

**FRESH FRUITS SELECTION RECOMMENDER**

**BY**

**LOH ZHAN HERNG**

**A REPORT**

**SUBMITTED TO**

**Universiti Tunku Abdul Rahman**

**in partial fulfillment of the requirements**

**for the degree of**

**BACHELOR OF COMPUTER SCIENCE (HONS)**

**Faculty of Information and Communication Technology**

**(Perak Campus)**

**JANUARY 2020**

## REPORT STATUS DECLARATION FORM


Title: Fresh Fruits Selection Recommender  
\_\_\_\_\_  
\_\_\_\_\_

Academic Session: January 2020

I LOH ZHAN HERNG  
(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in  
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

  
\_\_\_\_\_  
(Author's signature)

Verified by,

  
\_\_\_\_\_  
(Supervisor's signature)

Address:  
No 27, Persiaran Tasek Timur 14,  
Taman Mewah,  
31400 Ipoh, Perak

Aun YC  
\_\_\_\_\_  
Supervisor's name

Date: 24/04/2020

Date: 24/4/2020

**FRESH FRUITS SELECTION RECOMMENDER**

**BY**

**LOH ZHAN HERNG**

**A REPORT**

**SUBMITTED TO**

**Universiti Tunku Abdul Rahman**

**in partial fulfillment of the requirements**

**for the degree of**

**BACHELOR OF COMPUTER SCIENCE (HONS)**

**Faculty of Information and Communication Technology**

**(Perak Campus)**

**JANUARY 2020**

## DECLARATION OF ORIGINALITY

I declare that this report entitled “**FRESH FRUITS SELECTION RECOMMENDER**” is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : 

Name : Loh Zhan Herng

Date : 24/04/2020

## **ACKNOWLEDGEMENTS**

Firstly, I would like to thank my FYP supervisor, Dr. Aun Yichiet who have provided me guidance and opinions throughout my research project. I appreciate that Dr. Aun was willing spare his precious time to share his knowledge with patience which helped me a lot in this project. Next, I would like to thank my family members who gave me a lot support and encouragement in every aspect. Lastly, I would also thank my course mates who gave me guidance when I met difficulties and challenges. Without them, this project would not be conducted successfully.

## ABSTRACT

Fresh produces (fruits) quality prediction was the main idea of this project and it was hoped to serve as a fresh fruit selection recommender for common or industrial usage. With the difficulties of predicting the actual condition of the fruits through observing its external appearance as well as the internal quality factors, believing that Computer Vision could help us to solve the problem. In this project, the fruit domains involved were apple, banana and orange. Generally, the development mainly split into two different tasks or phases, fruit classification and fruit detection or localization. To identify and localize the fruit presented in one image or frame, another set of data image was collected to make annotations manually and these data were prepared to “feed” into the object detection API, which Faster R-CNN object detection model was configured as the training pipeline. A frozen inference graph was trained for fruit detection usage. Besides, to make predictions on the freshness states of these fruits, data images of two distinguishable freshness states of ‘Fresh’ and ‘Rotten’ was collected. Convolutional Neural Network (CNN) architecture was used to construct and train for classification model. In order to achieve desirable performance from the model, evaluations and analysis were made so as to make incremental improvements through using regularization method as well as transfer learning approach. To conclude the achievements of both model, the Faster R-CNN detection inference graph was well-trained as it identifies classes at most of the situations, as for the CNN classification model, it achieved the overall accuracy of 99.81% because of leveraging the pre-trained weights from VGG-16 model. Till current stage, the prototype was usable for limited fruit domains quality prediction. It was believed that it could be deployed for real-world usage if improvements and extended development were made on this system prototype.

# TABLE OF CONTENTS

|   |            |
|---|------------|
| <b>TITLE PAGE</b>   | <b>i</b>   |
| <b>DECLARATION OF ORIGINALITY</b>   | <b>ii</b>  |
| <b>ACKNOWLEDGEMENTS</b>   | <b>iii</b> |
| <b>ABSTRACT</b>   | <b>iv</b>  |
| <b>TABLE OF CONTENTS</b>  | <b>v</b>   |
| <b>LIST OF FIGURES</b>  | <b>ix</b>  |
| <b>LIST OF TABLES</b>   | <b>xi</b>  |
| <b>LIST OF ABBREVIATIONS</b>  | <b>xii</b> |
| <b>CHAPTER 1 INTRODUCTION</b>   | <b>1</b>   |
| 1.1 Introduction  | 1          |
| 1.2 Problem Statement   | 2          |
| 1.3 Background and Motivation   | 3          |
| 1.4 Project Objectives  | 5          |
| 1.5 Impact, Significance and Contribution   | 7          |
| 1.6 Proposed Approach   | 8          |
| 1.7 Highlight of the Project  | 9          |
| 1.8 Report Organization   | 10         |
| <b>CHAPTER 2 LITERATURE REVIEW</b>  | <b>11</b>  |
| 2.1 Related Fresh Produces Quality Estimation Techniques  | 11         |
| 2.1.1 Fruit freshness detection using sensor  | 11         |
| 2.1.2 Real-time smart fruit quality classification system through appearance and internal flavour factors | 12         |
| 2.1.3 Fresh produces grading using RGB-D and CNN  | 14         |

|  |   |           |
|--|---|-----------|
| 2.1.4                                  | Classical automated fruit grading system process using MATLAB         | 15        |
| 2.1.5                                  | Fruit freshness detection using Raspberry Pi                          | 17        |
| 2.2                                    | Critical Remarks and Comparison between Reviewed Techniques           | 18        |
| <b>CHAPTER 3 SYSTEM DESIGN</b>         |   | <b>20</b> |
| 3.1                                    | System Development Overview and General Work Procedures               | 20        |
| 3.2                                    | Fruit Classification  | 22        |
| 3.2.1                                  | Data Acquisition  | 22        |
| 3.2.2                                  | Data Pre-processing   | 23        |
| 3.2.3                                  | Network Architectures Construction and Training                       | 23        |
| 3.2.4                                  | Network Evaluation and Analysis                                       | 24        |
| 3.2.5                                  | Model Fine-Tuning   | 26        |
| 3.2.6                                  | Samples Prediction  | 26        |
| 3.3                                    | Fruit Detection/Localization  | 27        |
| 3.3.1                                  | Data Acquisition  | 27        |
| 3.3.2                                  | Manual Annotation on Images   | 28        |
| 3.3.3                                  | Data Pre-processing / Preparation                                     | 29        |
| 3.3.4                                  | Faster R-CNN Object Detection Model Training                          | 30        |
| 3.3.5                                  | Run Frozen Inference Graph  | 30        |
| 3.4                                    | System Flowchart  | 31        |
| 3.4.1                                  | Input Data (Still Images / Real-time Video)                           | 32        |
| 3.4.2                                  | Run with Frozen Inference Graph                                       | 32        |
| 3.4.3                                  | Search Bounding Boxes   | 33        |
| 3.4.4                                  | Perform Freshness States Prediction with CNN Classification Model     | 34        |
| 3.4.5                                  | Return Output to Front-end  | 34        |
| <b>CHAPTER 4 SYSTEM IMPLEMENTATION</b> |   | <b>35</b> |
| 4.1                                    | Evaluation and Analysis on Convolutional Neural Network Architectures | 35        |
| 4.1.1                                  | Simple CNN Build-from-scratch   | 36        |
| 4.1.2                                  | CNN with Regularization   | 39        |
| 4.1.3                                  | CNN with Image Augmentation   | 42        |
| 4.1.4                                  | Transfer Learning of Pre-trained CNN as Feature Extractor Performance | 45        |



|                  |   |           |
|------------------|---|-----------|
| 4.1.5            | Transfer Learning of Pre-trained CNN with Fine-tuning and Data Augmentation | 48        |
| 4.2              | Deduction on CNN Classification Model                                       | 51        |
| 4.3              | Evaluation on Faster R-CNN detection model                                  | 52        |
| 4.4              | Tools to Use  | 53        |
| 4.4.1            | Software/Platform   | 53        |
| 4.4.2            | Libraries   | 53        |
| 4.4.3            | Hardware  | 53        |
| 4.5              | System Performance Definition   | 55        |
| 4.6              | Verification Plan   | 56        |
| <b>CHAPTER 5</b> | <b>SYSTEM TESTING</b>   | <b>57</b> |
| 5.1              | System Testing Procedural Block Diagram                                     | 57        |
| 5.2              | Testing on Performance of Classification Model                              | 59        |
| 5.2.1            | Apple   | 59        |
| 5.2.2            | Banana  | 60        |
| 5.2.3            | Orange  | 61        |
| 5.3              | Testing on Performance of Detection Model                                   | 62        |
| 5.3.1            | Apple Detection   | 62        |
| 5.3.2            | Banana Detection  | 63        |
| 5.3.3            | Orange Detection  | 64        |
| 5.4              | Detect and Classify on Each Individual Fruit                                | 65        |
| 5.4.1            | Single Fruit Image Prediction   | 66        |
| 5.4.2            | Multiple Fruit Image Prediction   | 67        |
| 5.4.3            | Multi-class Fruit Image Prediction  | 68        |
| 5.4.4            | Real-time Video Prediction  | 69        |
| 5.5              | Discussion on Testing Issues and Dependencies                               | 70        |
| 5.6              | Extended Testing Phase Regarding Affecting Factors                          | 72        |
| 5.6.1            | Stacking / Unorderly Arranged   | 72        |
| 5.6.2            | Light Exposure in the Environment   | 74        |

|                                    |            |
|------------------------------------|------------|
| <b>CHAPTER 6 CONCLUSION</b>        | <b>78</b>  |
| 6.1 Project Review and Discussions | 78         |
| 6.2 Novelties and Contributions    | 80         |
| 6.3 Future Works                   | 80         |
| <b>BIBLIOGRAPHY</b>                | <b>81</b>  |
| <b>APPENDICES</b>                  | <b>A-1</b> |

## LIST OF FIGURES

|   |    |
|---|----|
| Figure 1.1 Project Development Overview                               | 8  |
| Figure 2.1 Sensor designed for fruit freshness detection              | 11 |
| Figure 2.2 Processed image to detect defection on fruits              | 12 |
| Figure 2.3 Classification model based on ANN for fruit grading system | 13 |
| Figure 2.4 One simple neural network and Two CNNs.                    | 15 |
| Figure 2.5 Flowchart of fruit grading process                         | 15 |
| Figure 2.6 Illustration of automated fruit grading system             | 16 |
| Figure 2.7 GUI presented with fruit defection                         | 17 |
| Figure 2.8 Project Equipment Setup                                    | 17 |
| Figure 3.1 System Development Overview                                | 20 |
| Figure 3.2 Fruit Classification Development Block Diagram             | 22 |
| Figure 3.3 Fruit Detection Development Block Diagram                  | 27 |
| Figure 3.4 Example of Apple Annotations                               | 28 |
| Figure 3.5 Example of Orange Annotations                              | 29 |
| Figure 3.6 Example of Banana Annotation                               | 29 |
| Figure 3.7 System Flowchart   | 31 |
| Figure 3.8 Example of Cropped Images from Original Image              | 33 |
| Figure 4.1 CNN Architecture 1   | 36 |
| Figure 4.2 CNN 1 Performance Line Graph                               | 36 |
| Figure 4.3 CNN Architecture 2   | 39 |
| Figure 4.4 CNN 2 Performance Line Graph                               | 39 |
| Figure 4.5 Example of Data Augmentation                               | 42 |
| Figure 4.6 CNN 3 Performance Line Graph                               | 43 |
| Figure 4.7 Network Architecture of VGG-16                             | 45 |
| Figure 4.8 CNN Architecture 4   | 46 |
| Figure 4.9 CNN 4 Performance Line Graph                               | 46 |
| Figure 4.10 Indicates the Trainable Layers                            | 48 |
| Figure 4.11 CNN 5 Performance Line Graph                              | 49 |
| Figure 5.1 System Testing Procedural Block Diagram                    | 57 |
| Figure 5.2 Testing Scenario 1   | 72 |
| Figure 5.3 Testing Scenario 2   | 73 |

|                               |    |
|-------------------------------|----|
| Figure 5.4 Testing Scenario3  | 75 |
| Figure 5.5 Testing Scenario 4 | 76 |
| Figure 5.6 Testing Scenario 5 | 77 |

## LIST OF TABLES

|   |    |
|---|----|
| Table 2.1 Comparison among Reviewed Techniques                  | 18 |
| Table 3.1 Confusion Matrix for Fruit Classification Predictions | 25 |
| Table 4.1 Data Splitting Table                                  | 35 |
| Table 4.2 Parameters with Respective Values                     | 35 |
| Table 4.3 Confusion Matrix for CNN 1                            | 37 |
| Table 4.4 Precision, Recall and F1-Scores for CNN 1             | 37 |
| Table 4.5 Performance Matrix Table for CNN 1                    | 38 |
| Table 4.6 Confusion Matrix for CNN 2                            | 40 |
| Table 4.7 Precision, Recall and F1-Score for CNN 2              | 40 |
| Table 4.8 Performance Metrics Table for CNN 2                   | 41 |
| Table 4.9 Confusion Matrix for CNN 3                            | 43 |
| Table 4.10 Precision, Recall, F1-Score for CNN 3                | 44 |
| Table 4.11 Performance Metrics for CNN 3                        | 44 |
| Table 4.12 Confusion Matrix for CNN 4                           | 47 |
| Table 4.13 Precision, Recall and F1-Score for CNN 4             | 47 |
| Table 4.14 Performance Metrics Table for CNN 4                  | 47 |
| Table 4.15 Confusion Matrix for CNN 5                           | 50 |
| Table 4.16 Precision, Recall and F1-Score for CNN 5             | 50 |
| Table 4.17 Performance Metrics Table for CNN 5                  | 50 |
| Table 4.18 Overall Performance Table                            | 51 |
| Table 5.1 Apple Single Image Classification Testing             | 59 |
| Table 5.2 Banana Single Image Classification Testing            | 60 |
| Table 5.3 Orange Single Image Classification Testing            | 61 |
| Table 5.4 Apple Detection Testing                               | 62 |
| Table 5.5 Banana Detection Testing                              | 63 |
| Table 5.6 Orange Detection Testing                              | 64 |
| Table 5.7 Single Fruit Image Prediction                         | 66 |
| Table 5.8 Multiple Fruit Image Prediction                       | 67 |
| Table 5.9 Multi-class Fruit Image Prediction                    | 68 |
| Table 5.10 Real-time Video Prediction                           | 69 |

## LIST OF ABBREVIATIONS

|             |                               |
|-------------|-------------------------------|
| <i>LED</i>  | Light-Emitting Diode          |
| <i>CV</i>   | Computer Vision               |
| <i>ANN</i>  | Artificial Neural Network     |
| <i>CNN</i>  | Convolutional Neural Network  |
| <i>RGB</i>  | Red Green Blue                |
| <i>HSV</i>  | Hue, Saturation, Value        |
| <i>HSL</i>  | Hue, Saturation, Lightness    |
| <i>DC</i>   | Direct Current                |
| <i>GUI</i>  | Graphical User Interface      |
| <i>LCD</i>  | Liquid-Crystal Display        |
| <i>CNN</i>  | Convolutional Neural Network  |
| <i>CPU</i>  | Central Processing Unit       |
| <i>GPU</i>  | Graphics Processing Unit      |
| <i>RAM</i>  | Random-Access Memory          |
| <i>ReLU</i> | Rectified Linear Unit         |
| <i>API</i>  | Application Program Interface |

## CHAPTER 1 INTRODUCTION

### 1.1 Introduction

Fresh fruits are generally known as fresh produces in the state of freshly harvested from fruit farms or orchards. They are the fleshy products of plant growth majority of their flesh parts are edible. There are up to hundred types of fruits existing in the world and their types are distinguishable through colours, shapes and also tastes. According to Better Health Channel (n.d.), fruits can be categorized into several types or classes, which includes apples and pears, citrus, stone fruit, tropical and exotic, berries, melons as well as tomatoes and avocados. Fresh and undamaged fruits are usually safe to consume by human or animals, but what is the length of time that the fruits are able to stay fresh in various circumstances or environments?

After a certain period of time, the freshness of the fruit may not be preserved. According to Ryan (n.d.), some of the spoilt fruits have obvious appearance such as growing of mould that human can easily inspect on their surface, yet there are also certain types of fruit that human needs to take a more hands-on method such as smelling its scent or feeling its body hardness to identify whether they are beginning to spoil or no longer fresh. Moreover, given a situation that certain fruits are stored in a cool place like refrigerator, their freshness are more likely to be maintained longer as the environment in the refrigerator keeps their moisture. On the other hand, if certain fruits like apples and oranges are kept under the room temperature, it will be a doubt that how long the fruits can maintain its freshness as moulds can grow easily in such condition of open-aired and room temperature. Moulds are the microscopic fungi that live and grow on plants surface.

## 1.2 Problem Statement

- Time-consuming and manpower intensive for fruit quality assurance process.

Providing a scenario in huge fruit distribution warehouse, it consists various types of fruits and up to tons of stocks to be distributed for fruit wholesalers. First and foremost, bearing with numerous tons of stocks, the quality control process could be harsh and expensive as it will be time-consuming and high manpower requirements. Although there are many fruit wholesalers are started to involve high-end machines for daily operations such as transportations, conveying and also packaging, yet the quality assurance of the fruits are still have to be maintained in a manual approach. Therefore, great efficiency of the whole system is still unreachable.

- Lack of references or measurement towards the fruit quality for common buyers and consumers.

Every buyers or consumers may have different interpretations and observations on checking the exact freshness states of particular fruit. To illustrate, people may interpret a rotten fruit to a still edible one or a partially fresh fruit to a rotten fruit, there was no absolute measurement for consumers to determine the current freshness states of fruits. As such, buyers who accidentally bought rotten fruits from the market may suffer from loss or even sickness if he/she consumed it. This should be concerned for safety consumption measurement.



### 1.3 Background and Motivation

Before the advancement of technologies during the past decades, the inspection of fresh produces quality can only be done through the observation of external factors. Each individual may have their own perspective regarding the judgement of whether the fruit is fresh or not fresh, it is quite subjective which depends to their past experience to predict the current condition of fruits.

In this modern era, technologies have evolved rapidly and thousands of real life problems could be solved by utilising the techniques or technologies invented by the contributors. In this aspect, fresh produces can be managed in an efficient way as the inventions of fruit grading system or fruit sorting system can often be seen in the agricultural industries. With the great assistance of these systems, the industry is able to grow due to the high efficiency of harvesting and categorising. Manpower could be reduced as the sorting of fruits in a manual way is no longer practicing in the industries.

Since the current technology has been developed up to this extends, there is a possibility that the quality of fresh produces can be estimated through deep learning techniques. The spoilt part of fruits usually have obvious characteristics or appearance, they can be extracted and learned so as to build a model for further evaluation.

According to (Vahab, A, et al, 2019), object detection applications are now popular to be implemented in various industries through using deep learning and computer vision to identify ideal objects from the input of images or video streams. Human-beings can identify objects because they were taught and learnt since they were born, unlike machines, they are unable to recognize any real-world object, unless we “teach” them through deep learning and computer vision. With the machines that can identify objects from given video streaming or image like human, believing that many tasks could be performed with automation for daily operations.

Discussing about the transfer learning technique, it is a machine learning approach that repurpose the task of the pre-trained model on another related task (Brownlee, 2017). The second task could be improved in terms of performance during the second task modelling due to the optimization given by transfer learning. On top of that, transfer learning can be classified into two common types which are develop model approach

and pre-trained model approach where the pre-trained model approach is commonly used in deep learning field. Transfer learning brings three potential benefits as compared to model training without transfer learning, which includes providing higher start for the initial model training, providing higher slope where the rate of improvement will become steeper during model training and lastly is higher asymptote where the model could reach higher performance than the original peak.

The motivation of conducting this research project was to help people and provide a guidance to people who are having issues in shopping fresh produces especially fruits through deep learning methods. Since now there was no standard reference or interpretation given by machine learning that whether a fruit is fresh or rotten, therefore this project was to build a deep learning model that can identify or evaluate the fruits' current freshness states. With such algorithm, many operations could be further extended or developed like building an automated fruit freshness caring system.

### 1.4 Project Objectives

The main goal of this research project was to perform detection as well as quality predictions for fresh produces (fruits) successfully through deep learning implementations with the input of still images or real-time video. The details were being discussed as follows: -

- To build Faster R-CNN, an object detection algorithm with customized datasets to perform detection on fruit samples.

Fruits were to be detected in any locations of the inputted images or videos. If there were existing fruit domains found, bounding boxes were expected to be drawn on top of the inputted sources. Despite of single fruit or multiple fruit in the inputted source, the system was expected to detect on the fruit domains.

- To develop a multi-class classification model to perform predictions on the current freshness states of specified fruits.

Convolutional Neural Networks (CNN) were usually built and trained for image classification tasks as they were commonly applied in the analysis of visual imagery and other backend systems in image classification. The freshness states concerned in this project only ranges in between two distinct categories, “Fresh” and “Rotten”. Training and evaluation of supervised learning models were to be conducted to perform predictions of the classes and conditions on given fruit samples.

- To utilize transfer learning approach on CNNs to achieve desired performance.

Transfer learning technique can be utilized when there is limited data source for training or testing. Hence, by leveraging the knowledge or so called the pre-trained weights from the pre-trained models, the performance and accuracy achieved of networks could also be enhanced from any “build-from-scratch” networks. With pre-trained weights from existing networks, in spite of the limited training data on hand, the desired

performance will still be reachable and the computing resources and time spent could be saved efficiently.

- To integrate both detection and classification models under the same process to serve as fresh fruit recommender to users.

Detection and classification both were expected to function alternately, by the way of localizing the detected fruits, classification task will further take place to predict on the fruit class and current freshness states. Once the back-end processes were done and output has returned to the front-end users, users will get to know where the fresh fruits were actually located or placed, which fruits were rotten that they should not make decision on that particular fruit.

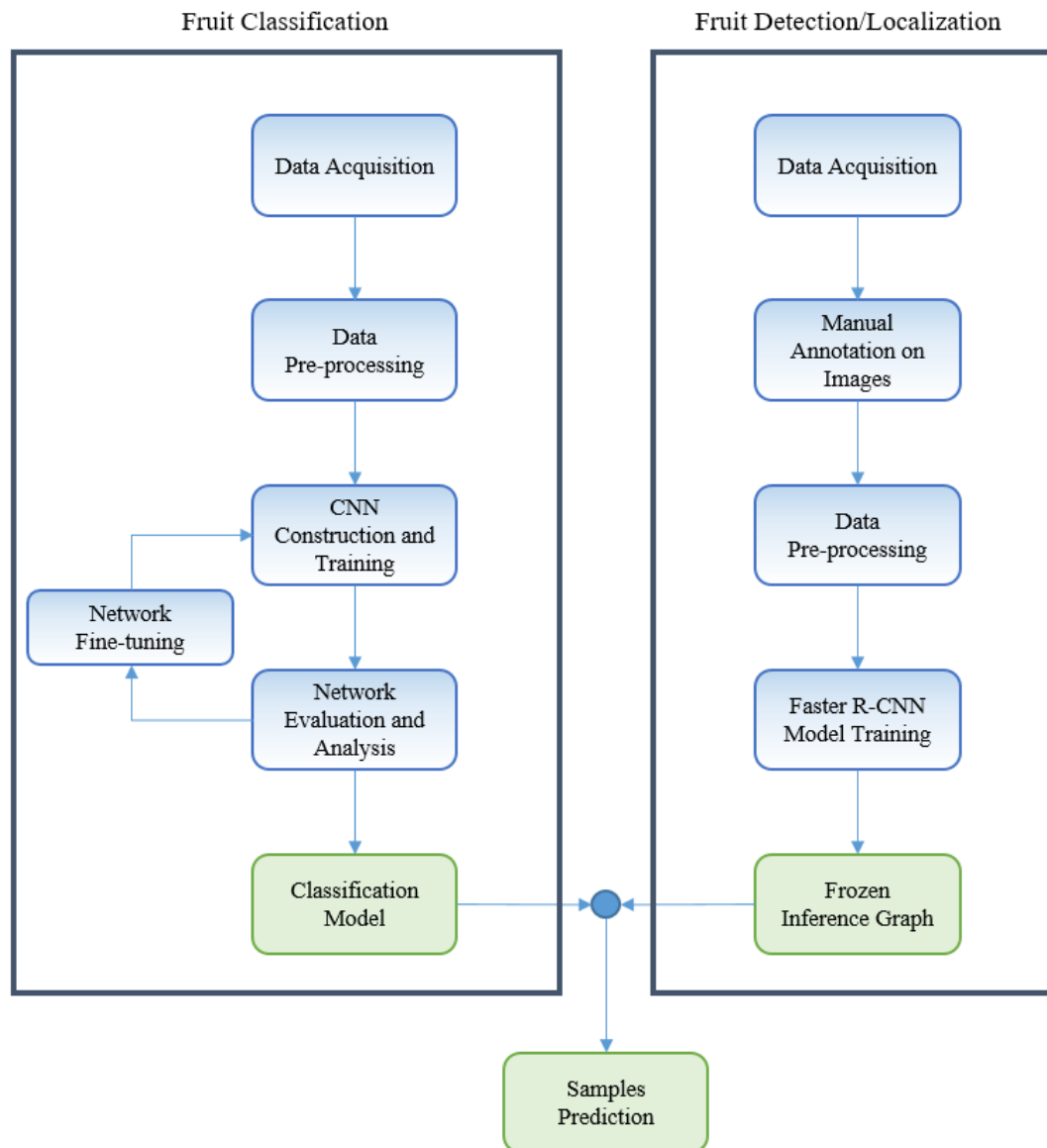
## **1.5 Impact, Significance and Contribution**

Systematic fruits retailing shops or supermarkets may have the problem in fruits storage management. Large amount and variety of fruits causes the issue of the fruits checking or examination process to be time-consuming and requires more manpower. Furthermore, if there is an existence of a spoilt fruit kept together with other fruits in a very close distance, the moulds of the spoilt fruit can be spread rapidly to other fruits as well. Subsequently, the nearby fruits will be infected and spoiled soon. The retailers may need to take much concern on the condition of fruits from time to time.

On the other hand, human as consumers are also having an issue on identifying the freshness of fruits through mere observation on fruits. If consumers consume mouldy or spoilt fruits accidentally, it might have the chance of them getting serious infection or illness. The problems are important and inevitable as fruits are considered as the necessity of daily consumption for human, nutrients, vitamins and fibre are essentials of human. To further explain on the solution applied to the real-life scenario, if consumers are able to take a snapshot of the fruit images before buying it, they can examine the current condition or freshness of the fruits taken in the photos. It could be beneficial to the consumers to avoid buying spoilt fruits after referring to the output of fruit quality prediction system on the mobile platform.

Thus, referring to the situations above for fruits retailers and consumers, a solution is needed to overcome the problem of estimating the freshness of fruits. Through the techniques of fruit quality prediction and evaluation, numerous problems could be solved in both perspectives, retailer perspective and also customer perspective.

## 1.6 Proposed Approach



*Figure 1.1 Project Development Overview*

The development was set into two different parts, one was for the part of fruit classification where it was expected to return the fruit class and current condition; while another part was for the fruit detection, it was expected that the trained model was able to identify and localize desired objects in an image or frame provided. Hence, to have a fresh fruit recommender, both models have to be integrated and functioned simultaneously. The system was expected to have three outputs return to the front-end users, the locations of the detected fruits, the condition predictions of each individual fruit and the total fruit counts value.

### 1.7 Highlight of the Project

The developed prototype was able to perform localization and make freshness predictions on each detected fruit where the fruit domains involved in this project were apple, banana and orange. For fruit detection algorithm, Tensorflow Object Detection API was used as the framework for training a custom dataset object detection model. The inference graph generated has achieved great performance that it was able to detect and localize the fruits found in the image and bounding boxes were drawn and presented to the front-end. On top of that, for fruit classification, CNN architecture was built and trained to tackle our problem, which was to recognizing fruit classes and current conditions. Transfer learning approach has been implemented by leveraging the pre-trained weights of VGG-16 model to train on our own fruit dataset. A surprising performance has achieved by the final model and it was chosen for the classification processes. To run the prototype, there were two ways of input data, through still images or through real-time video. Once the data was inputted into the system, after proceeding the back-end operations, like localization and predictions, the outputs would be presented at the front-end.

## **1.8 Report Organization**

In Chapter 1, the problem statement of this project and the motivation to conduct this research project was mentioned. Other than that, project scopes and objectives were also described, the proposed approach of project development and the highlight and achievement of the project were also mentioned.

In Chapter 2, it was about the reviews of the related past research or workings regarding fruit quality estimation and comparison in different aspect was made in a table form.

In Chapter 3, the system design was discussed where the project development flow was mentioned and a system flowchart was shown to clarify on how the system works.

In Chapter 4, discussion on the performance achieved by the CNN was made and comparison was done among them. The system specifications, system performance definition and verification plans were mentioned in this chapter.

In Chapter 5, it consisted of the process of testing, the outputs of the deliverables were shown in screenshots form and a deduction was made for this testing phase.

In Chapter 6, it consisted of the conclusion for this project, the highlighted novelties and contribution as well as the future development to be conducted were mentioned.



## CHAPTER 2 LITERATURE REVIEW

Deep learning and machine learning algorithm are widely utilized in object recognition and detection. We might emphasize on recognizing and detecting a malfunction or failure object through the applying the algorithms stated above. In our case, estimating or grading the fresh produces (fruits) will be the highlighted part in this project.

### 2.1 Related Fresh Produce Quality Estimation Techniques

#### 2.1.1 Fruit freshness detection using sensor

According to the previous research of Ozen (2018), the main working idea of the designed sensors are relied on the measurement of ion concentration of the fruit and vegetable samples given. The mechanism on function was generally controlled by four circuits, including processor supply circuit, liquid crystal display circuits and two measurement circuits plus a software. To improve the sensitivity of conducted measurements, the measurement circuit implement sensitive resistors in their design.

Three critical intervals were representing the freshness of the samples, the first interval represented the sample is fresh, the second interval represented the freshness of the sample has degraded and the third interval represented the sample has been rotten and inconsumable. Three LED lights were designed and to be displayed to users where these three lights were corresponds to their freshness, red indicated the third interval, blue indicated the second interval and lastly, green indicated the first interval.

On the other hand, two electrodes were built with the embedded measurement circuits stated above. By inserting the electrodes properly to the samples, the freshness could be measured through the ion concentration of the sample. Electrodes were expected to be inserted completely to the samples, measurement would be failed if failed to do so.

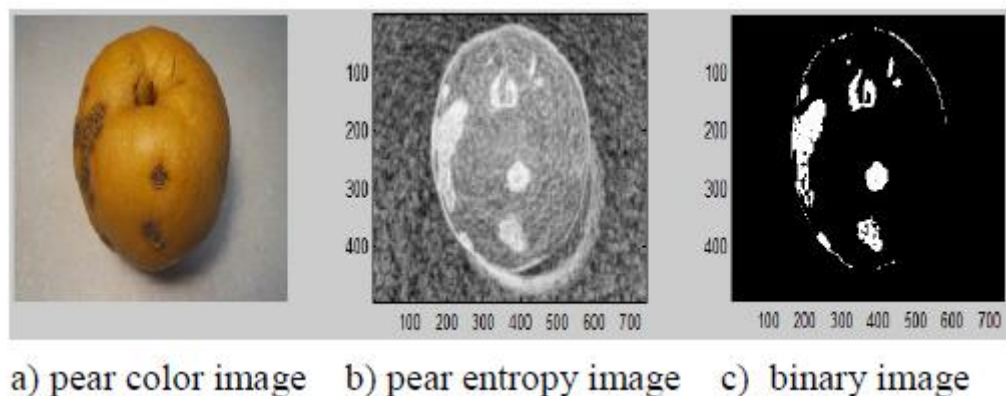


*Figure 2.1 Sensor designed for fruit freshness detection*

### 2.1.2 Real-time smart fruit quality classification system through appearance and internal flavour factors

As researched by Choi, et al. (2018), the method implemented requires the computation of the features from fruit appearance and the internal flavour factors. The fruit images were captured by CCD cameras installed in the measure chamber and undergone image segmentation process which consists of removing background image from main object as well as the outline detection of the fruit image using Sobel edge operator. Moving on, the computation of sizes, length and radius were performed. The calculation of fruit size and volume was completed using the first central moment (mean), the second central moment (variance) and the third central moment (skewness) and they compute the centre of distribution and its surrounding moment.

One of the highlighted part in this research was using entropy image analysis to detect defected fruits. According to the research, the external defects of a fruit can be estimated through its edge which can use edge extraction, colour difference on appearance using different colour distribution analysis and fruit entropy analysis. Total colour histogram of the fruit was calculated to analyse the probabilities of defection on each pixels. The defect areas have higher entropy compared to the normal areas of the fruits. According to Figure 2.2 shown below, the defect areas were presented in white when translated into binary image, it obviously differed from the non-defect areas.



*Figure 2.2 Processed image to detect defection on fruits*

Furthermore, NIR (Near Infrared) Spectroscopy Technologies was utilized in the aspect of internal quality analysis. The wavelength bands of near infrared is from 800 nm to

2.5  $\mu\text{m}$ . According to Yoon et al. (2014), reflected or transmitted light were to be analysed after irradiating near infrared rays on fresh produces can measure the internal conditions such as moisture level, sugar level, acidity and internal defection.

Artificial Neural Network (ANN) was used as the fruit grade classifier in this research. ANN feed-forward model was proposed to grade fruits by external appearance and internal flavour factors. Figure 2.3 shows the ANN based classification model for fruit grading system. Supervised learning model was built with 8 nodes (features) on input layer, 2 hidden layers and 3 nodes (fruit grades) on output layer.

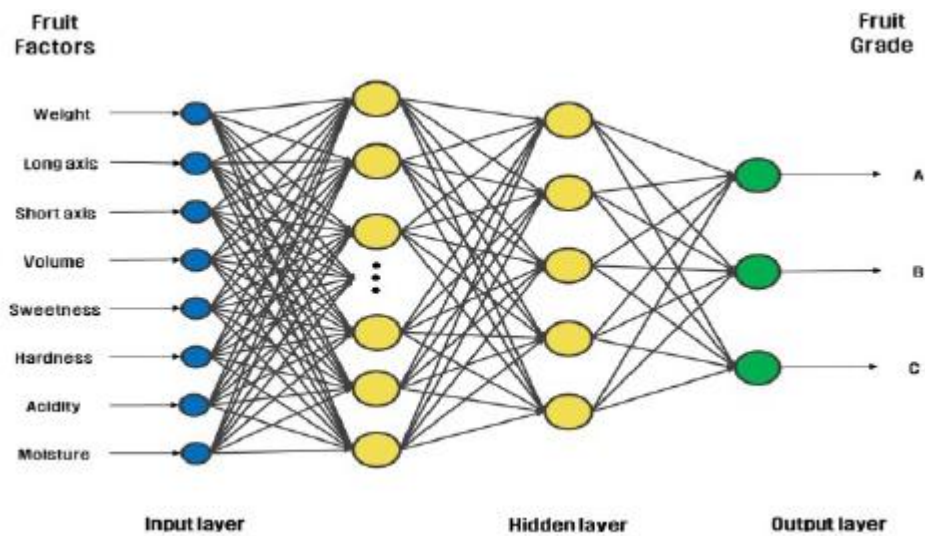
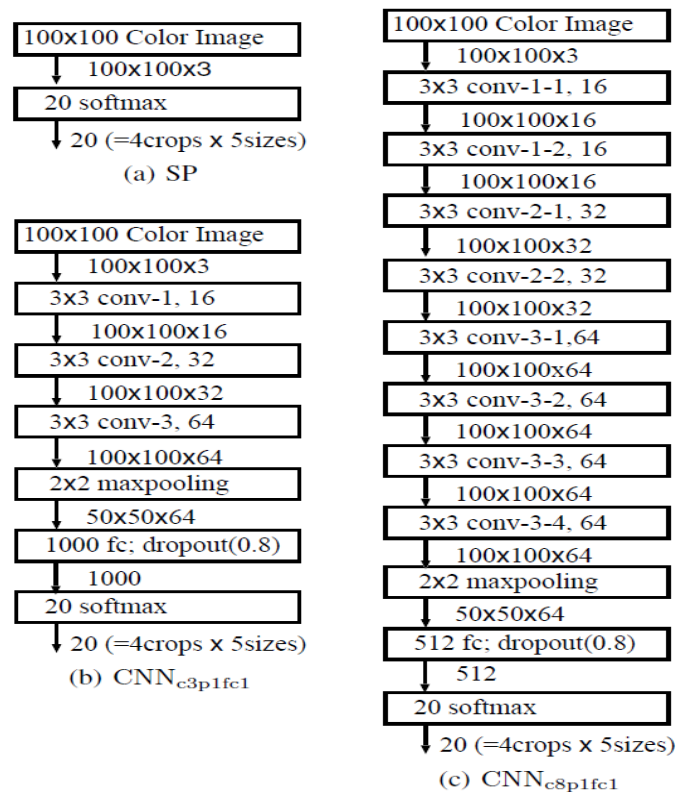


Figure 2.3 Classification model based on ANN for fruit grading system

### 2.1.3 Fresh produces grading using RGB-D and CNN

By referring the research of Nishi, Kurogi & Matsuo (2017), the proposed method was fruits and vegetables grading using RGB-D (RGB and depth) images along the implementation of Convolutional Neural Network (CNN). In this research, size was the main focus on the grading system.

During the first step, the proposed approach transformed the positions of pixels in RGB images so that the main object in 3D space would be placed at the position intermediately from the focal point, meaning of using the corresponding depth image. Moving on, three neural networks were constructed, a simple neural network (SP) and two CNNs, one of them was  $CNN_{c3p1fc1}$  which involved 3 convolutional layers, 1 max pooling layer and a fully connected layer, and another one was  $CNN_{c8p1fc1}$  which the settings were almost the same except that it was constructed with 8 convolutional layers. Adam optimizer was used as the learning algorithm. These neural networks were implemented in the Chainer framework. The reason of consideration on utilizing one max pooling layer was because of resizing the original image size to smaller sizes, from 100 x 100 into half of it, 50 x 50. Figure 2.4 shows the three neural networks stated above.

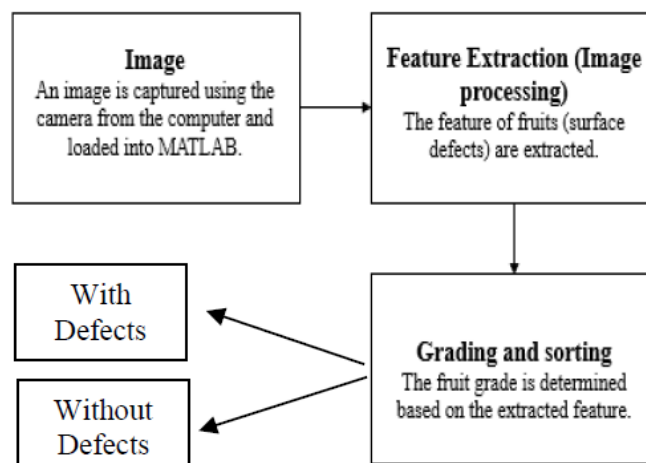


*Figure 2.4 One simple neural network and Two CNNs.*

To achieve rotation invariance, they used rotated images of the objects to train the networks which they rotated the images of the objects on y-axis (vertical direction) and z-axis (depth direction). As for the appearance of the objects, they emphasized on colours too, the performance of three colour spaces were evaluated, RGB, HSV and HSL.

#### 2.1.4 Classical automated fruit grading system process using MATLAB

According to the research by Ali & Wong (2017), an automated fruit grading system was proposed which MATLAB was the platform chosen for image processing. The general workflow of the fruit grading process is shown below.



*Figure 2.5 Flowchart of fruit grading process*

A simple workspace was built using simple image processing equipment for this project which they used the camera from PC to capture the lateral surface of fruits, a rotating desk purposed on placing the fruits which was connected to a 12V DC motor to enable auto rotation of rotating desk. The DC Motor was set by Arduino to rotate 180° twice so that the system can detect the first half and the second half external condition of the fruits placed on the rotating desk. Figure 2.6 shows the illustration of the fruit grading system.

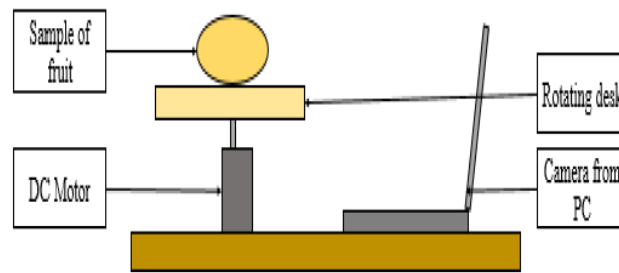
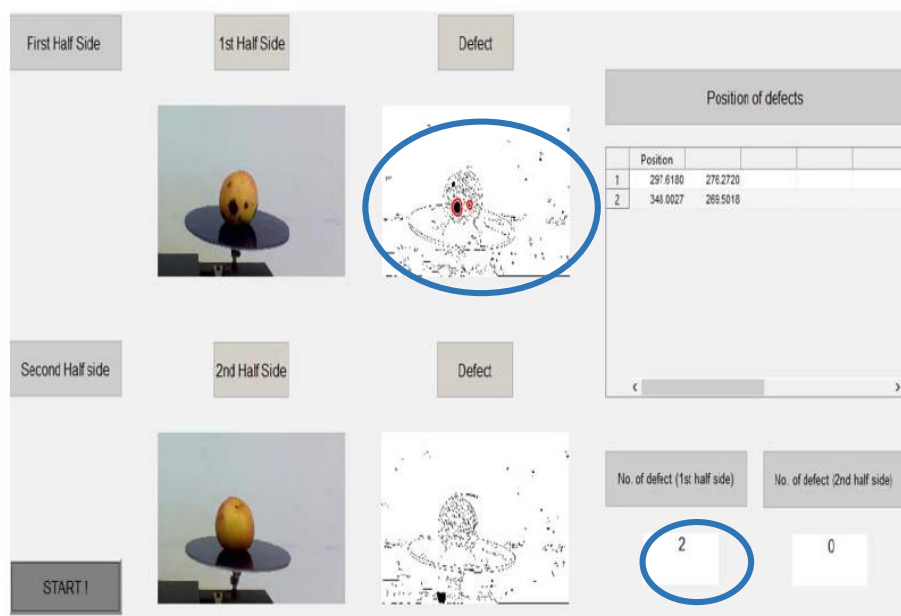


Figure 2.6 Illustration of automated fruit grading system

An image of fruit was captured and loaded into MATLAB workspace. The coloured input were required to be converted into grayscale using the function of “`rgb2gray(image)`” to transform true colour image RGB into grayscale intensity image and eventually transform it into binary image. Canny edge detection approach was applied to detect the edge of the fruits as well as to extract the boundary. Moreover, to handle the texture of the image of fruit, the discovered defect areas on the surface of fruits will be filled with a colour of black using the function of “`imfill()`”. Also, they worked on the drawing of red circle if any defect areas spotted from the analysed image, it will be presented on the GUI, showing the spot of the defect areas and number of defect. Figure 2.7 shows one of the example tested with the manually built workspace and presented on GUI with expected results.

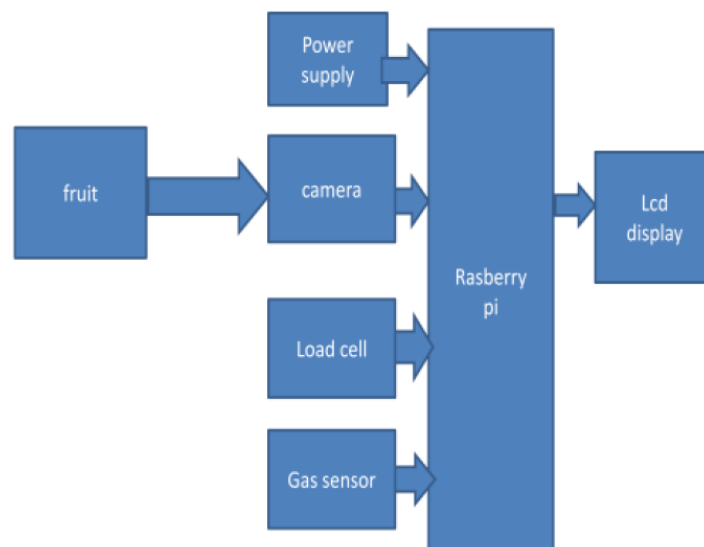


*Figure 2.7 GUI presented with fruit defection*

### 2.1.5 Fruit freshness detection using Raspberry Pi

This research was done by Jayasankar, et al. (2018) and it mainly described on how the freshness of fruits can be measured using Raspberry Pi. Several important components and concepts were involved in this research, Raspberry Pi 0W, Digital Image Processing, proximity sensor, load cell and gas sensor. Digital Image Processing was the application of algorithms to perform image processing on datasets or images given, open CV method has been used in this research. It was stated that edge detection algorithm has performed for the processing of digital images.

The program was executed in Raspberry Pi 0W under Raspbian OS. The system started from capturing the fresh produces images and the images were to be transmitted to process in open CV. Significant fruit features like colour, shape and size of fruit samples were extracted. Moving on, the captured image will be segmented using edge detection algorithm. Weight of the fruit samples can be identified using load cell and the average of the weight can be taken as one of the parameters under consideration for the fruit freshness measurement. Gas sensor was utilized to detect the gas present or sprayed in the fruits. The freshness percentage for different fruits would be presented and displayed through LCD display. Figure 2.8 shows the block diagram of the general workflow for this research.

*Figure 2.8 Project Equipment Setup*

## 2.2 Critical Remarks and Comparison between Reviewed Techniques

Each of techniques used have their own strengths and weaknesses. One of the obvious difference was the sensor detection requires the physical contact between the samples and the hardware while the others which utilise various image processing or deep learning techniques does not require real physical contact. Generally, only a few samples that they have chosen to include as their training and testing datasets, not all the variety of fruit samples. Furthermore, it can be deduced that the measurement of fruit quality is mostly based on their external appearance, this is due to the reason that the concern of the reviewed papers were focusing on the appearance of the fruit samples. Table 2.1 shows the summary of the reviewed paper.

*Table 2.1 Comparison among Reviewed Techniques*

| Index | Techniques                  | Training and Testing Factors     | Measuring / Evaluation unit | Accuracy achieved | Parameter                        | Datasets  |
|-------|-----------------------------|----------------------------------|-----------------------------|-------------------|----------------------------------|---|
| 2-1-1 | Sensor Detection            | Ion concentration of fruits      | Freshness threshold values  | Not stated        | -                                | Lemon, orange, strawberry, tomato, potato, pepper, apple and banana |
| 2-1-2 | Entropy Image Analysis, NIR | External appearance and internal | Grades (A, B, C)            | 97.4%             | 8 input nodes (fruit factors), 2 | Korean pears  |

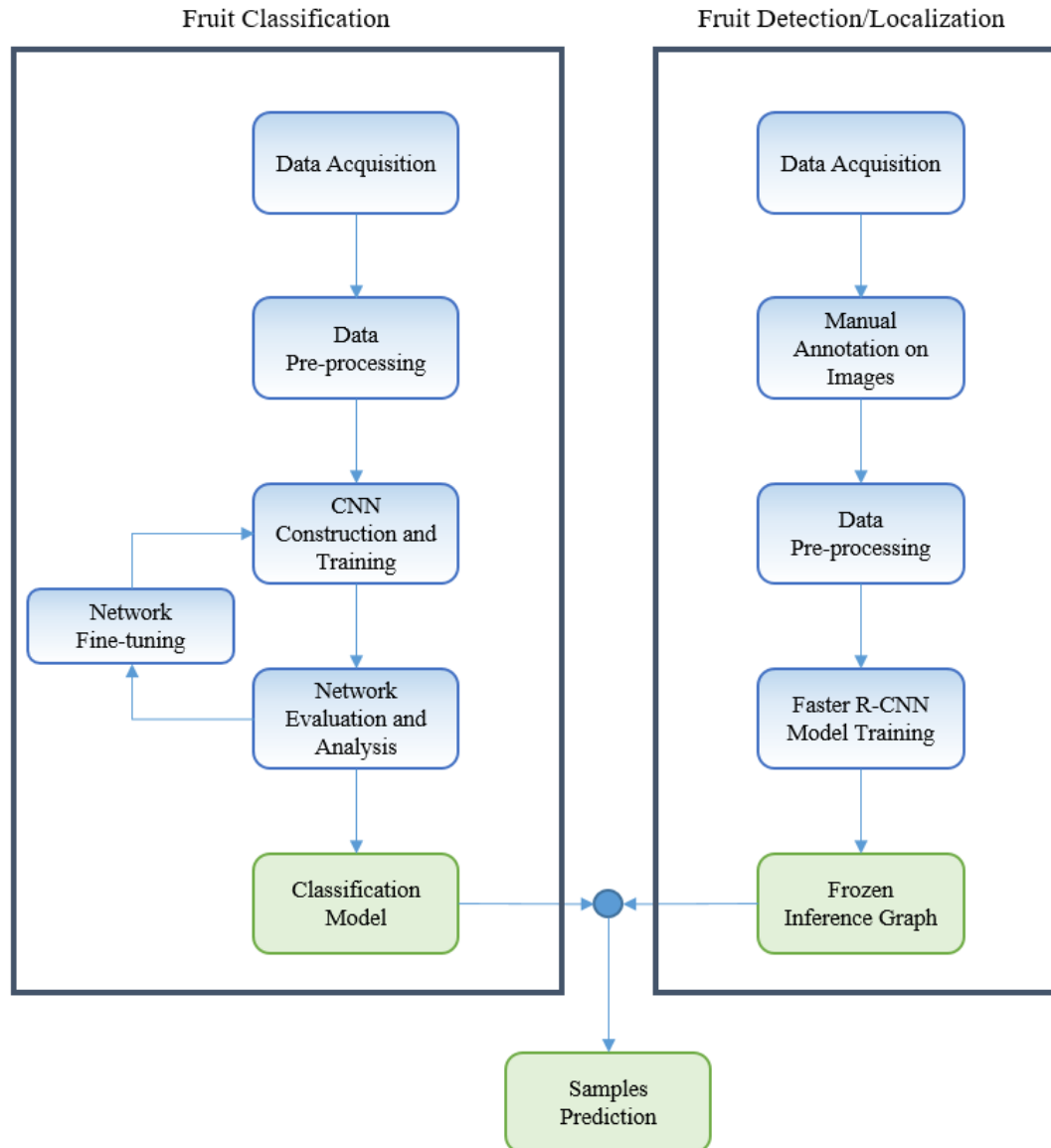


## CHAPTER 2 LITERATURE REVIEW

|       |   |                                 |   |               |  |  |
|-------|---|---------------------------------|---|---------------|--|--|
|       | Spectroscopy<br>Technologies, ANN         | flavour<br>factors              |   |               | hidden<br>layers, 3<br>output<br>nodes   |  |
| 2-1-3 | RGB-D and<br>CNN                          | Colour and<br>depth of<br>image | Sizes   | Not<br>stated | Simple<br>neural<br>network,<br>3<br>convolutional<br>layers,<br>and 8<br>convolutional<br>layers, | Apple,<br>bell<br>pepper,<br>orange<br>and<br>tomato |
| 2-1-4 | Automated<br>detection<br>using<br>MATLAB | Surface<br>defects or<br>decays | Position of<br>defects<br>and<br>number of<br>defects | Not<br>stated | -  | Apple and<br>mango                                   |
| 2-1-5 | Raspberry<br>Pi                           | Colour,<br>shape, size          | Freshness<br>Percentage                               | Not<br>stated | -  | Not stated   |

## CHAPTER 3 SYSTEM DESIGN

### 3.1 System Development Overview and General Work Procedures



*Figure 3.1 System Development Overview*

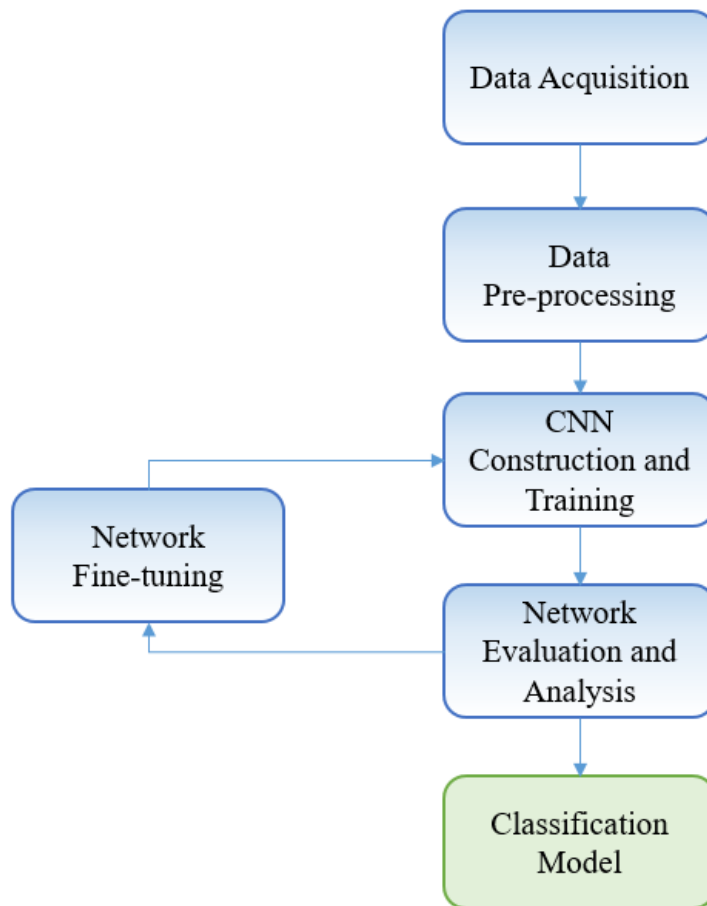
The general idea of the project was to predict the freshness states of fruits from inputted still images or real-time video with the implementation of machine learning. The system block diagram above has shown the structure of the overall project development. Generally, the development mainly split into two different tasks or phases, fruit classification and fruit detection or localization. To get desirable performance from prediction results, each of these activity blocks shown were the significant phases that

need to be implemented in a sequential way, starting from researching and collection of data to conduct evaluation and analysis.

For fruit classification task, the fruit classes involved were apple, banana and orange. Therefore, to make predictions and evaluations on the freshness states of these mentioned fruit classes, the raw images collected must have the features of their fresh and rotten states so as to ensure the neural networks get to learn significant features from them. For fruit detection task, the data collected were almost similar to the classification task, it ranged from single fruit image to multiple fruit image under the same fruit classes. Before proceeding to the detection algorithm, annotations must be made for each image to get the coordinates or location of fruit founded in that particular image. To ensure the detection results were successful, transfer learning from pre-trained detection models could be utilized, the pre-trained weights of models were open-source to be downloaded, developed by Tensorflow.

After the training phases, a CNN classification model and a frozen inference graph for detection were expected to run inference on both still images and real-time video execution with desirable outcomes.

### 3.2 Fruit Classification



*Figure 3.2 Fruit Classification Development Block Diagram*

The system block diagram above has shown the work procedures on developing CNN classification model. More details were to be discussed in the following sections.

#### 3.2.1 Data Acquisition

First things first, data collection was one of the important phase because the outcomes of the following phases were highly dependent on the data collected for the usage on network training and evaluation. For fruit classification task, raw RGB images were collected from the source of Kaggle.com, author named Kalluri, S. R. The dataset

consists of SIX classes, where these were the fruit domains covered in this project, “Fresh Apple”, “Fresh Banana”, “Fresh Orange”, “Rotten Apple”, “Rotten Banana” and “Rotten Orange”. The data acquired was exactly suitable for modelling our problem domain, having three types of fruits as well as having two distinguishable freshness states, ‘Fresh’ and ‘Rotten’. On top of that, the acquired images were all single fruit images from all sides of capturing, top, bottom and middle parts.

### **3.2.2 Data Pre-processing**

After acquiring datasets from relevant source, file paths declarations towards respective classes were done beforehand. Next, dataset was split into training, validation and testing sets with the size of (8640, 1080, 1080), ratio 8:1:1. Each class contained 1440 training images, 180 validation images and 180 testing images. To input of neural network requires a specific input size of images, therefore, all images were then loaded into numpy arrays with the targeted size of (150, 150). Class labels were encoded using Label Encoder which corresponds to their respective classes.

### **3.2.3 Network Architectures Construction and Training**

With Tensorflow and Keras libraries, building neural network architectures could be done easily. Adding convolutional layers, pooling layers and fully connected layers in a proper sequence with specified input dimensions and output dimensions was very important in order to have a great performance classification model. Adding more layers will just deepen and increase the complexity of the network and requires more time and resource for compiling and training. Besides, the parameters set in the layers such as filter size, kernel size and activation function should be set properly.

To achieve desired performance from this classification model, the simple network constructed was insufficient to reach the expectations. Thus, regularization techniques were also utilized in network constructions and refining training data. Dropout layers were added in between fully connected layers, the existing nodes in the fully connected layers would have a probability to be dropped from the continuation of network training so as to prevent over-fitted problems. For all dropout layers in any constructed networks,

the dropout probability was set to 0.3, meaning it has 30% of chance being dropped out from the network.

On top of that, transfer learning method was attempted and implemented for network training. Instead of building and training the network from scratch, leveraging the pre-trained weights from existing pre-trained models will often achieve desirable outcomes. There are various pre-trained models available on Keras, such as VGG, ResNet and InceptionV3. By fine-tuning specific layers like input and output layers, setting with pre-trained weights and prepared with sufficient training data, the model could be trained to satisfy or solve the particular tasks.

### **3.2.4 Network Evaluation and Analysis**

To check and verify whether a model was well-trained, the observation on the accuracies and losses in each epoch was conducted throughout the training process. Great accuracy as well as low loss score were inevitably significant for great performance model. Overfitting and underfitting problems of the models were usually caused by the reasons of either the network was constructed excessively complicated or overly simple. Poor performance can be observed obviously from the models such that the predicted results will not always be tallied with the expected results.

Therefore, to observe the pattern of the trained models, Matplotlib library was used to plot the accuracy and loss points across each training epoch during the CNNs training. Through analysing the pattern shown in the graph, the results will indicate the performance of the networks, whether it was underfitting, overfitting or just right. With the reference of the plotted graphs, the fine-tuning of the classification model architecture will be necessary to make improvements on the network.

Moreover, computing the confusion matrix can evaluate the network performance as well. It summarized the overall predictions of the classification based on the trained models. With confusion matrix, true positive (TP), true negative (TN), false positive (FP) and false negative (FN) were resulted after the computation and these values can be brought forward so that classification rate or accuracy scores can be calculated.

Table 3.1 Confusion Matrix for Fruit Classification Predictions

|        |               | Predicted         |                   |                   |                   |                    |                   |
|--------|---------------|-------------------|-------------------|-------------------|-------------------|--------------------|-------------------|
|        |               | Fresh Apple       | Fresh Banana      | Fresh Orange      | Rotten Apple      | Rotten Banana      | Rotten Orange     |
| Actual | Fresh Apple   | P <sub>FAFA</sub> | P <sub>FBFA</sub> | P <sub>FOFA</sub> | P <sub>RAFA</sub> | P <sub>RBFA</sub>  | P <sub>ROFA</sub> |
|        | Fresh Banana  | P <sub>FAFB</sub> | P <sub>FBFB</sub> | P <sub>FOFB</sub> | P <sub>RAFB</sub> | P <sub>RBFB</sub>  | P <sub>ROFB</sub> |
|        | Fresh Orange  | P <sub>FAFO</sub> | P <sub>FBFO</sub> | P <sub>FOFO</sub> | P <sub>RAFO</sub> | P <sub>RBFO</sub>  | P <sub>ROFO</sub> |
|        | Rotten Apple  | P <sub>FARA</sub> | P <sub>FBRA</sub> | P <sub>FORA</sub> | P <sub>RARA</sub> | P <sub>RBRA</sub>  | P <sub>RORA</sub> |
|        | Rotten Banana | P <sub>FARB</sub> | P <sub>FBRB</sub> | P <sub>FORB</sub> | P <sub>RARB</sub> | P <sub>RB RB</sub> | P <sub>RORB</sub> |
|        | Rotten Orange | P <sub>FARO</sub> | P <sub>FBRO</sub> | P <sub>FORO</sub> | P <sub>RARO</sub> | P <sub>RBRO</sub>  | P <sub>RORO</sub> |

Other than that, a classification report could be generated to conduct further analysis. By generating this report, scikit-learn library was imported to measure the overall performance of predictions from a classification model. Precision, recall, f1-score as well as the overall accuracy achieved of this model will be computed as shown.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

$$Accuracy = \frac{True\ Positive + True\ Negative}{TruePositive + TrueNegative + FalsePositive + FalseNegative}$$

### 3.2.5 Model Fine-Tuning

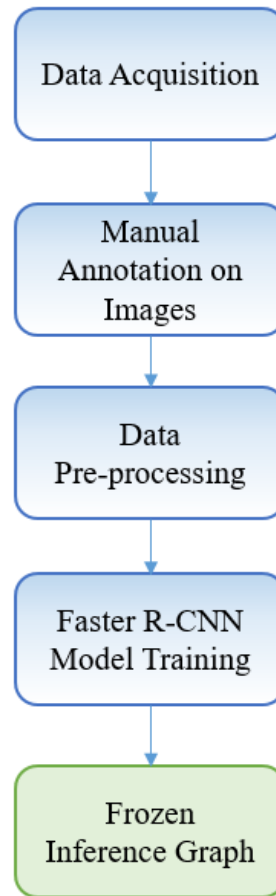
Fine-tuning models would be necessary for the project development. By adding or removing convolutional layers, observing how the models were being trained in terms of accuracy percentage and loss rate. This process could be dependent to the results from the model evaluation process. There were chances that the overfitting models were trained in a complex CNN architecture, perhaps reducing some unnecessary layers may reduce the overfitting problem to smaller extent. It would be the similar way to solve the underfitting models by adding convolutional and pooling layers.

### 3.2.6 Samples Prediction

In this phase, some of the fruit images gathered from internet or external sources were being tested with the classification models to predict on respective classes. The output results would be in the range of predefined labels.



### 3.3 Fruit Detection/Localization



*Figure 3.3 Fruit Detection Development Block Diagram*

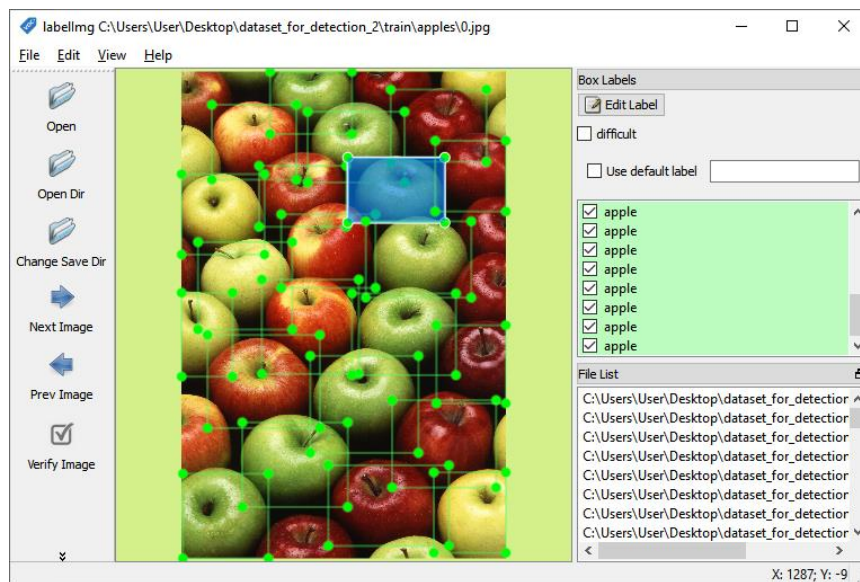
The system block diagram above has shown the work procedures on training Faster R-CNN detection model. More details were to be discussed in the following sections.

#### 3.3.1 Data Acquisition

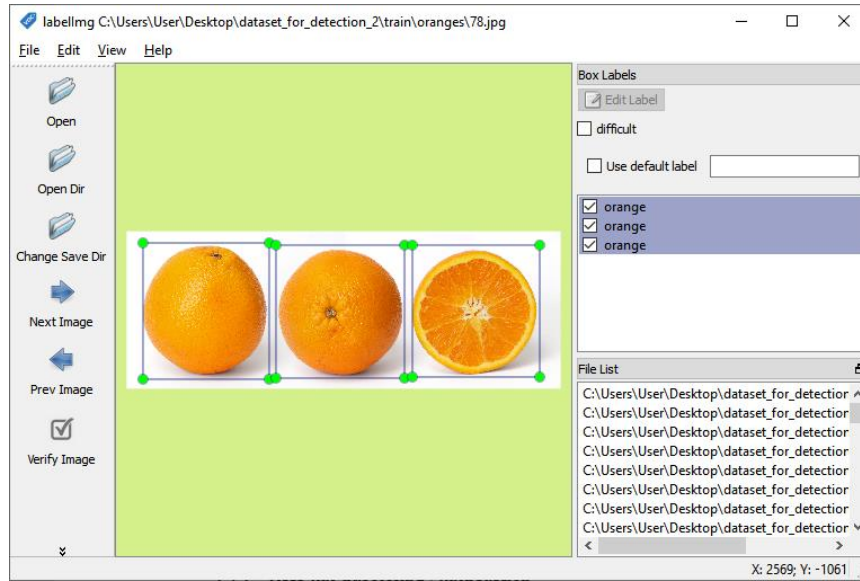
The dataset was collected as named FIDS30, prepared by Marko. The raw fruit images acquired were in RGB with random sizes, ranged from single fruit images to multiple fruit images. The specific fruit domains involved were extracted from other insignificant classes, only involving “Apple”, “Banana” and “Orange”.

### 3.3.2 Manual Annotation on Images

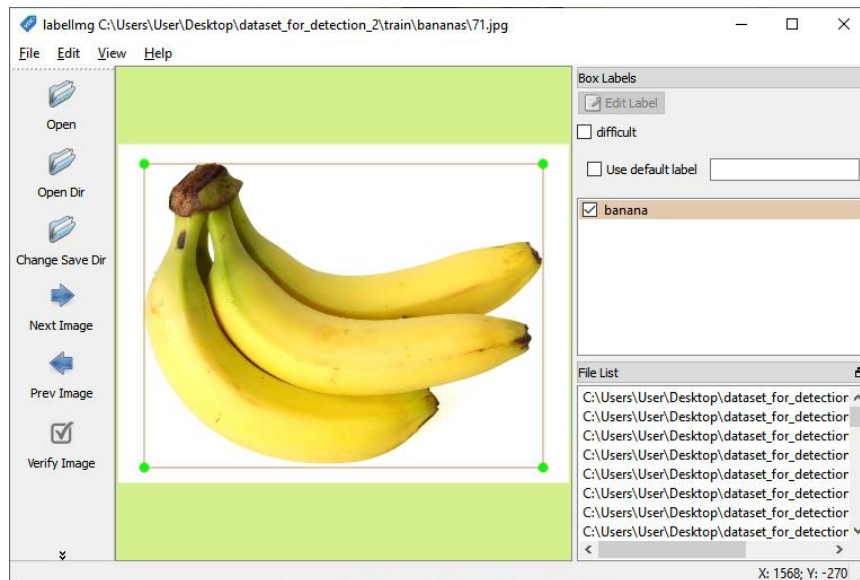
To standardize the raw images size, all images were resized to (800, 600). From the raw fruit images acquired, bounding boxes were annotated manually on each fruit found in images. After performing annotations on each image, all details of each image were being saved into xml files, representing that particular image file. The xml file consisted of the original image filename, the image size includes width, height and depth, coordinates of bounding boxes, xmin, ymin, xmax and ymax with respective fruit labels, representing the location of fruits in that particular image. The xml files were important for further processes. This phase was conducted with the application named “LabellImg”, an annotation tool that can generate xml files after annotating bounding boxes on inputted images.



*Figure 3.4 Example of Apple Annotations*



*Figure 3.5 Example of Orange Annotations*



*Figure 3.6 Example of Banana Annotation*

### 3.3.3 Data Pre-processing / Preparation

Each individual xml file generated from fruit images were then unified into one csv file (annotation file) with details sequentially listed. The annotation and images files were then together converted into TFRecord files which corresponded to train.record and test.record. They were both binary files which contains the encoded image (.jpg) file and annotated bounding box details, storing them into each one train and test files would

fasten the data loading process for later steps. Other than that, a label map file was also generated to map every object class names to an integer, which represented the encoded fruit image files respectively, such as 1 to 'apple', 2 to 'banana', and 3 to 'orange'. At the end of this process, there were three important files that we should get through previous mentioned steps, including train and test TFRecord files and label map file.

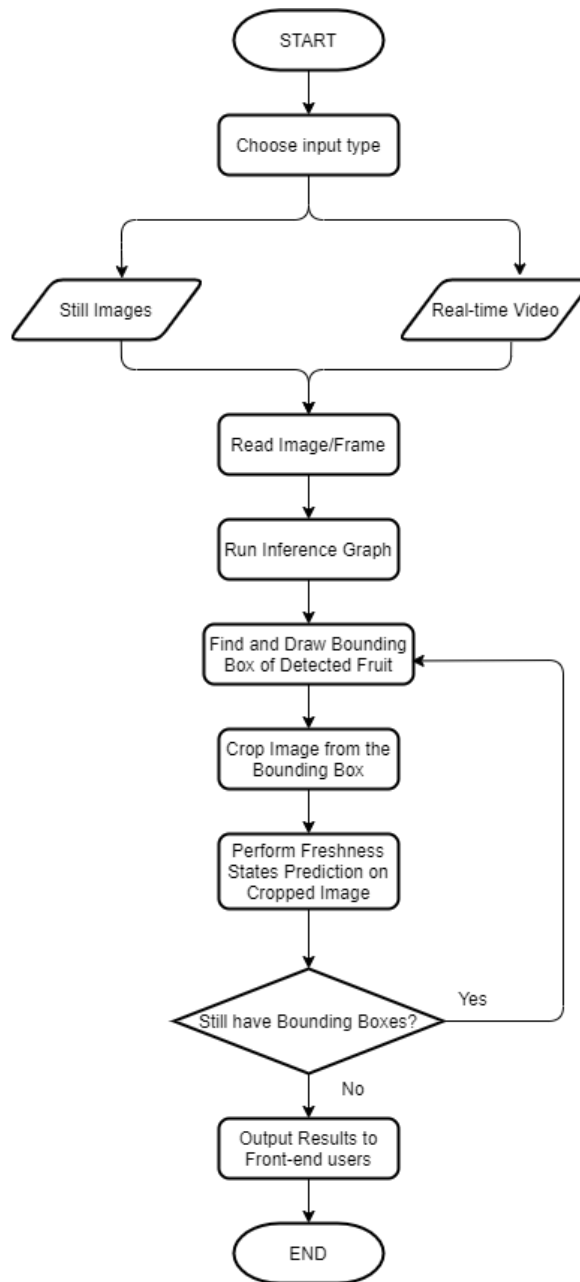
### **3.3.4 Faster R-CNN Object Detection Model Training**

In this phase, Tensorflow Object Detection API was implemented, with customized datasets, a detection model that resolve on the specified fruit domain detection can be built. Instead of training the detection model from scratch, transfer learning approach was utilized the pre-trained weights from a Faster R-CNN object detection model, Inception V2. The object detection API was an open-source framework that has already developed on top of Tensorflow as template for other future usages. Therefore, with sufficient training data of raw images with combined annotation file, compiling and training the network could be done easily by modifying simple configurations such as the output classes of inference graph

### **3.3.5 Run Frozen Inference Graph**

The inference graph generated was ready for object detection testing. Provided testing images to run on this inference graph, the expected output would be each fruit domain detected were drawn with bounding boxes, appended with their respective class names and the detection scores.

### 3.4 System Flowchart



*Figure 3.7 System Flowchart*

After getting both classification and detection models done, the next step was to program the sequence of executing both models to do their particular job. Therefore, to illustrate how the system works, an illustration of the system flowchart shown above has clearly indicated the process flow step by step, from start to end of process. This would be the core part of the whole project, outputs were to be observed at the end of

program execution. At the front-end, users can expect to observe the detected fruit classes, its current condition and the cumulative fruit counts outputted from the system.

### 3.4.1 Input Data (Still Images / Real-time Video)

Firstly, users have to choose the input type in between still images or real-time video for the freshness states prediction. If still images were chosen, the system could predict and return the outcome in few seconds, because still images were counted as one frame only, the program only needs to handle on one frame, thus the prediction process would be faster. As compared with real-time video input, the prediction process would be continuously executed until the user terminates the process manually. Due to the frame-by-frame process taken in the system, the system supported machine needs to have sufficient computing power and resource for high load computations in continuity.

### 3.4.2 Run with Frozen Inference Graph

Moving the next step, after the input data was chosen, the data would be read into Numpy arrays where the dimensions of data such as width, height and dimensionality were kept originally. After that, the image or frame proceed to run with the frozen inference graph, which was the well-trained fruit detection model. There were several outputs generated and packed into dictionary keys from the inference graph, but we only take concern on the significant ones, such as **the scores upon object detection** and **the normalized coordinates of bounding boxes on detected objects**. The score was set to a threshold value of 0.70, meaning that if the detection score was unreached to this threshold, the particular detection would be abandon automatically due to the unconvincing results. For those detection scores which exceeded 0.70, they were appended to a score list for further usage. Because of the output was in dictionary keys, each of the qualified score has their respective index, pointing its location in this dictionary class. Therefore, with the score list, finding each selected object would be rather easier by iterating their indexes. In short, the detection scores was filtered and appended into a list so to find the respective indexes which having good detection scores, and append all into an index list for next steps.

### 3.4.3 Search Bounding Boxes

Now, having an index list, the normalized coordinates of bounding boxes could be found through indexing. The acquired data from the dictionary key were xmin, xmax, ymin and ymax, they were all coordinates in normalized form, so they were not appropriate to be plotted directly on the loaded image.

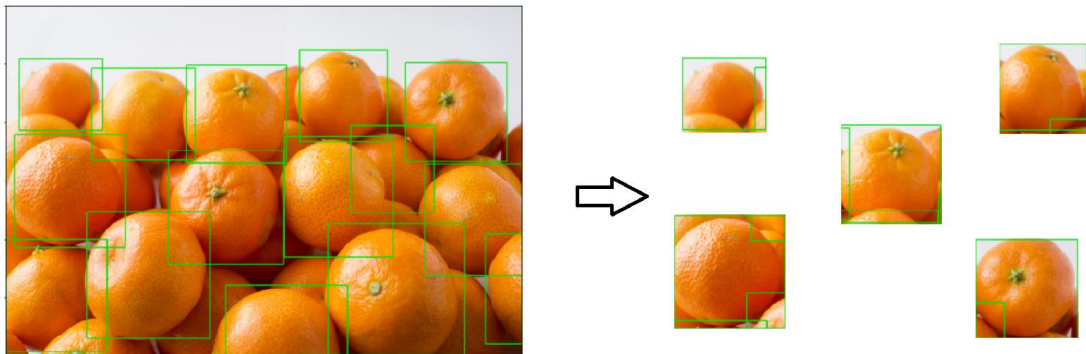
$$\text{Left} = \text{xmin} * \text{Original Image Width}$$

$$\text{Right} = \text{xmax} * \text{Original Image Width}$$

$$\text{Top} = \text{ymin} * \text{Original Image Height}$$

$$\text{Bottom} = \text{ymax} * \text{Original Image Height}$$

The normalized coordinates were then converted to the actual coordinates and they were all eligible to be plotted. As shown above, there were four coordinates, meaning that it can form into a square shape. To justify this, the detected fruit is now bounded with the square shape, therefore to make freshness states prediction on each fruit individually, each of these square needs to be cropped out.



*Figure 3.8 Example of Cropped Images from Original Image*

#### **3.4.4 Perform Freshness States Prediction with CNN Classification Model**

In this stage, the cropped images were then proceeded with predictions with the well-trained classification model. The predictions returned from the model were to be converted into text form and printed above each bounding box respectively, representing their exact fruit class and current condition.

#### **3.4.5 Return Output to Front-end**

The outputs were the fruit class and freshness states prediction, bounding boxes drawn on detected fruits and the cumulative fruit count value.



## CHAPTER 4 SYSTEM IMPLEMENTATION

### 4.1 Evaluation and Analysis on Convolutional Neural Network Architectures

The following table shows how the dataset was managed into different sets.

*Table 4.1 Data Splitting Table*

| <b>Data</b>     | <b>Quantity / Size</b> |
|-----------------|------------------------|
| Training Sets   | 8640                   |
| Validation Sets | 1080                   |
| Testing Sets    | 1080                   |

The following table shows the important parameters, hyperparameters or information regarding the CNNs that would be used for model training.

*Table 4.2 Parameters with Respective Values*

| <b>Parameters / Hyperparameters</b> | <b>Value</b>              |
|-------------------------------------|---------------------------|
| Input shape of data/images          | (150, 150, 3)             |
| Batch size                          | 30                        |
| Training epochs                     | 30 or 100                 |
| Output classes                      | 6                         |
| Loss function                       | Categorical Cross Entropy |
| Optimizer                           | Adam                      |
| Learning rate                       | 0.001                     |

The performance metrics used to evaluate the models were as follows:

- Accuracy
- Precision
- Recall
- F1-Score

### 4.1.1 Simple CNN Build-from-scratch

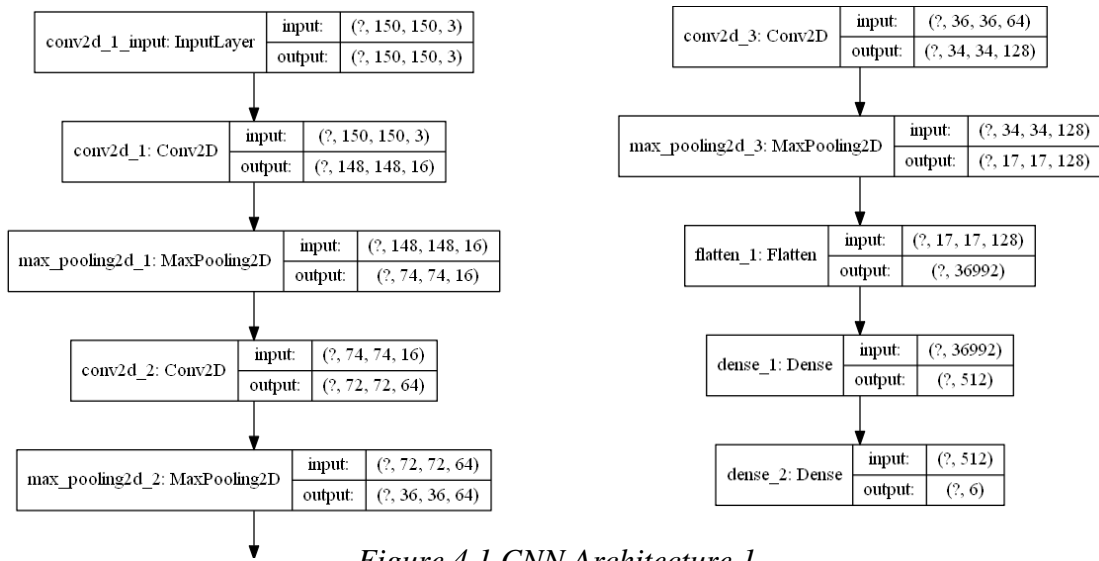


Figure 4.1 CNN Architecture 1

For the network shown above, it was built simple with three convolutional layers for significant feature extractions, followed by two max pooling layers in between of convolutional layers for down sampling the output feature maps. One fully connected layer (flatten layer) to flatten output from the last convolutional layer and proceeded to dense layers to acquire final output of the network, which are the fruit classes involved.

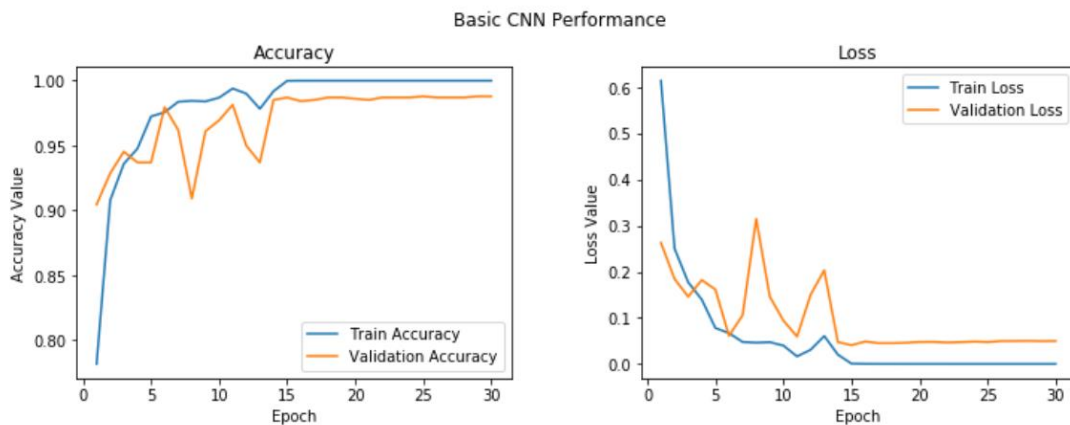


Figure 4.2 CNN 1 Performance Line Graph

The figure above shows how the CNN model was trained. Through observing the line graph patterns, the train accuracy and loss were normal and fine throughout 30 epochs. Yet, as observing the validation accuracy and loss, the pattern of both graphs were not

consistently growing like train. From the 15<sup>th</sup> epoch, the line graph started to following like train accuracy and loss, inference can be made that starting from the 15<sup>th</sup> epoch, the model was started to memorizing instead of learning features. Therefore, this basic CNN model was overfitted. To tackle with newly met data, this model may not be capable to handle, poor performance resulted with evaluation on new data.

*Table 4.3 Confusion Matrix for CNN 1*

|        |               | Predicted   |              |              |              |               |               |
|--------|---------------|-------------|--------------|--------------|--------------|---------------|---------------|
|        |               | Fresh Apple | Fresh Banana | Fresh Orange | Rotten Apple | Rotten Banana | Rotten Orange |
| Actual | Fresh Apple   | <b>180</b>  | 0            | 0            | 0            | 0             | 0             |
|        | Fresh Banana  | 0           | <b>180</b>   | 0            | 0            | 0             | 0             |
|        | Fresh Orange  | 1           | 0            | <b>174</b>   | 1            | 1             | 3             |
|        | Rotten Apple  | 1           | 1            | 1            | <b>175</b>   | 0             | 2             |
|        | Rotten Banana | 0           | 1            | 0            | 1            | <b>178</b>    | 0             |
|        | Rotten Orange | 0           | 0            | 1            | 3            | 0             | <b>176</b>    |

*Table 4.4 Precision, Recall and F1-Scores for CNN 1*

|               | Precision | Recall | F1-Score | Support |
|---------------|-----------|--------|----------|---------|
| Fresh Apple   | 0.99      | 1.00   | 0.99     | 180     |
| Fresh Banana  | 0.99      | 1.00   | 0.99     | 180     |
| Fresh Orange  | 0.99      | 0.97   | 0.98     | 180     |
| Rotten Apple  | 0.97      | 0.97   | 0.97     | 180     |
| Rotten Banana | 0.99      | 0.99   | 0.99     | 180     |
| Rotten Orange | 0.97      | 0.98   | 0.98     | 180     |
| Total No.     |           |        |          | 1080    |

*Table 4.5 Performance Matrix Table for CNN 1*

| <b>Performance Metrics</b> | <b>Score</b> |
|----------------------------|--------------|
| Accuracy                   | 0.9843       |
| Precision                  | 0.9843       |
| Recall                     | 0.9843       |
| F1-Score                   | 0.9842       |

This network seemed to have quite accurate result, yet the observable floating scenario of validation accuracy and loss were unacceptable for great performance classification. The model was memorizing the data instead of learning significant features, presenting overfitting problem.

### 4.1.2 CNN with Regularization

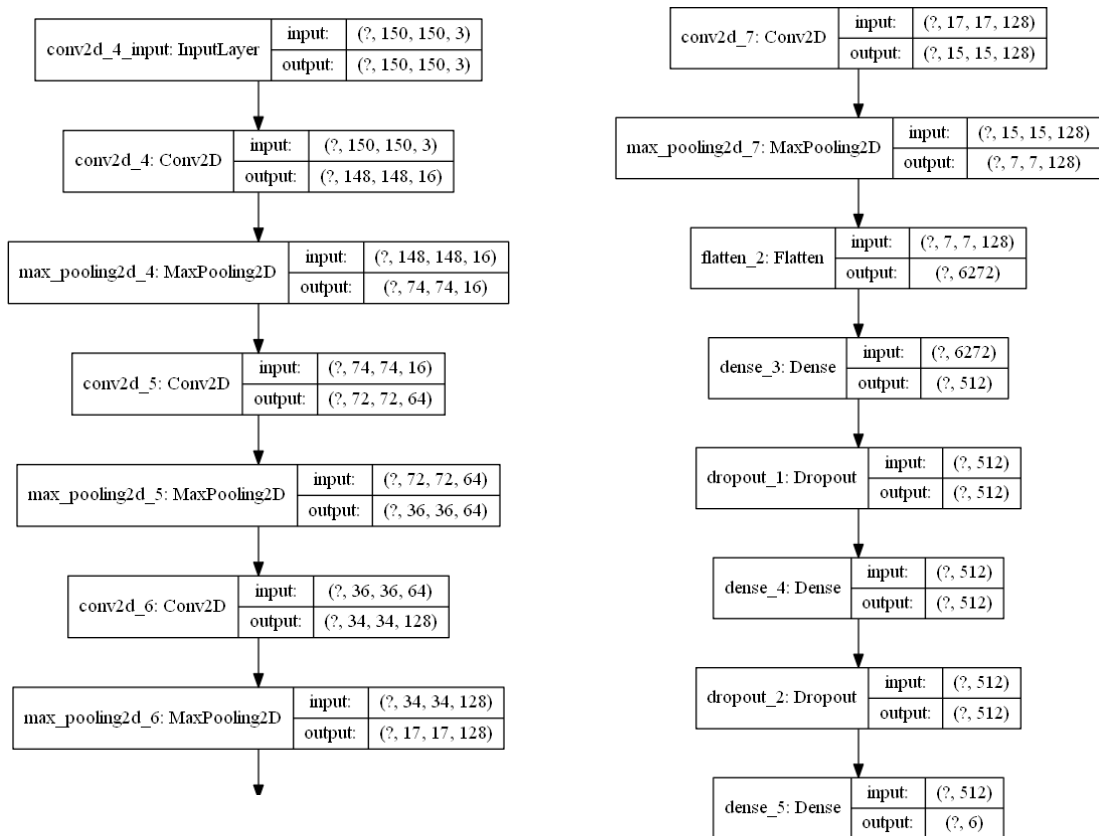


Figure 4.3 CNN Architecture 2

Regularization method was used to improve the overall performance from the previous model, this was the so called under the model fine-tuning phase. Adding dropout layers with parameter,  $p=0.3$  after dense layers to provide 0.3 probability to collect or pick up the output of each node from previous layers.

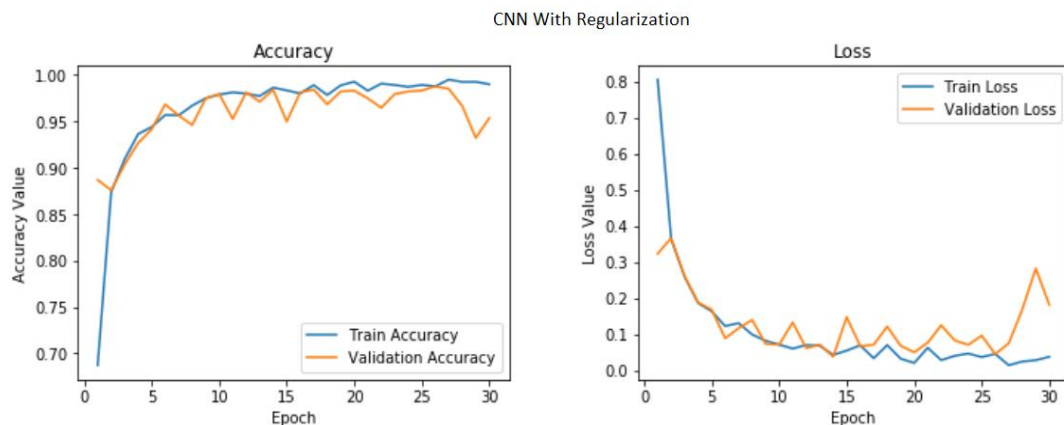


Figure 4.4 CNN 2 Performance Line Graph

From the line graphs shown above, the patterns of both graph seemed to be likely match for both train and validation accuracies and losses. There were no critical rise or drop significantly after each epoch as compared to the last trained model. Yet, during the last few epochs, unexpected drop of validation accuracy and rising of validation loss were presented. Overall, the model was started to grow in a manageable behavior, therefore the regularization has slightly decreased the overfitting problem brought previously.

*Table 4.6 Confusion Matrix for CNN 2*

|        |               | Predicted   |              |              |              |               |               |
|--------|---------------|-------------|--------------|--------------|--------------|---------------|---------------|
|        |               | Fresh Apple | Fresh Banana | Fresh Orange | Rotten Apple | Rotten Banana | Rotten Orange |
| Actual | Fresh Apple   | <b>180</b>  | 0            | 0            | 0            | 0             | 0             |
|        | Fresh Banana  | 1           | <b>174</b>   | 0            | 1            | 1             | 3             |
|        | Fresh Orange  | 4           | 0            | <b>171</b>   | 0            | 0             | 5             |
|        | Rotten Apple  | 2           | 0            | 0            | <b>175</b>   | 3             | 0             |
|        | Rotten Banana | 0           | 0            | 0            | 0            | <b>180</b>    | 0             |
|        | Rotten Orange | 0           | 0            | 3            | 8            | 0             | <b>169</b>    |

*Table 4.7 Precision, Recall and F1-Score for CNN 2*

|               | Precision | Recall | F1-Score | Support |
|---------------|-----------|--------|----------|---------|
| Fresh Apple   | 0.96      | 1.00   | 0.98     | 180     |
| Fresh Banana  | 1.00      | 0.97   | 0.98     | 180     |
| Fresh Orange  | 0.98      | 0.95   | 0.97     | 180     |
| Rotten Apple  | 0.95      | 0.97   | 0.96     | 180     |
| Rotten Banana | 0.98      | 1.00   | 0.99     | 180     |
| Rotten Orange | 0.95      | 0.94   | 0.95     | 180     |
|               |           |        |          | 1080    |

*Table 4.8 Performance Metrics Table for CNN 2*

| <b>Performance Metrics</b> | <b>Score</b> |
|----------------------------|--------------|
| Accuracy                   | 0.9713       |
| Precision                  | 0.9716       |
| Recall                     | 0.9713       |
| F1-Score                   | 0.9712       |

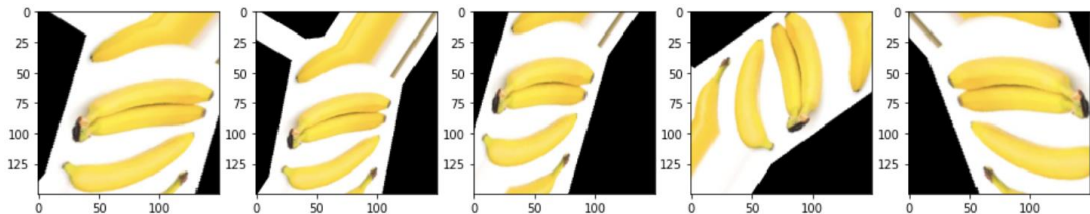
### 4.1.3 CNN with Image Augmentation

Image augmentation is also a regularization method to improve the proficiency of CNN models. It artificially re-processes and creates training images through different processing methods in order to increase the amount used of data for model training as neural networks with larger training data tends to achieve desirable performance. Hence, Image Data Generator API in Keras was used as the configurations were easy to implement for training data.

The configurations applied were as follows:

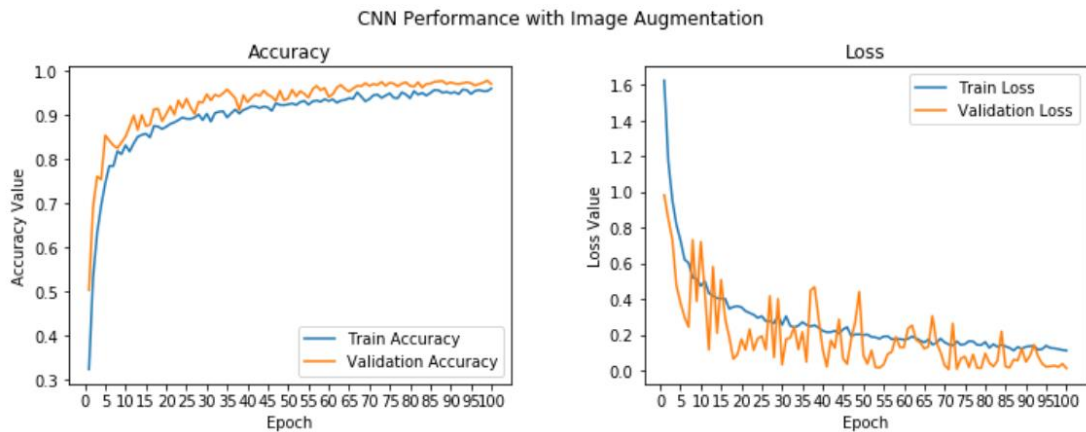
- Rescale =  $1./255$
- Zoom range = 0.3
- Rotation range = 50
- Width shift range = 0.2
- Height shift range = 0.2
- Shear range = 0.2
- Horizontal flip = True
- Fill mode = 'nearest'

After the images were augmented, the outcomes of new data would be as follows:



*Figure 4.5 Example of Data Augmentation*





*Figure 4.6 CNN 3 Performance Line Graph*

The model was trained for 100 epochs and it actually trained well as the train and validation accuracies were increased in a regular manner but the pattern was seemed to be overly followed to the training accuracy line, it can be also inferred by memorizing data features. As for the train and validation losses graph, the validation loss was abnormally changing in growth manner, a few big wave changes could be observed.

*Table 4.9 Confusion Matrix for CNN 3*

|        |               | Predicted   |              |              |              |               |               |
|--------|---------------|-------------|--------------|--------------|--------------|---------------|---------------|
|        |               | Fresh Apple | Fresh Banana | Fresh Orange | Rotten Apple | Rotten Banana | Rotten Orange |
| Actual | Fresh Apple   | <b>180</b>  | 0            | 0            | 0            | 0             | 0             |
|        | Fresh Banana  | 0           | <b>180</b>   | 0            | 0            | 0             | 0             |
|        | Fresh Orange  | 0           | 0            | <b>177</b>   | 0            | 0             | 3             |
|        | Rotten Apple  | 15          | 0            | 1            | <b>163</b>   | 1             | 0             |
|        | Rotten Banana | 0           | 0            | 0            | 0            | <b>180</b>    | 0             |
|        | Rotten Orange | 0           | 0            | 10           | 1            | 0             | <b>169</b>    |

*Table 4.10 Precision, Recall, F1-Score for CNN 3*

|               | <b>Precision</b> | <b>Recall</b> | <b>F1-Score</b> | <b>Support</b> |
|---------------|------------------|---------------|-----------------|----------------|
| Fresh Apple   | 0.92             | 1.00          | 0.96            | 180            |
| Fresh Banana  | 1.00             | 1.00          | 1.00            | 180            |
| Fresh Orange  | 0.94             | 0.98          | 0.96            | 180            |
| Rotten Apple  | 0.99             | 0.91          | 0.95            | 180            |
| Rotten Banana | 0.99             | 1.00          | 1.00            | 180            |
| Rotten Orange | 0.98             | 0.94          | 0.96            | 180            |
|               |                  |               |                 | 1080           |

*Table 4.11 Performance Metrics for CNN 3*

| <b>Performance Metrics</b> | <b>Score</b> |
|----------------------------|--------------|
| Accuracy                   | 0.9713       |
| Precision                  | 0.9726       |
| Recall                     | 0.9713       |
| F1-Score                   | 0.9712       |

Based on the results of the predictions from the confusion matrix, we can observe that the performance of the classification model was improving increasingly. This is because the wrong predictions were still under the same fruit domain, such as predicting rotten apple as fresh apple, fresh orange as rotten orange. They were still under the same fruit type, therefore it was almost close to the perfect prediction model.

#### 4.1.4 Transfer Learning of Pre-trained CNN as Feature Extractor Performance

Using transfer learning approach, the pre-trained weights can be leveraged and train on new models to achieve better performance compared to constructing and training neural networks from scratch. Hence, it has to be mentioned that the VGG-16 will be the popular model that has been trained on a huge dataset named ImageNet which consisted of diverse categories for the purpose of recognition and classification tasks.

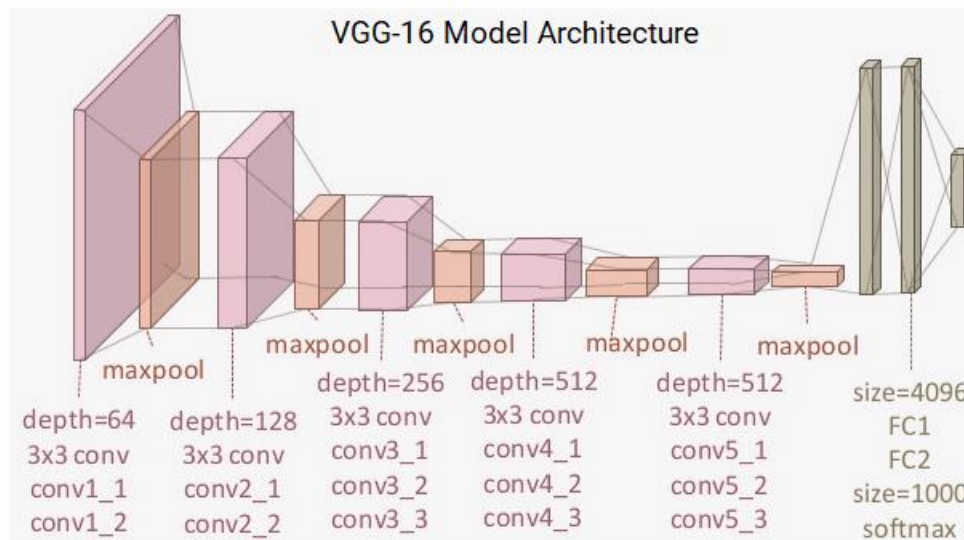


Figure 4.7 Network Architecture of VGG-16

According to the figure shown, the architecture of VGG-16 was presented in a deep and complex network that consisted of 13 convolutional layers with 3x3 filters and each followed by the max pooling layers, two fully connected layers of 4096 units and a dense layer of 1000 units. The 1000 units signifies the total number of categories found in the ImageNet database.

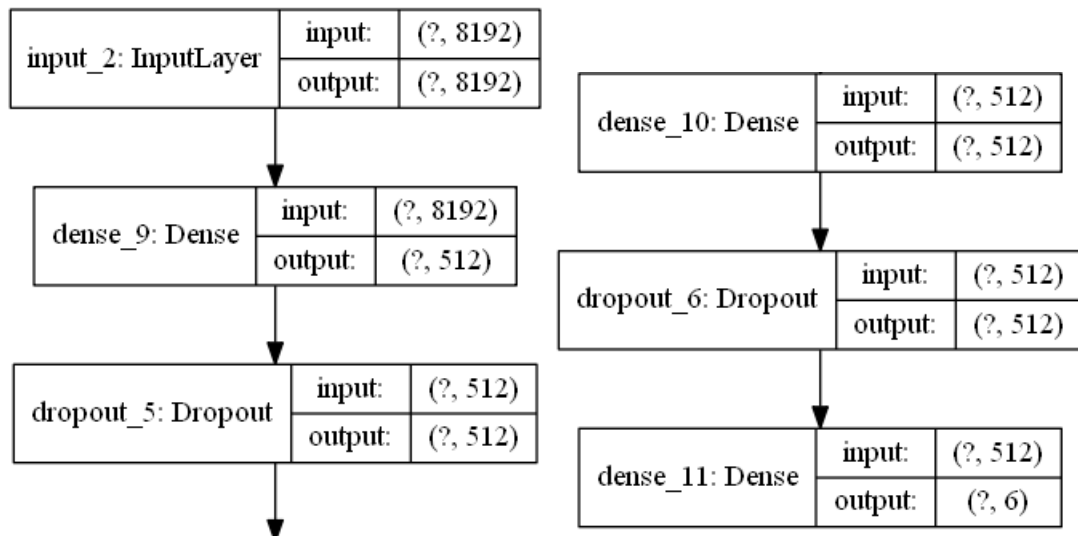


Figure 4.8 CNN Architecture 4

In this project, we altered the fully connected layers to smaller units, 512 and the classification layer to the unit of 6, which indicates the number of fruit classes for output. Therefore, with the pre-trained weights of VGG-16 and altering the last few layers, the new model that tackles our classification problem was proceeded for training.

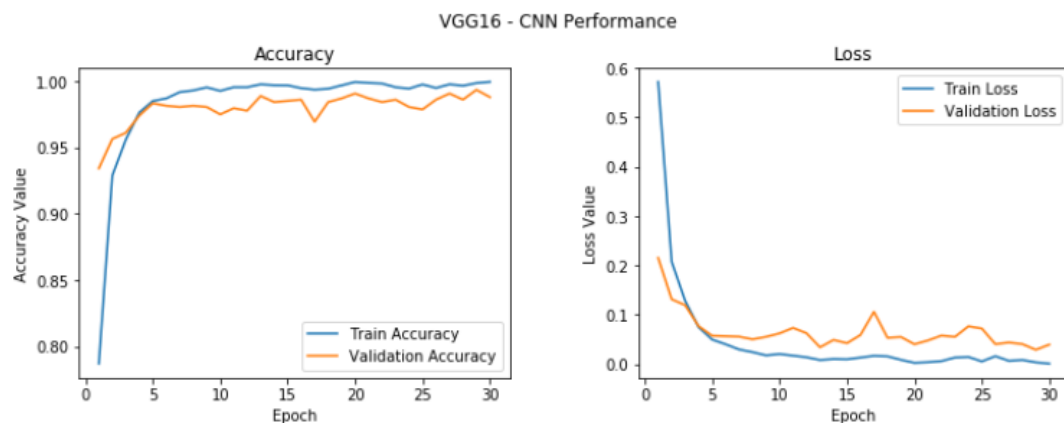


Figure 4.9 CNN 4 Performance Line Graph

The model was trained in 30 epochs. The growth pattern of both line graphs were in normal manner, showing the model was well-trained, no sudden uprising or down falling after each epoch. VGG-16 pre-trained weights enhanced the model training, using less time and less computing resources to achieve the performance of deepen network.

Table 4.12 Confusion Matrix for CNN 4

|        |               | Predicted   |              |              |              |               |               |
|--------|---------------|-------------|--------------|--------------|--------------|---------------|---------------|
|        |               | Fresh Apple | Fresh Banana | Fresh Orange | Rotten Apple | Rotten Banana | Rotten Orange |
| Actual | Fresh Apple   | <b>179</b>  | 0            | 0            | 0            | 0             | 1             |
|        | Fresh Banana  | 0           | <b>179</b>   | 0            | 0            | 1             | 0             |
|        | Fresh Orange  | 0           | 0            | <b>179</b>   | 0            | 0             | 1             |
|        | Rotten Apple  | 3           | 0            | 0            | <b>169</b>   | 0             | 8             |
|        | Rotten Banana | 0           | 1            | 0            | 0            | <b>179</b>    | 0             |
|        | Rotten Orange | 0           | 0            | 2            | 2            | 0             | <b>176</b>    |

Table 4.13 Precision, Recall and F1-Score for CNN 4

|               | Precision | Recall | F1-Score | Support |
|---------------|-----------|--------|----------|---------|
| Fresh Apple   | 0.99      | 0.94   | 0.96     | 180     |
| Fresh Banana  | 0.99      | 0.99   | 0.99     | 180     |
| Fresh Orange  | 0.99      | 0.99   | 0.99     | 180     |
| Rotten Apple  | 0.99      | 0.94   | 0.96     | 180     |
| Rotten Banana | 0.99      | 0.99   | 0.99     | 180     |
| Rotten Orange | 0.95      | 0.98   | 0.96     | 180     |
|               |           |        |          | 1080    |

Table 4.14 Performance Metrics Table for CNN 4

| Performance Metrics | Score  |
|---------------------|--------|
| Accuracy            | 0.9824 |
| Precision           | 0.9826 |
| Recall              | 0.9824 |
| F1-Score            | 0.9824 |

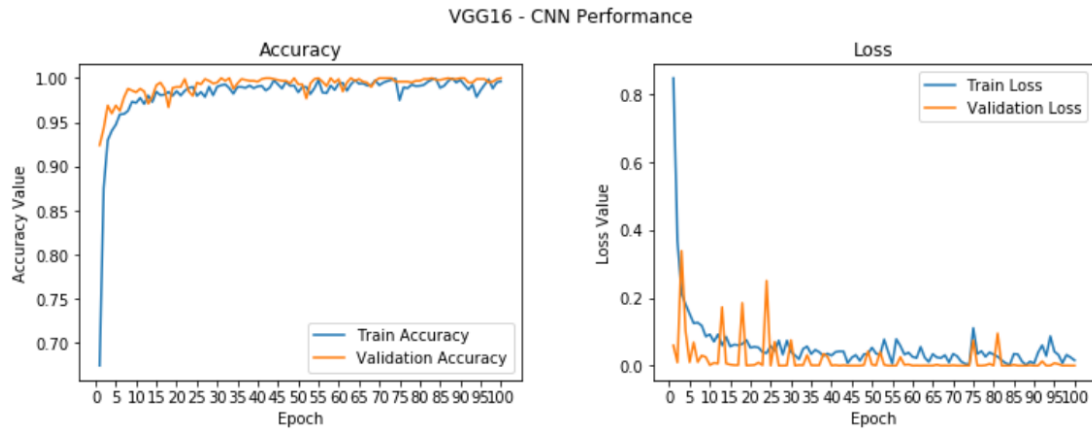
#### 4.1.5 Transfer Learning of Pre-trained CNN with Fine-tuning and Data Augmentation

Based on the results given from the previous transfer learning model, the predictions has already met the expectation level, achieving overall 98% of accuracy in classification task. Perhaps, we could attempt on data augmentation to generate more training data and do fine-tuning on certain layers to make further enhancements.

Firstly, setting the 4<sup>th</sup> and 5<sup>th</sup> block to be trainable so that the weights for these layers will update along back-propagation in each training iteration. The data augmentation configurations were used back like the previous ones and the model architecture will be also the identical to the last transfer learning model. After “unfreezing” the layers to be trainable, the 4<sup>th</sup> and 5<sup>th</sup> blocks were set trainable to true and the network will indicate as such, shown in the next figure.

|    | Layer Type   | Layer Name   | Layer Trainable |
|----|--|--------------|-----------------|
| 0  | <keras.engine.topology.InputLayer object at 0x7f26c86b2518>  | input_1      | False           |
| 1  | <keras.layers.convolutional.Conv2D object at 0x7f277c9fc080> | block1_conv1 | False           |
| 2  | <keras.layers.convolutional.Conv2D object at 0x7f26c86b26d8> | block1_conv2 | False           |
| 3  | <keras.layers.pooling.MaxPooling2D object at 0x7f26c86e6c88> | block1_pool  | False           |
| 4  | <keras.layers.convolutional.Conv2D object at 0x7f26c867dc18> | block2_conv1 | False           |
| 5  | <keras.layers.convolutional.Conv2D object at 0x7f26c8690f28> | block2_conv2 | False           |
| 6  | <keras.layers.pooling.MaxPooling2D object at 0x7f26c869e5c0> | block2_pool  | False           |
| 7  | <keras.layers.convolutional.Conv2D object at 0x7f26c863f828> | block3_conv1 | False           |
| 8  | <keras.layers.convolutional.Conv2D object at 0x7f26c863f128> | block3_conv2 | False           |
| 9  | <keras.layers.convolutional.Conv2D object at 0x7f26c86607b8> | block3_conv3 | False           |
| 10 | <keras.layers.pooling.MaxPooling2D object at 0x7f26c83d7d68> | block3_pool  | False           |
| 11 | <keras.layers.convolutional.Conv2D object at 0x7f26c83fd358> | block4_conv1 | True            |
| 12 | <keras.layers.convolutional.Conv2D object at 0x7f26c83fddd8> | block4_conv2 | True            |
| 13 | <keras.layers.convolutional.Conv2D object at 0x7f26c839da20> | block4_conv3 | True            |
| 14 | <keras.layers.pooling.MaxPooling2D object at 0x7f26c83ac1d0> | block4_pool  | True            |
| 15 | <keras.layers.convolutional.Conv2D object at 0x7f26c834e978> | block5_conv1 | True            |
| 16 | <keras.layers.convolutional.Conv2D object at 0x7f271a15eb38> | block5_conv2 | True            |
| 17 | <keras.layers.convolutional.Conv2D object at 0x7f26c8371d68> | block5_conv3 | True            |
| 18 | <keras.layers.pooling.MaxPooling2D object at 0x7f26c8314b00> | block5_pool  | True            |
| 19 | <keras.layers.core.Flatten object at 0x7f26c828bda0>         | flatten_1    | True            |

Figure 4.10 Indicates the Trainable Layers



*Figure 4.11 CNN 5 Performance Line Graph*

Based on the line graphs above, the results achieved were favorable. As compared to the result of the first basic CNN that was built initially, the classification model seemed to be improved a lot from overfitting problem. Regularization method like adding dropout layers and increase artificial training data using data augmentation approach has been implemented before training this final model. Although the overall accuracy achieved from the first network was already above expectations, yet the line graphs indicated the problem of overfitting. The accuracy achieved may be high yet when the model encounters new and strange data, or data that is very dissimilar from the training data during the real-time predictions, somehow the model may not have the capabilities to make accurate prediction.

Table 4.15 Confusion Matrix for CNN 5

|        |               | Predicted   |              |              |              |               |               |
|--------|---------------|-------------|--------------|--------------|--------------|---------------|---------------|
|        |               | Fresh Apple | Fresh Banana | Fresh Orange | Rotten Apple | Rotten Banana | Rotten Orange |
| Actual | Fresh Apple   | <b>180</b>  | 0            | 0            | 0            | 0             | 0             |
|        | Fresh Banana  | 0           | <b>179</b>   | 0            | 0            | 1             | 0             |
|        | Fresh Orange  | 0           | 0            | <b>180</b>   | 0            | 0             | 0             |
|        | Rotten Apple  | 0           | 0            | 0            | <b>180</b>   | 0             | 0             |
|        | Rotten Banana | 0           | 1            | 0            | 0            | <b>179</b>    | 0             |
|        | Rotten Orange | 0           | 0            | 0            | 0            | 0             | <b>180</b>    |

Table 4.16 Precision, Recall and F1-Score for CNN 5

|               | Precision | Recall | F1-Score | Support |
|---------------|-----------|--------|----------|---------|
| Fresh Apple   | 1.00      | 1.00   | 1.00     | 180     |
| Fresh Banana  | 0.99      | 0.99   | 0.99     | 180     |
| Fresh Orange  | 1.00      | 1.00   | 1.00     | 180     |
| Rotten Apple  | 1.00      | 1.00   | 1.00     | 180     |
| Rotten Banana | 0.99      | 0.99   | 0.99     | 180     |
| Rotten Orange | 1.00      | 1.00   | 1.00     | 180     |
|               |           |        |          | 1080    |

Table 4.17 Performance Metrics Table for CNN 5

| Performance Metrics | Score  |
|---------------------|--------|
| Accuracy            | 0.9981 |
| Precision           | 0.9981 |
| Recall              | 0.9981 |
| F1-Score            | 0.9981 |



## 4.2 Deduction on CNN Classification Model

*Table 4.18 Overall Performance Table*

| <b>CNN Model</b>  | <b>Overall Accuracy</b> |
|---|-------------------------|
| CNN Build-from-scratch  | 0.9843                  |
| CNN with Regularization   | 0.9713                  |
| CNN with Image Augmentation   | 0.9713                  |
| Transfer Learning of VGG-16   | 0.9824                  |
| <b>Transfer Learning of VGG-16 with Fine-tuned layers and Data Augmentation</b> | <b>0.9981</b>           |

The performance of the classification model was incrementally improved from one to one. The final model, which was applied with transfer learning from VGG-16 pre-trained weights and fine-tuned last few layers. Based on the overall accuracies achieved by each model, they only have minor difference of 0.01 to 0.02 among each other. By just looking at the values attained, it might not be sufficient to do much analysis, the observation on how the models were being trained and evaluated with validation sets was significant.

To infer the reason that each model achieves higher than 95% accuracy, it was due to the whole dataset was prepared clean, only a single object falls on the white background, making the dataset acquired contains less noises like complicated background. Therefore, to further make predictions on these clean images, it can be rather easier than inputting images consisted of complicated backgrounds or other noises. Hence, in order to tackle real-world problems, the classification model must achieve astonishing performance so that it will have practical usage for targeted users in real world.

Through analyzing each detail return from the models training, the last trained model would be chosen for further classification tasks because of the remarkable prediction results after being tested with testing sets.

### 4.3 Evaluation on Faster R-CNN detection model

The aim of building this detection model was to localise and identify fruit domains from the inputted data. The fruit detection model was built on top of the open source Tensorflow Object Detection API and Faster R-CNN was chosen as the detection model architecture. To achieve accurate detection outcome, transfer learning approach was utilized where a pre-trained model on COCO datasets, **Faster R-CNN Inception V2** was chosen.

Both confidence score and Intersection over Union (IoU) played an significant role in determining whether a detection in an image was true positive or false positive. For confidence score, it is a probability that the computed bounding boxes contain an actual object that the model wants to predict. For IoU, it is interpreted as the area of overlapped divided by the union area of predicted bounding box ( $B_P$ ) and a ground-truth box ( $B_{GT}$ ).

$$IoU = \frac{Area(Bo_x_p \cap Bo_x_{GT})}{Area(Bo_x_p \cup Bo_x_{GT})}$$

If the confidence score of the detection reaches a predefined threshold value and find one that has the highest IoU value with the detection among all ground truths, if there is an existence of ground truth, then the detection is a true positive.

For this project, the threshold value was set manually to 0.70 because it may be an optimum value to prevent any miss-detection from actual object. Therefore, the confidence score that has lower value than the threshold value of 0.70, it will be known as false positive. Bounding boxes were drawn for these positively detected objects to indicate and localise the actual object presented in the image or frame.

### 4.4 Tools to Use

#### 4.4.1 Software/Platform

The software/platform involved in this project development:

1. Jupyter Notebook (Anaconda)
2. Google Colaboratory
3. LabelImg – used for image annotation
4. IP Camera – used for real-time video capturing

#### 4.4.2 Libraries

The libraries involved and utilized in the development in both Jupyter Notebook and Google Colaboratory environments:

1. Tensorflow v1.15.0
2. Keras
3. Scikit-Learn
4. OpenCV

#### 4.4.3 Hardware

The hardware involved throughout the implementation and testing of the project:

1. Laptop

|                  |  |
|------------------|--|
| Brand            | Asus ROG Strix G                             |
| Operating System | Windows 10 Home – 64-bit                     |
| Processor        | Intel® Core™ i7-9750H CPU @ 2.60GHz (12CPUs) |
| RAM              | 24GB DDR4 2666MHz                            |
| GPU              | NVIDIA GeForce GTX 1660 Ti                   |
| Storage          | 512GB SSD                                    |

2. Mobile Phone

- Act as real-time video capturing tool.
- Any Android OS mobile phone that has network connectivity, meaning able to connect to Wi-Fi or sharing mobile hotspots.
- Supports up to Android Version 7.0 and above.
- Has a working camera.
- Pre-installed with IP Camera, could be found in Google Play Store.

#### 4.5 System Performance Definition

There were two matters which were significantly concerned on this project:

- **Performance/Speed of the deliverable system**

The system was expected to be performed in an automation approach on classifying fruit images and predict its freshness state. Therefore, returning results within a few seconds would be important as users are prone to fast feedback systems. Providing results within seconds will be helpful for users in every aspects especially the system will be utilized on business or daily usage. Thus, the system was built and designed to have great performance with the implementation deep learning techniques. Improvements should be made if any unnecessary delays observed during the execution of the system.

- **Accuracy of the deliverable system**

For the accuracy on classifying fruits and freshness prediction in this system, it was expected to be accurate and precise. Hence, comparisons would be made for a few CNN model during the implementation phase of the system in order to make better improvement for this system.

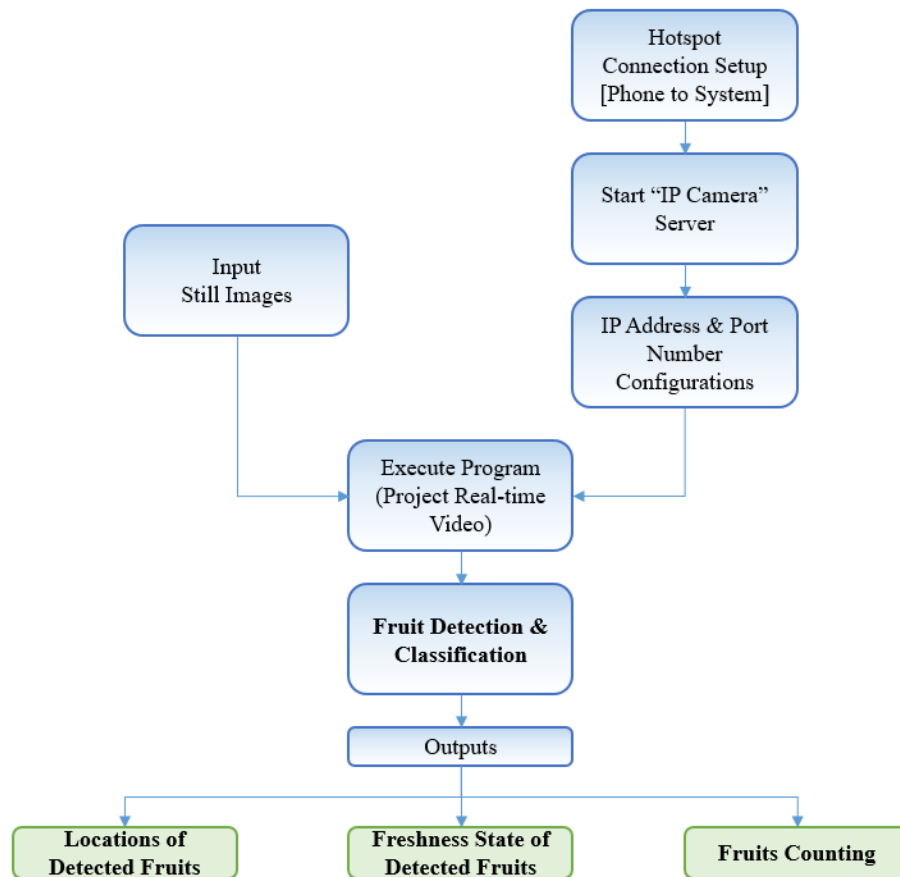
#### 4.6 Verification Plan

The system developed was expected to identify the fruit classes as well as its current freshness states based on the input of still images or real-time video. However, different circumstances might affect the final results of prediction. Hence, certain verification plans were set to assure the final system brings out outstanding performance.

1. A single/multiple fruit image is inputted for freshness state prediction, the system is expected to return the correct fruit class as well as its current condition to the front-end.
2. A single/multiple fruit image is inputted for detection and counting purpose, the system is expected to detect the location of fruit accurately by drawing bounding boxes and return the value of fruit count to the front-end.
3. A single/multiple fruit image is inputted for freshness state prediction, detection and counting, the system is expected to draw bounding boxes on detected fruit(s), return accurate fruit class and current condition as well as the value of fruit count to the front-end.
4. A real-time video is inputted into the system, the system is expected to draw bounding boxes on detected fruit(s), return accurate fruit class and current condition as well as the value of fruit count to the front-end.
5. A random image is inputted that if the object is out of the range of fruit domains involved, the object will not be detected and further classify.

## CHAPTER 5 SYSTEM TESTING

### 5.1 System Testing Procedural Block Diagram



*Figure 5.1 System Testing Procedural Block Diagram*

There were two input ways to run on this detection and classification tasks. For the first one, input still images. This was just simple, selecting images that user wanted to input from the local directories. Secondly, input real-time video into the system. A few steps need to be taken before proceeding further processes.

To input still images into the system:

1. Select images from local directories.
2. Execute the program.

To project/cast real-time video into the system:

1. Get an Android mobile phone that supports up to Android Version 7.0 and above.
2. Install IP Camera from Google Play Store.
3. Setup the hotspot connection, make sure the phone and the system were connected under the same network.
4. Launch into IP Camera and start Server.
5. Observe the IP address and port number shown.
6. Configure the input IP address and port number in the system, make sure they are both identical to the application.
7. Execute the program.

Once the program is started, the inputted images or frames are to be processed into fruit detection and classification models. There are three outputs can be expected to return from the system, which are mainly the locations of detected fruits, the freshness states of each individual fruit and the fruit counting value.








## 5.2 Testing on Performance of Classification Model






Images were found randomly from internet sources to test on the final classification model, the fruit class and current condition were the expected prediction outcome. In this case, five apple, banana and orange fruit images were chosen randomly for testing.

### 5.2.1 Apple

*Table 5.1 Apple Single Image Classification Testing*






| Index | Images  | Actual       | Predicted           |
|-------|---|--------------|---------------------|
| 1     |   | Fresh Apple  | <b>Fresh Apple</b>  |
| 2     |  | Rotten Apple | <b>Rotten Apple</b> |
| 3     |  | Fresh Apple  | <b>Fresh Apple</b>  |
| 4     |  | Rotten Apple | <b>Rotten Apple</b> |
| 5     |  | Fresh Apple  | <b>Fresh Apple</b>  |
|       |   |              | <b>5/5 Correct</b>  |

**5.2.2 Banana***Table 5.2 Banana Single Image Classification Testing*

| Index | Images  | Actual        | Predicted            |
|-------|---|---------------|----------------------|
| 1     |    | Fresh Banana  | <b>Fresh Banana</b>  |
| 2     |    | Fresh Banana  | <b>Fresh Banana</b>  |
| 3     |    | Rotten Banana | <b>Rotten Banana</b> |
| 4     |  | Rotten Banana | <b>Rotten Banana</b> |
| 5     |  | Fresh Banana  | <b>Fresh Banana</b>  |
|       |   |               | <b>5/5 Correct</b>   |

### 5.2.3 Orange


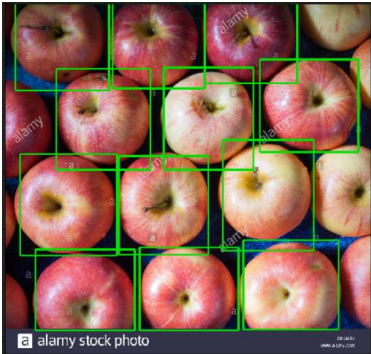

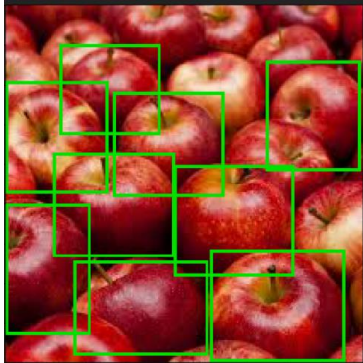

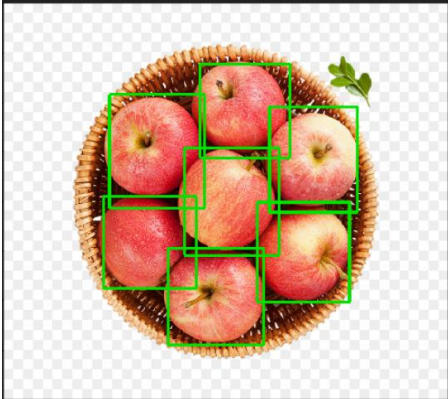
*Table 5.3 Orange Single Image Classification Testing*

| Index | Images  | Actual        | Predicted            |
|-------|---|---------------|----------------------|
| 1     |    | Fresh Orange  | <b>Fresh Orange</b>  |
| 2     |    | Rotten Orange | <b>Rotten Orange</b> |
| 3     |    | Rotten Orange | <b>Rotten Orange</b> |
| 4     |  | Rotten Orange | <b>Rotten Orange</b> |
| 5     |  | Fresh Orange  | <b>Fresh Orange</b>  |
|       |   |               | <b>5/5 Correct</b>   |

5.3 Testing on Performance of Detection Model


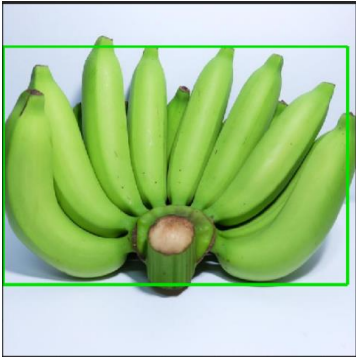

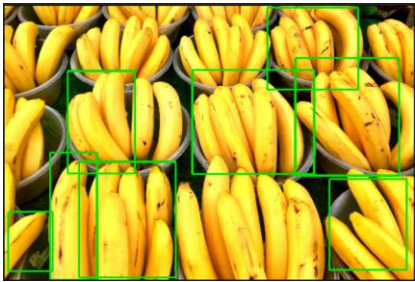

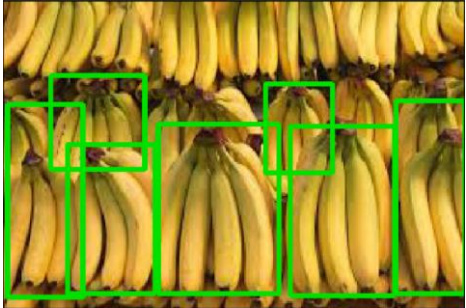
5.3.1 Apple Detection

Table 5.4 Apple Detection Testing

|   | Original Images   | Output   |
|---|---|--|
| 1 |   |   |
| 2 |  |  |
| 3 |  |  |

5.3.2 Banana Detection

Table 5.5 Banana Detection Testing

|   | Original Images   | Output   |
|---|---|--|
| 1 |    |    |
| 2 |  |  |
| 3 |  |  |



### 5.3.3 Orange Detection

*Table 5.6 Orange Detection Testing*

|   | Original Images   | Output   |
|---|---|--|
| 1 |    |    |
| 2 |  |  |
| 3 |  |  |

#### **5.4 Detect and Classify on Each Individual Fruit**

In this stage, testing was conducted on whether the models trained were qualified to be used as the recommender for front-end users to choose their desired products/fruits. In order to have a precise performance on both detection and classification, the detection inference graph and the classification model was programmed to work alternately and the prediction results were to be observed. The output concerns were whether the fruits detected were being localized and drawn with bounding boxes, whether the fruit class and freshness states were predicted accurately, and whether the system returned the exact amount of fruit counting value that was identical to the inputted image.




A few test cases were listed to test on the actual performance after integrating both detection and classification models in the system.

For still images:

1. Input a single fruit image for prediction.
2. Input a multiple fruit image for predictions.
3. Input a multi-class fruit image for predictions

### 5.4.1 Single Fruit Image Prediction

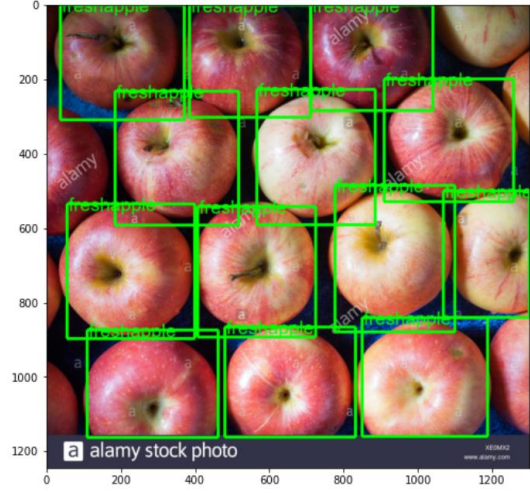
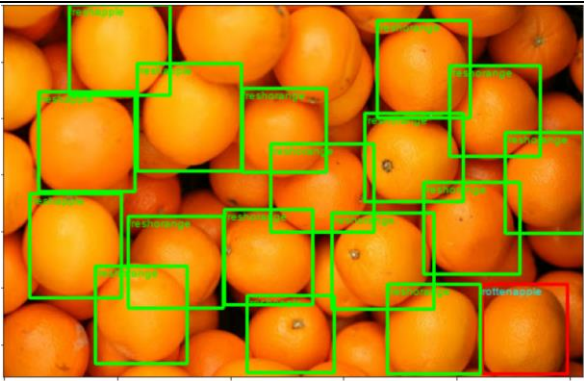
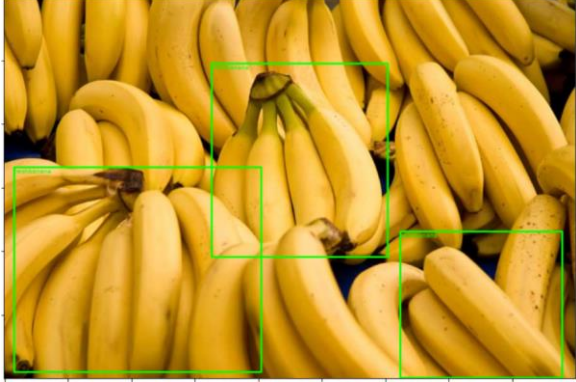
*Table 5.7 Single Fruit Image Prediction*

| Image   | Predictions  | Comment |
|---|--|---------|
|    | <p>Fruit Detected: Fresh Banana</p> <p>Fruit count: 1</p>  | Correct |
|   | <p>Fruit Detected: Rotten Apple</p> <p>Fruit Count: 1</p>  | Correct |
|  | <p>Fruit Detected: Rotten Orange</p> <p>Fruit Count: 1</p> | Correct |



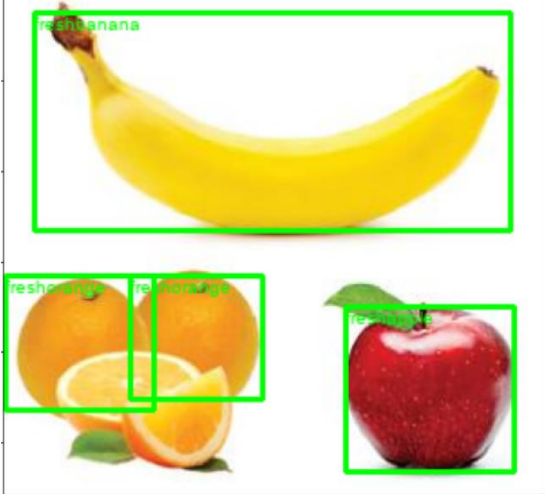
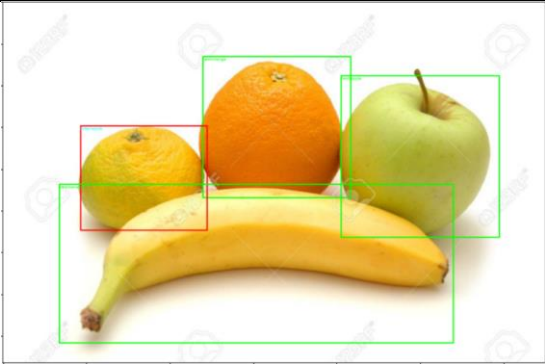
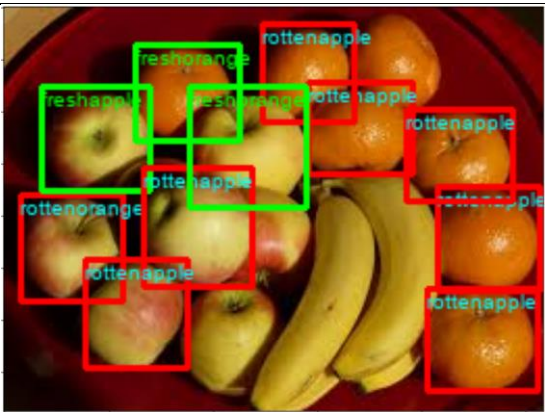
### 5.4.2 Multiple Fruit Image Prediction

*Table 5.8 Multiple Fruit Image Prediction*

| Image   | Predictions  | Comment   |
|---|--|---|
|    | <p>Fruit Detected:<br/>Fresh Apple (13)</p> <p>Fruit count: 13</p>   | Correct   |
|  | <p>Fruit Detected:<br/>Fresh Orange (12)<br/>Fresh Apple (5)<br/>Rotten Apple (1)</p> <p>Fruit Count: 18</p> | Wrong,<br>should be<br>all fresh<br>oranges.                    |
|  | <p>Fruit Detected:<br/>Fresh Banana (3)</p> <p>Fruit Count: 3</p>  | Partially<br>Correct,<br>fruit<br>count<br>should be<br>larger. |

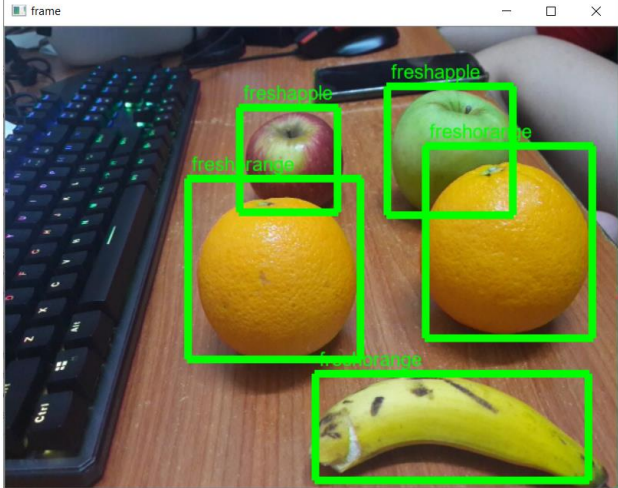
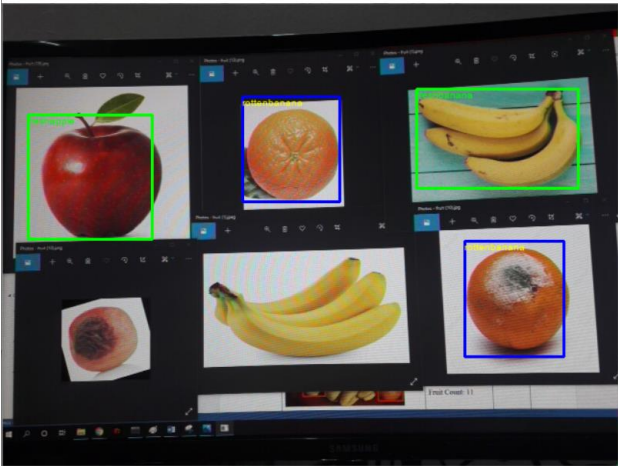
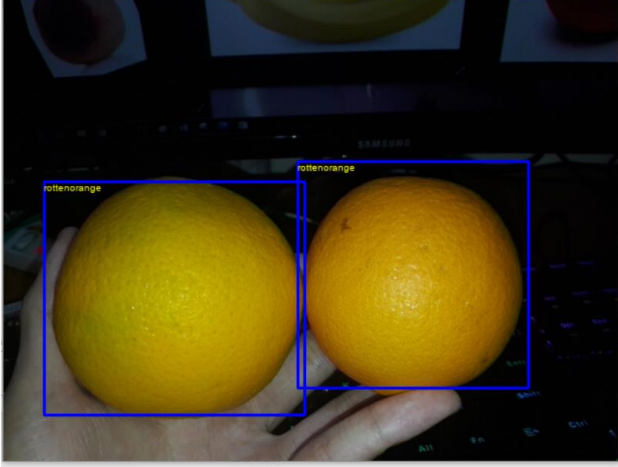
### 5.4.3 Multi-class Fruit Image Prediction

*Table 5.9 Multi-class Fruit Image Prediction*

| Image   | Predictions   | Comment                                  |
|---|---|--|
|    | <p>Fruit Detected:</p> <p>Fresh Banana (1)</p> <p>Fresh Orange (2)</p> <p>Fresh Apple (1)</p> <p>Fruit count: 4</p>                           | Correct                                  |
|  | <p>Fruit Detected:</p> <p>Fresh Orange (1)</p> <p>Fresh Apple (1)</p> <p>Fresh Banana (1)</p> <p>Rotten Apple (1)</p> <p>Fruit Count: 4</p>   | Partially Correct                        |
|  | <p>Fruit Detected:</p> <p>Fresh Apple (1)</p> <p>Fresh Orange (2)</p> <p>Rotten Apple (7)</p> <p>Rotten Orange (1)</p> <p>Fruit Count: 11</p> | Critically Wrong, inaccurate predictions |

## 5.4.4 Real-time Video Prediction

Table 5.10 Real-time Video Prediction

| Frame   | Predictions  | Comment   |
|---|--|---|
|    | <p>Fruit Detected:</p> <p>Fresh Orange (3)</p> <p>Fresh Apple (2)</p> <p>Fruit count: 5</p>                          | <p>Partially Correct, banana classified as orange</p>           |
|   | <p>Fruit Detected:</p> <p>Fresh Apple (1)</p> <p>Fresh Banana (1)</p> <p>Rotten Banana (2)</p> <p>Fruit Count: 4</p> | <p>Wrong, both detection and classify wrong, inaccurate</p>     |
|  | <p>Fruit Detected:</p> <p>Rotten Orange (2)</p> <p>Fruit Count: 2</p>  | <p>Partially Correct, detection correct but classify wrong.</p> |

### 5.5 Discussion on Testing Issues and Dependencies

There were several factors that bring issues towards the system to function inefficiently and below expectations in terms of performance. The following points stated the factors that will affect the overall system functionality:

- Stacking or overlapping objects in one single image or frame

If the fruits are arranged properly but not overly stacking with each other, the detection model should be able to identify each individual object in one frame. This is due to the reason of the stacked objects may not be detected well when running with the inference graph because of getting lower confidence score, causing the back-end algorithm to ignore the existence of the actual object, which is a False Positive. Although the objects presented at the most front part or obvious part could be detected easily, but the remaining objects hidden behind were ignored “innocently”.

- Resolutions of input image or frame

The resolution of image or frame will indeed affect the performance of the system significantly. Given a low resolution image for human-being to identify and make evaluations on it, perhaps they are unable to do so because the visual representation brought by the image is too blur, less pixels or highly compressed. If such data is inputted into the algorithm, perhaps the effectiveness could not be observed because the models are unable to make accurate predictions.

- Light intensity or light exposure in the environment of image or frame

This issue was raised during the testing of real-time video prediction. The “sensitivity” would be affected if the objects are projected with dimmer lights, as lights are not irradiate towards the object, causing the visibility of the object decrease. Thus, no detected objects are found when image or frame is inputted to further processes. In contrast, if the environment has good light intensity, objects are more likely to be

detected since the objects are clearly visible for detection, greater detection and classification results are to be expected. Hence, the light exposure has to be assured in a sufficient and optimum level since this is a controllable element, human-beings can project lights or LED whenever there is insufficient light exposure.

- Requires high-end specifications hardware support for system execution

The back-end processes requires high computational power and resources that most of the common devices or machines are less likely to have optimal support to execute the system. For inputting still images for prediction, common machines are still manageable to allocate sufficient resource and runtime to produce the output to front-end. Yet, as for the real-time prediction, only high-end devices that bear with sufficient computing resources like CPU and RAM are recommended for real-time execution. The execution during real-time requires synchronization between input frames and predictions, meaning that once an object is identified inside the frame, bounding boxes must be shown for detected object, make further predictions on cropped image part and return the output simultaneously. These processes are heavy weightages and requires more runtime throughout the process. If a common machine attempts to run on this system, it may suffer from lagging issues since there is insufficient resource allocation, then the execution is no longer efficient and meaningful.

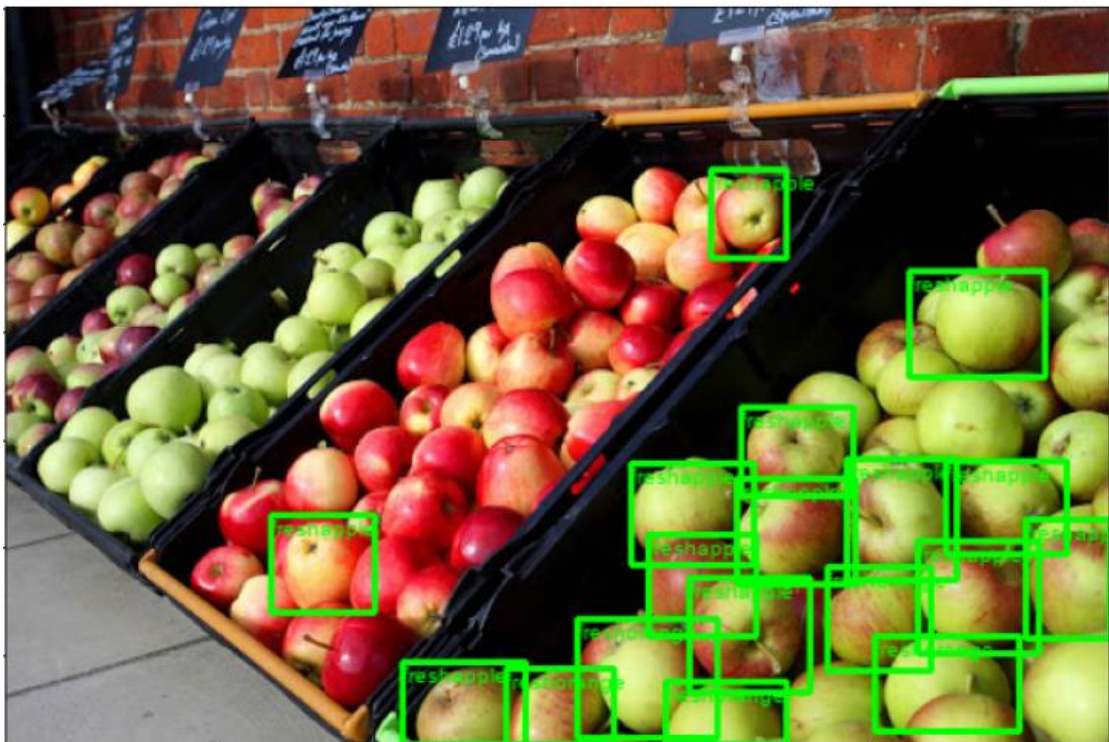


## 5.6 Extended Testing Phase Regarding Affecting Factors

At the previous sub-chapter, we have discussed on the factors that will somehow affect the overall performance of the system either is minor affected or immense affected. Hence, in order to verify the effectiveness of the developed prototype, the testing measurements were elevated to test on how the external factors will heavily affect the accuracy of the predictions towards the inputted images or frames.

### 5.6.1 Stacking / Unorderly Arranged

Firstly, we tested on several images that the fruits were **arranged in a stacking structure**. Let us observe the outcome from the prototype and analyse on it.



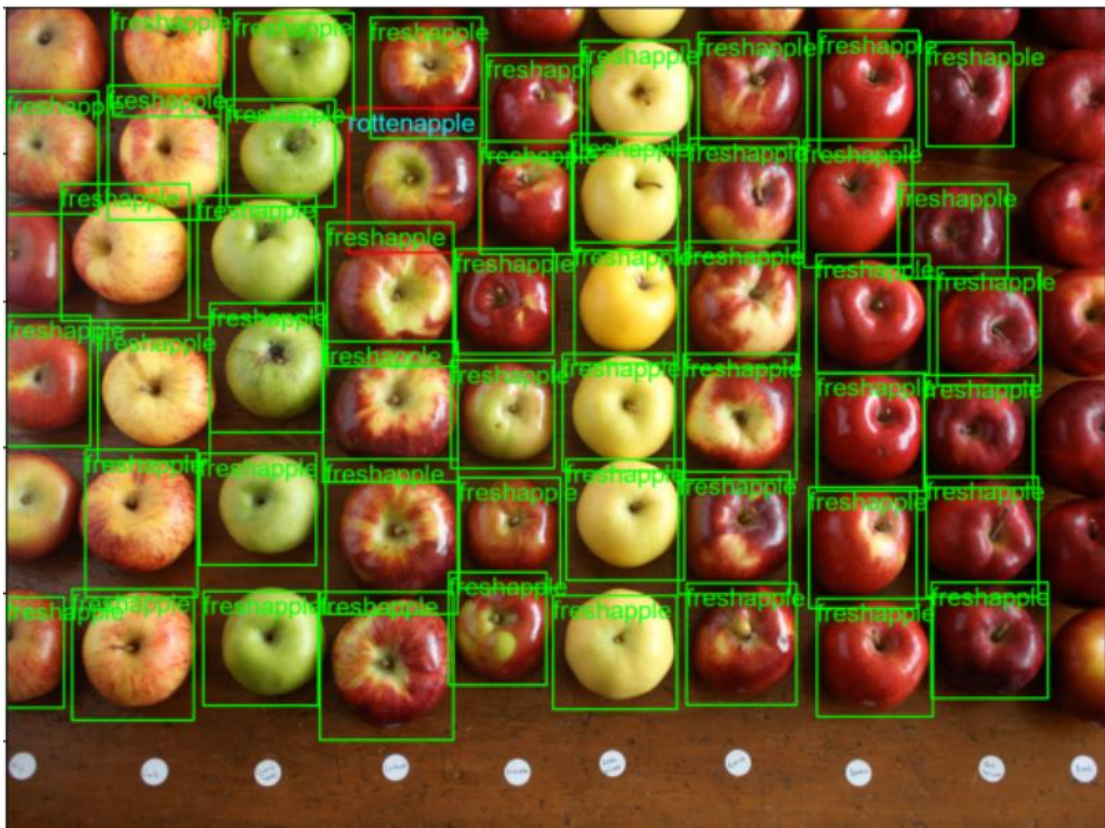
*Figure 5.2 Testing Scenario 1*

Returned output:

**Fresh Apple detected (13), Fresh Orange detected (5), Total count (18)**

By observing the figure shown above, the apples were arranged unorderedly in the basket and the detection inference graph can only detect on the apples that were presented at the front location, whereas the apples at the back were not been detected by the system. This was because of perhaps the illustrations for the front apples were clearer for the inference graph to interpret, while the back apples have slightly poorer illustrations in terms of clearness of the fruit edges. Plus, the overly stacking condition will also affect the effectiveness of the inference graph to detect on available fruits. Once the fruits were not being detected, it could not proceed to freshness states prediction, then the effectiveness has lost.

Therefore, there were two ways to overcome the factor, which is to **change the camera view from the side view to the top view**, and also **arranging these fruits to at least have small spaces in between each of them**. The reason of proposing these solutions can be observed in the next figure.



*Figure 5.3 Testing Scenario 2*

Returned output:

**Fresh Apple detected (50), Rotten Apple detected (1), Total count (51)**

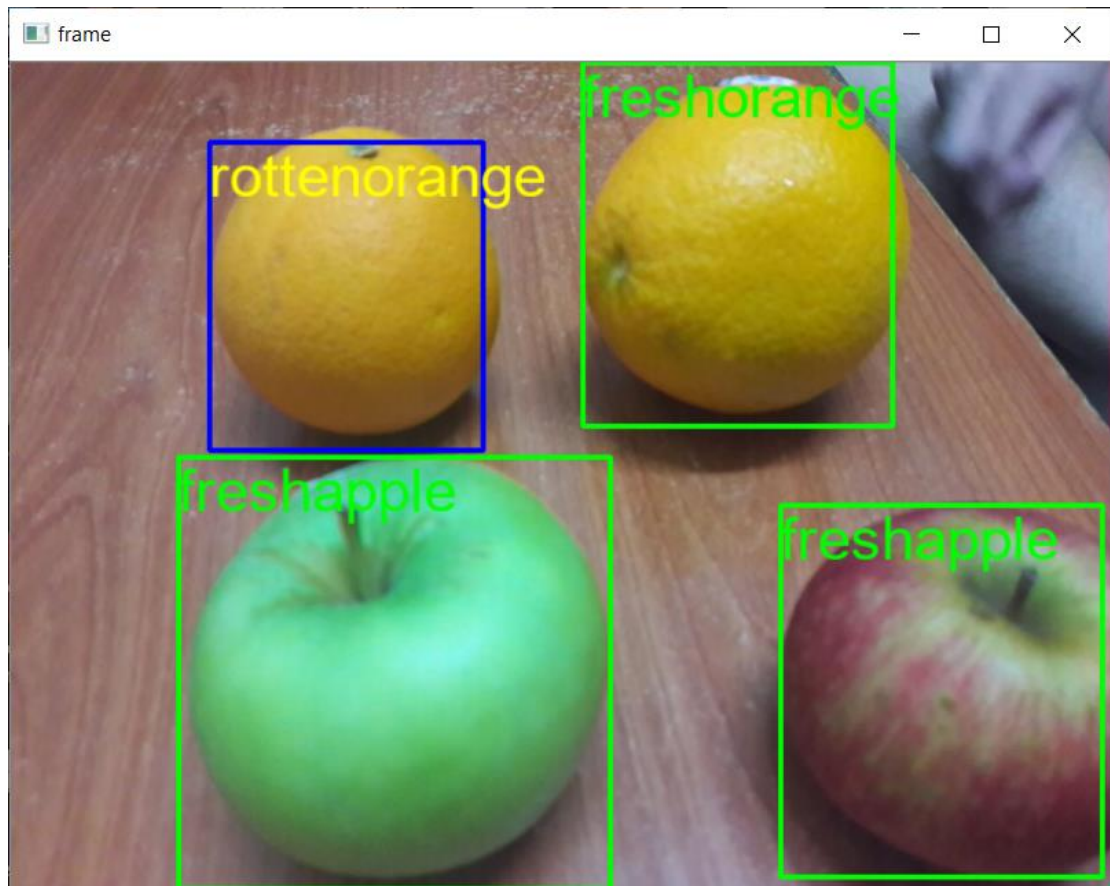
Based on the results shown in the figure above, the fruit detection and classification prototype could be claimed to have a reliable and outstanding performance. Comparing with both figures and their respective results, we could observe that the apples from the top view with sequentially arranged versus the apples from the side view with randomly stacking, there was a huge difference in terms of effectiveness of prototype in between of them. The most significant point was the system prototype can easily **perform great predictions on the fruits that were well-arranged and with high visibility for each individual fruit**. Thus, an inference can be made that the overly stacking fruits were hardly to be detected by the existing system prototype.

### 5.6.2 Light Exposure in the Environment

Environmental lighting was also an important factor that will disturb the effectiveness of the prototype. This can be explained by providing an assumption, if human eyes are unable to see the objects in dark, then the computers or machines can barely identify an object from the frame. For this experiment, it was conducted with the input of real-time video in different lighting conditions, dark and bright. The outputs were marked down synchronously in the system and returned to the front-end.

Firstly, the experimental environment was set to ‘bright’ lighting condition, so to have good visibility for objects.



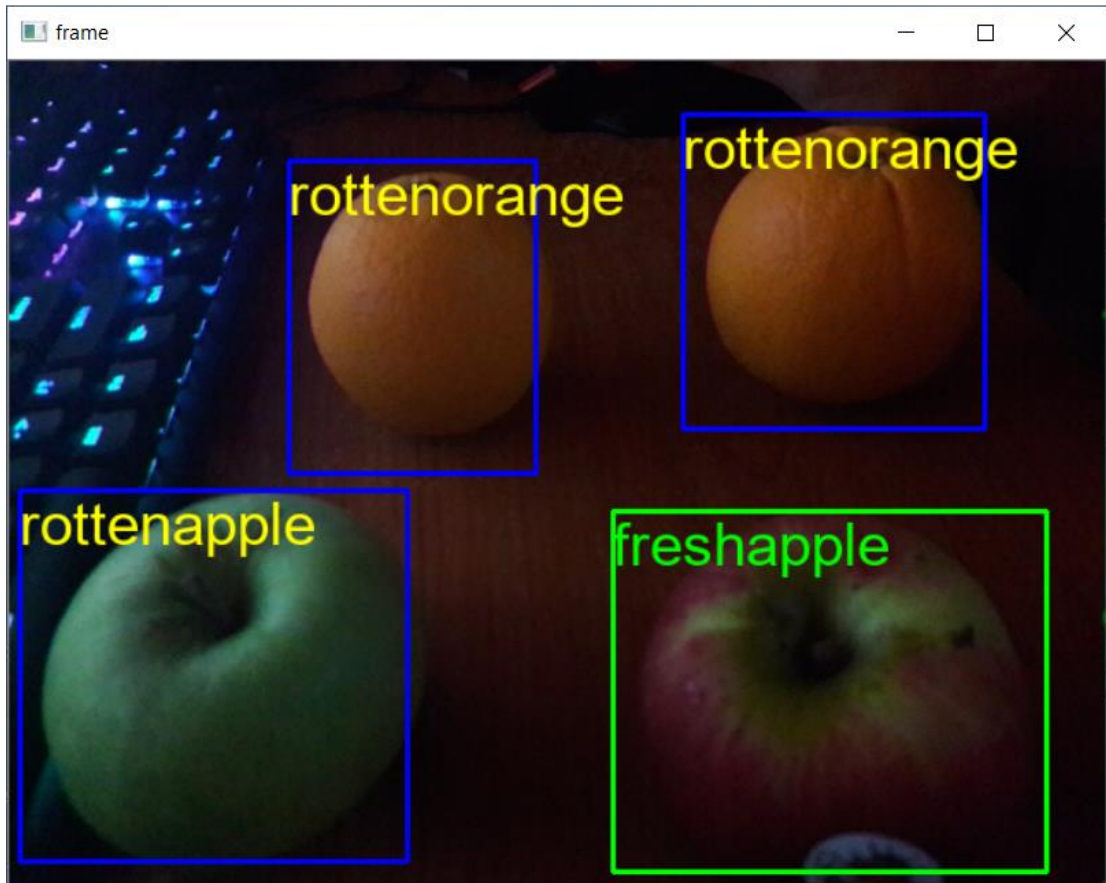


*Figure 5.4 Testing Scenario3*

Returned output:

**Fresh Apple detected (2), Fresh Orange detected (1), Rotten Orange detected (1),  
Fruit count (4)**

This video frame was captured in the **bright lighting condition**. Therefore, we could observe that all fruits were being detected and further quality predictions were made and shown to the front-end. It has less issue when environment has sufficient lights to make objects visible.

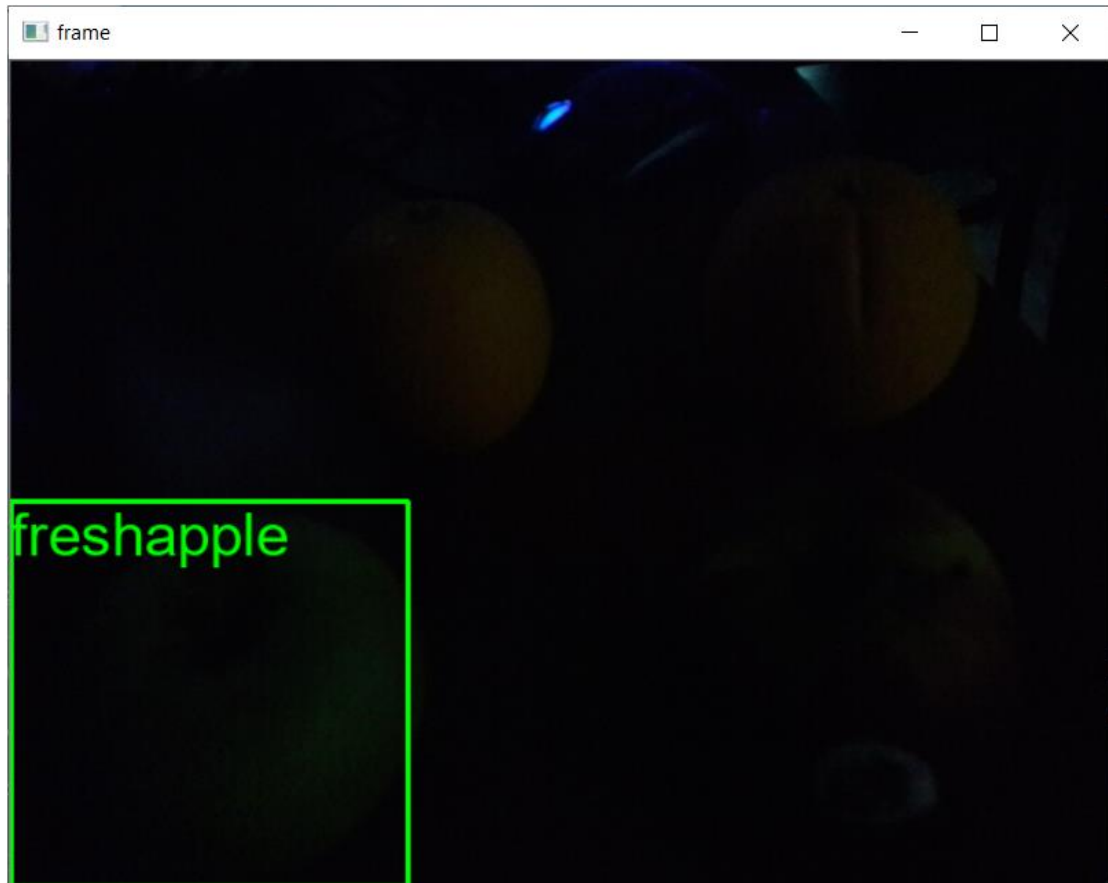


*Figure 5.5 Testing Scenario 4*

Returned output:

**Fresh Apple detected (1), Rotten Orange detected (2), Rotten Apple detected (1),  
Fruit count (4)**

This video frame was captured in the **moderate lighting condition**. Although the fruits were still visible for camera to capture, yet the classification results were somehow been affected because of the light exposure on fruits. Hence, it can be deduced that the effectiveness of the system was affected in moderate lighting condition, the visibility of fruits were maintained yet the important features of external appearance of fruits were less visible, causing the classification results to be inaccurate.



*Figure 5.6 Testing Scenario 5*

Returned output:

**Fresh Apple detected (1), Fruit count (1)**

Lastly, this video frame was captured in the **dark lighting condition**. It was obvious to observe that the detection inference graph can only detect on one fruit among four fruits. The reason may be because of the low confidence score acquired from the inference graph, causing the fruits to be ignored. Not to mention on the computer vision, even for human eyes to observe, we can only barely see and identify the colours of fruits spotted from the dark environment. Hence, the prototype was not functional for dark environments, making the effectiveness of the system decrease when predicting fruits.

To conclude on this experiment, lighting condition was indeed an important factor that will heavily cause consequences on how the system functions.

## CHAPTER 6 CONCLUSION

### 6.1 Project Review and Discussions

In a nutshell, to review the whole project, it has to mention that the fruit domains involved were only categorised into three types, apple, banana and orange. The aim was to detect fruits if found in image or frame and further make prediction on each individual fruit to get their current freshness states. For practically usage in real life, it can be utilised as a fresh fruit recommender, providing guidance for targeted users to select on ideal choice of fruits.

The structure of this project development can be set into two parts, fruit detection and fruit classification. For fruit detection, Faster R-CNN model architecture was used to generate the inference graph where transfer learning approach was used from Inception V2 model consisted of pre-trained weights on COCO datasets. The detection model was developed using Tensorflow Object Detection API, an open-source framework that was readily to be built with any custom datasets. Therefore, with sufficient training data and annotation file, plus modifying several configurations, a detection model on custom dataset is ready to build. On the other hand, as for the fruit classification, the image data acquired were the distinct freshness states of the fruit domains, categorised into 'Fresh' and 'Rotten' states. For common image classification or recognition problem, Convolutional Neural Network (CNN) was widely used as it can achieve outstanding performance compared to traditional machine learning models. During network training, incremental improvements has been made to enhance the performance for the capability to encounter new data. Therefore, a few network architectures and regularization method have been attempted and lastly leveraging the pre-trained weights from VGG-16 to train on self-dataset. To evaluate their performance throughout the training epochs, line graphs and several performance metrics were done for further analysis.

The detection inference graph generated has achieved remarkable results that it has mostly the successive detections when still images or sequential frames were inputted. To assure that the detection algorithm functions well, certain criteria has to be complied with where the details has already been discussed in the previous chapter. Moreover,

for the classification model, it was trained to predict on the fruit class and current freshness states. The fine-tuned transfer learning model has achieved the astonishing prediction results and it was chosen as the best performing model to process on the classification tasks. As computed in the confusion matrix using the testing data for predictions, the accuracy achieved was 99.81% which was high notable for classification results.

Till this stage, the stated objectives were achieved. The fruit detection inference graph and the transfer learning classification model were constructed and built successfully. The detection and classification models were integrated under the same process to work alternately, so as to serve as the fresh fruit recommender for front-end users.

There were implementation issues and challenges met throughout the project development. Firstly, acquiring suitable datasets for the usage of object detection model was found difficult as the unusable images were inspected and filtered manually. In addition, the preparation of annotation file was highly time-consuming and requires good patience especially for multiple object images. All boxes were annotated manually and image after image. Moreover, the hardware limitations had restricted the range of development. To further explain on it, the CNN training has spent much time for a common machine to develop, time has been allocated for the training and waiting process because of deep network training. If a high-end machine is acquired, the development can be proceeded quicker as more resources are available.

Lastly, I feel that I have learnt a lot throughout the project. In the field of computer vision, knowing that how images or data are interpreted and proceeding to further processes, seems most of the problems in real world could be solved using Artificial Intelligence. The motive of conducting this research project was to help and provide guidance to people who are having issues in shopping fresh produces especially fruits. This could save up their time in observing and choosing the ideal product if the system could be deployed for market usage in the future.

### 6.2 Novelties and Contributions

To highlight novelties in this project, the detection on fruits with the outputs of their current freshness states were mentionable. So far, only fruit detection applications were observable, yet the feature of classify the 'Fresh' or 'Rotten' for fruits is still awaiting for deployments. Therefore, this feature has been implemented in this project in the initial phase which was classifying the freshness states of fruits. Moreover, either still images or videos can be inputted into the system and the ideal outcomes were showing the detected fruits, predictions of freshness states on each fruit and fruit counts. The idea of fresh fruit selection recommender is now under pre-phase development, because of the limited fruit domains involved. For current phase, although only three types of fruits are now available to perform freshness prediction, it is confident that this prototype has its practical usage in real world scenario.

### 6.3 Future Works

For current stage, the fruits involved were only three types, apple, banana and orange. Although the overall system is now workable, the trained models somehow had achieved outstanding performance, the system is still unconvincing to be published and provide convenience for targeted users. Hence, a lot of improvements were planned to make the system to have a better market value, working on desirable functions with great proficiencies. To make the system can be utilized more practically, most importantly, the fruit class involved has to be increased. To explain on this, if the model can only detect on three types of fruit, users might not prone to use it, because they did not find effectiveness from it.

To further develop on more fruit classes in this system, more data images of other fruit class has to be acquired and it may consume a lot of time and effort. To gather the freshness states of fruits, developers have to keep an eye on it consistently, record and capture their behaviours or external appearances. With sufficient data for modelling, great performance of the system can be expected, since the previous works has achieved proper usability.

## BIBLIOGRAPHY

- Ali & Wong, 2017. *Automated Fruit Grading System*. Kuala Lumpur, IEEE.
- Anonymous, n.d.. Fruit and vegetables. *Better Health Channel*.
- Anonymous, n.d.. *How Do I Store My Fruits and Veggies So That They Last as Long as Possible?*. [Online]  
Available at: <https://www.halfyourplate.ca/fruits-and-veggies/store-fruits-veggies/>  
[Accessed 5th November 2019].
- Brownlee, J., 2017. *Machine Learning Mastery*. [Online]  
Available at: <https://machinelearningmastery.com/transfer-learning-for-deep-learning/>  
[Accessed 19 November 2019].
- Choi, Cho, Kim & Choi, 2018. *A Real-Time Smart Fruit Quality Grading System Classifying by External Appearance and Intenal Flavor Factors*. Lyon, IEEE.
- Grant, A., n.d.. *Understanding Different Fruit Types*. [Online]  
Available at: <https://www.gardeningknowhow.com/edible/fruits/fegen/different-fruit-types.htm>  
[Accessed 6th November 2019].
- Jayasankar, K. et al., 2018. Fruit Freshness Detection Using Raspberry Pi. *International Journal of Pure and Applied Mathematics*, Volume 119, pp. 1685-1691.
- Kalluri, S. R., 2018. *Kaggle*. [Online]  
Available at: <https://www.kaggle.com/sriramr/fruits-fresh-and-rotten-for-classification>  
[Accessed 10 October 2019].
- Kemiklioglu, E. & Ozen, O., 2018. Design of a Sensor to Detect Fruit Freshness. *International Journal of Scientific & Technology Research*, Volume 4.
- Nishi, T., Kurogi, S. & Matsuo, K., 2017. *Grading Fruits and Vegetables Using RGB-D Images and Convolutional Neural Network*. Honolulu, IEEE.
- Ryan, T., n.d.. *How to Tell If a Fruit Is Spoiled*. [Online]  
Available at: <https://www.livestrong.com/article/473318-how-to-tell-if-a-fruit->

is-spoiled/

[Accessed 29 October 2019].

Sarkar, D. (., 2018. *A Comprehensive Hands-on Guide to Transfer Learning with Real-World Applications in Deep Learning*. [Online]

Available at: <https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning-212bf3b2f27a>

[Accessed 10 November 2019].

Škrjanec, M., 2013. *Vicos*. [Online]

Available at: <https://www.vicos.si/Downloads/FIDS30>

[Accessed February 2020].

Thomson, J. R., 2017. *This is What Eating Moldy Fruit Does to Your Body*. [Online]

Available at: [https://www.huffpost.com/entry/moldy-fruit-okay-to-eat\\_n\\_59402f8ee4b003d5948b6f72](https://www.huffpost.com/entry/moldy-fruit-okay-to-eat_n_59402f8ee4b003d5948b6f72)

[Accessed 5th November 2019].



# APPENDICES

## FINAL YEAR PROJECT WEEKLY REPORT

(Project I / Project II)

|   |                   |
|---|-------------------|
| Trimester, Year: Trimester 3, Year 3              | Study week no.: 1 |
| Student Name & ID: Loh Zhan Herng 16ACB02230      |                   |
| Supervisor: Dr. Aun Yichiet                       |                   |
| Project Title: Fresh Fruits Selection Recommender |                   |

|   |
|---|
| <b>1. WORK DONE</b> <ul style="list-style-type: none"><li>Refining Project Issues</li></ul> |
| <b>2. WORK TO BE DONE</b> <ul style="list-style-type: none"><li>Research studies</li></ul>  |
| <b>3. PROBLEMS ENCOUNTERED</b> <p>-</p>   |
| <b>4. SELF EVALUATION OF THE PROGRESS</b> <p>Still fine, need to catch up soon.</p>         |



Supervisor's signature



Student's signature

# FINAL YEAR PROJECT WEEKLY REPORT

(Project I / Project II)

|   |                   |
|---|-------------------|
| Trimester, Year: Trimester 3, Year 3              | Study week no.: 3 |
| Student Name & ID: Loh Zhan Herng 16ACB02230      |                   |
| Supervisor: Dr. Aun Yichiet                       |                   |
| Project Title: Fresh Fruits Selection Recommender |                   |

## 1. WORK DONE

- Refining Project Issues

## 2. WORK TO BE DONE

- Research studies
- Solve classification problem

## 3. PROBLEMS ENCOUNTERED

-

## 4. SELF EVALUATION OF THE PROGRESS

Still fine, need to catch up soon.



Supervisor's signature



Student's signature

# FINAL YEAR PROJECT WEEKLY REPORT

(Project I / Project II)

|   |                   |
|---|-------------------|
| Trimester, Year: Trimester 3, Year 3              | Study week no.: 5 |
| Student Name & ID: Loh Zhan Herng 16ACB02230      |                   |
| Supervisor: Dr. Aun Yichiet                       |                   |
| Project Title: Fresh Fruits Selection Recommender |                   |

## 1. WORK DONE

- Refining Project Issues
- Solve classification problem

## 2. WORK TO BE DONE

- Research studies
- Make improvement on model

## 3. PROBLEMS ENCOUNTERED

-

## 4. SELF EVALUATION OF THE PROGRESS

Still fine, need to catch up soon.



Supervisor's signature



Student's signature

# FINAL YEAR PROJECT WEEKLY REPORT

(Project I / Project II)

|   |                   |
|---|-------------------|
| Trimester, Year: Trimester 3, Year 3              | Study week no.: 7 |
| Student Name & ID: Loh Zhan Herng 16ACB02230      |                   |
| Supervisor: Dr. Aun Yichiet                       |                   |
| Project Title: Fresh Fruits Selection Recommender |                   |

## 1. WORK DONE

- Refining Project Issues
- Solve classification problem

## 2. WORK TO BE DONE

- Research studies
- Focus on object detection studies

## 3. PROBLEMS ENCOUNTERED

-

## 4. SELF EVALUATION OF THE PROGRESS

Still fine, need to catch up soon.



Supervisor's signature



Student's signature

# FINAL YEAR PROJECT WEEKLY REPORT

(Project I / Project II)

|   |                   |
|---|-------------------|
| Trimester, Year: Trimester 3, Year 3              | Study week no.: 9 |
| Student Name & ID: Loh Zhan Herng 16ACB02230      |                   |
| Supervisor: Dr. Aun Yichiet                       |                   |
| Project Title: Fresh Fruits Selection Recommender |                   |

## 1. WORK DONE

- Refining Project Issues
- Solve classification problem

## 2. WORK TO BE DONE

- Research studies
- Attempt on building Faster R-CNN detection model

## 3. PROBLEMS ENCOUNTERED

Needs better understand in localization in deep learning field.

## 4. SELF EVALUATION OF THE PROGRESS

Still fine, need to catch up soon.



Supervisor's signature



Student's signature

# FINAL YEAR PROJECT WEEKLY REPORT

(Project I / Project II)

|   |                    |
|---|--------------------|
| Trimester, Year: Trimester 3, Year 3              | Study week no.: 11 |
| Student Name & ID: Loh Zhan Herng 16ACB02230      |                    |
| Supervisor: Dr. Aun Yichiet                       |                    |
| Project Title: Fresh Fruits Selection Recommender |                    |

## 1. WORK DONE

- Refining Project Issues
- Fruit classification model is trained
- Object detection model is trained

## 2. WORK TO BE DONE

- Set test cases to do system testing and verifications

## 3. PROBLEMS ENCOUNTERED

-

## 4. SELF EVALUATION OF THE PROGRESS

Still fine, need to catch up soon.



Supervisor's signature



Student's signature

# FINAL YEAR PROJECT WEEKLY REPORT

(Project I / Project II)

|   |                    |
|---|--------------------|
| Trimester, Year: Trimester 3, Year 3              | Study week no.: 13 |
| Student Name & ID: Loh Zhan Herng 16ACB02230      |                    |
| Supervisor: Dr. Aun Yichiet                       |                    |
| Project Title: Fresh Fruits Selection Recommender |                    |

## 1. WORK DONE

- Refining Project Issues
- Fruit classification model is trained
- Object detection model is trained
- Test on the outcomes of the algorithm

## 2. WORK TO BE DONE

- Reporting

## 3. PROBLEMS ENCOUNTERED

-

## 4. SELF EVALUATION OF THE PROGRESS

Still fine, need to catch up soon.



Supervisor's signature



Student's signature

# FRESH FRUIT SELECTION RECOMMENDER

Prepared by: Loh Zhan Heng      Supervised by: Dr. Aun Yichiet

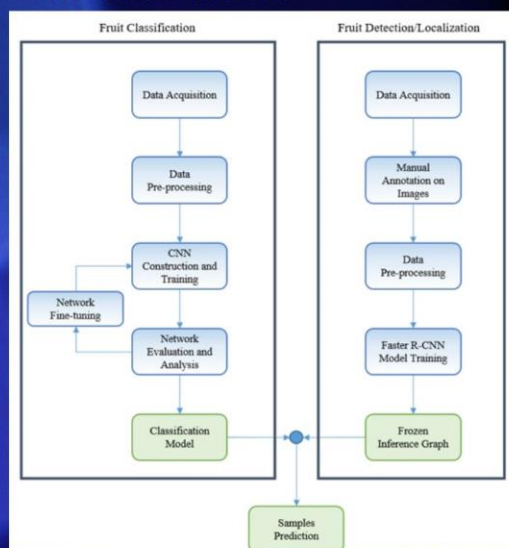
## ~ Problem Statement ~

- Time-consuming and manpower intensive for fruit quality assurance process
- Lack of references or measurement towards the fruit quality for common buyers and consumers

## ~ Problem Objectives ~

- To build Faster R-CNN with customized datasets
- To develop a multi-class classification to perform predictions
- To utilize transfer learning approach on CNNs
- To integrate both detection and classification models

## ~ Research Method ~



## ~ Results ~

- **Fruit Detection** : achieved reliable detection performance for multiple objects.
- **Fruit Classification** : an accuracy of 99.81% was achieved for freshness states prediction.

## ~ Discussion ~

The prototype is able to detect and perform freshness prediction on the specified fruit domains, including apple, banana and orange. Whether the input is an image or a video frame, the system is able to receive and proceeds with back-end processes and return the outputs to the front-end users. The outputs are the locations of detected fruits, the freshness states of each individual fruit and the total fruit counts value.

# Specially done with Deep Learning and Computer Vision techniques.







Feedback Studio x Turnitin x +

turnitin.com/newreport.asp?eq=1&eb=1&esm=8&oid=1306145741&m=0&svr=43&r=91.01850789976824&lang=en\_us&bypass\_cv=1

preferences

turnitin Originality Report

Processed on: 24-Apr-2020 10:41 +08  
ID: 1306145741  
Word Count: 12192  
Submitted: 1

## Fresh Fruits Selection Recommender

By Loh Zhan Heng

| Similarity by Source |    |
|----------------------|----|
| Similarity Index     | 5% |
| Internet Sources:    | 1% |
| Publications:        | 3% |
| Student Papers:      | 3% |

Document Viewer

include quoted include bibliography excluding matches < 8 words

mode: show highest matches together Change mode

**CHAPTER 1 INTRODUCTION 1.1 Introduction** Fresh fruits **are** generally known **as** fresh produces **in** 37

the state of freshly harvested from fruit farms or orchards. They are the fleshy products of plant growth majority of their flesh parts are edible. There are up to hundred types of fruits existing in the world and their types are distinguishable through colours, shapes and also tastes. According to Better Health Channel (n.d.), fruits can be categorized into several types or classes, which includes apples and

**pears, citrus, stone fruit, tropical and exotic, berries, melons** as well as **tomatoes and avocados**. Fresh **and** undamaged **fruits** 15

are usually safe to consume by human or animals, but what is the length of time that the fruits are able to stay fresh in various circumstances or environments?

**After a certain period of time**, the freshness **of the fruit** **may** not **be** preserved. According to 22

Ryan (n.d.), some of the spoilt fruits have obvious appearance such as growing of mould that human can easily inspect on their surface, yet there are also certain types of fruit that human needs to take a more hands-on method such as smelling its scent or feeling its body hardness to identify whether they are beginning to spoil or no longer fresh. Moreover, given a situation that certain fruits are stored in a cool place like refrigerator, their freshness are more likely to be maintained longer as the environment in the refrigerator keeps their moisture. On the other hand, if certain fruits like apples and oranges are kept under the room temperature, it will be a doubt that how long the fruits can maintain its freshness as moulds can grow easily in such condition of open- aired and room temperature. Moulds are the microscopic fungi that live and grow on plants surface. 1.2 Problem Statement ? Time-consuming and manpower intensive for fruit quality assurance process. Providing a scenario in huge fruit distribution warehouse, it consists various types of fruits and up to tons of stocks to be distributed for fruit wholesalers. First and foremost, bearing with numerous

**1** < 1% match (publications)  
[Mohammed A. H. Ali, Kelvin Wong Thai, "Automated fruit grading system", 2017 IEEE 3rd International Symposium in Robotics and Manufacturing Automation \(ROMA\), 2017](#)

**2** < 1% match (publications)  
["Computing Algorithms with Applications in Engineering", Springer Science and Business Media LLC, 2020](#)

**3** < 1% match (Internet from 11-Jan-2020)  
<https://hub.packtpub.com/how-to-leverage-transfer-learning-using-pretrained-cnn-models-tutorial/>

**4** < 1% match (publications)  
["Advances in Multimedia Information Processing - PCM 2018", Springer Science and Business Media LLC, 2018](#)

**5** < 1% match (publications)  
[Shawon Ashraf, Ivan Kadery, Md Abdul Ahad Chowdhury, Tahsin Zahin Mahbub, Rashedur M. Rahman, "Fruit Image Classification Using Convolutional Neural Networks", International Journal of Software Innovation, 2019](#)

**6** < 1% match (student papers from 15-Oct-2015)  
[Submitted to Higher Education Commission Pakistan](#)

|  |            |                            |                  |
|--|------------|----------------------------|------------------|
| <b>Universiti Tunku Abdul Rahman</b>   |            |                            |                  |
| <b>Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)</b> |            |                            |                  |
| Form Number: FM-IAD-005  | Rev No.: 0 | Effective Date: 01/10/2013 | Page No.: 1 of 1 |




## FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

|                                     |                                     |
|-------------------------------------|-------------------------------------|
| <b>Full Name(s) of Candidate(s)</b> | Loh Zhan Herng                      |
| <b>ID Number(s)</b>                 | 16ACB02230                          |
| <b>Programme / Course</b>           | Bachelor of Computer Science (HONS) |
| <b>Title of Final Year Project</b>  | Fresh Fruits Selection Recommender  |

| <b>Similarity</b>   | <b>Supervisor's Comments<br/>(Compulsory if parameters of originality exceeds the limits approved by UTAR)</b> |
|---|--|
| <b>Overall similarity index:</b> <u>5%</u><br><br><b>% Similarity by source</b><br>Internet Sources: <u>1%</u> %<br>Publications: <u>3%</u> %<br>Student Papers: <u>3%</u> %  |  |
| <b>Number of individual sources listed of more than 3% similarity:</b> <u>0</u>   |  |
| <b>Parameters of originality required and limits approved by UTAR are as Follows:</b><br>(i) Overall similarity index is 20% and below, and<br>(ii) Matching of individual sources listed must be less than 3% each, and<br>(iii) Matching texts in continuous block must not exceed 8 words<br><i>Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.</i> |  |

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

*Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.*

  
 \_\_\_\_\_  
 Signature of Supervisor  
  
 Name: AUN YC  
 \_\_\_\_\_  
 Date: 24/4/2020  
 \_\_\_\_\_

\_\_\_\_\_  
 Signature of Co-Supervisor  
  
 Name: \_\_\_\_\_  
 \_\_\_\_\_  
 Date: \_\_\_\_\_  
 \_\_\_\_\_



# UNIVERSITI TUNKU ABDUL RAHMAN

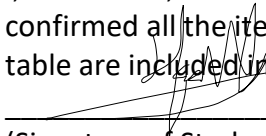
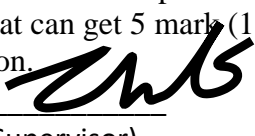
## FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (KAMPAR CAMPUS)

### CHECKLIST FOR FYP2 THESIS SUBMISSION

|                 |                 |
|-----------------|-----------------|
| Student Id      | 16ACB02230      |
| Student Name    | Loh Zhan Herng  |
| Supervisor Name | Dr. Aun Yichiet |

| TICK (✓) | DOCUMENT ITEMS   |
|----------|--|
|          | Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item. |
| ✓        | Front Cover  |
| ✓        | Signed Report Status Declaration Form  |
| ✓        | Title Page   |
| ✓        | Signed form of the Declaration of Originality  |
| ✓        | Acknowledgement  |
| ✓        | Abstract   |
| ✓        | Table of Contents  |
| ✓        | List of Figures (if applicable)  |
| ✓        | List of Tables (if applicable)   |
|          | List of Symbols (if applicable)  |
| ✓        | List of Abbreviations (if applicable)  |
| ✓        | Chapters / Content   |
| ✓        | Bibliography (or References)   |
| ✓        | All references in bibliography are cited in the thesis, especially in the chapter of literature review   |
| ✓        | Appendices (if applicable)   |
| ✓        | Poster   |
| ✓        | Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)   |

\*Include this form (checklist) in the thesis (Bind together as the last page)

|  |   |
|--|---|
| <p>I, the author, have checked and confirmed all the items listed in the table are included in my report.</p> <p></p> <p>(Signature of Student)</p> <p>Date: 24/04/2020</p> | <p>Supervisor verification. Report with incorrect format can get 5 mark (1 grade) reduction.</p> <p></p> <p>(Signature of Supervisor)</p> <p>Date: 24/4/2020</p> |
|--|---|