D(2,2))
     # Flatten the results to feed into a DNN
     model.add(Conv2D(512, (3,3), activation='relu'))
     model.add(MaxPooling2D(2,2))

     model.add(Flatten())
     model.add(Dropout(0.5))


     model.add(Dense(512, activation='relu'))
     model.add(Dense(64, activation="relu"))
     model.add(Dense(6, activation='softmax'))
```

Figure 3.6: Baseline Model Implementation.

A **sequential** approach is suited for a simple stack of layers with one input tensor

and one output tensor for each layer. In our model there are four **convolutional** layers. Each of them having a different number of (3,3) filters. The higher the number, the smaller the extracted feature will be. We have used "ReLU" as activation function in these convolutional layers. This activation function helps us to set the negative numbers generated while applying filters to zero as it gives a output from 0 to x. The input image resolution is set to 150 pixels by 150 pixels and the channel is set as 3 as the inputs are RGB images.

Pooling is responsible for lowering the complicated feature's spatial size. The output of a neuron cluster at one layer is pooled and combined into a single neuron at the following layer. The maximum value from each cluster of neurons from the previous layer is used in max pooling. We have used a (2,2) filter here.

Flattening is the process of converting data into a one-dimensional array for input into the next layer. The output of the convolutional layers is flattened to create a single long feature vector. It is also linked to the final classification model, which is referred to as a fully-connected layer.

Dropout layers are used to simply avoid the overfit while training a model. The range of rate lies into 0 to 1. 0.5 is set as dropout.

The **dense layers** are generally fully connected layers. These layers are connected deeply, this means that each neuron in the dense layer receives input from all neurons in the layer before it. Dense layers are usually used to change the dimensions of vector.

The summary of our model can be shown as below:

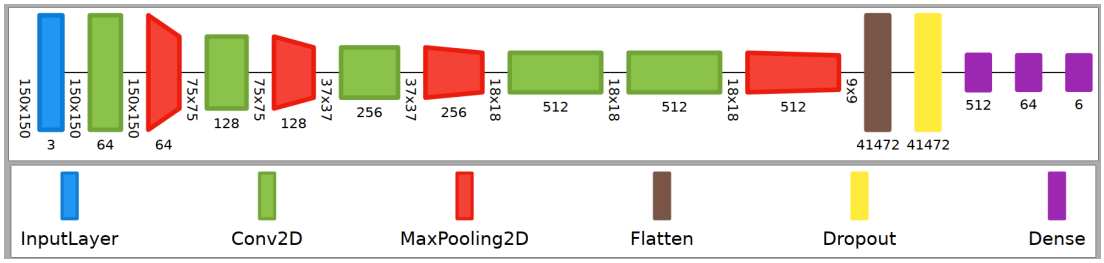
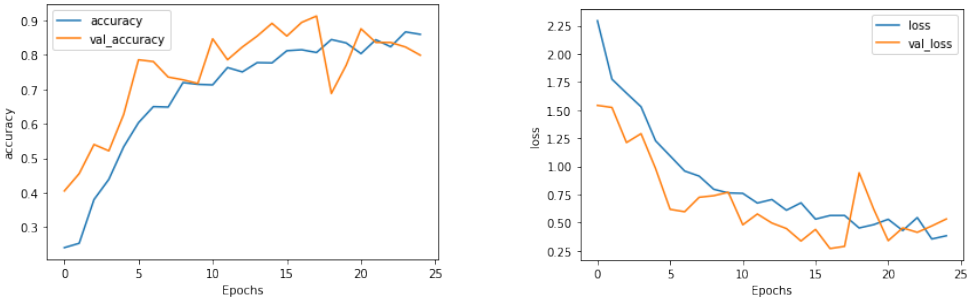


Figure 3.7: Summary of the model.

3.3.5 HyperParameter Tuning

Hyperparameter optimization or tuning is the challenge of selecting a collection of appropriate hyperparameters for a learning algorithm in machine learning. A

hyperparameter is a value for a parameter that is used to influence the learning process. Other factors, such as node weights, are, on the other hand, learned. Initially we were not getting the best results from our model. As it's a simple CNN classifier and the dataset is also small comparatively so a lot of changes in hyperparameters were needed to achieve the best accuracy. Previously the accuracy and loss of our baseline model was something like that:



(a) Accuracy (b) Loss

Figure 3.8: Initial Accuracy & Loss.

1. **Image input size:** Selecting the optimal size for input image was our first challenge. We tried by changing various input image size. The model is now working on a image size of 150x150 pixel.

2. **Number of Conv2D layers:** We experimented by changing the numbers of Conv2D layers and found that the model works best when it has four Conv2D layers. If we use more than that, then the model overfits with the dataset and learns unnecessary features.

3. **Conv2D filter size:** The higher the filter size, the detailed features are learned by the model. We have used filters of size starts from 64 and upto 512. If we go more than that, the model overfits.

4. **Dropout:** Dropout is also used to reduce the overfit problem. Dropout can be ranged from 0 to 1. We have used 0.5 here.

5. **Optimizer:** Previously we have used Stochastic Gradient Descent (SGD) as our optimizer but now we are using Adam as optimizer cause Adam performs better in classfication than SGD.

6. **Learning rate scheduler and Callbacks declaration:** Learning rate

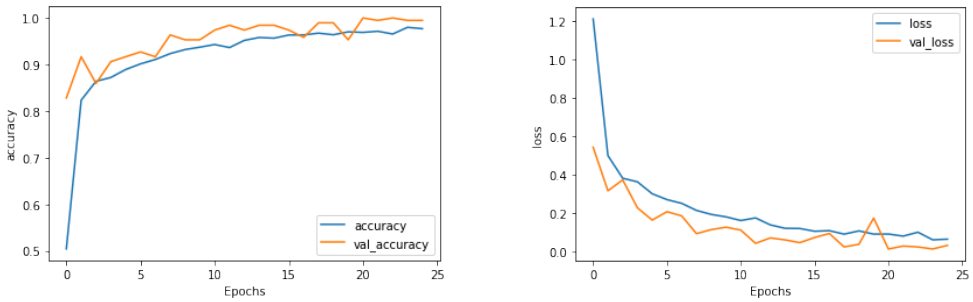
Figure 3.9: Callbacks declaration to save the best weight.

works on two parameters, one of them is scheduler. This actually takes the epoch number and current learning rate as input and returns a new learning rate as output. When training a model, lowering the learning rate as the training advances is often beneficial. We have used a user defined function that reduces the learning rate at each epoch exponentially.

We have declared a callbacks method which monitors the validation accuracy and saves only when the validation accuracy is maximum.

7. **Number of epochs:** We trained the model from five epochs to fifteen epochs and found the best accuracy by training it with seven epochs.

By applying this kind of tuning the accuracy and loss of our baseline model improved gradually.

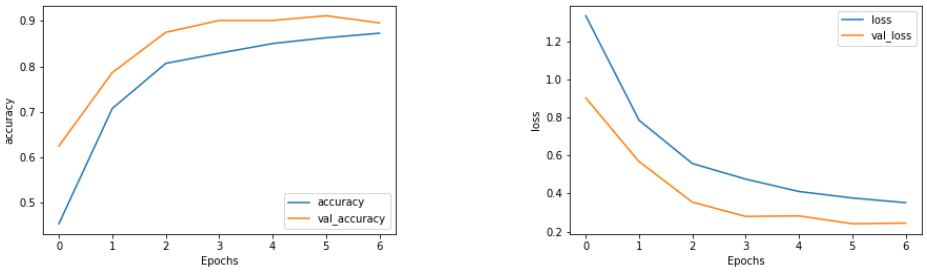


(a) Accuracy (b) Loss

Figure 3.10: Intermediate Accuracy & Loss.

3.3.6 Prediction

After training the model with our train dataset and validate the loss and accuray using test dataset, some sample images are used to check output of or classifier. We have also created a WEB APP based on streamlit and used this trained



(a) Accuracy (b) Loss

Figure 3.11: Final Accuracy & Loss.

model's weight to predict the class of an image. It supports one image at a time of "PNG", "JPG" or "JPEG" format and can predict it's class.

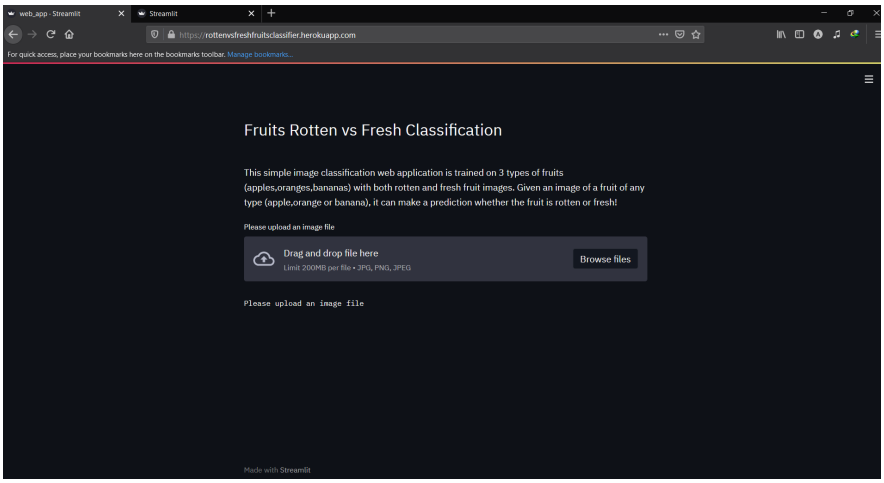


Figure 3.12: Web App interface.

3.3.7 Improvement

The baseline model that is described so far doesn't achieve the best available result for our dataset. This can be for various reason. First of all this is a simple CNN classifier, secondly the dataset doesn't have diversity of images. So we have tried to train some pre-trained model with our dataset and improve the result. The pre-trained models are:

1. **Resnet50:** ResNet50 is a ResNet variation of 48 Convolution layers, 1 Max-Pool layer, and 1 Average Pool layer. There are 3.8 x 109 floating point operations in it. It's a popular ResNet model. Because of the framework that ResNets presented it was made possible to train ultra deep neural networks and by that i mean that i network can contain hundreds or thousands

of layers and still achieve great performance. It was trained on a dataset containing over 14 millions of images. [20]

2. **MobileNetV2:** MobileNet try for the highest level of accuracy while using the least amount of memory and computer power possible. As a result, they are a particularly fast network family to utilize for image processing. MobileNet models use depthwise separable convolutions, which divide a normal convolution into two parts: a depthwise convolution that convolves a single filter with each input channel, and a pointwise convolution that does a weighted sum of the convolution outputs. The depthwise separable convolutions can express more features with fewer convolution parameters by employing weighted sums [21].

3. **VGG16:** VGG16 is a convolutional neural network model that achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes [22].

4. **Inception:** On the ImageNet dataset, Inception v3 is a commonly used image recognition model that has been demonstrated to achieve higher than 78.1 percent accuracy. The model represents the result of several ideas explored over time by a number of researchers [23].
   The performance comparison of all the models will be discussed in the next chapter.

3.3.8 Implementation

All the steps of our workflow has been described in the above sections. A simple CNN model is implemented using TensorFlow's high level API. A simple WEB app is developed using streamlit which can predict the class of a fruit in real time.

3.4 Conclusion

We have described the whole framework and all the steps in this chapter. We described about CNN and how we have built an simple CNN model to predict the class of a fruit. We also create a WEB app in addition to our project so that it

can be used by anyone in real life. As mentioned earlier, some other pre-trained models also used to maximize the accuracy. The comparisons among all these models will be shown in the next chapter. An analysis of the accuracy and loss of all the models will be shown also.

Chapter 4

Results and Discussions

4.1 Introduction

We are going to show the prediction results of our baseline model in this section and also we will analysis the performance of our model. Besides, as mentioned earlier, we have tried to maximize the accuracy rate for our project that's why we have used some pre trained model for transfer learning. We will make comparisons among the performance of all the models.

4.2 Dataset Description

This dataset is originated from Kaggle. Our dataset has two sub categories. The train data and the test data. Train data was used to train the models and test data was for validating the results. Each sub categories has six classes which are also our prediction classes. There are 10901 images in the train class and 2698 images in the test class and 13599 images in total. Some saple images of dataset is added here if figure 4.1 and 4.2

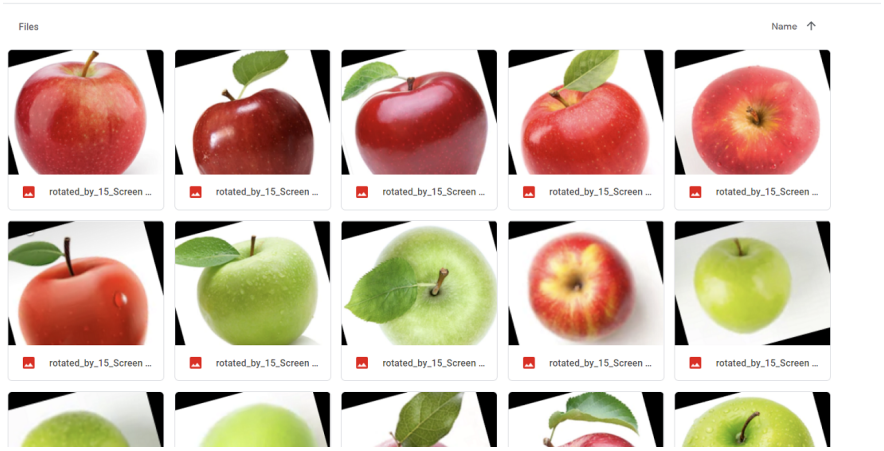


Figure 4.1: Dataset Sample (1).

Figure 4.2: Dataset Sample (2).

4.3 Impact Analysis

We have tried to build an automated system for monitoring fruits' quality. Fruits' quality can be checked by this project and this will help to assure quality control of an organization.

Besides this will help a factory to reduce human cost and errors, as well as it will save time.

4.3.1 Social and Environmental Impact

This project can come handy for people of all class in a society. As there is a Web App part, so anyone can take advantage of this without knowing what is going on in background. But he can use to classify from a huge number of fruits. As this is a classifier project, so this can be used to select rotten fruits from a huge number of fruits and if this rotten fruits can be dumped in a proper way after removing from the lot, then it will help to prevent environmental pollution.

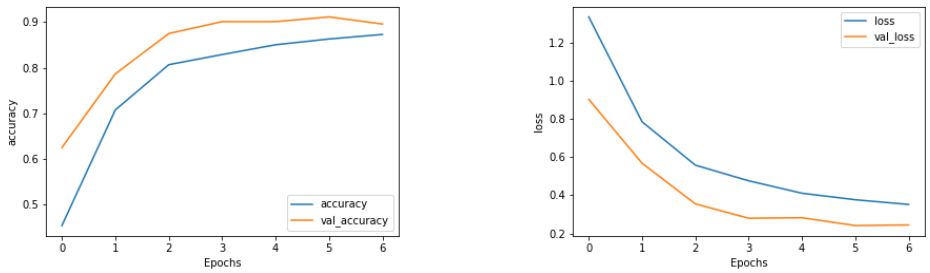
4.3.2 Ethical Impact

This project has some ethical consequences too. For instance, we can think of a fruit selling store. If the quality of fruits can be monitored in the store, then the seller won't be able to sell bad/rotten fruits to people.

4.4 Evaluation of Framework

Five models are taken into consideration. The final accuracy and loss graph will be shown for the models.

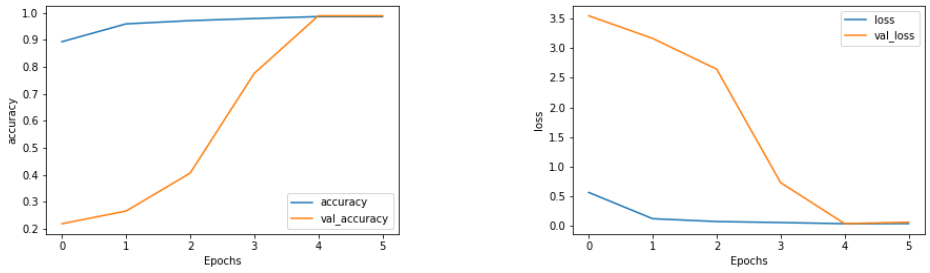
1. **Baseline Model:** The final accuracy, loss for the model is shown in figure 4.3



(a) Accuracy (b) Loss

Figure 4.3: Final Accuracy & Loss of Baseline model.

2. **ResNet50 Model:** The final accuracy, loss for the model is shown in figure 4.4



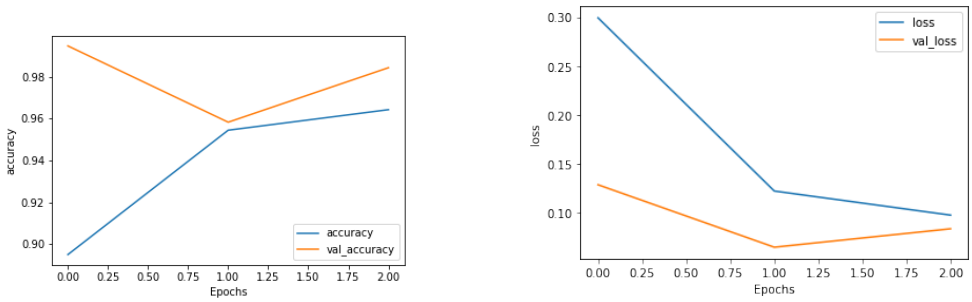
(a) Accuracy (b) Loss

Figure 4.4: Final Accuracy & Loss of ResNet50 model.

3. **MobileNetV2 Model:** The final accuracy, loss for the model is shown figure 4.5

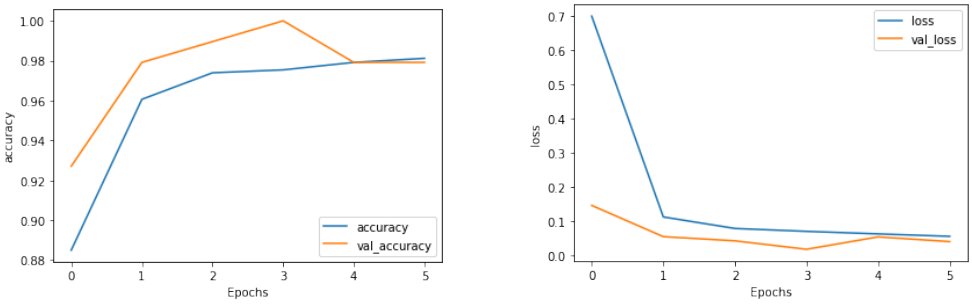
4. **Inception V3 Model:** The final accuracy, loss for the model is shown figure 4.6

5. **VGG16 Model:** The final accuracy, loss for the model is shown in figure 4.7



(a) Accuracy (b) Loss

Figure 4.5: Final Accuracy & Loss of MobileNetV2 model.



(a) Accuracy (b) Loss

Figure 4.6: Final Accuracy & Loss of InceptionV3 model.

A combined overview of loss and accuracy of all the models can be shown like this in figure 4.8 and figure 4.9

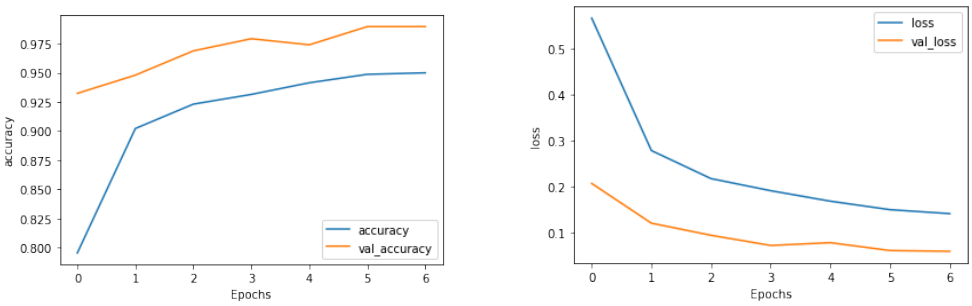
4.5 Evaluation of Performance

4.5.1 Metrices used for evaluation

AUC - ROC curve: The AUC - ROC curve is a performance metric for classification problems with varying threshold settings. AUC represents the degree or measure of separability, while ROC is a probability curve. It indicates how well the model can distinguish between classes [24].

Confusion Matrix: An N x N matrix is used to evaluate the performance of a classification model, where N is the number of target classes. The matrix compares the actual goal values to the machine learning model's predictions [25].

The table of Precision, Recall, F1 Score and Accuracy of all the models will be shown in this section. Besides we will add confusion matrix and AUC - ROC



(a) Accuracy (b) Loss

Figure 4.7: Final Accuracy & Loss of VGG16 model.

Figure 4.8: Epoch Accuracy.

curve for each of the model.

1. **Baseline Model:** Show in table 4.1 and figure 4.10

	Precision	**Recall**	**F1 Score**	**Support**
freshapples	0.86	0.96	0.90	395
freshbanana	0.96	0.98	0.97	381
freshoranges	0.90	0.96	0.93	388
rottenapples	0.86	0.85	0.85	601
rottenbanana	0.98	0.96	0.97	530
rottenoranges	0.93	0.79	0.85	403
accuracy		0.91		2698
macro average	0.91	0.92	0.91	2698
weighted average	0.91	0.91	0.91	2698

Table 4.1: Performance analysis of Baseline model.

2. **Resnet50 Model:** Shown in 4.2 and figure 4.11
3. **MobileNetV2 Model:** Shown in table 4.3 and figure 4.12
4. **Inception V3 Model:** Shown in table 4.4 and figure 4.13
5. **VGG16 Model:** Shown in table 4.5 and figure 4.14

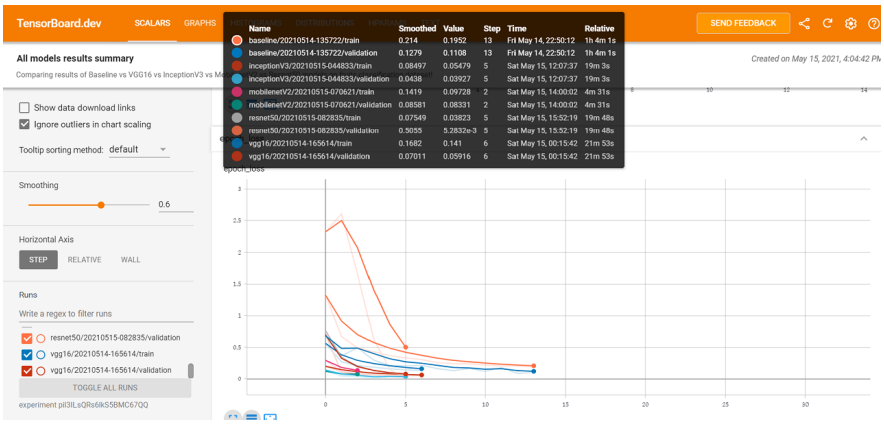
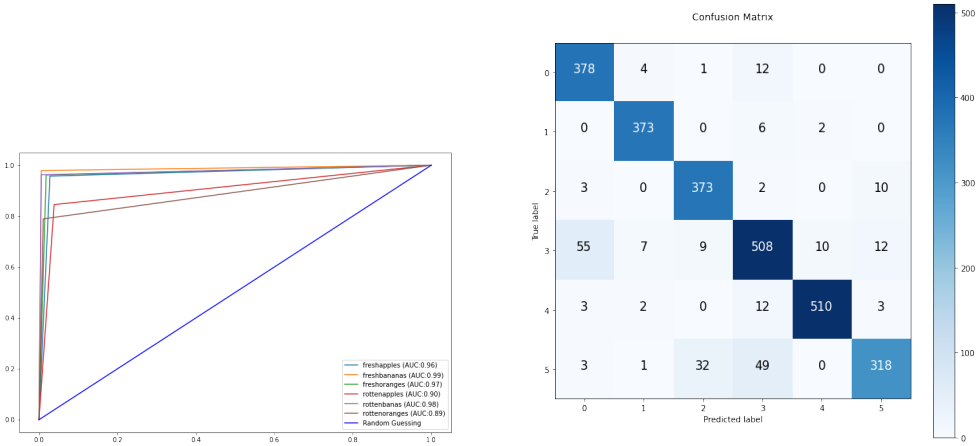


Figure 4.9: Epoch Loss.



(a) ROC-AUC Curve

(b) Confusion Matrix

Figure 4.10: ROC-AUC Curve & Confusion Matrix of Baseline model.

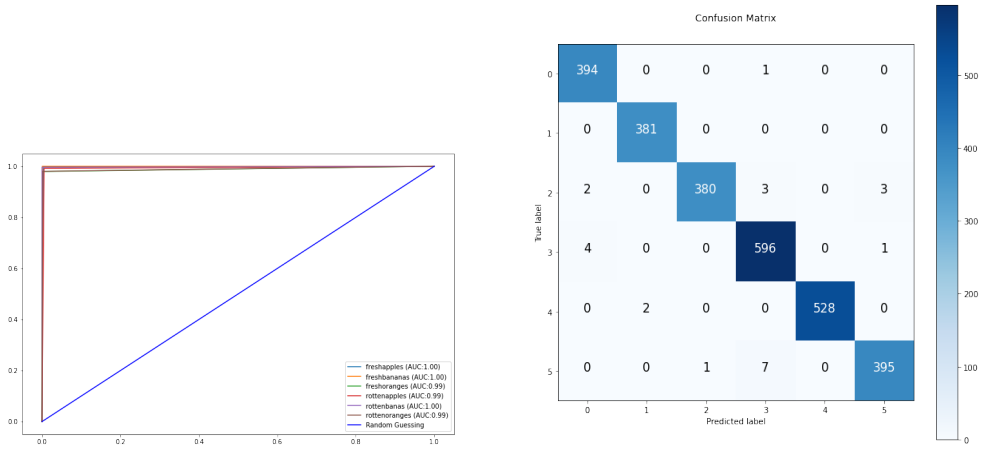
4.5.2 Overall Performance

Overall performance of all the models can be shown through table 4.6 We can see ResNet50 and Inception V3 is performing the best on our dataset. The validation accuracy is very high for both of them. But if we have a look on the ROC-AUC curve for them, we can see a better performance from ResNet50 model. So, performance can be maximized by using ResNet50 model.

Here's a comparison between the precision and recall of our ResNet50 model and MobileNetV2 model from reference [10] in table 4.7 and table 4.8.

	Precision	Recall	F1 Score	Support
freshapples	0.98	1.00	0.99	395
freshbanana	0.99	1.00	1.00	381
freshoranges	1.00	0.98	0.99	388
rottenapples	0.98	0.99	0.99	601
rottenbanana	1.00	1.00	1.00	530
rottenoranges	0.99	0.98	0.99	403
accuracy		0.99		2698
macro average	0.99	0.99	0.99	2698
weighted average	0.99	0.99	0.99	2698

Table 4.2: Performance analysis of ResNet50 model.



(a) ROC-AUC Curve

(b) Confusion Matrix

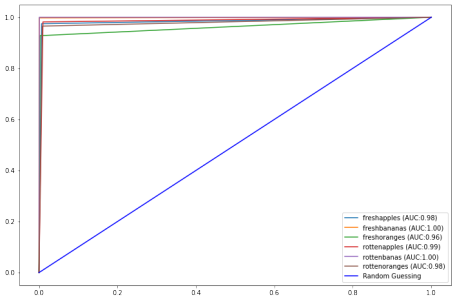
Figure 4.11: ROC-AUC Curve & Confusion Matrix of ResNet50 model.

4.6 Prediction

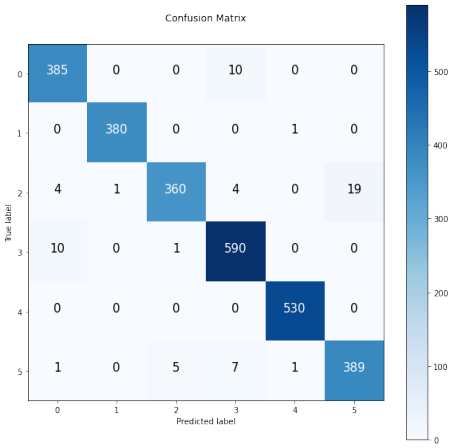
We have tried with some random images to check the prediction results. We have marked the fresh fruits as black and rotten fruits as red in our model output. Sample outputs are shown in figure 4.15, 4.16, 4.17 and 4.18.

4.7 Conclusion

From the discussions in this section, it can be shown the accuracy and correctness of our model is pretty high. This study was related to the previous literature. The performance of the baseline model was very much moderate but we have achieved a better performance result as well as prediction results using the dataset and by improving our model with some pre-trained models.

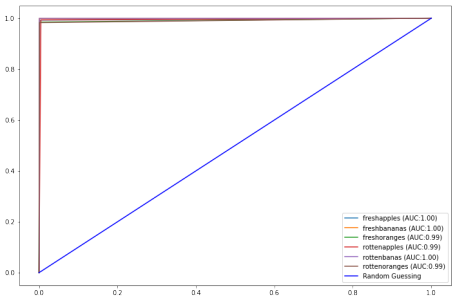


(a) ROC-AUC Curve

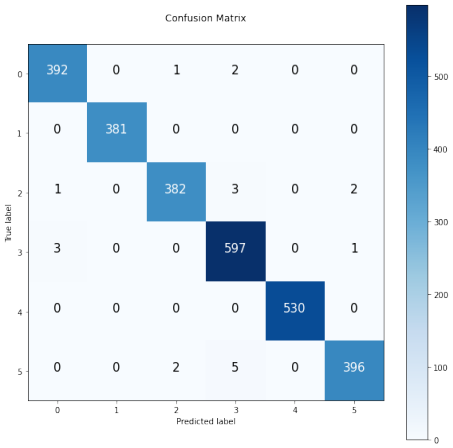


(b) Confusion Matrix

Figure 4.12: ROC-AUC Curve & Confusion Matrix of MobileNetV2 model.

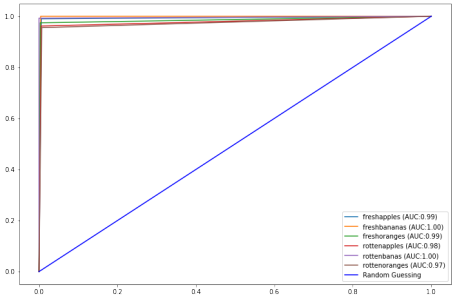


(a) ROC-AUC Curve

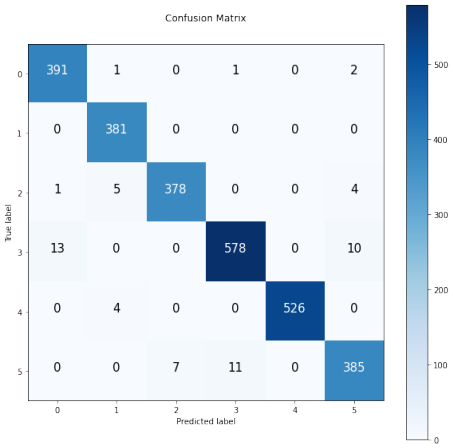


(b) Confusion Matrix

Figure 4.13: ROC-AUC Curve & Confusion Matrix of InceptionV3 model.



(a) ROC-AUC Curve



(b) Confusion Matrix

Figure 4.14: ROC-AUC Curve & Confusion Matrix of VGG16 model.

	Precision	Recall	F1 Score	Support
freshapples	0.96	0.97	0.97	395
freshbanana	1.00	1.00	1.00	381
freshoranges	0.98	0.93	0.95	388
rottenapples	0.97	0.98	0.97	601
rottenbanana	1.00	1.00	1.00	530
rottenoranges	0.95	0.97	0.96	403
accuracy		0.98		2698
macro average	0.98	0.97	0.98	2698
weighted average	0.98	0.98	0.98	2698

Table 4.3: Performance analysis of MobileNetV2 model.

	Precision	Recall	F1 Score	Support
freshapples	0.99	0.99	0.99	395
freshbanana	1.00	1.00	1.00	381
freshoranges	0.99	0.98	0.99	388
rottenapples	0.98	0.99	0.99	601
rottenbanana	1.00	1.00	1.00	530
rottenoranges	0.99	0.98	0.99	403
accuracy		0.99		2698
macro average	0.99	0.99	0.99	2698
weighted average	0.99	0.99	0.99	2698

Table 4.4: Performance analysis of InceptionV3 model.

	Precision	Recall	F1 Score	Support
freshapples	0.97	0.99	0.98	395
freshbanana	0.97	1.00	0.99	381
freshoranges	0.98	0.97	0.98	388
rottenapples	0.98	0.96	0.97	601
rottenbanana	1.00	0.96	1.00	530
rottenoranges	0.96	0.99	0.96	403
accuracy		0.98		2698
macro average	0.98	0.98	0.98	2698
weighted average	0.98	0.98	0.98	2698

Table 4.5: Performance analysis of VGG16 model.

Name	Accuracy
Baseline Model	91%
ResNet50 Model	99%
Inception V3 Model	99%
MobileNet V2 Model	98%
VGG16 Model	98%

Table 4.6: Overall Accuracies of all model.

	Precision	Recall	Support
freshapple	0.98	1.00	395
freshbanana	0.99	1.00	381
freshoranges	1.00	0.98	388
rottenapple	0.98	0.99	601
rottenbanana	1.00	1.00	530
rottenoranges	0.99	0.98	403
		Total test data	2698

Table 4.7: Precision and Recall of ResNet50 Model.

	Precision	Recall	Support
freshapple	0.91	0.98	395
freshbanana	0.99	0.98	381
freshoranges	0.99	0.95	388
rottenapple	0.98	0.95	601
rottenbanana	0.97	0.99	530
rottenoranges	0.98	0.96	403
		Total test data	2698

Table 4.8: Precision and Recall of MobileNet50 Model from Reference [10]

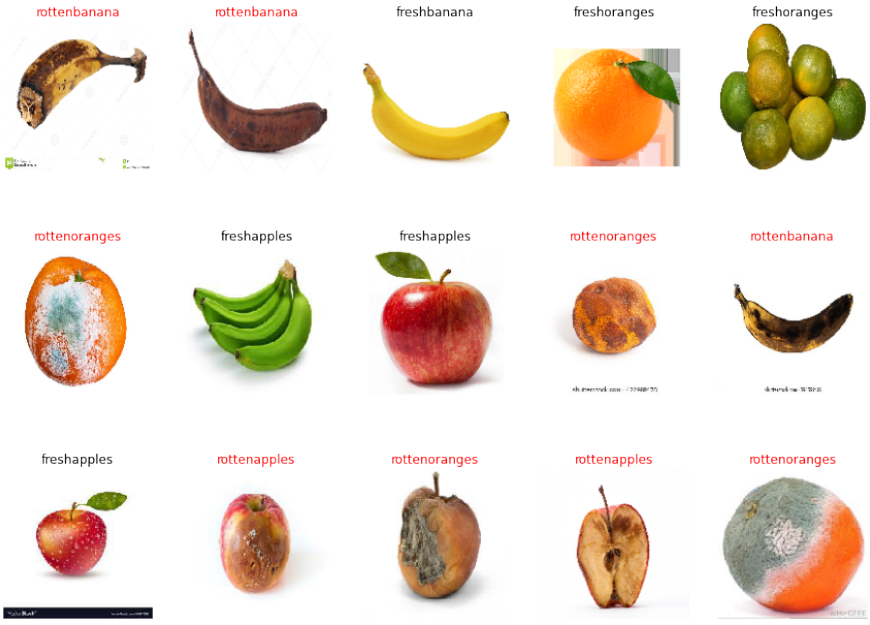


Figure 4.15: Output in model.

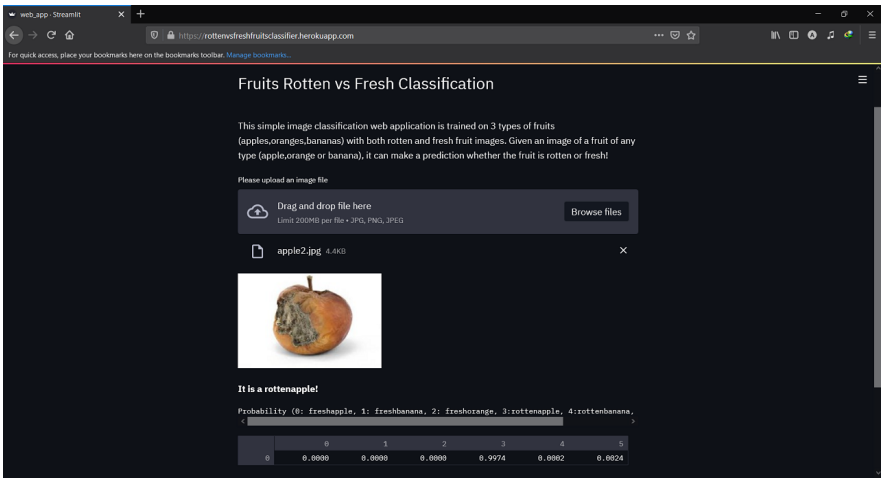


Figure 4.16: Web app output (1).

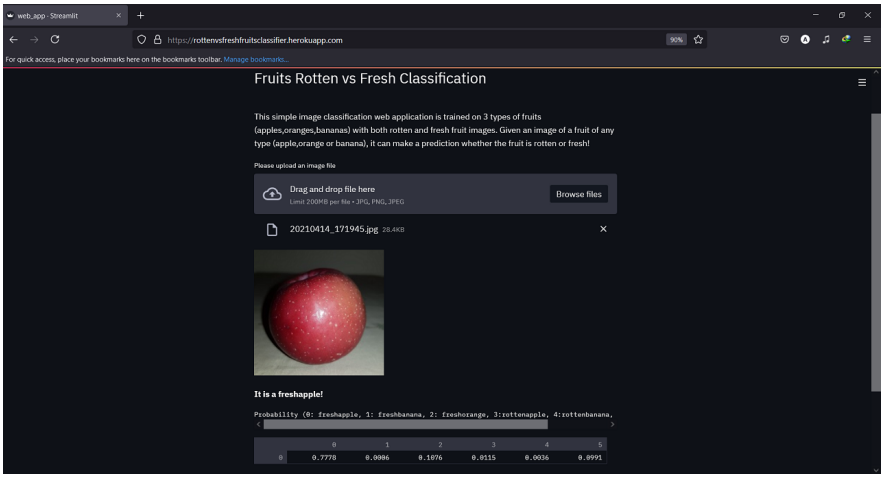


Figure 4.17: Web app output (2).

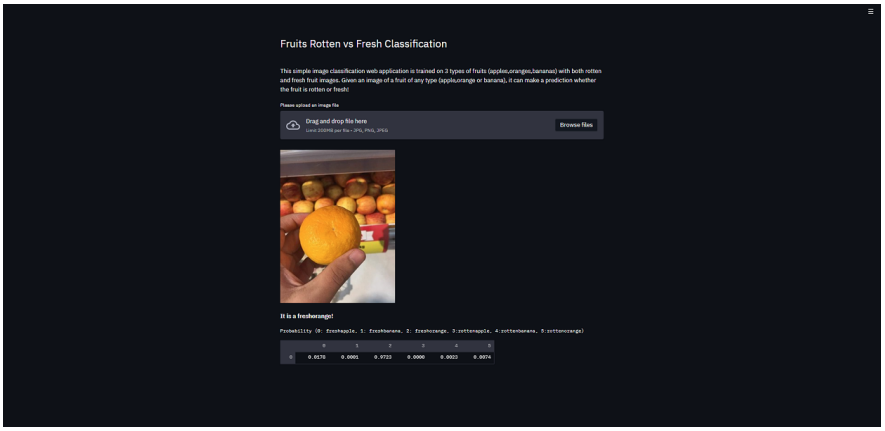


Figure 4.18: Web app output (3).

Chapter 5

Conclusion

5.1 Conclusion

Manual selection of fruits from a huge lot is not that efficient and at the same time tiring and time consuming. Wrong choices of fruits will affect a factory greatly in long run. We have tried to develop a CNN based model to detect and classify fresh and rotten fruits. Although the baseline model performed moderately on a comparatively small dataset, but we got our highest accuracy by using some pre-trained models. This is because even though the dataset have around 13600 images but there is not that diversity of images. On the other hand, the pre-trained models are trained on huge datasets consisting of thousands of classes. So their abilities to find the smallest pattern is very much higher than the baseline one. The differences of the classification results of all the models is presented in the result analysis chapter.

In addition we have developed a Web based application using streamlit, that gives flexibility to a user that he can classify a fruit without knowing what is happening in the model or he doesn't even need access to the model.

5.2 Future Work

We faced some difficulties while developing the project. One of the main problems was the skewed dataset. The performance of the model can be maximize by training it with a better dataset in future. Secondly, a computer with a good graphics processing unit is a must for this kind of trainings. But because of limited resources, this couldn't be done. Thankfully Google Colab was there to save the day.

And if we come to the web app, this can be improved also. This is a simple python based web app that was developed using a built in library named Streamlit. But for resource limitation, we couldn't host our most accurate model ResNet50 rather we are using MobileNetV2.

References

[1] *Deep learning definition*, https://www.investopedia.com/terms/d/deep-learning.asp, (Accessed on 05/19/2021) (cit. on p. 1).

[2] *What is the difference between cnn and deep nn, in machine learning? - quora*, https://www.quora.com/What-is-the-difference-between-CNN-and-deep-NN-in-machine-learning, (Accessed on 05/19/2021) (cit. on p. 1).

[3] *Convolutional neural network with python code explanation | convolutional layer | max pooling in cnn*, https://www.analyticssteps.com/blogs/convolutional-neural-network-cnn-graphical-visualization-code-explanation, (Accessed on 05/19/2021) (cit. on p. 7).

[4] *Convolutional neural network - wikipedia*, https://en.wikipedia.org/wiki/Convolutional_neural_network, (Accessed on 05/19/2021) (cit. on p. 7).

[5] *What is tensorflow: Deep learning libraries and program elements explained*, https://www.simplilearn.com/tutorials/deep-learning-tutorial/what-is-tensorflow, (Accessed on 05/19/2021) (cit. on p. 7).

[6] *Keras - wikipedia*, https://en.wikipedia.org/wiki/Keras, (Accessed on 05/19/2021) (cit. on p. 7).

[7] D. Unay, B. Gosselin, O. Kleynen, V. Leemans, M.-F. Destain and O. Debeir, 'Automatic grading of bi-colored apples by multispectral machine vision,' *Computers and electronics in agriculture*, vol. 75, no. 1, pp. 204–212, 2011 (cit. on p. 8).

[8] M. P. Arakeri *et al.*, 'Computer vision based fruit grading system for quality evaluation of tomato in agriculture industry,' *Procedia Computer Science*, vol. 79, pp. 426–433, 2016 (cit. on p. 8).

[9] C. Nandi, B. Tudu and C. Koley, 'Machine vision based techniques for automatic mango fruit sorting and grading based on maturity level and size,' in *Sensing Technology: Current Status and Future Trends II*, Springer, 2014, pp. 27–46 (cit. on p. 8).

[10] T. Ananthanarayana, R. Ptucha and S. C. Kelly, 'Deep learning based fruit freshness classification and detection with cmos image sensors and edge processors,' *Electronic Imaging*, vol. 2020, no. 12, pp. 172–1, 2020 (cit. on pp. 8, 29, 33).

[11] M. Satone, S. Diwakar and V. Joshi, 'Automatic bruise detection in fruits using thermal images,' *International Journal*, vol. 7, no. 5, 2017 (cit. on p. 8).

[12] F. Vega and M. Torres, 'Automatic detection of bruises in fruit using bio-speckle techniques,' in *Symposium of Signals, Images and Artificial Vision-2013: STSIVA-2013*, IEEE, 2013, pp. 1–5 (cit. on p. 9).

[13] Y. Zhang and L. Wu, 'Classification of fruits using computer vision and a multiclass support vector machine,' *sensors*, vol. 12, no. 9, pp. 12 489–12 505, 2012 (cit. on p. 9).

[14] A. Kumar, G. Gill *et al.*, 'Automatic fruit grading and classification system using computer vision: A review,' in *2015 Second International Conference on Advances in Computing and Communication Engineering*, IEEE, 2015, pp. 598–603 (cit. on p. 9).

[15] *Cnn explainer*, https://poloclub.github.io/cnn-explainer/, (Accessed on 05/19/2021) (cit. on p. 10).

[16] *Fauna image classification using convolutional neural network | by kavish sanghvi | analytics vidhya | medium*, https://medium.com/analytics-vidhya/fauna-image-classification-using-convolutional-neural-network-30df9e25a010, (Accessed on 05/19/2021) (cit. on p. 10).

[17] *Fruits fresh and rotten for classification | kaggle*, https://www.kaggle.com/sriramr/fruits-fresh-and-rotten-for-classification, (Accessed on 05/19/2021) (cit. on p. 14).

[18] *Image data preprocessing*, https://keras.io/api/preprocessing/image/, (Accessed on 05/19/2021) (cit. on p. 15).

[19] M. Jogin, M. Madhulika, G. Divya, R. Meghana, S. Apoorva *et al.*, 'Feature extraction using convolution neural networks (cnn) and deep learning,' in *2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, IEEE, 2018, pp. 2319–2323 (cit. on p. 16).

[20] K. He, X. Zhang, S. Ren and J. Sun, *Deep residual learning for image recognition*, 2015. arXiv: 1512.03385 [cs.CV] (cit. on p. 22).

[21] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L.-C. Chen, *Mobilenetv2: Inverted residuals and linear bottlenecks*, 2019. arXiv: 1801.04381 [cs.CV] (cit. on p. 22).

[22] K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, 2015. arXiv: 1409.1556 [cs.CV] (cit. on p. 22).

[23] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, *Rethinking the inception architecture for computer vision*, 2015. arXiv: 1512.00567 [cs.CV] (cit. on p. 22).

[24] *Understanding auc - roc curve | by sarang narkhede | towards data science*, https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5#:~:text=AUC%20%2D%20ROC%20curve%20is%20a,ca

pable%20of%20distinguishing%20between%20classes., (Accessed on 05/19/2021) (cit. on p. 27).

[25] *Confusion matrix for machine learning*, https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/#:~:text=A%20Confusion%20matrix%20is%20an,by%20the%20machine%20learning%20model., (Accessed on 05/19/2021) (cit. on p. 27).