Building & Applying Convolutional Neural Network for Classifying Fruits, Vegetables & Its Ripeness

Rian Dwi Putra
MSc Data Analytics
National College of Ireland
Dublin, Ireland
x22108637@student.ncirl.ie

Suman Shreyas
MSc Data Analytic
National College of Ireland
Dublin, Ireland
x21169641@student.ncirl.ie

Sandra Pereppadan Ignatious
MSc Data Analytics
National College of Ireland
Dublin, Ireland
x21195463@student.ncirl.ie

Lilian Ifeoma Enwereobi
MSc Data Analytics
National College of Ireland
Dublin, Ireland
x20255322@student.ncirl.ie

Abstract—Due to an increasing use chemical in the agriculture industry the quality of fruits has been declining. It is challenging for the sellers to separate fresh fruits from rotten oes. This study proposes the use of deep learning networks to perform classification between fresh and rotten fruits. It will make use of networks such as MobileNet-v2 and Inception v3. Data is collected from platforms such as Kaggle, analyzed, preprocessed, and used in model training. To evaluate the model, metrics like accuracy, precision, recall, f1 score are used. The modelling is performed in such a way that the models can be deployed on small devices with low computation power, thus, making them more usable. Overall, neural network seems like a promising option to solve this problem with the help of tools like Python programming language and its various libraries that support deep learning operations.

Keywords – fruits & vegetables, MobileNet-v2, Inception-v3

I. INTRODUCTION

Fresh and Rotten fruit classification is an important challenge in agriculture industry because of the increasing demand due to the rise in population and therefore an increasing suppy. This leads to the usage of harmful chemicals such as insecticides, pesticides etc. and fertilizers to grow the crops. These chemicals increase the production by a large quantity and hence the farmers cannot be stopped from using them. But this also results in an increase of the percentage of the rotten fruits present in the total produce.

This phenomena is especially common in countries like India where there are no legal laws to check the chemicals used in production and everything is available in the market to be used. Also, the farmers in third world countries are not educated enough to know the damage these chemicals cause. Hence, it becomes necessary to segregate the rotten produce from the fresh one. Right now, it's mostly done manually with the involvement of human labour. There is very little to no technical intervention in this process. This study demonstrates the use of deep learning technologies in agriculture sector i.e., the classification of rotten and fresh fruits.

Deep learning image classification techniques are being used in various fields including medicine, security etc. but it is not as widely used in the agriculture sector. One of the reasons is that agriculture lacks behind in technology due to

the population involved not being educated enough. Hence, technology fails to penetrate into this sector. Other reason is the lack of technical equipment required to work high-tech. This is especially true in the Asian countries. Hence this study proposes the use of deep learning technologies that can run on mobile devices or devices that don't have a high computation power.

This study proposes the use of small deep neural networks which can be easily deployed on mobile devices and used by everyone. Previous experiments that are done in this field use complex neural network design which require a high computation power and hence those models could not be deployed. This paper aims at training the models on devices which have high computation power and use GPUs, TPUs[1], etc. to train but can be quantified[2] to run on devices not having the same.

This will solve the problem of both the seller and consumers. The sellers which include big business that sell large quantities and small shops that sell to the local public, etc. all will be benefitted. Both of them will save resources, money and time on segregating the rotten produce from the fresh produce. This is also result in an increase of quality of the produce which will then increase their business. The consumers will also get good quality produce delivered to them.

The paper has a literature review section where research done on similar topics has been thoroughly review and critiqued. The next section is of the methodology which has the details of steps required to build the solution. Next is the results sections which discusses the results of the experiments in contrast to the other research done. Finally, there is future scope and references.

II. Literature Review

This section provides an overview of the past research regarding the use of technology in classification of fruits and vegetables from disease, rotten or not and the technologies that have been proven effective for performing these tasks.

In the agriculture packaging industry, to determine the quality of fruits and vegetables being packed, one good method is to determine it by analyzing the skin of the fruits and vegetables. [3] have been using traditional machine learning algorithm to analyze the quality of fruits. They are using Support Vector Machines [4] to perform classification using the features extracted from the images of the fruits. The images of the fruits are labeled as rotten or fresh and the features are extracted from then in the form of matrices and passed to the algorithm to perform classification. However, this method only gives the accuracy of about 70% which is less because fruit classification needs to be accurate to 97% or more to be usable. This can be improved with help of deep learning algorithms which learn image features better than classical machine learning algorithms.

Another challenge that comes with the classification of fresh and rotten fruits is the similarity of the features and shapes of disease marks, defects, and insect marks on the fruits. All of this makes it difficult for any computational algorithm to classify the images. To combat this [5] have used laser backscattering spectrographic images that are obtained from semiconductor laser. They have done extensive image processing to obtain sharp quality images. They have used AlexNet[6] to perform the classification and obtained 92% accuracy. Using 500 images of apple, 250 with defects and the other 250 of normal apples. There are decent, considering the data size and model. These results can be improved with increasing the data size and changing the model to a better new generation model. Also, this algorithm cannot be deployed due to its large size, hence there is need to find a smaller and more effective algorithm.

The impact of deep learning in agriculture is growing every day. New image processing techniques are being discovered specifically for agriculture related datasets. [7] shows the various image techniques used in agriculture today that include image recognition, segmentation etc. These also affect the image collection techniques such as collecting images with a good pixel information and higher quality images on which transformations can be applied without distorting the quality of the images. Augmentation

techniques such as zoom in/out, rotation, pixel distortion etc. are also used for agriculture datasets.

Mangosteen fruit classification for defects was performed [8] using Convolutional Neural Networks (CNN)[9]. They have used 4-fold cross validation to validate the model and the mean accuracy of the folds obtained was 97%. They have used approximately 1000 images for train and 120 images for testing, which includes 90 fine images and 30 with defects. The image size is 512 X 512, and the total number of epochs are 50. This can be improved by increasing the size of the images and using newer models like MobileNet-v2, etc.

[10] have used MobileNets to classify various images of fruits and vegetables images taken as a snapshot in a mobile. This model is deployed on a smartphone device which has less computation power than a traditional system but still gives an accuracy of around 92%. They have trained it on the images that are captured from the same device only. To increase the performance of this model external images from other data sources can also be used. Some researchers are researching how different types of images can contribute to the field of deep learning, especially agriculture for which they have started generating custom datasets. [11] have created a dataset with original and augmented images of 8 different fruits. This can be used in the further research of deep learning in agriculture.

Many more deep learning experiments have been performed on rotten and fresh fruit classification. [12] has used fruits of 10 different classes with the total of 5658 images. They have used 5 different CNN models and the best performance is given by Inceptionv3[13] by giving the highest accuracy of 97.34%. [14] have used various pre-trained models and then trained them on their dataset and have also created a custom model and trained it. They have done this on apples, bananas and oranges and have outperformed the pretrained models using their custom model. Their model gives an accuracy of 97.82% with adam optimizer [15]. The performance stagnated after 50 epochs and starts decreasing when batch size is increased to more than 30. They have proposed a simple network with 3 convolutional layers and a final fully connected layer along with batch normalization and max pooling after every layer.

Another effective way of increasing the performance of any CNN or even a multi-layer perceptron is to perform data preprocessing and augmentation to improve the quality and quantity of data. Some of them involve performing complex mathematical operations on the data which can be images or in tabular format. [16] performed complex statistical operations like GrabCut algorithm, Hue Saturation Value with Adaptive Gaussian Thresholding, Laplacian O Gaussian, etc. on a fresh vs rotten fruit dataset containing 3000 images for training, testing and validation that were trained on VGG16[17] architecture and improved the validation accuracy from 80.01% to 89.97% and testing accuracy from 82.31% to 91.33%.

III. Data Mining Methodology

Fresh and rotten fruit classification is an important task especially with the growing use of pesticides, insecticides and other chemicals that are used in farming today. Also, the use of chemical enhancers such as fertilizers are causing decay in the fruit production. Due to this it becomes essential for both the consumers and sellers to classify between fresh and rotten fruits. The sellers want to purchase only fresh fruits as they are the once that the consumer will purchase from them. Due to mass production of fruits, we need a system that can classify fresh vs. rotten fruits.

Right now the fruits are segregated manually but with the methods proposed in this paper, segregation can be done just by using few cameras and computers. This paper aims to achieve 95% precision in classification. Implementation of the algorithm will be performed with Python programming language as it has a wide support for deep learning algorithms which will be used in this. Cloud computing resources such as GCP, AWS etc. will also be used.

3.1. Data Selection

Classifying images between fruits and vegetables using Convolutional Neural Networks (CNNs) is a type of computer vision application. In this application, the CNN is trained on a large dataset of fruit and vegetable images to recognize and classify these images into one of two categories: fruits or vegetables.

- Classifying Fruit

```
#Importing necessary packages
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import math
import numpy as np
import matplotlib.pyplot as plt
import logging
logger = tf.get_logger()
logger.setLevel(logging.ERROR)
```

```
#Parameters
BATCH_SIZE = 32
IMAGE_SIZE = 100
```

```
#loading in and preprocessing the train and test datasets
train_dataset = tf.keras.preprocessing.image_dataset_from_directory(
    "kaggle/input/fruits/fruits-360_dataset/fruits-360/Training",
    shuffle=True,
    image_size=(IMAGE_SIZE, IMAGE_SIZE),
    batch_size=BATCH_SIZE
)
test_dataset = tf.keras.preprocessing.image_dataset_from_directory(
    "kaggle/input/fruits/fruits-360_dataset/fruits-360/Test",
    shuffle=True,
    image_size=(IMAGE_SIZE, IMAGE_SIZE),
    batch_size=BATCH_SIZE
)
```

```
Found 67692 files belonging to 131 classes.
Found 22688 files belonging to 131 classes.
```

Figure 1. Classifying fruit

- Classifying Vegetable

```
#Importing necessary packages
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import tensorflow_hub as hub
import math
import numpy as np
import matplotlib.pyplot as plt
import logging
logger = tf.get_logger()
logger.setLevel(logging.ERROR)
```

```
#Parameters
BATCH_SIZE = 32
IMAGE_SIZE = 224
```

```
#loading in and preprocessing the train and test datasets
train_dataset = tf.keras.preprocessing.image_dataset_from_directory(
    "C:\\Users\\lilia\\Downloads\\vegetable\\VegetableImages\\train",
    shuffle=True,
    image_size=(IMAGE_SIZE, IMAGE_SIZE),
    batch_size=BATCH_SIZE,
    seed = 123 #Random Seed for reproducibility
)
test_dataset = tf.keras.preprocessing.image_dataset_from_directory(
    "C:\\Users\\lilia\\Downloads\\vegetable\\VegetableImages\\test",
    shuffle=True,
    image_size=(IMAGE_SIZE, IMAGE_SIZE),
    batch_size=BATCH_SIZE,
    seed = 123 #Random Seed for reproducibility
)
```

```
Found 15000 files belonging to 15 classes.
Found 3000 files belonging to 15 classes.
```

Figure 2. Classifying vegetable

To classify an input image, it is first pre-processed to ensure consistency and then passed through the trained CNN. The output of the CNN is a probability score for each category, which is used to determine the final classification of the image.

- Classifying its ripeness

The next step is selecting data for training deep learning algorithms. The proposed method collects images of fruits that are rotten and fresh. Data scraping is performed using selenium to scrape data of apple, oranges and bananas. Images of rotten and fresh fruits are also collected from Kaggle. Images of each fruit of each category, i.e. rotten and fresh are collected which will be distributed between train and test sets.

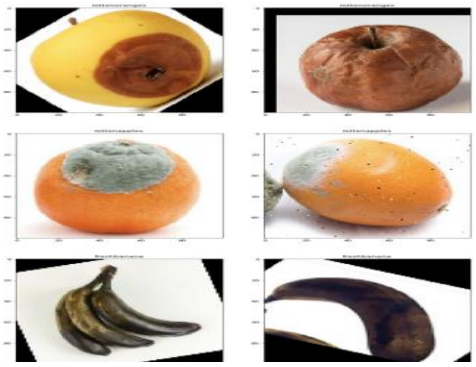

Figure 3: Rotten Fruits

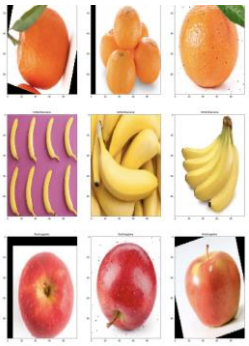

Figure 4: Fresh Fruits

In train data there are 1693 fresh apple images, 1581 fresh banana images, 1466 fresh orange images and 2342 rotten apple images, 2224 rotten banana images, 1595 rotten oranges images. In test there are 395 fresh apple images, 381 fresh banana images, 388 fresh orange images and 601 rotten

apple images, 530 rotten banana images, 403 rotten oranges image. In order to train a deep learning model image size has to be made constant.

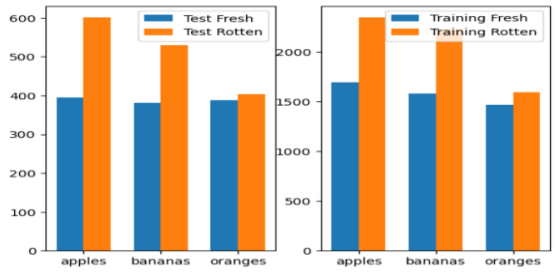

Figure 5: Distribution of Data

3.2 Data Pre-processing

This is a classification problem. Hence, first step towards data pre-processing is creating labels for all the classes. This is binary classification, so there are only 2 classes. The images are resized to a fixed size of 32X 3 and labels i.e. 1 and 0 are assigned to each image. Images are loaded as arrays or tensors in the neural network. Various types of transformations such as zoom in/out, rotate, flip, etc. Are also applied to the dataset. These transformations reduce the bias in the data and help the model generalize better. These transformation will reduce the biases of having the camera at same distance or same angle by zoom in/out of the image and rotating it. These are also known as data augmentation techniques. They help in increasing the size of the dataset. Normalizing the dataset so that images have zero mean and one variance. Finally the data size is increased to 1600 images of fresh fruit for each fruit and 2300 rotten images for each fruit. Total train images are 10700.

3.3 Modelling

After data pre-processing, next task is to model the dataset. It starts with splitting the dataset into train and validation sets. This paper proposes splitting it in 3:1 ratio. The batch size is kept 64 considering the resources and models are trained on GPU. Validation split during training is kept .2. Loss function is cross entropy loss.

The proposed method uses Convolutional Neural Networks(CNN) to model the data. It starts by using a simple CNN network consisting of few Conv 2D layers and few fully connected layers. Initially, the model is trained for 50 epochs, which will be increased or decreased based on the model performance. The proposed method also uses Inceptionv3 to model the dataset. Inception V3 is used due to its high learning capabilities while keeping the computation size low. Due to its 1X1 convolutional blocks Inceptionv3 uses less computational resources as compared to a normal deep neural network and with its 3X3 and 5X5 blocks it learns the spatial features as well. The following approach also uses Mobilenetv2 because of its small size which results in less compute time and high precision. The custom CNN network takes less computation time but does not provide results similar as inception v3 or mobilenet v2. Mobilenet v2 performs better than inceptionv3 due to its smaller size and network structure.

"Downloading model" and "transfer learning" refer to two different steps involved in using pre-trained Convolutional Neural Network (CNN) models like Inception V3 and MobileNet.

- Downloading required model

"Downloading model" typically refers to the process of obtaining a pre-trained CNN model from an online source. These models are typically trained on large datasets like ImageNet, which contain millions of labelled images. By downloading a pre-trained model, you can save a significant amount of time and resources that would be required to train your own model from scratch.

Downloading the Pre-Trained Model

```
CLASSIFIER_URL = 'https://tfhub.dev/google/imagenet/inception_v3/classification/5'
IMAGE_RES = 100

model = tf.keras.Sequential([
    hub.KerasLayer(CLASSIFIER_URL, input_shape=(IMAGE_RES, IMAGE_RES, 3))
])
```

Transfer Learning using Tensorflow Hub

```
URL = 'https://tfhub.dev/google/imagenet/inception_v3/feature_vector/5'
feature_extractor = hub.KerasLayer(URL, input_shape=(IMAGE_RES, IMAGE_RES, 3))
```

```
feature_extractor.trainable = False
```

Figure 6. Downloading Inception v3

"Transfer Learning" on the other hand, refers to the process of fine-tuning the downloaded model for a specific task. This involves modifying the architecture of the model

and adjusting its parameters to improve its performance on a specific dataset. This step is necessary because the pre-trained model is trained on a generic dataset and may not perform well on a new dataset without some fine-tuning.

Downloading the Pre-Trained Model

```
CLASSIFIER_URL = 'https://tfhub.dev/google/tf2-preview/mobilenet_v2/classification/4'
IMAGE_RES = 224

model = tf.keras.Sequential([
    hub.KerasLayer(CLASSIFIER_URL, input_shape=(IMAGE_RES, IMAGE_RES, 3))
])
```

Transfer Learning using Tensorflow Hub

```
URL = 'https://tfhub.dev/google/tf2-preview/mobilenet_v2/feature_vector/4'
feature_extractor = hub.KerasLayer(URL, input_shape=(IMAGE_RES, IMAGE_RES, 3))
```

```
feature_extractor.trainable = False
```

Figure 7. Downloading MobileNet v2

In the context of Inception V3 and MobileNet, downloading the model would involve obtaining the pre-trained model files from a source like TensorFlow Hub or the official TensorFlow website. Model building or transfer learning would then involve modifying the model architecture and training it on a specific dataset to improve its accuracy and performance on a specific image classification task.

- Model Building

The first step in this process is to obtain a pre-trained model that has been trained on a large dataset of images, such as MobileNet. This model has already learned to recognize many different features of images and can be used as a starting point for the fruit and vegetable classification task.

Model Building

I. Setting up the layer

```
model = tf.keras.Sequential([
  feature_extractor,
  tf.keras.layers.Dense(131)
])
```

II. Compiling the model

```
#Compiling using Adam Optimizer
model.compile('adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy'])
```

Figure 8. Model Building

Next, the pre-trained model is fine-tuned on a smaller dataset of labelled fruit and vegetable images. This involves adjusting the parameters of the model to make it more sensitive to the unique features of the fruit and vegetable images, such as shape, size, and colour.

During fine-tuning, the weights of some layers in the pre-trained model are frozen, while others are retrained using the smaller dataset. This allows the model to leverage the pre-existing knowledge learned from the large dataset while still adapting to the specific characteristics of the fruit and vegetable images.

After fine-tuning is complete, the accuracy of the model is evaluated on a separate validation dataset, and adjustments can be made to the model architecture or fine-tuning process if necessary. Once the model achieves satisfactory accuracy, it can be used to classify new fruit and vegetable images.

Overall, model building for classifying fruit and vegetable images using a pre-trained model involves adapting an existing deep learning model to the specific classification task through fine-tuning on a smaller dataset. This approach can be more efficient and effective than training a deep learning model from scratch.

This study uses Adam (Adaptive Moment Estimation) for optimization algorithm. Adam for gradient-based optimization. It is a popular choice for training deep neural networks because of its several advantages, which include:

a) Adaptive learning rate: Adam adapts the learning rate for each parameter in the network based on the gradient magnitudes, which can lead to faster convergence and better generalization.

b) Momentum: Adam uses a momentum term to keep track of past gradients, which helps the optimizer continue in the same direction when the gradients change direction.

c) Bias correction: Adam uses bias correction to correct the estimate of the first and second moments of the gradients. This helps to initialize the parameters at the beginning of the optimization process.

d) Efficient memory usage: Adam stores the estimates of the first and second moments of the gradients as moving averages, which requires less memory than storing the gradients themselves.

e) Robustness to noisy gradients: Adam is robust to noisy gradients, which can occur when training deep neural networks with large datasets.

Overall, Adam is a powerful optimization algorithm that can lead to faster convergence and better generalization for deep neural networks. It is widely used in deep learning research and has been shown to be effective in a wide range of applications.

- Classifying its ripeness by Inception v3 and MobileNet v2

This process involves using deep learning algorithms, such as Convolutional Neural Networks (CNNs), to analyse the features of images of fruits and vegetables and make predictions about their ripeness level. These algorithms are trained on large datasets of labelled images, with each image categorized as ripe, unripe or overripe.

During the training process, the deep learning model learns to recognize and distinguish the visual cues that indicate ripeness level, such as colour, texture, and shape. Once trained, the model can be used to classify the ripeness level of new images of fruits and vegetables with high accuracy.

The proposed CNN network uses two convolution layers and two fully connected layers. Max pooling layer is used after each convolution layer. Activation function used is tanh. This network is trained for 50 batches and has a batch size of 64.

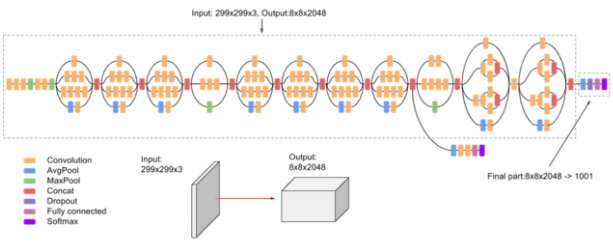

Figure 9. Custom CNN Model

- Inception v3

The Inception V3 model has over 23 million trainable parameters and consists of 42 convolutional layers, along with fully connected layers and global average pooling. The architecture includes multiple parallel branches that process different scales of input images, allowing it to capture a wide range of features at different levels of abstraction.

Inception V3 was pre-trained on the ImageNet Large Scale Visual Recognition Challenge dataset, which contains over 1 million labelled images spanning 1000 categories. The model achieved state-of-the-art results on the dataset, with a top-5 accuracy of 93.33%.

Inceptionv3 is trained for 10 epochs and starts getting overfitted. The other parameters remain same. The model weights of Inceptionv3 are loaded using pytorch

- MobileNet v2

MobileNet V2 is a pre-trained Convolutional Neural Network (CNN) model designed for efficient mobile and embedded vision applications. It is the second version of the MobileNet architecture, which was developed by Google.

The MobileNet V2 model has fewer parameters and is more lightweight than many other CNN models, making it ideal for use on mobile devices or embedded systems. It uses a combination of depth wise separable convolutions and linear bottleneck layers to reduce the computational cost and memory footprint of the model, while maintaining high accuracy.

MobileNet V2 was pre-trained on the ImageNet dataset, a large-scale dataset containing over 1 million labeled images spanning 1000 categories. The model achieved state-of-the-art results on the dataset, with a top-1 accuracy of 72.0% and a top-5 accuracy of 90.2%.

MobileNet V2 is a popular model for transfer learning, where it is fine-tuned on smaller datasets for specific image classification tasks. Its efficient design and high accuracy make it well-suited for use in mobile and embedded vision

applications, such as object detection, facial recognition, and augmented reality.

Mobilenetv2 is also trained for 10 epochs because it starts overfitting after that.

Table 1.

Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	

3.4 Evaluation

This study has utilized transfer learning and used a pre-trained model, MobileNet v2, on fruits360 dataset to classify different classes of fruits and vegetables.

MobileNet is a state-of-the-art convolutional neural network developed by Google that uses a very efficient neural network architecture that minimizes the amount of memory and computational resources needed, while maintaining a high level of accuracy. MobileNet is ideal for mobile devices that have limited memory and computational resources. The MobileNet classifier had an average accuracy of roughly 99% when identifying images of single fruits or vegetables which is very impressive compared to another CNN which created from scratch, on the same dataset, which achieved an accuracy of about (95%). It proves to be a powerful model.

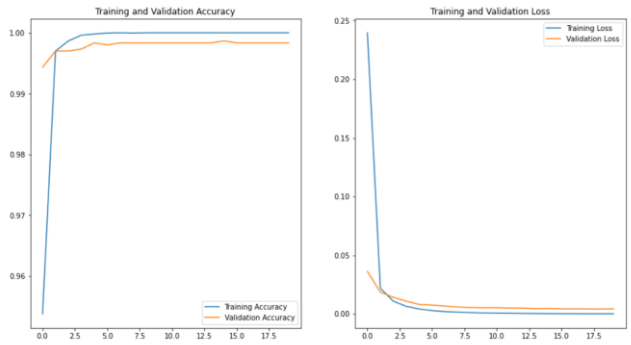

Figure 10. Accuracy and Loss between Training and Validation in MobileNet v2

Figure 11. Accuracy and Loss between Training and Validation in Inception v3

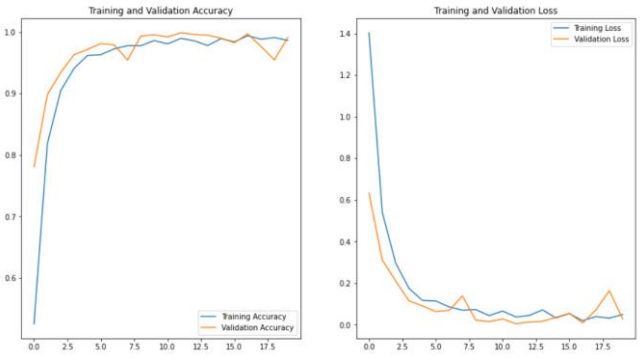

Figure 12. Accuracy and Loss between Training and Validation in CNN which built from scratch

This study built a Convolutional Neural Network model to classify different classes of fruits and vegetables. The classifier has an average accuracy of roughly 95% when identifying images of single fruits or vegetables which is very impressive. CNN's are typically best suited to image classification tasks.

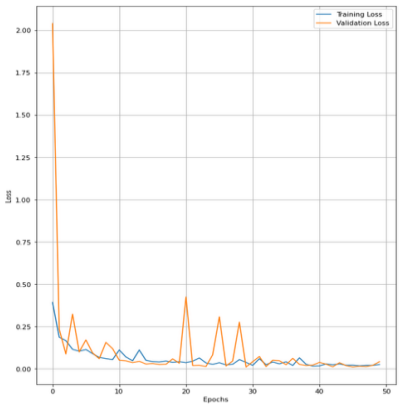

Figure 13: Training Loss vs Validation Loss for Ripeness

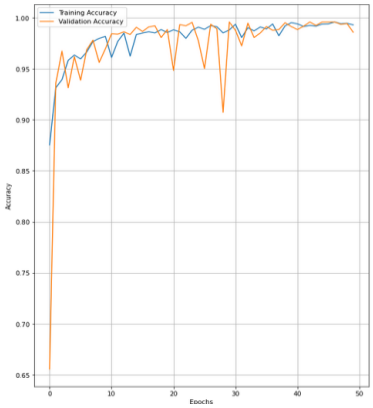

Figure 14: Training Accuracy vs. Validation Accuracy for Ripeness

The evaluation metrics used for this model is accuracy, precision, recall and f1 score. Accuracy gives a general idea about the model's performance while precision gives an overview about the correctness of the model. In this case, precision tells whether the fruits that are classified as fresh are actually fresh. Recall gives a true positive rate which in this case will be the number of fresh fruits that could be classified out of the total fresh fruits.

Table 3.2: Results

Algorithm	Accuracy	Precision	Recall	F1
CNN	89.1	.6	.65	.62
Inception V3	95.2	.8	.83	.82
MobileNet V2	98.1	1	.92	1

3.5 Deployment

These models can be deployed on small IoT devices having enough computation power or can be deployed computers. There can be live classification, directly taking data from camera or on a recorded video.

IV. Results

The approach proposed in this paper to solve the problem of rotten and fresh fruit classification on scale to help sellers and consumers uses deep learning models. This methodology has chosen deep learning models by maintain a balance between performance and resource requirement. Models like Mobilenet v2 can be easily deployed on small devices and can be used at a scale as they do not require a lot of computation power.

As we can see in the table 3.1 that Mobilenet v2 has given good scores on all the metrics. This will impact the agriculture industry on a large scale. Manual efforts will be saved that were previously required to distinguish between the fresh and rotten fruits. This will save the money of the sellers that buy fruits from farmers which can then be used to invest in other tasks such as faster transportation, better storage facility etc.

This method creates a template for training various different deep learning architectures just by tweaking some parameters. This allows for easy training and evolution of the models.

V. Conclusions & Future Works

The proposed method researched upon various techniques that can be used to solve the classification problem and finally deciding to use deep learning. In future, new deep learning models can be implemented on this dataset. More techniques of image augmentation can be applied on the dataset to make it larger and help the model generalize better. Deployment can be done on small IoT devices to test the performance.

To solve the challenge of fresh and rotten fruit classification the methods proposed in this paper demonstrate an effective approach. These deep learning models can be deployed on a device, tested and improved by tweaking its parameters.

References

[1] Kosaian, J. and Phanishayee, A. (2022) *A study on the intersection of GPU utilization and CNN inference*, *arXiv.org*. Available at: https://arxiv.org/abs/2212.07936 (Accessed: April 28, 2023).

[2] Huang, Z., Lam, H. and Zhang, H. (2022) *Quantifying epistemic uncertainty in deep learning*, *arXiv.org*. Available at: https://arxiv.org/abs/2110.12122 (Accessed: April 28, 2023).

[3] L. Wang, A. Li, and X. Tian, "Detection of Fruit Skin Defects Using Machine Vision System," *IEEE Xplore*, Nov. 01, 2013. https://ieeexplore.ieee.org/abstract/document/6961088

[4] L. Wang, A. Li, and X. Tian, "Detection of Fruit Skin Defects Using Machine Vision System," *IEEE Xplore*, Nov. 01, 2013. https://ieeexplore.ieee.org/abstract/document/6961088

[5] A. Wu, J. Zhu, and T. Ren, "Detection of apple defect using laser-induced light backscattering imaging and convolutional neural network," *Computers & Electrical Engineering*, vol. 81, p. 106454, Jan. 2020, doi: https://doi.org/10.1016/j.compeleceng.2019.106454.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, May 2012, doi: https://doi.org/10.1145/3065386.

[7] Prakash, K & Saravanamoorthi, P & Sathishkumar, Mr & M.Parimala, Rishwanth. (2017). A Study of Image Processing in Agriculture. International Journal of Advanced Networking and Applications. 9. 3311-3315.

[8] L. M. Azizah, S. F. Umayah, S. Riyadi, C. Damarjati, and N. A. Utama, "Deep learning implementation using convolutional neural network in mangosteen surface defect detection," *IEEE Xplore*, Nov. 01, 2017. https://ieeexplore.ieee.org/document/8284412 (accessed Apr. 28, 2023).

[9] K. O'Shea and R. Nash, "An Introduction to Convolutional Neural Networks," *arXiv:1511.08458 [cs]*, Dec. 2015, Available: https://arxiv.org/abs/1511.08458

[10] R. M. Bocanegra, J. A. L. Sotelo, H. Satizabal, and A. P. Uribe, "Fruit and Vegetable Information System Using Embedded Convolutional Neural Networks," *IEEE Xplore*, Nov. 01, 2019. https://ieeexplore.ieee.org/document/9037057 (accessed Apr. 28, 2023).

[11] N. Sultana, M. Jahan, and M. S. Uddin, "An extensive dataset for successful recognition of fresh and rotten fruits," *Data in Brief*, vol. 44, p. 108552, Oct. 2022, doi: https://doi.org/10.1016/j.dib.2022.108552.

[12] M. S. Miah, T. Tasnuva, M. Islam, M. Keya, Md. R. Rahman, and S. A. Hossain, "An Advanced Method of Identification Fresh and Rotten Fruits using Different Convolutional Neural Networks," *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, Jul. 2021, doi: https://doi.org/10.1109/icccnt51525.2021.9580117.

[13] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," *arXiv.org*, 2015. https://arxiv.org/abs/1512.00567

[14] S. S. S. Palakodati, V. R. Chirra, Y. Dasari, and S. Bulla, "Fresh and Rotten Fruits Classification Using CNN and Transfer Learning," *Revue d'Intelligence Artificielle*, vol. 34, no. 5, pp. 617–622, Nov. 2020, doi: https://doi.org/10.18280/ria.340512.

[15] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv.org*, Dec. 22, 2014. https://arxiv.org/abs/1412.6980

[16] S. Nuanmeesri, L. Poomhiran, and K. Ploydanai, "Improving the Prediction of Rotten Fruit Using Convolutional Neural Network," *International Journal of Engineering Trends and Technology*, vol. 69, no. 7, pp. 51–55, Jul. 2021, doi: https://doi.org/10.14445/22315381/ijett-v69i7p207.