Non-Destructive Carabao Mango Sorter and Grader based on Physical Characteristics using Machine Learning

---

A Thesis
Presented to the Faculty of the
Department of Electronics and Computer Engineering
Gokongwei College of Engineering
De La Salle University

In Partial Fulfillment of the
Requirements for the Degree of
Bachelor of Science in Computer Engineering

---

by

BANAL Kenan A.
BAUTISTA Francis Robert Miguel F.
HERMOSURA Don Humphrey L.
SALAZAR Daniel G.

November, 2025

De La Salle University

THESIS APPROVAL SHEET

This thesis entitled **Non-Destructive Carabao Mango Sorter and Grader based on Physical Characteristics using Machine Learning**, prepared and submitted by:

BANAL, Kenan A.
BAUTISTA, Francis Robert Miguel F.
HERMOSURA, Don Humphrey L.
SALAZAR, Daniel G.

with group number AISL-1-2425-C5 in partial fulfillment of the requirements for the degree of **Bachelor of Science in Computer Engineering**, (**BS-CPE**) has been examined and is recommended for acceptance and approval.

PANEL OF EXAMINERS

____

Dr. Donabel de Veas Abuan
Chair

____

Dr. Alexander Co Abad
Member

____

Engr. Dino Dominic F. Ligutan
Member

____

Dr. Reggie C. Gustilo
Adviser

Date: ______________________________

De La Salle University

ACKNOWLEDGMENT

We would like to first acknowledge God the almighty for blessing and guiding us in this thesis paper and to our family members for their support and encouragement. Furthermore, we would like to thank our current adviser Dr. Reggie C. Gustilo for his support, guidance, and patience throughout our thesis. We would also like to thank our previous adviser Dr. Ana Antoniette Illahi for helping us with the thesis proposal and the proposal defense. Furthermore, we would like to thank our external advisor Dr. Cecilia Reyes for giving her experienced insight and guidance in research related to Carabao mangoes in the Philippines. Moreover, we would like to thank our panelists Dr. Donabel de Veas Abuan, Dr. Alexander Co Abad, and Engr. Dino Ligutan for their comments and suggestions to improve our thesis. Additionally, we would like to thank our previous panelist member Engr. Jose Martin Maningo for giving advice during the thesis proposal defense. Lastly, we would like to thank the farmers Jerry Bravante, Ivan Joseph Palma, Ailen Q. Redome, and Jesus Redome for participating in our thesis paper and sharing their experience in mango farming.

ABSTRACT

Current machine learning systems for Carabao mango sorting and grading primarily classify mangoes based on individual physical characteristics such as size, bruises, and ripeness. However, limited research has explored systems that can prioritize these characteristics according to user-defined preferences with customizable weighting. This study introduces a flexible Carabao mango grading and sorting system that integrates machine learning with a user-adjustable weighting mechanism, enabling dynamic prioritization or exclusion of ripeness, size, and bruises based on specific requirements. Different machine learning methods were evaluated for classifying ripeness and bruises separately. The dataset consisted of both publicly available images and researchers' own Carabao mango images, with a data split of 70-15-15 for training, validation, and testing, respectively. Convolutional Neural Network (CNN) models, particularly EfficientNetV2, achieved optimal performance for ripeness and bruise classification with accuracy scores of 98% and 99%, respectively. To validate these results, a comparative analysis between the best-performing model and expert evaluations was conducted, yielding an overall agreement accuracy of 91.02%. For size classification, OpenCV method demonstrated an accurate performance, with measured area percent difference of 4.51% to the manual measurement by getting its length and width, respectively. Finally, the image acquisition system, consisting of an Raspberry Pi (RPi) with a camera module and conveyor belt setup, successfully demonstrated the proposed grading and sorting process using the developed linear grading formula.

Index Terms—Machine Learning, Carabao mango, Bruises, Ripeness, Microcontrollers.

TABLE OF CONTENTS

Thesis Approval Sheet **ii**

Acknowledgment **iv**

Abstract **v**

Table of Contents **vi**

List of Figures **xii**

List of Tables **xiv**

Abbreviations and Acronyms **xv**

Notations **xvi**

Glossary **xvii**

Listings **xviii**

Chapter 1 INTRODUCTION **1**
- 1.1 Background of the Study 2
- 1.2 Prior Studies 4
- 1.3 Problem Statement 5
- 1.4 Objectives and Deliverables 6
  - 1.4.1 General Objective (GO) 6
  - 1.4.2 Specific Objectives (SOs) 7
  - 1.4.3 Expected Deliverables 7
- 1.5 Significance of the Study 9
  - 1.5.1 Technical Benefit 10
  - 1.5.2 Social Impact 11
  - 1.5.3 Environmental Welfare 11
- 1.6 Assumptions, Scope, and Delimitations 11
  - 1.6.1 Assumptions 11
  - 1.6.2 Scope 12
  - 1.6.3 Delimitations 12

Chapter 2 LITERATURE REVIEW **14**
2.1 Existing Work 15
2.1.1 Deep Learning Classification Algorithms 19
2.2 Lacking in the Approaches 21
2.3 Summary 22

Chapter 3 THEORETICAL CONSIDERATIONS **24**
3.1 Introduction 25
3.2 Relevant Theories and Models 25
3.3 Technical Background 26
3.4 Conceptual Framework Background 26
3.5 Software Concepts 27
3.5.1 Thresholding 27
3.5.2 Object Size Calculation 28
3.5.3 Convolutional Neural Network 29
3.6 Hardware Concepts 29
3.6.1 Camera Module 29
3.6.2 4 Channel Relay 30
3.6.3 Gear Ratio 31
3.7 Summary 31

Chapter 4 DESIGN CONSIDERATIONS **32**
4.1 Introduction 33
4.2 Engineering Standards 33
4.2.1 Electrical Certifications 33
4.2.2 Safety of Machinery 33
4.2.3 Safety Requirements for Technology Equipment 34
4.2.4 Open-source Software Compliance 34
4.3 System Architecture 34
4.4 Hardware Considerations 36
4.4.1 General Prototype Framework 36
4.4.2 Prototype Flowchart 37
4.4.3 Prototype 3D Model 39
4.4.4 Hardware Specifications 39
4.4.4.1 Raspberry Pi 39
4.4.4.2 Raspberry Pi Camera 42
4.4.4.3 DC Motor 44
4.4.4.4 MicroSD Card 45
4.4.4.5 LED Lights 46
4.4.4.6 Power Supply 47

4.4.4.7 4 Channel Relay Module 49
4.4.4.8 RPi Power Supply 50
4.4.4.9 Mini Conveyor Single Narrow 52
4.4.4.10 Mini Conveyor Double Narrow 53
4.5 Software Considerations 54
4.5.1 PyTorch 54
4.5.2 OpenCV 54
4.5.3 CustomTkinter 55
4.6 User Interface 55
4.7 Summary 55

Chapter 5 METHODOLOGY **57**
5.1 Introduction 60
5.2 Research Approach 60
5.3 Hardware Design 60
5.3.1 Mango Position 61
5.4 Software Design 61
5.4.1 Machine Learning Methods 64
5.4.2 Optimizer 64
5.4.3 Data Loading Optimization 65
5.4.4 Data Transfer Optimization 65
5.4.5 Mixed Precision Training 66
5.4.6 Adaptive learning Rate Schedulers 66
5.4.7 CrossEntropy Loss with Label Smoothing 67
5.4.8 Early Stopping and Checkpointing 67
5.4.9 Input Resolution 68
5.4.10 Regularization 68
5.5 Data Collection Methods 69
5.6 Testing and Evaluation Methods 70
5.6.1 Data Augmentation and Splitting 73
5.6.2 Comparative Test of CNN Models 76
5.6.3 Benchmarking Best CNN Model on +10k Mango Dataset 77
5.6.4 Classification Report 79
5.6.4.1 Confusion Matrix 80
5.6.4.2 Precision 81
5.6.4.3 Recall 81
5.6.4.4 F1 Score 81
5.6.4.5 Accuracy 82
5.6.5 Ripeness Training and Testing 82

5.6.5.1 Green . . . 83
5.6.5.2 Yellow_Green . . . 83
5.6.5.3 Yellow . . . 84
5.6.6 Bruises Training and Testing . . . 84
5.6.6.1 Stem End Rots . . . 85
5.6.6.2 Dendritic Spot . . . 85
5.6.6.3 Anthracnose . . . 85
5.6.6.4 Sapburn . . . 85
5.6.6.5 Skin Browning . . . 86
5.6.6.6 Lenticel Spot . . . 86
5.6.7 Size Determination . . . 86
5.6.7.1 Determining the Ranges for Mango Sizes Based on Area 86
5.6.7.2 Estimating the Carabao Mango Size . . . 87
5.6.7.3 K-Means Classification . . . 88
5.6.7.4 Quantile-Based Classifier . . . 89
5.7 Mango Formula with User Priority . . . 90
5.8 Expert Evaluation Methodology . . . 91
5.9 Summary . . . 91

Chapter 6 RESULTS AND DISCUSSIONS **95**
6.1 Training and Testing Results of the Model . . . 98
6.1.1 Ripeness Classification Results . . . 98
6.1.1.1 Naive Bayes . . . 98
6.1.1.2 KMeans . . . 100
6.1.1.3 KNN . . . 100
6.2 Achieving the Highest Accuracy in CNN Models . . . 102
6.2.1 Analyzing the Accuracy of Different CNN Network . . . 102
6.2.2 Analysis of Table 6.6 and Table 6.5 . . . 107
6.2.3 Analysis of Confusion Matrix together with Validation Loss and Accuracy . . . 111
6.2.3.1 Ripeness Classification . . . 111
6.2.3.2 Bruises Classification . . . 117
6.2.3.3 CNN . . . 121
6.2.4 Bruises Classification Results . . . 125
6.2.4.1 CNN . . . 125
6.3 Comparative Analysis: Model Performance vs. Expert Benchmark . . . 128
6.3.1 Comparative Results . . . 128
6.4 Size Determination Results . . . 134
6.4.1 Actual and Estimated Length . . . 134

6.4.2 Actual and Estimated Width 135
6.4.3 Calculated Area and Estimated Area 136
6.4.4 Summarized Size Results 137
6.5 Formula with User Priority 139
6.6 Physical Prototype 141
6.6.1 Version 1: Barebone with Black Conveyor Sheets 141
6.6.2 Version 2: Enclosed with White Conveyor Sheets and Physical Sorter 141
6.7 Software Application 144
6.7.1 Version 1: Progress Bar with Black Conveyor Sheets 144
6.7.2 Version 2: Improved UI without Progress Bar 146
6.7.3 Mango Image Sorting 147
6.7.4 Error Handling 147
6.7.5 Sample UI Outputs 148
6.8 Summary 148

Chapter 7 CONCLUSIONS, RECOMMENDATIONS, AND FUTURE DIRECTIVES 152
7.1 Concluding Remarks 153
7.1.1 Objectives Achieved 153
7.1.1.1 GO: To develop a user-priority-based grading and sorting system for Carabao mangoes, using machine learning and computer vision techniques to assess ripeness, size, and bruises. 153
7.1.1.2 SO1: To make an image acquisition system with a conveyor belt for automatic sorting and grading mangoes. 153
7.1.1.3 SO2: To get the precision, recall, F1 score, confusion matrix, and train and test accuracy metrics for classifying the ripeness and bruises with an accuracy score of at least 90%. 154
7.1.1.4 SO3: To create a microcontroller-based system to operate the image acquisition system, control the conveyor belt, and process the mango images through machine learning. 154
7.1.1.5 SO4: To grade mangoes based on user priorities for size, ripeness, and bruises. 154
7.1.1.6 SO5: To classify mango ripeness based on image data using machine learning algorithms such as kNN, k-mean, and Naïve Bayes. 155

7.1.1.7 SO6: To classify mango size based on image data by getting its length and width using OpenCV, geometry, and image processing techniques. 155

7.1.1.8 SO7: To classify mango bruises based on image data by employing machine learning algorithms. 155

7.2 Contributions 155

7.3 Recommendations 156

7.4 Future Prospects 157

References **158**

Appendix A STUDENT RESEARCH ETHICS CLEARANCE **162**

Appendix B REVISIONS TO THE PROPOSAL **164**

Appendix C REVISION TO THE FINAL **170**

Appendix D QUESTIONNAIRE TO THE EXPERT **173**

Appendix E CERTIFICATE FROM FARMERS **180**

Appendix F DATASET VALIDATION **188**

LIST OF FIGURES

1.1 Carabao Mangoes at Different Ripeness Stages (Guillermo et al., 2019) . . . 2

2.1 Prototype for Grading Mangoes (Adam et al., 2022) . . . 15
2.2 System Block Diagram (Samaniego Jr. et al., 2023) . . . 16
2.3 Background Removal of Mango (Tomas et al., 2022) . . . 18

3.1 Theoretical Framework Diagram. . . . 25
3.2 Conceptual Framework Diagram. . . . 26

4.1 Hardware Schematic . . . 35
4.2 Prototype Framework . . . 37
4.3 Prototype Main Flowchart . . . 38
4.4 Initial 3D Model of the Prototype . . . 40
4.5 Raspberry Pi 4 Model B . . . 41
4.6 Raspberry Pi Camera Module Version 2 . . . 43
4.7 12 Volt DC Gear Motor . . . 44
4.8 SanDisk Ultra MicroSD Card . . . 45
4.9 LED Light Strip . . . 46
4.10 Bench Power Supply . . . 48
4.11 4 Channel Relay Module . . . 49
4.12 Power Supply for the RPi . . . 50
4.13 Single Narrow Mini Conveyor . . . 52
4.14 Double Narrow Mini Conveyor . . . 53

5.1 conveyor positioning . . . 62
5.2 Image Acquisition Dimensions . . . 63
5.3 Camera Setup . . . 69
5.4 Carabao Mangoes Image Dataset Collection . . . 70
5.5 Sample Mango Image . . . 71
5.6 Collecting Mango on a Farm . . . 72
5.7 CNN Ripeness 70-15-15 Image Datasplit . . . 74
5.8 CNN Bruises 70-15-15 Image Datasplit . . . 75
5.9 Bruises Image Datasplit . . . 79
5.10 Ripeness Image Datasplit . . . 80
5.11 Carabao Mango Ripeness Stages (Bureau of Agriculture and Fisheries Product Standards, 2004) . . . 83
5.12 Different Kinds of Mango Defects (Scott Ledger and Cooke, 2000) . . . 84

5.13 Length and Width of Mango (Bureau of Agriculture and Fisheries Product Standards, 2006) 87

6.1 Ripeness Confusion Matrix using Naive Bayes 99
6.2 Ripeness Confusion Matrix using KMeans 101
6.3 Ripeness Confusion Matrix using KNN 102
6.4 Ripeness Training and Testing of EfficientNet-B5 113
6.5 Ripeness Training and Testing of EfficientNet-B6 115
6.6 Ripeness Training and Testing of EfficientNetV2-B3 116
6.7 Bruises Training and Testing of EfficientNet-B2 118
6.8 Bruises Training and Testing of EfficientNet-B3 120
6.9 Bruises Training and Testing of EfficientNetV2-B3 122
6.10 EfficientNetV2-B3 Ripeness Confusion Matrix 125
6.11 EfficientNetV2-B3 Ripeness Accuracy and Loss Graph 125
6.12 EfficientNetV2-B3 Bruises Confusion Matrix 127
6.13 EfficientNetV2-B3 Bruises Accuracy and Loss Graph 128
6.14 Bar Graph of Actual vs Estimated Length 134
6.15 Bar Graph of Actual vs Estimated Width 135
6.16 Bar Graph of Actual vs Estimated Area 136
6.17 List of Size Results 137
6.18 Size Area Classification with cm^2 Gap 138
6.19 Tested 42 Mangoes 138
6.20 Only Bruises as a None Zero Value 139
6.21 Only Ripeness and Bruises as a None Zero Value 140
6.22 Only Ripeness as a None Zero Value 140
6.23 Version 1 of the Prototype 142
6.24 Hardware View 143
6.25 Version 2: Improved Prototype 145
6.26 Version 1 of the RPi's User Interface 146
6.27 Version 2 of the RPi's User Interface 147
6.28 Mango Image Data Sorting 148
6.29 Error Messages 149
6.30 Error message for Letter as Input 149
6.31 Help Page UI 150
6.32 Sample Ripeness and Bruises Results 151

LIST OF TABLES

1.1 Expected Deliverables per Objective 8
1.1 Expected Deliverables per Objective 9

2.1 Comparison of Existing Studies 18
2.2 Comparison of Sorting Algorithm Models 21

5.1 Summary of methods for reaching the objectives 58
5.2 Confusion Matrix Example 80

6.1 Summary of methods for achieving the objectives 96
6.2 Ripeness Classification Report using Naive Bayes 99
6.3 Ripeness Classification Report using KMeans 100
6.4 Ripeness Classification Report using KNN 101
6.5 CNN Training Results for Ripeness using GPU 103
6.6 CNN Bruises Results for bruises using CPU 104
6.7 Accuracy of Different CNN Models 104
6.8 Test Accuracy of Different EfficientNet Version 1 and 2 105
6.9 EfficientNetV2-B3 Ripeness Classification Report with Precision: 0.9848, Recall: 0.9847, F1 Score: 0.9847 124
6.10 EfficientNetV2-B3 Bruises Classification Report with Precision: 0.9887, Recall: 0.9886, F1 Score: 0.9886 127
6.11 Expert Classification Results for Mango Phenotypic Traits from Professional 1 129
6.12 Expert Classification Results for Mango Phenotypic Traits from Professional 2 130
6.13 Expert Classification Results for Mango Phenotypic Traits from Professional 3 132

ABBREVIATIONS

AC	Alternating Current	13
CNN	Convolutional Neural Network	v
CPU	Central Processing Unit	41
GPU	Graphics Processing Unit	77
GUI	Graphical User Interface	55
KNN	K-Nearest Neighbors	26
LED	Light Emitting Diode	25
RESNET	Residual Network	102
RPI	Raspberry Pi	v
UI	User Interface	55
VGGNET	Visual Geometry Group Network	76

NOTATION

$B\,(P)$	Bruises User Priority/Weight	90
$b\,(p)$	Bruises AI Prediction	90
$R\,(P)$	Ripeness User Priority/Weight	90
$r\,(p)$	Ripeness AI Prediction	90
$S\,(P)$	Size User Priority/Weight	90
$s\,(p)$	Size AI Prediction	90
$D\,(p, d, f)$	Real World Dimension	28
p	Pixel Dimension	28
d	Distance from Camera to Object	28
f	Focal Length	28
g	Green Mango Ripeness	128
yg	Yellow_Green Mango Ripeness	128
y	Yellow Mango Ripeness	128
b	Bruised Mango	128
nb	Non-bruised Mango	128

392 # GLOSSARY

393	Adam	An optimizer that computes adaptive learning rates for each parameter, combining the advantages of two other extensions of stochastic gradient descent.
394	AdamW	A variant of Adam that decouples the weight decay from the gradient update, which often leads to better generalization and more stable convergence.
395	bruises	The darkened black or brown region on the mango's skin resulting from impact, compression, or over-ripening, indicating tissue damage beneath the surface.
396	ripeness	The stage at which a mango has developed its optimal color, texture, flavor, and aroma for consumption.
397	Carabao mango	A popular variety of mango grown in the Philippines, known for its sweet and juicy flesh.
398	accuracy score	A performance metric that measures the overall proportion of correct predictions made by a machine learning model.
399	confusion matrix	A table that summarizes the performance of a classification model, showing the number of true positives, true negatives, false positives, and false negatives.
400	machine learning	A subset of Artificial Intelligence that enables systems to learn and improve from data.
401	computer vision	The use of cameras and algorithms to provide imaging-based inspection and analysis.
402	microcontroller	A small computing device that controls other parts of a system such as sensors.
403	Precision	A performance metric that reflects the percentage of instances classified as positive that are truly positive.
404	recall	A performance metric that measures the proportion of actual positive instances that the model correctly identified.
405	User Priority-Based Grading	A customizable grading system where users can assign weights to grading factors.

LISTINGS

5.1 Datasplit Logs 94
6.1 Sorting the Mangoes 144

Chapter 1

INTRODUCTION

1.1 Background of the Study

Carabao mango *(Mangifera indica L.)* is a variety of a mango that is found and cultivated in the Philippines. It is known for its sweet signature taste that was recognized sweetest in the world in the Guinness Book of World Records in 1995. The mango was named after the national animal of the Philippines, a native breed of buffalo. On average, it is 12.5 cm in length and 8.5 cm in diameter, having a bright yellow color when ripe as seen in Figure 1.1 (Knight et al., 2009). It is often cultivated during late May to early July (Bayogan and Secretaria, 2019).

Likewise, the Philippines produced an estimated 596.34 thousand metric tons of mangoes during the April to June 2023 quarter, marking an 11.4 percent increase from the 535.43 thousand metric tons harvested in the same three-month period of 2022. Of this total output, the mango variety accounted for the vast majority at 495.06 thousand metric tons, or 83.0 percent of the nation's entire mango production (Philippine Statistics Authority, 2023).

Fig. 1.1 Carabao Mangoes at Different Ripeness Stages (Guillermo et al., 2019)

This shows that mangoes are a highly valued fruit in the Philippines as it is not only the country's national fruit but also amongst the leading agricultural exports of the country,

De La Salle University

ranking only third below bananas and pineapples. This gives the country the 9th slot amongst the leading exporters of Mangoes across the world. Attributed to this ranking is the country's export of both fresh and dried mangoes, as well as low tariff rates. This allows the country to export a large quantity of the fruit in countries such as Singapore, Japan, and the USA as they can enter duty free markets provided by the World Trade Organization and Japan. Due to this, the mangoes have become a major source of income to an estimated 2.5 million farmers in the country (Centino et al., 2020).

Before mangoes are sold in markets, they first undergo multiple post-harvest processes. This is to ensure that the mangoes that arrive in markets are utmost quality before being sold to consumers. Moreover, it ensures that mangoes are contained and preserved properly such that they do not incur damages and/or get spoiled on its transportation to the market . Processing of the mango involves pre-cooling, cleaning, waxing, classification, grading, ripening, packaging, preservation, storage, packing, and transportation (Patel et al., 2019).

Among the processes that mangoes undergo, classification and grading is important as it allows the manufacturer to separate mangoes with good qualities versus mangoes with poor qualities. According to a study by (Lacap et al., 2021), size, length, width, volume, density, indention, and grooves are aspects that determine the maturity of mangoes. These traits are being checked along with the ripeness of the mango, sightings of bruise injury, and cracks on the fruit as these aspects affect the sellability of the fruit as well as the chances of it getting spoiled sooner.

Previous studies have been made to automate the sortation process of the mangoes. Among these is a research done by Abbas et al. (2018), which focuses on classification of mangoes using their texture and shape features. They do this by, first, acquiring an image of the mango using a digital camera. Then, these images are fed to the MaZda package, which is a software originally developed for magnetic resonance imaging. Within the MaZda package is the B11 program, which uses Principal Component Analysis, Linear Discriminant Analysis, Nonlinear Discriminant Analysis, and texture classification to extract features from the mango, which in this case are the length, width, and texture. This data is then compared to a database in order to classify any given mango (Abbas et al., 2018).

Another study is done by Rizwan Iqbal and Hakim (2022), which classifies mangoes based on their color, volume, size, and shape This is done by making use of Charge Coupled Devices, Complementary Metal-Oxide Semiconductor sensors, and 3-layer CNN. To classify the mangoes, images are first captured and preprocessed to be used as a data set (Rizwan Iqbal and Hakim, 2022). This data set is then augmented to be used as a model for the 3-layer CNN. After extracting the features of the mango, the 3-layer CNN is used as a method for their classification as it can mimic the human brain in pattern recognition, and process data for decision making. This is important as some mangoes have very subtle differences which make it difficult to differentiate them.

1.2 Prior Studies

A paper written by Amna et al. (2023), designed an automated fruit sorting machine based on the quality through an image acquisition system and CNN. Furthermore, the results of the paper show that the image processing detection score was 89% while that of the tomatoes was 92% while the CNN model had higher validity of 95% for mangoes and 93% for tomatoes. 15%, while the percentage of distinction between the two groups was reported to be 5% respectively (Amna et al., 2023). Despite the high accuracy score in detecting mango defects, the fruit sorting system only sorts based on the mango defects and not on ripeness, and weight.

Furthermore, the article presented by Guillergan et al. (2024) designed an Automated Carabao mango classifier, in which the mango image database is used to extract the features like size, area along with the ratio of the spots for grading using Naïve Bayes Model. For the results, the Naïve Bayes' model recognized large and rejected mangoes with 95% accuracy and the large and small/medium difference with a 7% error, suggesting an application for quality differentiation and sorting in the mango business industry. Despite the high accuracy of classifying Carabao mangoes, the researchers used a high quality DSLR camera for the image acquisition system without any microcontroller to control the mangoes (Guillergan et al., 2024).

1.3 Problem Statement

As mangoes are among the top exports of the Philippines (Centino et al., 2020), assessing the physical deformities is a necessity. The physical deformities of the mango can determine the global competitiveness of the country. Having higher quality exports can often lead to gaining competitive edge, increase in demand, increase export revenues, and becoming less susceptible to low-wage competition (D'Adamo, 2018). In order to increase the quality of mango fruit exports, a key post-harvest process is done, which is sorting and grading. Mango sorting and grading then becomes important to determine which batches are of high quality and can be sold for a higher price, and which batches are of low quality and can only be sold for a low price (Tai et al., 2024). Traditionally, fruit sorting and grading is inefficient as it is done manually by hand. Some tools are used such as porous ruler to determine fruit size and color palette for color grading. However, among the problems encountered in the process of manually sorting and grading mangoes are susceptibility to human error and requiring a number of laborers to do the task.

With the current advancements in technology, some researchers have already taken steps to automate the process of sorting and grading mangoes. However, these attempts would often only consider some of the aspects pertaining to size, ripeness, and bruises but not dynamically change the method of sorting and grading. Furthermore, most of the journal articles have a fix static method in grading and sorting the mangoes. This means that it doesn't take into consideration the user's priority when grading and sorting the mangoes. Lastly, not all research approaches were able to implement a hardware for their algorithm, limiting their output to only a software implementation and not an embedded system. As such the proposed system would assess the quality of the mango based on all the mentioned mango traits, namely size, bruises, and ripeness while also taking into consideration being non-destructive and the user's priority when grading and sorting the mangoes. These aspects are important because, as was previously mentioned, there is a need to develop a user priority based mango sorter that takes into account all these aspects at the same time while being non-destructive.

1.4 Objectives and Deliverables

1.4.1 General Objective (GO)

- GO: To develop a user-priority-based grading and sorting system for Carabao mangoes, using machine learning and computer vision techniques to assess ripeness, size, and bruises. ;

1.4.2 Specific Objectives (SOs)

- SO1: To make an image acquisition system with a conveyor belt for automatic sorting and grading mangoes. ;
- SO2: To get the precision, recall, F1 score, confusion matrix, and train and test accuracy metrics for classifying the ripeness and bruises with an accuracy score of at least 90%.;
- SO3: To create a microcontroller-based system to operate the image acquisition system, control the conveyor belt, and process the mango images through machine learning. ;
- SO4: To grade mangoes based on user priorities for size, ripeness, and bruises. ;
- SO5: To classify mango ripeness based on image data using machine learning algorithms such as kNN, k-mean, and Naïve Bayes. ;
- SO6: To classify mango size based on image data by getting its length and width using OpenCV, geometry, and image processing techniques. ;
- SO7: To classify mango bruises based on image data by employing machine learning algorithms.

1.4.3 Expected Deliverables

Table 1.1 shows the outputs, products, results, achievements, gains, realizations, and/or yields of the Thesis.

TABLE 1.1 EXPECTED DELIVERABLES PER OBJECTIVE

Objectives	**Expected Deliverables**
GO: To develop a user-priority-based grading and sorting system for Carabao mangoes, using machine learning and computer vision techniques to assess ripeness, size, and bruises.	• To develop a Carabao mango grading and sorting system. • To grade Carabao mangoes into three categories based on ripeness, size, and bruises using machine learning. • To integrate sensors and actuators to control the conveyor belt and image acquisition system.
SO1: To make an image acquisition system with a conveyor belt for automatic sorting and grading mangoes.	• To make an image acquisition system with a camera and LED light source. • To build a flat belt conveyor for moving the mangoes.
SO2: To get the precision, recall, F1 score, confusion matrix, and train and test accuracy metrics for classifying the ripeness and bruises with an accuracy score of at least 90%.	• To use a publicly available dataset of at least 10,000 mango images for classification of ripeness and bruises.
SO3: To create a microcontroller-based system to operate the image acquisition system, control the conveyor belt, and process the mango images through machine learning.	• To develop an intuitive UI where users can start and stop the system. • To implement a priority-based grading system with sliders for ripeness, bruises, and size.
SO4: To grade mangoes based on user priorities for size, ripeness, and bruises.	• To utilize a linear combination formula as the overall mango score, where each classification level contributes a grade, weighted by the priority assigned to the three properties. • To assign score values for each classification level of the mango.

Continued on next page

TABLE 1.1 EXPECTED DELIVERABLES PER OBJECTIVE

Objectives	Expected Deliverables
SO5: To classify mango ripeness based on image data using machine learning algorithms such as kNN, k-mean, and Naïve Bayes.	• To train a machine learning model such as kNN, k-means, or Naïve Bayes capable of classifying mango ripeness based on the image color. • To gather a dataset of annotated images with ripeness labels. • To obtain an evaluation report of performance metrics of the model.
SO6: To classify mango size based on image data by getting its length and width using OpenCV, geometry, and image processing techniques.	• To develop an image processing algorithm capable of determining mango size using OpenCV, NumPy, and imutils. • To classify mangoes based on size into small, medium, and large based on measurements.
SO7: To classify mango bruises based on image data by employing machine learning algorithms.	• To train a machine learning model such as capable of distinguishing bruised and non-bruised mangoes. • To train a machine learning model such as kNN, k-means, and Naïve Bayes capable of assessing the extent of bruising on the mangoes if it is significant or partial. • To gather a dataset of annotated images based on bruises. • To obtain an evaluation report of performance metrics of both CNN and other machine learning models.

1.5 Significance of the Study

Automating the process of sorting and grading mangoes increases efficiency and productivity for the user which would in effect remove human error in sorting and grading and decrease the human labor and time taken to sort and grade the mangoes. This is especially important for farmers with a large amount of fruit such as mangoes and a lesser labor force.

A recent study showed that their automated citrus sorter and grader using computer vision can reduce the human labor cost and time to sort and grade when comparing the automated citrus sorter and grader to manual human labor (Chakraborty et al., 2023).

Another benefit to the automation of sorting and grading mangoes is the improvement in quality control. This implies that compared to human labor, automating sorting and grading mangoes can uniformly assess the quality of mangoes based on size, color, and bruises, ensuring that the expected grade and high-quality mangoes reach the consumer. By accurately identifying substandard mangoes, the system helps in reducing waste and ensuring that only marketable fruits are processed further.

Likewise, the scalability of automating sorting and grading mangoes is simpler, especially for lower labor force farmers with large volumes of mangoes. Because of the possibility of large-scale operations by automating sorting and grading mangoes, farmers can now handle large volumes of mangoes, making them suitable for commercial farms and processing plants.

1.5.1 Technical Benefit

1. The development of an automated Carabao mango sorter would increase the quality control of classifying Carabao mango based on ripeness, size, and bruising.
2. The accuracy in sorting Carabao mangoes will be significantly improved while reducing the errors due to human factors in manual sorting.
3. The automated Carabao mango sorter carefully sorts the mangoes while ensuring that they remain free from bruising or further damage during the process

1.5.2 Social Impact

1. The reduction in manual labor creates opportunities in maintenance and technologies in the automated Carabao mango sorter.
2. The automated Carabao mango sorter system improves Carabao mango standards and enhances the satisfaction of the buyers and the customers through guaranteeing consistent Carabao mango grade.
3. Opportunity to increase sales and profit for the farmers through consistent quality and grade Carabao mangoes while reducing the physical labor to sort it.

1.5.3 Environmental Welfare

1. With the utilization of non-destruction methods of classifying Carabao mangoes together with an accurate sorting system, overall waste from Carabao mangoes is reduced and the likelihood of improperly sorted mangoes is decreased.
2. Automation of sorting and grading Carabao mangoes promotes sustainable farming practices.

1.6 Assumptions, Scope, and Delimitations

1.6.1 Assumptions

1. The Carabao mangoes are from the same source together with the same variation
2. The Carabao mangoes do not have any fruit borer and diseases

3. All the components do not have any form of defects
4. The prototype would have access to constant electricity/power source.
5. The Carabao mangoes to be tested would be in the post-harvesting stage and in the grading stage.
6. The image-capturing system would only capture the two sides of the mango which are the two largest surface areas of the skin.

1.6.2 Scope

1. The prototype would be specifically designed to grade and sort Carabao Mangoes based on only ripeness, size, and visible skin bruises.
2. The mangoes used as the subject will be solely sourced from markets in the Philippines.
3. The Carabao mangoes would be graded into three levels.
4. The prototype will be using a microcontroller-based system locally stored on the device itself to handle user interaction.
5. Computer vision algorithms to be used will include image classification.

1.6.3 Delimitations

1. The project would only be able to perform sorting and grading on one specific fruit which is the Carabao mango and will not be able to sort other types of mangoes.

2. Additionally, the project prototype will only be able to capture, sort, and grade one mango subject at a time which means the mangoes have to be placed in the conveyor belt in a single file line for accurate sorting.

3. For the bruises, the system will only be able to detect external bruises and may not identify the non-visible and internal bruises.

4. The system does not load the mangoes onto the conveyor belt itself. Assistance is required to put mangoes into the conveyor belt to start the sorting process

5. The prototype will be powered using Alternating Current (AC) power and will be plugged into a wall socket which is only suitable for indoor use.

Chapter 2

LITERATURE REVIEW

De La Salle University

2.1 Existing Work

Adam et al. (2022) developed a ripeness grader for Carabao mangoes. The Carabao mango ripeness grade calculated based on object and color detection which were written in microcontroller. These are the systems designed by the researchers that consists of Raspberry Pi 4, Arduino Uno, camera, touch screen LCD, MQ3 gas sensor, ventilation system as seen on Figure 2.1 The proposed system was able to ascertain an overall reliability of 95% which means that the specified objective of ascertaining the ripeness level of the mangoes was met with success. However, accuracy and reliability of the software system are there since the hardware design does not seem to be workable when one must deal with the scores of mangoes. In addition, the design of the hardware does not integrate any form of physical automating, say like the conveyor belt. Besides, the hardware system only works efficiently when deciding the ripeness grade of mangoes separately.

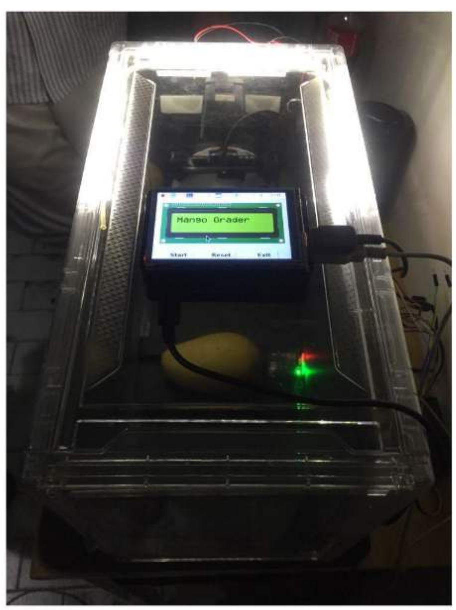


Fig. 2.1 Prototype for Grading Mangoes (Adam et al., 2022)

A study done by Samaniego Jr. et al. (2023) supports and has relevant information concerning the aforementioned topic. They proposed a fully-perovskite photonic system which has the capability to identify and sort or grade mango based on features such as color, weight and, conversely, signs of damages. Some of the techniques in image processing that the researchers used included image enhancement, image deblurring, edge detection using MATLAB and Arduino as well as color image segmentation. Likewise the system block diagram containing these equipment used are seen on Figure 2.2. By carrying out the multiple trials on the device they achieved a classification speed of 8.132 seconds and an accuracy of 91.2%. The proponents' metrics used for the ratings were speed wherein the results were rated "excellent" while the accuracy rating given was "good". One of the limitations of the paper is that the researchers were only limited to the color, texture, and size of the Carabao mango

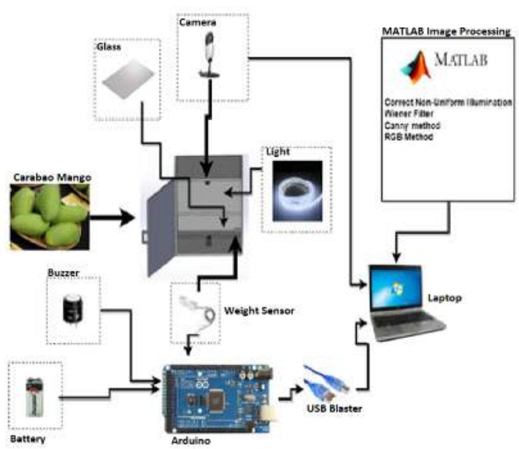


Fig. 2.2 System Block Diagram (Samaniego Jr. et al., 2023)

Furthermore, Guillergan et al. (2024) designed an Automated Carabao mango classifier, in which the mango image database is used to extract the features like weight, size, area along with the ratio of the spots for grading using Naïve Bayes Model. Concerning the quantitative test design, one had to control and experiment with various methods of image processing that would improve the likelihood of improved classification. Their methodology entailed sample collection from 300 Carabao mangoes, picture taking using a DSLR camera, and feature deconstruction for categorization. The system prototype and the software were designed with the programming language C# with integration of Aforge. NET routines. The performance of this model was checked with the help of the dataset containing 250 images, precision, recall, F-score key indicators were used. The investigation discovered that the Naïve Bayes' model recognized large and rejected mangoes with 95% accuracy and the large and small/medium difference with a 7% error, suggesting an application for quality differentiation and sorting in the mango business industry. The limitations they encountered was they were not able to achieve the highest accuracy after using a high quality DSLR camera and the fact that the researchers were not able to incorporate the use of microcontrollers.

Another study by Tomas et al. (2022) proposed an SVM-based system for classifying the maturity stages of bananas, mangoes, and calamansi. With the use of 1729 images of bananas together with 711 mango images and 589 calamansi, the researchers were able to achieve a high accuracy score of above 90% for all fruits. Some pre-processing techniques used to get this high accuracy are the change in hue, saturation, and value channels in the mango image. One of the pre-processing methods (background removal) is shown on Figure 2.3 To better understand the harvest time of mangoes, the paper by Abu et al. (2021) examined the association of the harvest season with seasonal heat units, rainfall, and physical fruit attributes for Haden, Kent, Palmer, and Keitt mango varieties to establish export and domestic market maturity standards. For the results of the paper, it shows that temperature, rainfall, and physical characteristics have a reliable, non-destructive indicators for determining mango maturity. This shows that physical characteristics and temperature are important when exporting fruits such as mangoes.

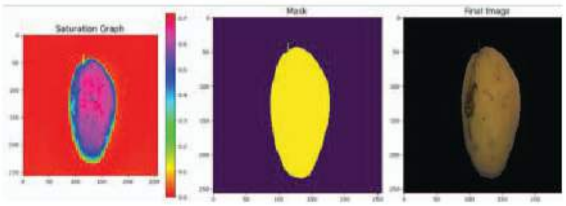


Fig. 2.3 Background Removal of Mango (Tomas et al., 2022)

TABLE 2.1 COMPARISON OF EXISTING STUDIES

Existing Study	**Limitations**	**Accuracy Rating**
Adam et al. (2022)	No physical automation, not suitable for large amounts of mangoes, only classifies ripeness and only a sample size of 10 mangoes.	95%
Samaniego Jr. et al. (2023)	Focuses only on color and size.	91.2%
Guillergan et al. (2024)	Relies on high-quality DSLR cameras, and limited automation due to not integrating microcontrollers.	95%
Supekar and Wakode (2020)	No physical automation implemented. Ripeness, size, and shape-based classification achieved 100%, 98.19%, and 99.20% accuracy respectively on their own. However, errors occurred when taking into account all these aspects together for grading mangoes, causing an accuracy rating deduction.	88.88%

Previous studies on mango grading have achieved an accuracy rating of up to 95%, as shown in Table 2.1. However, these studies either relied on a small sample size, which limits statistical significance, or utilized expensive equipment, which may be impractical. In light of this, the researchers have set a target accuracy rating of greater than or equal to 90%. This target ensures that the system being developed is comparable to, or better than, existing studies that used larger sample sizes or assessed multiple mango traits at the same time. Furthermore, this research aims to distinguish itself by not only maintaining or exceeding the 90% accuracy rating but also incorporating a graphical user interface (GUI) for selective priority-based mango classification. The system will integrate both software and hardware components, and it will evaluate a greater number of mango traits for grading purposes.

2.1.1 Deep Learning Classification Algorithms

Researchers have implemented various artificial intelligence algorithms in order to determine the optimal and most effective method for sorting mangoes. One of the algorithms that was used in the classification of mangoes was the CNN or Convolutional Neural Networks. A study done by Zheng and Huang (2021) explored the effectiveness of CNN, specifically in classifying mangoes through image processing. The system that the researchers developed graded mangoes into four groups which was based on the Chinese National Standard. These mangoes were examined by their shape, color uniformity, and external defects. The system that was developed had an impressive accuracy of 97.37% in correctly classifying the mangoes into these grading categories Support Vector Machine was also one of the classification algorithms that was implemented to detect flaws in mangoes. In that study by Veling (2019), SVM was used in the classification of diseases from mangoes. The study used 4 different diseases/defects for testing. The diseases were Anthracnose, Powdery Mildew, Black Banded, and Red Rust. and provided 90% accuracy for both the leaves and the fruit

In the study done by Schulze et al. (2015), Simple Linear Regression, Multiple Linear

Regression, and Artificial Neural Network models were all studied and compared for the purpose of size-mass estimation for mango fruits. The researchers found that the Artificial Neural Network yielded a high accuracy rating for mass estimation and for mango classification based on size with a success rate of 96.7%. This is attributed to the Artificial Neural Network model's ability to learn both linear and nonlinear relationships between the inputs and the outputs. However, a problem can occur with the use of the model, which is overfitting. This issue occurs when the model is overtrained with the data set such that it will start to recognize unnecessary details such as image noise which results in poor generalization when fed with new data. With this in mind, additional steps will be necessary to mitigate the issue. Another research article written by Alejandro et al. (2018) implements a method for sorting and grading Carabao mangoes. This research focuses on the use of Probabilistic Neural Network, which is another algorithm that is used for pattern recognition and classification of objects. For this study, the researchers focused on the area, color, and the black spots of the mango for their Probabilistic Neural Network model. Their research using the model yielded an accuracy rating of 87.5% for classification of the mangoes which means it is quite accurate for classifying mangoes within the predefined categories. However, problems were encountered with the use of the model when trying to identify mangoes that did not fit the predefined size categories of small, medium, and large. This means that the PNN model may become challenged when presented with a mango with outlying traits or traits that were very different from the data set.

TABLE 2.2 COMPARISON OF SORTING ALGORITHM MODELS

Sorting Algorithm Model	Accuracy Rating	Criteria	Problems Encountered
Convolution Neural Network	97.37%	shape, color, defects	Minor blemishes affected the accuracy.
Support Vector Machine	90%	mango defects and diseases	The model is sensitive to noise, which requires intensive image preprocessing.
Artificial Neural Network	96.7%	for mango size and mass	Overfitting
Probabilistic Neural Network	87.5%	for mango area, color, and black spots	Difficulty in identifying mangoes that have outlying features or did not fit the predefined categories

2.2 Lacking in the Approaches

The majority of past researchers such as Amna et al. (2023) and Guillermo et al. (2019) were able to implement a fruit and mango sorter together with an accurate AI algorithm to detect the ripeness defects. This means that none of the previous research papers were able to integrate an interchangeable user-priority-based grading together with size, ripeness, and bruises using machine learning for Carabao mango sorter and grader. Our research however would implement an automated Carabao mango sorter in terms of size, ripeness, and bruises with its own UI, conveyor belt, DC motors, and bins for collecting the different ripeness and defect grade of the Carabao mango.

2.3 Summary

To reiterate, there is an innovative gap that needs to be filled with regards to the process of sorting and grading Carabao mangoes. The traditional methods for conducting this process manually by hand, by a porous ruler, by a sugar meter, and by a color palette can be prone to human error and expensive costs due to the number of laborers required to do the task. On the other hand, although researchers have already taken steps to automate the process of mango sorting and grading, there is still a need for an implementation that takes into account size, ripeness, and bruises altogether whilst being non-destructive with its own user-priority-based grading and sorting and having its own embedded system. The research articles shown above show the different computer vision and CNN approaches for sorting and classifying mangoes. For example, a system created by Adam et al. (2022) was more focused on ripeness detection. Samaniego Jr. et al. (2023) considered photonic systems for grading mango fruit based on color and weight. On the other hand, Guillermo et al. (2019) implemented the Naïve Bayes classification model on mangoes with high accuracy, which thereby did not include any microcontroller. There was an attempt to study each of those parameters separately and that is why the multifactorial approach was not used. With this in mind, the system being proposed does exactly what was mentioned, to implement a non-destructive and automated sorting and grading system for Carabao mangoes that takes into account size, ripeness, and bruises altogether using machine learning, as well as having its own embedded system. This system will be mainly composed of a conveyor belt, servo motors, a camera, microcontrollers, and an LCD display for the user interface. By doing so, the system should be able to improve the efficiency and productivity of mango sorting and grading, remove the effect of human error and reduce time consumption. The studies also provided critical insights regarding the effective algorithms that can be used in classification stages in image processing. The use of CNN had the most accuracy with manageable potential challenges. Lastly, by scaling the implementation, the overall export quality of the Carabao mangoes can be improved.

Chapter 3

THEORETICAL CONSIDERATIONS

3.1 Introduction

Likewise, the purpose of this chapter is to go through the important theories in developing the prototype together with training and testing the machine learning model.

3.2 Relevant Theories and Models

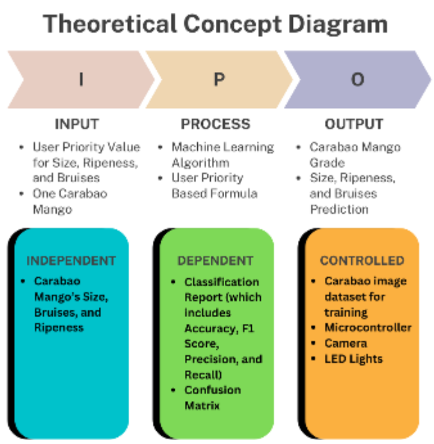


Fig. 3.1 Theoretical Framework Diagram.

The theoretical framework seen in figure 3.1 follows the IPO (Input-Process-Output) Model for a Carabao Mango Sorting System. The Input section includes user-defined priority values for size, ripeness, and bruises, along with a single mango for analysis. The Process section highlights the use of a machine learning algorithm and a user-priority-based formula to classify the mango. The Output consists of the mango's grade, predicted size, ripeness, and bruises. Below the IPO model, the diagram categorizes variables into three groups: Independent (mango's size, ripeness, and bruises), Dependent (classification report with accuracy, precision, recall, and confusion matrix), and Controlled (image dataset, microcontroller, camera, and Light Emitting Diode (LED) lights).

De La Salle University

3.3 Technical Background

At its core, the system will be using machine learning concepts pertaining to Convolutional Neural Network (CNN) and OpenCV, and may use other algorithms such as Naive Bayes and k-Nearest Neighbors (KNN) to supplement the classification tasks, particularly for assessing mango ripeness, bruise detection, and size determination. The system will be built on an embedded framework, integrating a Raspberry Pi microcontroller to control the Raspberry Pi camera, actuators, LED lights, and motors. A user-friendly GUI will also be utilized to ensure users can customize the prioritization of the mango sorting system.

3.4 Conceptual Framework Background

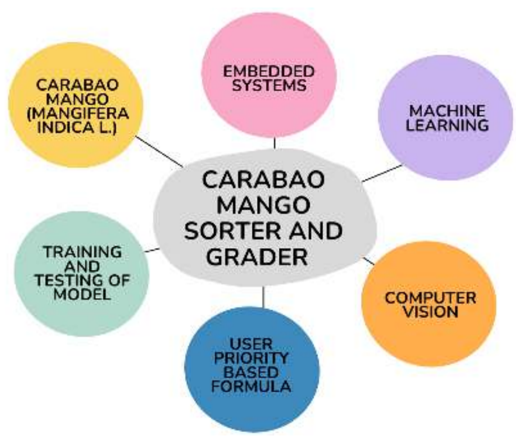


Fig. 3.2 Conceptual Framework Diagram.

The conceptual framework seen in figure 3.2 illustrates the key components involved in the Carabao Mango Sorter and Grader system. At the center, the system is represented as the core element, surrounded by six interconnected components: Carabao Mango (Mangifera indica L.), Embedded Systems, Machine Learning, Computer Vision, User

Priority-Based Formula, and Training and Testing of the Model. These elements represent the different technologies, methodologies, and considerations required for the development and operation of the sorter and grader. The diagram provides an overview of how various disciplines contribute to the project's functionality.

3.5 Software Concepts

3.5.1 Thresholding

Thresholding is a computer vision image segmentation technique that is used to separate objects from their surroundings by converting a grayscale image to binary. The conversion is done by choosing a certain threshold intensity value. It is usually done by assigning pixels with an intensity higher than the threshold are mapped to one value (commonly white), and pixels with an intensity lower than the threshold are mapped to another (commonly black). The result of this technique is in a high-contrast image that makes it easy to detect the object's boundary and shape in the image.

In this project, two types of thresholding were applied:

- Absolute Difference Thresholding – This method involves computing the absolute difference between two images. The first image is one of the object, and the other of the same background without the object. The result isolates only the pixels that have changed between the two images, thus isolating the mango from its background successfully.

- Binary Thresholding – Once the difference image has been created, binary thresholding is used. A threshold value is employed to threshold the difference image into a binary image. Values greater than the threshold are made white (foreground), and values less than that are made black (background). This creates a clear silhouette of the mango, which is appropriate for size estimation and contour detection.

3.5.2 Object Size Calculation

Object size calculation is the calculation of a certain object's true size from image data. This is essential in computer vision systems to efficiently process object features in real-time. In this research, the size of the Carabao mango is estimated through image measurement techniques based on geometric principles and camera calibration.

$$\text{Real World Dimension} = \frac{\text{Pixel Dimension} \times \text{Distance from Camera to Object}}{\text{Focal Length}} \tag{3.1}$$

$$D\left(p, d, f\right) = \frac{p \cdot d}{f} \tag{3.2}$$

where $D\left(p, d, f\right)$ is the real world dimension of the object, p is the pixel dimension of the object, d is the distance from the camera to the object, and f is the focal length of the camera. This relationship follows from the pinhole camera model, where the real-world dimension is proportional to the image dimension and the ratio of distance to focal length Badali et al. (2005).

After capture and preprocessing of the image, the binary image so obtained is processed with contour detection to find the largest object, which is assumed to be the mango. The contour is then bounded with a minimum-area bounding box, and pixel-based length and width are calculated using Euclidean distance between the corner points.

This size estimation method offers a consistent and efficient way of taking the measurements with only standard camera input, providing consistency in classification and reducing the necessity for physical measuring devices.

3.5.3 Convolutional Neural Network

Convolutional Neural Networks are a class of deep learning models is commonly used in analyzing visual data. CNNs are particularly effective in image classification tasks due to their ability to automatically extract and effectively learn the spatial hierarchies of features directly from the pixels of a given image. This makes it highly suitable for functions such as object detection and, in the case of this study, image classification.

CNN usually applies filters to input images. These filters are designed to detect local patterns such as edges, textures, and color gradients. The network is able to learn more patterns as the data goes through the layers. This enables it to recognize effectively the characteristics that it is looking for.

The use of CNNs in this study allows for accurate, automated classification of mango images which contributes to the development of a reliable, non-destructive grading system that minimizes human error and ensures consistent quality assessment

3.6 Hardware Concepts

3.6.1 Camera Module

The camera module serves as the main image acquisition tool in the mango sorter and grader system. Its role is to capture clear, high-resolution images of each mango as it moves along the conveyor. These images are critical for analyzing physical traits like ripeness, bruising, and size through computer vision and machine learning techniques.

The camera is directly connected to the Raspberry Pi, which manages both image capture and processing. It is fixed in position to ensure consistent distance and angle for all images. It is also paired with a lighting system to provide a consistent lighting for the images. The system captures images of both the top and bottom sides of each mango to ensure a more accurate grading. The prototype integrates the Raspberry Pi Camera Module Version 2. This camera is chosen for its 8MP resolution which is critical in capturing real-time images. Another reason for integrating this camera is because of its compatibility with the Raspberry Pi 4, and reliability in capturing detailed images needed for accurate classification. It is also cost effective and lightweight which is important for the prototype.

3.6.2 4 Channel Relay

The relay module in this project is used to control the direction and movement of the motors that operate the conveyor system and mango sorting mechanism. As an electrically operated switch, the relay allows the low-power signals from the Raspberry Pi to safely manage the higher voltage and current required by the DC motors.

For the prototype, the relay module is responsible for changing the polarity of motor connections which enables the motors to rotate in both forward and reverse directions. This will drive the conveyor belt system. This is essential for moving mangoes along the conveyor, rotating them for the top and bottom image capture, and directing them to the appropriate bin based on their grade.

De La Salle University

3.6.3 Gear Ratio

In this prototype, gear ratios are used to control the rotational speed of the conveyor belts that move and rotate the mango. A gear ratio of 1:3 was applied, meaning the motor gear completes one full rotation for every three rotations of the driven gear. This is also done in order to avoid overspeeding and make sure that the conveyor belt moves in a controlled manner. This setup slows down one belt relative to the other, creating a differential speed between the left and right belts. As a result, the mango rotates in place while being moved forward. This rotation is essential for capturing both the top and bottom views of the mango for accurate classification and grading.

3.7 Summary

Overall, chapter 3 establishes key concepts and theoretical considerations that form the foundation of the Carabao mango sorter and grading system. It discusses and connects each component together, explaining how each component such as the RaspberryPi and DC motors work together to create a system that utilizes machine learning and computer vision techniques to classify mangoes based on user priority.

Chapter 4

DESIGN CONSIDERATIONS

4.1 Introduction

Likewise, the objective of chapter 4 is to describe the researcher's design consideration when developing and testing the prototype. For an overview of the design of the prototype, the researchers considered different computer vision models in classifying the ripeness and bruises together with other algorithms to determine the size of the mango. Likewise, the hardware design was also taken into consideration where the physical design of the conveyor belt was taken into account.

4.2 Engineering Standards

4.2.1 Electrical Certifications

The UL Listed certification indicates that the Raspberry Pi power supply has been tested and approved by Underwriters Laboratories (UL), meeting safety standards for both the United States and Canada under certification number E330985. This certification ensures that the power supply complies with established requirements for electrical safety, insulation, and protection against potential fire hazards. It also carries an Efficiency Level VI rating, which represents the highest energy efficiency standard set by the U.S. Department of Energy (DOE) for external power supplies, ensuring minimal energy loss and optimized performance.

4.2.2 Safety of Machinery

The ISO 13850:2015 – Safety of Machinery (Emergency Stop Function, Principles for Design) standard defines the safety requirements for emergency stop functions in machinery.

It specifies that emergency stop devices must be clearly visible, easily accessible, and capable of quickly and safely halting machine operations in the event of a malfunction or hazard. For the prototype, the stop button is located at the bench power supply and the RPi.

4.2.3 Safety Requirements for Technology Equipment

The IEC 62368-1:2018 / ISO 62368-1:2018 – Safety Requirements for Audio/Video, Information, and Communication Technology Equipment standard establishes international safety guidelines for modern electronic devices and their power supplies. It replaces older standards (IEC 60065 and 60950) with a hazard-based safety engineering approach, ensuring that equipment in the prototype like the RPi power supply and bench power supply are designed to prevent electrical shock, overheating, and fire risks.

4.2.4 Open-source Software Compliance

The ISO/IEC 5230e - Open-source Software Compliance ensures that organizations using open-source components in their products maintain proper documentation, license traceability, and transparency in software management. For components in the prototype like the RPi operating system, which rely on open-source ecosystems, compliance with this standard promotes responsible use and distribution of software, reducing legal and security risks associated with open-source code.

4.3 System Architecture

The system architecture is represented through a block diagram, showcasing modules such as image acquisition, preprocessing, feature extraction, machine learning model, and grading output. Each module is described in detail, emphasizing its role in the overall system. For instance, the image acquisition module uses high-resolution cameras to capture mango images, while the preprocessing module enhances image quality for better feature extraction.

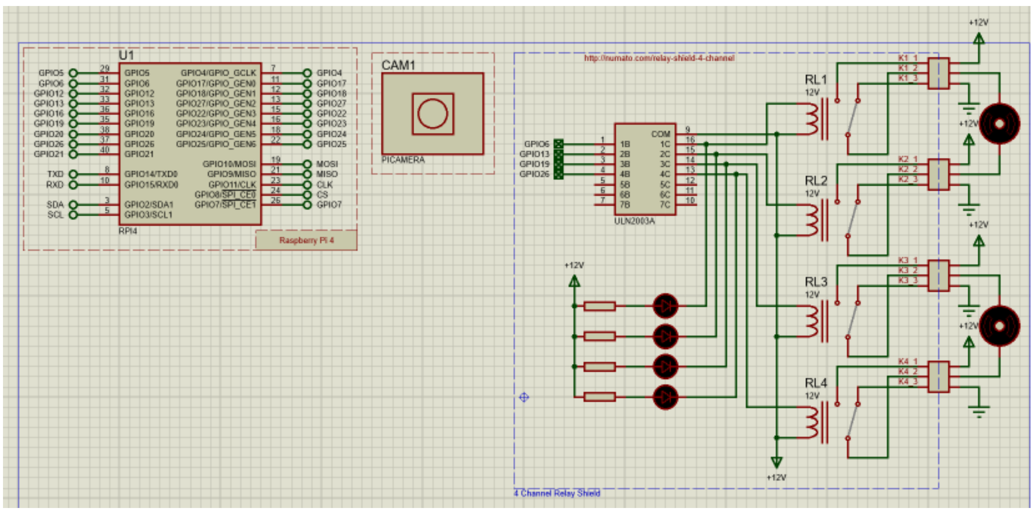


Fig. 4.1 Hardware Schematic

In figure 4.1 presents the electronic circuit diagram, designed using Proteus. The diagram illustrates a system where a Raspberry Pi 4 serves as the central control unit, managing four motors through a relay mechanism. The Raspberry Pi 4, represented by a rectangular box on the left, showcases various pin connections, including GPIO pins, power supply pins (5V and 3V3), ground pins (GND), and communication pins (TXD, RXD, SDA, SCL).

In the center of the diagram, an 18-pin integrated circuit labeled ”ULN2803A” is depicted. This component, a Darlington transistor array, likely functions as a buffer, providing the necessary current to drive the relays. Four relays, designated as RL1, RL2, RL3, and RL4, are positioned on the right side of the diagram, each connected to a motor (represented by a circle with an "M" inside) and a +12V power source. Additionally, four resistors are placed between the ULN2803A and the relays, serving to limit current. The circuit section containing these resistors is labeled "4 Channel Relay Driver," indicating its purpose.

The camera module is labeled "PICAMERA" is located in the top center of the diagram. It is represented by a square with a circle inside, symbolizing the camera lens. The camera module is connected to the Raspberry Pi 4 through the CSI (Camera Serial Interface) pins. The overall circuit is designed for a 12V system, with the +12V power supply indicated at various points. The Raspberry Pi 4's GPIO pins are used to control the relays.

4.4 Hardware Considerations

The hardware components include high-resolution cameras, lighting systems for consistent image capture, and microcontrollers like Raspberry Pi or Arduino for system control, actuators like DC motors to move the mangoes. The choice of hardware is justified based on cost, performance, and compatibility with the software framework.

4.4.1 General Prototype Framework

The Figure 4.2 presents the overall prototype layout of the automated Carabao mango sorter and grader. The diagram illustrates the flow of operations from mango loading onto the conveyor belt to sorting them. It illustrates the major elements of the system, that is, the image acquisition area, lighting system, camera module, Raspberry Pi controller, and mechanical actuators. The layout illustrates how all the subsystems work together to ensure mangoes are scanned, processed, sorted based on ripeness, size, and bruises, and eventually

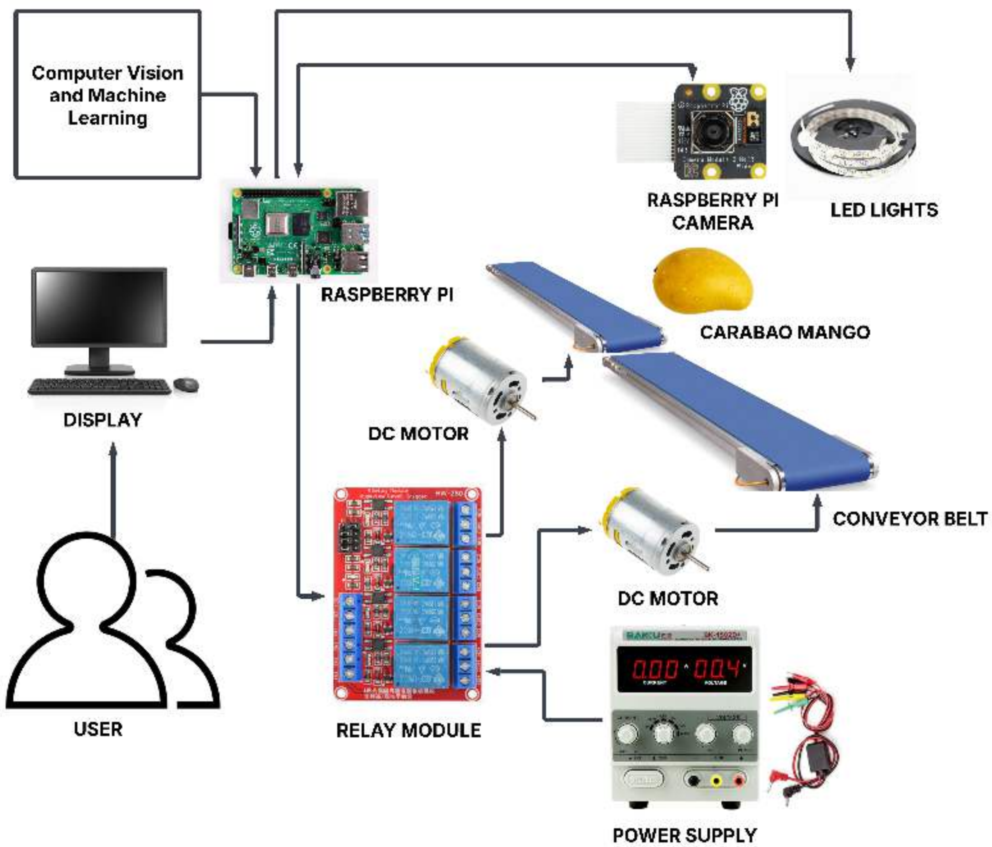


Fig. 4.2 Prototype Framework

945 sorted based on the calculated priority score. The layout served as the basis for actual prototype development.

4.4.2 Prototype Flowchart

The flowchart in Figure 4.3 represents the overall operational logic of the mango grading and sorting system. The process starts with system initialization, where the camera and lighting modules are switched on and the machine learning algorithms are initialised. The

De La Salle University

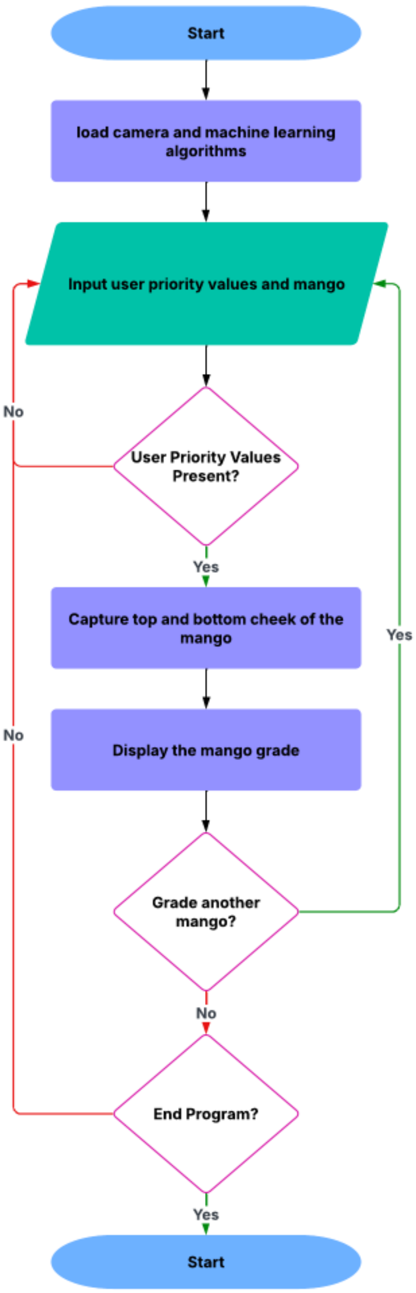


Fig. 4.3 Prototype Main Flowchart

input of the user priority values as well as the detection of the mango on the conveyor belt triggers the capture of both the top and bottom cheek of the mango. The captured image is processed using machine learning algorithms to determine its ripeness, size, and bruises. Depending on these classifications along with priority weights given by the user, the system calculates an overall score. Once this calculation is done, the mango is routed to the respective bin through the respective actuator. Having this logical sequence is important to know the system's decision-making and automation process.

4.4.3 Prototype 3D Model

Figure 4.4 shows the first 3D model of the initial physical prototype developed for the sorting and grading system. This model shows the skeleton of the system and where the conveyor system is going to be placed strategically in order to flip the mango for image acquisition. It is useful for where the hardware components would be arranged and assembled. This 3D model helped the researchers visualize the spacing, alignment, and where to mount parts before assembling the prototype making sure all electronic and mechanical components are effectively integrated.

4.4.4 Hardware Specifications

4.4.4.1 Raspberry Pi

The Raspberry Pi 4 Model B serves as the central processing unit of the prototype, chosen for its compact form factor, affordability, and substantial computational capability required for image processing and machine learning tasks. The board's essential features include GPIO pins for connecting sensors, actuators, and relays, along with USB and HDMI ports

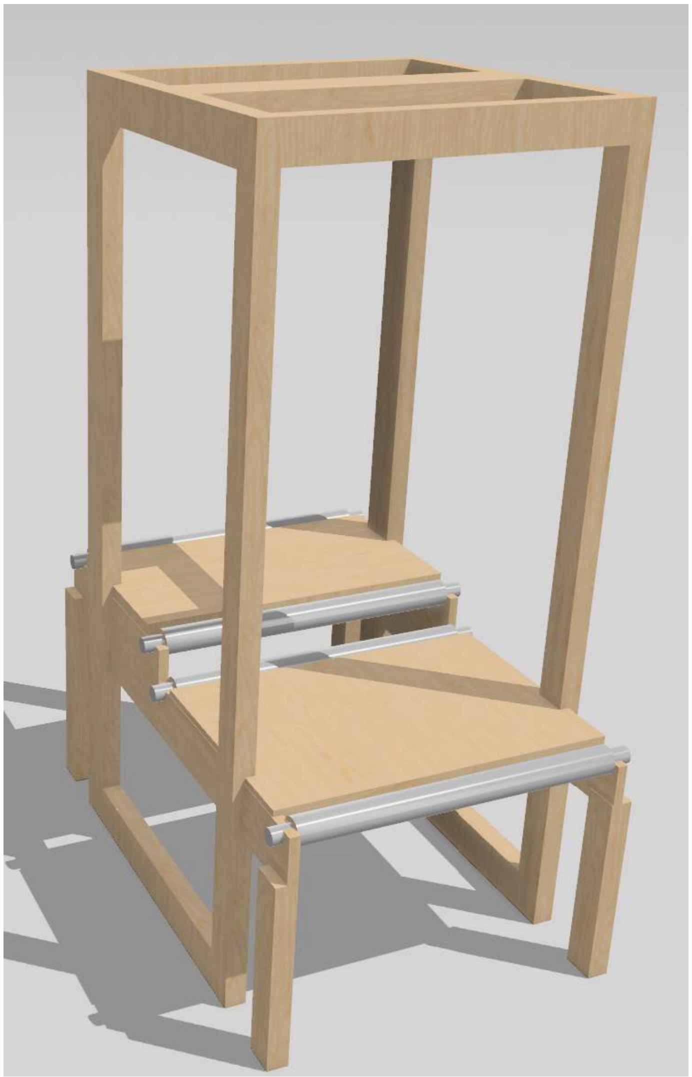


Fig. 4.4 Initial 3D Model of the Prototype

Fig. 4.5 Raspberry Pi 4 Model B

for peripheral integration. Its support for a full operating system enables it to efficiently manage both the user interface and the core control logic of the mango grading system.

Specifications:

- SoC: Broadcom BCM2711
- Central Processing Unit (CPU): Quad-core ARM Cortex-A72 (64-bit)
- Clock Speed: 1.5 GHz (base, overclockable)
- RAM: 8GB LPDDR4-3200 SDRAM
- Wireless: Dual-band 2.4 GHz / 5 GHz Wi-Fi (802.11ac)
- Bluetooth: Bluetooth 5.0 (BLE support)
- Ethernet: Gigabit Ethernet (full throughput)

- USB: 2 x USB 3.0 ports and 2 x USB 2.0 ports
- Video Output: 2 x micro-HDMI ports (supports 4K @ 60Hz, dual 4K display capability)
- Audio: 3.5mm audio/video composite jack
- Storage: MicroSD card slot (supports booting via SD card or USB)
- GPIO: 40-pin GPIO header (backward-compatible with older models)
- Camera/Display: CSI (camera) and DSI (display) ports
- Power Input: USB-C (5V/3A recommended)
- Power Consumption: 3W idle, up to 7.5W under load

4.4.4.2 Raspberry Pi Camera

This high-quality camera module is specifically engineered for the Raspberry Pi platform, offering 8-megapixel still image capture and video recording capabilities at 1080p (30fps), 720p (60fps), and 480p (90fps). It incorporates a fixed-focus lens with a 62.2-degree diagonal field of view and a 1/4-inch optical format. Compatibility with Python libraries like Picamera and OpenCV facilitates seamless image capture and processing. Its selection was driven by its small size, straightforward integration, and capacity for high-resolution imaging.

Specifications:

- Sensor: Sony IMX219PQ 8-megapixel CMOS sensor.
- Still Images Resolution: 8 MP (3280 x 2464 pixels).

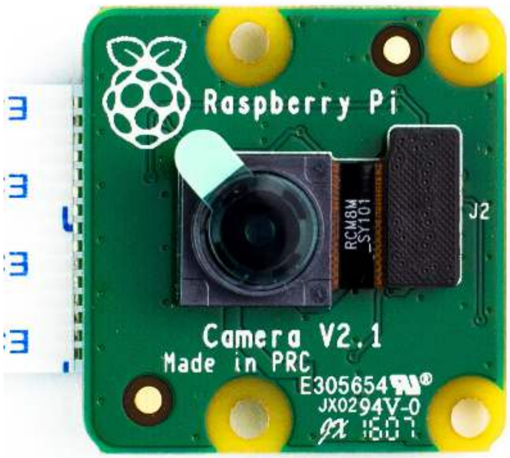


Fig. 4.6 Raspberry Pi Camera Module Version 2

- Video Resolution: Supports up to 1080p @ 30fps, 720p @ 60fps, and 480p @ 90fps.
- Focus: Fixed-focus lens (manual focus adjustment not supported without physical modification).
- Lens Size: 1/4-inch optical format.
- Field of View (FoV): Diagonal 62.2 degrees.
- Interface: Connected via 15-pin ribbon cable to the Raspberry Pi's CSI (Camera Serial Interface) port.
- APIs/Libraries: Supports Python libraries such as Picamera and OpenCV for image capture and processing.
- Dimensions: 25 mm x 24 mm x 9 mm.

4.4.4.3 DC Motor

Fig. 4.7 12 Volt DC Gear Motor

This compact 12V DC gear motor delivers high torque and operates quietly, making it suitable for robotics, automation, and industrial control systems. Its spur gear design ensures a high reduction ratio for enhanced torque. Engineered for continuous duty, it maintains low power consumption during standard operation and offers reliability under high-temperature conditions.

Specifications:

- Gearbox Type: Spur gear design
- Operating Voltage: 12V (operational range: 6-12V)
- No-load Current Consumption: 0.8A
- Rated Current Draw: 3A (under standard load)

- No-load Speed: 282 RPM (maximum)
- Operating Speed: 248 RPM (under rated load)
- Torque Output: 18 kg-cm (rated)
- Stall Torque: 60 kg-cm (maximum)
- Power Rating: 50W (maximum)
- Unit Weight: 350 grams

4.4.4.4 MicroSD Card

Fig. 4.8 SanDisk Ultra MicroSD Card

This compact, high-capacity SanDisk Ultra MicroSD card provides secure digital storage for devices like digital cameras, smartphones, and tablets. Its high-speed data transfer rate is optimal for handling large files such as images and videos. The card was chosen for the prototype's storage system due to its substantial capacity, reliable data protection, and user-friendly design.

Specifications:

- Capacity: 256GB
- Type: MicroSDXC (Secure Digital eXtended Capacity)
- Form Factor: MicroSD (11mm x 15mm x 1mm)
- File System: Pre-formatted exFAT

4.4.4.5 LED Lights

Fig. 4.9 LED Light Strip

The LED strips were implemented to deliver uniform illumination for image capture, which is crucial for precise color representation and feature extraction. Their selection was based on exceptional energy efficiency, extended operational lifespan, and consistent light output quality.

Specifications:

- Power Input: 5V DC (USB-powered, compatible with laptops, power banks, or USB adapters).
- Waterproof Design: Suitable for indoor/outdoor use.
- LED Type: SMD 2835 (surface-mount diodes for high brightness and efficiency).
- Color Type: White (cool white)
- Length: 1m
- Beam Angle: 120°
- Operating Temperature: -25°C to 60°C.
- Storage Temperature: -40°C to 80°C.

4.4.4.6 Power Supply

This bench power supply is an adaptable and regulated source that delivers stable voltage and current for diverse electronic projects. Designed for testing purposes, it enables precise setting of voltage and current parameters. Its versatility, user-friendly operation, and accurate control capabilities led to its selection.

Specifications:

- Type: SMPS (Switch-Mode Power Supply)

Fig. 4.10 Bench Power Supply

- Input: 110V AC, 50/60Hz (U.S. Standard)
- Output Range: 0-30V DC / 0-5A DC
- Voltage Precision: ±0.010V (10 mV) resolution
- Current Precision: ±0.001A (1 mA) resolution
- Power Precision: ±0.1W resolution
- Weight: 5 lbs (2.27 kg)
- Dimensions: 11.1” x 4.92” x 6.14” (28.2 cm x 12.5 cm x 15.6 cm)
- Maximum Power: 195W
- Power Source: AC input only

4.4.4.7 4 Channel Relay Module

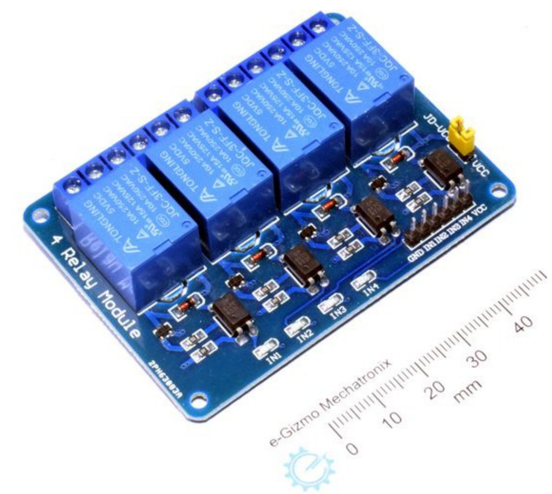


Fig. 4.11 4 Channel Relay Module

This compact and versatile relay board enables control of multiple devices through a single microcontroller. It was chosen for its small footprint, operational simplicity, and capacity to manage several devices concurrently. Designed for compatibility with microcontrollers like Arduino and Raspberry Pi, it integrates smoothly into the prototype.

Specifications:

- Operating Voltage: 5V DC (compatible with Arduino, Raspberry Pi, and other microcontrollers).
- Number of Relays: 4 independent channels.
- Relay Type: Electromechanical (mechanical switching).
- Max AC Load: 10A @ 250V AC (resistive).

- Max DC Load: 10A @ 30V DC (resistive).
- Contact Type: SPDT (Single Pole Double Throw) - NO (Normally Open), NC (Normally Closed), COM (Common).
- Dimensions: 50mm x 70mm x 20mm
- Weight: 50-80 grams.
- Status LEDs: Individual LEDs for each relay (indicates ON/OFF state).
- Input Pins: 4 digital control pins (one per relay).
- Output Terminals: Screw terminals for connecting loads (NO/NC/COM).

4.4.4.8 RPi Power Supply

Fig. 4.12 Power Supply for the RPi

This official Raspberry Pi power supply is optimally designed for the Raspberry Pi 4 Model B, compatible with all its memory variants. Delivering 5.1V at 3A via a USB-C connector, it ensures reliable performance. The OKdo-branded unit provides stable power suitable for the Raspberry Pi 4, other single-board computers, and mobile devices, and includes comprehensive over-temperature protection with global plug compatibility.

Features:

- Compatible with Raspberry Pi 4 Model B
- Color: Black
- USB-C connector
- US Plug
- Over temperature protection
- Short circuit protection
- Over current protection
- Over voltage protection

Specifications:

- Input Voltage: 100-264V AC
- Input Frequency Range: 47-63Hz
- Input Current: 600mA Max
- Output Voltage: 5.1V DC
- Output Current: 3A
- Power Rating: 15.3W

- Output Connector: USB Type C
- Output Cable Length: 1.5M
- Number of Outputs: 1
- Unload Standby Power: 0.1W
- Max Ripple Noise: 50-240mVp-p

4.4.4.9 Mini Conveyor Single Narrow

Fig. 4.13 Single Narrow Mini Conveyor

This miniature conveyor system facilitates the creation of compact factory setups for presentations and prototyping. The single narrow configuration is particularly suited for small-scale automation tasks and experimental applications.

Specifications:

- Belt Dimensions: 43.4 x 9 x 9 cm (L x W x H)
- Chassis Dimensions: 46 x 10.5 x 11 cm (L x W x H)

- Type: Single narrow conveyor
- Application: Prototyping and miniature factory setups

4.4.4.10 Mini Conveyor Double Narrow

Fig. 4.14 Double Narrow Mini Conveyor

This miniature conveyor system enables the development of small-scale factory environments for demonstrations and prototyping. The double narrow version offers increased length to accommodate more sophisticated automation processes and continuous operation requirements.

Specifications:

- Belt Dimensions: 85.5 x 9 x 9 cm (L x W x H)
- Chassis Dimensions: 88 x 10.5 x 11 cm (L x W x H)
- Type: Double narrow conveyor
- Application: Extended prototyping and miniature factory setups

4.5 Software Considerations

The software stack includes Python for programming PyTorch for machine learning and OpenCV for image processing. These tools are selected for their robustness, ease of use, and extensive community support, ensuring efficient system development.

4.5.1 PyTorch

PyTorch is an open-source deep-learning framework used in this project for implementing and running the convolutional neural networks responsible for classifying mango ripeness and detecting bruises. Its dynamic computational graph and GPU acceleration support made it an ideal choice for real-time image classification. Its simplicity and flexibility also allowed for easy integration with the Raspberry Pi which is important as it is the main processing unit for the system.

4.5.2 OpenCV

Open Source Computer Vision Library or OpenCV is utilized in the system for all image processing tasks, particularly in preprocessing steps such as background subtraction, thresholding, edge detection, and contour analysis. These operations are essential for calculating the real-world dimensions of the mango. OpenCV was utilized primarily because of its diverse set of functions, performance optimization, and ease of use making it a core tool for enabling accurate and fast computer vision processing within the prototype.

4.5.3 CustomTkinter

CustomTkinter is a modern alternative to the standard Tkinter library, and is used to build the graphical user interface (GUI) of the system. It provides a more polished and customizable visual appearance while retaining the simplicity of Tkinter. With features such as styled buttons, frames, and labels, CustomTkinter allowed for the creation of a user-friendly interface that supports real-time display of classification results, priority scoring inputs, and system status updates.

4.6 User Interface

A User Interface (UI) is designed to display grading results, system status. Wireframes illustrate the layout, ensuring usability and accessibility for operators. Likewise, a Graphical User Interface (GUI) is also used to allow users to customize the system's grading priorities.

4.7 Summary

This chapter outlines the foundational design and engineering decisions for the automated mango grading system. The design process prioritized creating a scalable, efficient, and user-friendly system, guided by established engineering standards for safety and compliance. These standards include UL Listing for the power supply, ISO 13850 for the emergency stop function, and IEC 62368-1 for the safety of the technology equipment.

The system architecture is built around a RPi 4 Model B as the central controller, which manages a network of hardware components. The core hardware includes a RPi Camera for image acquisition, 12V DC gear motors to drive the conveyor belts, a 4-channel relay module for motor control, and LED strips to ensure consistent lighting. A detailed hardware schematic and a 3D model were created to plan the integration of these electronic and mechanical parts effectively.

On the software side, the system leverages a robust stack including PyTorch for running the deep learning models, OpenCV for image processing tasks like size determination, and CustomTkinter to build an intuitive GUI. This GUI allows operators to input grading priorities and view results. The overall operational logic, from mango detection and image capture to classification and sorting, is defined by a clear system flowchart. In summary, this chapter details the careful selection and integration of both hardware and software components to form a coherent, safe, and functional prototype.

Chapter 5

METHODOLOGY

De La Salle University

TABLE 5.1 SUMMARY OF METHODS FOR REACHING THE OBJECTIVES

Objectives	**Methods**	**Locations**
GO: To develop a user-priority-based grading and sorting system for Carabao mangoes, using machine learning and computer vision techniques to assess ripeness, size, and bruises.	1. Hardware design: Build an image acquisition system with a conveyor belt, LED lights, and Raspberry Pi Camera 2. Software design: Coded a Raspberry Pi application to grade and sort the Carabao mangoes	Sec. 5.2 on p. 60
SO1: To make an image acquisition system with a conveyor belt for automatic sorting and grading mangoes.	1. Hardware implementation: Design and build an image acquisition system prototype	Sec. 5.3 on p. 60
SO2: To get the precision, recall, F1 score, confusion matrix, and train and test accuracy metrics for classifying the ripeness and bruises with an accuracy score of at least 90%.	1. Performance testing: Train and test the machine learning algorithm for classifying bruises and ripeness 2. Data collection: Gather our own Carabao mango dataset together with an online dataset	Sec. 5.5 on p. 70

Continued on next page

Continued from previous page

Objectives	Methods	Locations
SO3: To create a microcontroller-based system to operate the image acquisition system, control the conveyor belt, and process the mango images through machine learning.	1. Algorithm development: To develop a code for the image acquisition system 2. Hardware design: To design a schematic for the microcontroller based system	Sec. 5.3 on p. 60
SO4: To grade mangoes based on user priorities for size, ripeness, and bruises.	1. Formula development: Formulated an equation based on the inputted user priority and the predicted mango classification	Sec. 5.7 on p. 90
SO5: To classify mango ripeness based on image data using machine learning algorithms such as kNN, k-mean, and Naïve Bayes.	1. Performance testing: Train and test the machine learning algorithm for classifying bruises	Sec. 5.6.6 on p. 84
SO6: To classify mango size based on image data by getting its length and width using OpenCV, geometry, and image processing techniques.	1. Performance testing: Train and test the machine learning algorithm for classifying ripeness	Sec. 5.6.5 on p. 82
SO7: To classify mango bruises based on image data by employing machine learning algorithms.	1. Accuracy testing: Get the percent accuracy testing for getting the length and width of the Carabao mango	Sec. 5.6.7 on p. 86

5.1 Introduction

The methodology for this research outlines the development of the Carabao Mango sorter using machine learning and computer vision. The sorting system uses a conveyor belt system which delivers the mangoes into the image acquisition system. This system captures the image of the mangoes which will then be going through the various stages of image processing and classification into grades which will depend on the priority of the user. This methodology ensures that the grading of the mangoes will be accurate while being non-destructive.

5.2 Research Approach

This study applies the experimental approach for research in order to develop and properly test the proposed system. The experimental approach of the methodology will allow the researchers to fine-tune the parameters and other factors in the classification of mangoes in order to get optimal results with high accuracy scores while maintaining the quality of the mangoes. This approach will also allow for real-time data processing and classification which will improve the previous static grading systems. To efficiently design and build the prototype, the researchers employed a Scrum agile methodology for managing the two main clusters of the prototype which are the software and hardware design.

5.3 Hardware Design

The prototype consists of hardware and software components for automated mango sorting and grading purposes. The hardware includes the conveyor belt system used to transfer mangoes from scanning to sorting smoothly. A camera and lighting system are able to collect high-resolution images for analysis. The DC motors and stepper motors are responsible for driving the conveyor belt and sorting actuators. The entire system is controlled by a microcontroller RPi, coordinating actions of all components. Sorting actuators then direct mangoes into selected bins based on their classification to make sorting efficient.

5.3.1 Mango Position

In the image acquisition system, the mango is always positioned above the camera and parallel to the metallic rollers and gap. This is so that the size classification would be consistent for both image capturing attempts. Once the mango has already been graded, the mango would exit the image acquisition system parallel to the metallic rollers and parallel to the long conveyor belt. In the case that the mango would go towards the small conveyor belt, it would be perpendicular to the small conveyor belt.

Figure 5.1 shows the position of the mango from the image acquisition system which are the mangoes labeled 1 and 2. When the mango is already graded, it would be sorted using the T sorter seen on mangoes 3, 3.1, and 3.2.

5.4 Software Design

For the programming language used for the prototype and training and testing the CNN model, Python was used for training and testing the CNN model and it was also used in the microcontroller to run the application containing the UI and CNN model. PyTorch was the main library used in using the EfficientNet model that is used in classifying the ripeness

De La Salle University

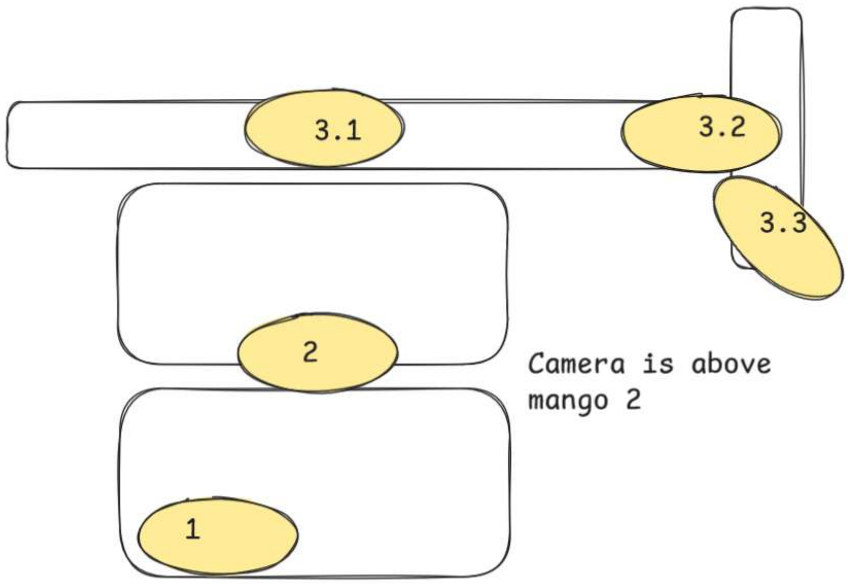
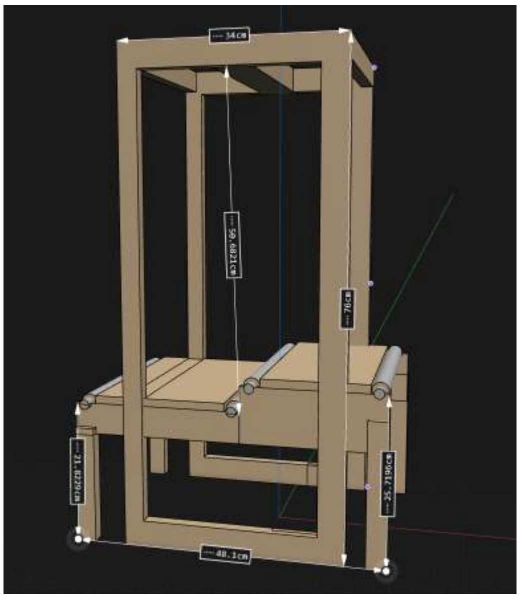


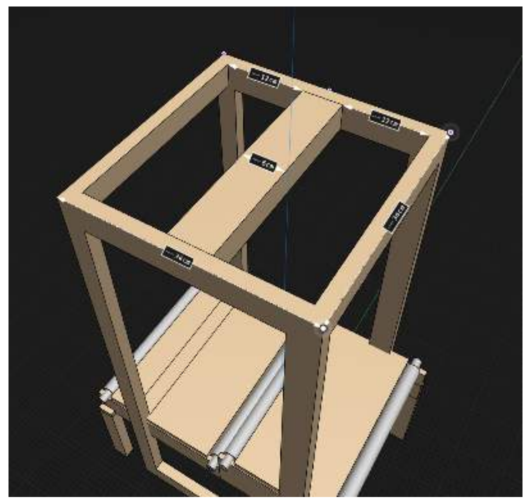
Fig. 5.1 conveyor positioning

and bruises of the mango. Likewise, tkinter is the used library when designing the UI in Python.

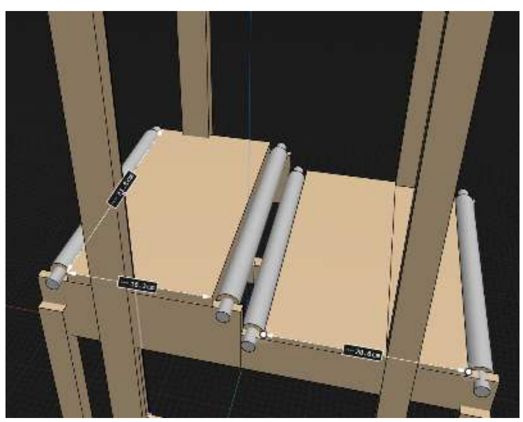
Furthermore, the rest of the software components are of utmost importance to mango classification. Image processing algorithms in OpenCV and CNN models extract features such as color, size, and bruises that are known to determine quality parameters of mangoes. Mangoes are classified based on ripeness and defects by using machine learning algorithms, which further enhances accuracy using deep learning techniques. A user interface (UI) is designed for users to control and observe the system in real time. Finally, the interface programming of the microcontroller provides the necessary synchronization between sensors, actuators, and motors throughout the sorting operation scenario.



(a) Left View



(b) Top View



(c) Right View

Fig. 5.2 Image Acquisition Dimensions

5.4.1 Machine Learning Methods

The processed dataset is to be then used to create models using a variety of machine learning methods. For a comprehensive evaluation, the processed dataset was used to train and test a variety of machine learning models. The training included Convolutional Neural Network (CNN), k-Nearest Neighbors (k-NN), Naive Bayes, and k-Mean clustering and various Efficientnet models. This comparative analysis was conducted to benchmark the performance of the deep learning approach against traditional machine learning algorithms.

5.4.2 Optimizer

Choosing the correct optimizer critically impacts both the convergence speed and the generalization ability of deep neural networks. The widely used Adam optimizer employs adaptive learning rates for each parameter, adjusting them according to the first- and second-order moments of gradients. However, Adam implements weight decay as a part of gradient updates, which couples regularization and optimization in a way that can hamper generalization. AdamW was developed to decouple weight decay from the adaptive gradient update. Specifically, in AdamW, weight decay is applied directly to the parameters after the Adam update, leading to improved generalization and often more robust performance in large-scale tasks. Extensive benchmark comparisons reveal that AdamW outperforms standard Adam, especially when it comes to image classification or language modeling tasks with deep architectures (Loshchilov and Hutter, 2017).

5.4.3 Data Loading Optimization

Efficient data loading is a vital but often underestimated aspect of deep learning. In frameworks like PyTorch, the num_workers parameter of the DataLoader determines how many subprocesses are used to fetch batches of data in parallel. Setting num_workers >0 enables multiprocessing, which prefetches batches and keeps the GPU occupied without idling, especially for large datasets or CPU-intensive augmentations. When misconfigured, however, the CPU can become a bottleneck, or resource contention may lead to unexpected slowdowns. The ideal number of workers depends on many factors: CPU and memory resources, dataset I/O demands, and the complexity of any required preprocessing. Practically, practitioners start with a low value for num_workers, gradually increasing while monitoring CPU utilization and GPU occupancy, always balancing throughput gains against system constraints (Migacz, 2020).

5.4.4 Data Transfer Optimization

Data transfer from host (CPU) to device (GPU) is a significant performance consideration during training, particularly as model and batch sizes grow. PyTorch and similar frameworks provide pin_memory and non_blocking options to optimize these transfers. When data is loaded with pin_memory=True, it is allocated in page-locked (pinned) memory, which prevents the operating system from swapping it to disk and enables direct memory access (DMA) from the GPU, reducing latency. Setting non_blocking=True in transfer calls further allows these memory copies to be overlapped with computation, eliminating host-thread blocking and enabling concurrent initiation of multiple transfers. Together, these settings can cut data transfer times and better exploit GPU concurrency. However, misuse, such as excessive pinned memory allocation, can reduce overall system stability due to increased physical memory pressure (Moens, 2024).

5.4.5 Mixed Precision Training

Mixed precision training is now a near-standard approach for accelerating deep learning, especially on modern GPUs equipped with specialized compute units, such as NVIDIA Tensor Cores, that can handle reduced numerical precision efficiently. By employing 16-bit floating point (FP16 or BF16) arithmetic for most operations and retaining 32-bit (FP32) precision for critical accumulations and weight updates, mixed precision training achieves two main benefits: faster computation throughput and decreased memory footprint. This allows for increased model or batch sizes and faster experimentation cycles, while, with proper loss scaling, preserving model convergence and final accuracy (Markidis and et al., 2018).

5.4.6 Adaptive learning Rate Schedulers

Adaptive learning rate schedules can profoundly affect both convergence speed and the ability of a model to generalize. The cosine annealing schedule cyclically adjusts the learning rate from a maximum to a minimum according to a cosine function, periodically "restarting" back to the initial value. This warm restart strategy prevents the learning rate from decaying to zero too rapidly and encourages exploration of flatter minima in the loss surface, thereby enhancing generalization. Cosine annealing with restarts is widely cited as a simple but effective modification over static or monotonic decay schedules, giving superior performance across various deep learning domains from computer vision to language modeling (Loshchilov and Hutter, 2016).

5.4.7 CrossEntropy Loss with Label Smoothing

Using CrossEntropy loss with label smoothing addressed the issue of overconfidence in predictions. Standard CrossEntropy encourages the model to assign near-absolute probability to the correct class, which can lead to poor generalization, especially when classes are ambiguous or noisy. Label smoothing redistributes a small fraction of probability mass to incorrect classes, effectively softening the target distribution. This discourages the model from becoming overly confident, reduces variance in predictions, and improves robustness against mislabeled or borderline samples (Guo and et al., 2024; Szegedy et al., 2016)

5.4.8 Early Stopping and Checkpointing

Overfitting is a major concern in deep learning, as models with high capacity can easily memorize the training data without learning to generalize to new inputs. Early stopping is a widespread technique wherein training is halted when performance on a held-out validation set ceases to improve, rather than after a fixed number of epochs. This prevents the model from entering the overfitting regime. Model checkpointing complements early stopping by routinely saving the model's parameters and, optionally, optimizer states, ensuring recoverability in the event of hardware failure and enabling the best-performing model on validation metrics to be retained, rather than simply the last epoch's snapshot (Hussein and Shareef, 2024; Lee et al., 2024).

De La Salle University

5.4.9 Input Resolution

The spatial resolution of input images materially affects both computational cost and prediction accuracy in deep learning, especially for vision tasks. Higher input resolutions can theoretically yield better performance, as more visual detail is made available to the model, but this often comes at the expense of increased memory and higher training times, sometimes forcing smaller batch sizes and less efficient optimization. Conversely, reducing input resolution can dramatically decrease resource requirements, permitting faster development and larger batch sizes, but at a potential loss of accuracy, especially for tasks that demand fine-grained spatial detail (Richter et al., 2020).

5.4.10 Regularization

Regularization techniques combat overfitting, and two of the most prominent in deep learning are dropout and drop path, also called “stochastic depth”. Dropout randomly deactivates a subset of neurons or weights during each training iteration, preventing any single unit from becoming indispensable and encouraging redundancy in representation. Drop path extends this principle by stochastically skipping entire layers or blocks during training, particularly in architectures with skip connections such as ResNet. This approach reduces the effective depth of the model during training while maintaining full depth at inference, acting as an implicit model ensemble and further strengthening generalization (Huang et al., 2016).

5.5 Data Collection Methods

For data collection, publicly available datasets were used along with our own gathered dataset. to gather the images of the mangoes the setup seen in Figure 5.3 was used to film the mangoes for about 5 seconds each side. Using a python script every 20th frame per second was extracted. The collected images were then sorted into the following directories for use in training the model: non-bruised, bruised, green, yellow-green, and yellow.

Fig. 5.3 Camera Setup

For the setup of the captured Carabao mangoes, the height of the camera to the white flat surface is 26 cm which can be seen on Figure 5.3. Furthermore, the Samsung S24's camera is used for capturing both cheeks of the Carabao mango. Initially, the Carabao mangoes would be unripe and green and each day the Carabao mangoes would be pictured until they are yellow ripe. Likewise, Figure 5.4 shows the 8 kilogram green Carabao mangoes from the Bicol region. The same mangoes from Bicol are seen on the Figure 5.4. Note that the mangoes were individually captured one at a time at both cheek sides as a video format

(a) Boxes of Carabao Mangoes (b) Table of Carabao Mangoes

Fig. 5.4 Carabao Mangoes Image Dataset Collection

which can be seen on Figure 5.5.

For the farm one of our members went to interview the head farmer (Jerry Bravante) as seen on Figure 5.6, it is located at Ibaan, Batangas. He as 50 years of experience being a farmer and 20 years of experience in quality standards of different mango fruits variations such as Carabao, Pico, Indian, and Apple. Additionally, the farm has a total of 4 hectares.

5.6 Testing and Evaluation Methods

In a bid to ensure the mango sorting and grading system is accurate and reliable, there is intensive testing conducted at different levels. Unit testing is initially conducted on each

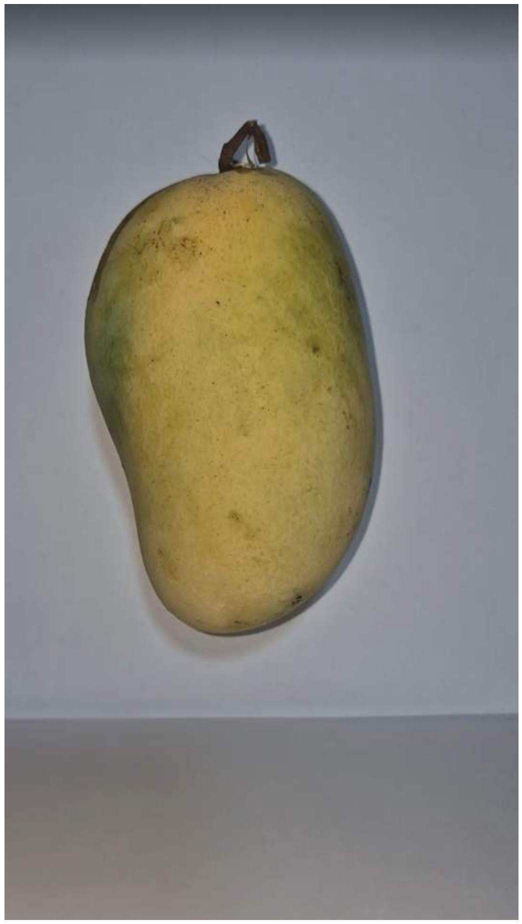


Fig. 5.5 Sample Mango Image

component separately, for instance, the conveyor belt, sensors, and cameras, to ensure that each of the components works as expected when operating separately. After component testing on an individual basis, integration testing is conducted to ensure communication between hardware and software is correct to ensure the image processing system, motors, and sorting actuators work in concert as required. System testing is conducted to conduct overall system performance testing in real-world conditions to ensure mangoes are accurately and efficiently sorted and graded.

De La Salle University

(a) Collecting Carabao Mangoes

(b) Carabao Mango Tree

(c) Sack of Carabao Mangoes

Fig. 5.6 Collecting Mango on a Farm

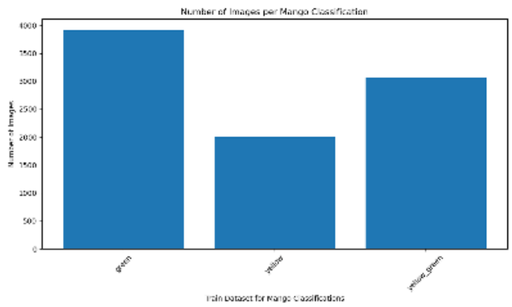
For the training, everything was done on a laptop, specifically the Acer Predator Helios 16 (PH16-71, 2023 model). The technical specifications of this unit are: Intel Core i9-13900HX processor, NVIDIA RTX 4070 GPU with 8GB VRAM, and 32GB DDR5 RAM running at 5600MHz.

5.6.1 Data Augmentation and Splitting

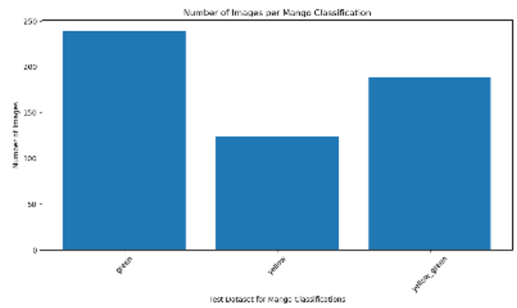
For the used methods to increase the Carabao mango image dataset, data augmentation techniques such as rotation, flipping, Gaussian blur, brightness adjustment, noise, crop, and resizing of the images were done. Note that the split ratio of the dataset is 70-15-15 where it refers to the training, testing, and validation as seen on the Listing 5.1.

The dataset for mango classification was organized into five categories: bruised, not bruised, green, yellow-green, and yellow. To ensure robust model training and evaluation, the dataset was initially split into training (70%), validation (15%), and test (15%) sets using PyTorch's automated splitting functions. Following standard practice in deep learning (Perez, Wang, 2017), only the training set was augmented to increase sample diversity and improve generalization, while the validation and test sets remained unaltered to preserve their role as unbiased evaluation benchmarks.

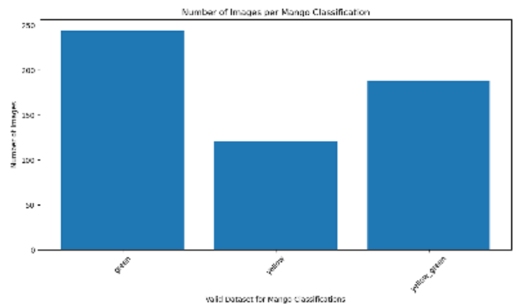
The validation set contains a balanced representation of the five mango classes. In the bruise-based categories shown on Table 5.7, the distribution shows slightly more bruised samples ($\sim$260) compared to not bruised ($\sim$240). On the other hand, for the ripeness-based categories as shown in Tables 5.8, green has the highest count ($\sim$250), followed by yellow-green ($\sim$175), and yellow ($\sim$125). This distribution ensures that the validation set provides a fair assessment of the model's performance across both damage-related and ripeness-related classifications.



(a) Training Dataset



(b) Test Dataset

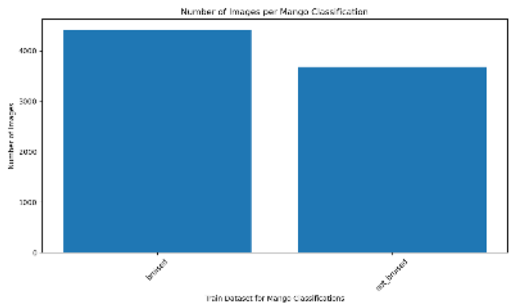


(c) Validation Dataset

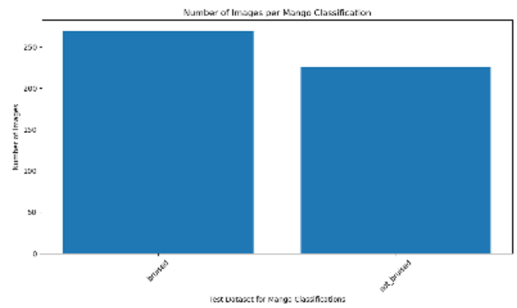
Fig. 5.7 CNN Ripeness 70-15-15 Image Datasplit

The test set mirrors the validation set in structure, maintaining proportional representation across classes. Approximately 260 bruised samples and (∼240) not bruised samples are included as seen in Figure 5.7. For the ripeness categories seen in Table 5.8, green (∼225), yellow-green (∼175), and yellow (∼125) are represented. This balanced distribution allows for reliable final evaluation of the trained CNN model, ensuring that results are not biased toward any single class.

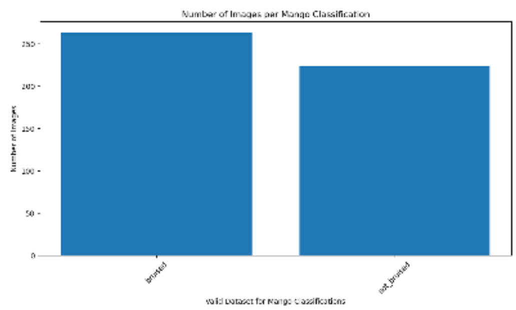
The training set underwent augmentation to artificially expand the dataset and introduce variability. Augmentation techniques included transformations such as rotation, flipping, scaling, and brightness adjustments. After augmentation, the dataset contained approximately 5,100 bruised and 4,900 not bruised samples as seen in Table 5.8. For the



(a) Training Dataset



(b) Test Dataset



(c) Validation Dataset

Fig. 5.8 CNN Bruises 70-15-15 Image Datasplit

ripeness categories, green had the highest representation (~4,200), followed by yellow-green (~3,400), and yellow (~2,600) as seen in Table 5.7. This augmentation step increased the training set size substantially, shifting the dataset distribution from 70-15-15 to approximately 90-5-5. Such a shift is expected, as augmentation only affects the training set, thereby increasing its relative proportion.

Augmenting only the training set is a widely accepted best practice in deep learning. According to Shorten and Khoshgoftaar (2019), data augmentation enhances model robustness by simulating real-world variability, but applying it to validation or test sets would artificially inflate accuracy by exposing the model to transformed versions of already-seen data. Similarly, Goodfellow et al. (2016) emphasize that evaluation datasets must remain unseen, original, and unaltered to provide a true measure of generalization. Perez and Wang (2017) further demonstrate that augmentation is most effective when applied exclusively to training data, as it improves performance without compromising the integrity of evaluation.

The dataset preparation process therefore ensures that the CNN model is trained on a large, diverse, and augmented training set, while validation and test sets remain unaltered and representative. This methodology aligns with established best practices in computer vision research, supporting both robust training and fair evaluation of the mango classification model

5.6.2 Comparative Test of CNN Models

To identify the most suitable CNN architecture for grading Carabao mangoes, multiple CNN models were evaluated under fixed experimental parameters. Each model was trained for 15 epochs with an input image size of 224×224 pixels, a batch size of 32, and the Adam optimizer set at a learning rate of 0.001. Data preprocessing included resizing, normalization using ImageNet mean and standard deviation, and augmentation techniques such as random horizontal and vertical flips, random rotations, and Gaussian blur, which were applied exclusively to the training set. The validation and test sets remained unaugmented to ensure unbiased evaluation.

The performance of several CNN architectures, including EfficientNetV1, EfficientNetV2, Visual Geometry Group Network (VGGNet), AlexNet, ResNet50, GoogleNet, MobileNetV2, and DenseNet121 was first compared. Based on these results, a more detailed comparison was then conducted within the EfficientNet family, versions V1 and V2, to determine the most effective variant for the task.

No advanced optimization techniques such as early stopping, learning rate schedulers, or mixed precision training were employed. This decision was intentional to maintain fairness across all experiments and to ensure that the only variable factor influencing performance was the network architecture itself. Ripeness classification models were trained using a Graphics Processing Unit (GPU), while bruise classification models were trained on a CPU to compare training times and assess the impact of hardware constraints on accuracy. Model performance was evaluated using precision, recall, F1-score, accuracy, resource utilization, and elapsed training time.

5.6.3 Benchmarking Best CNN Model on +10k Mango Dataset

As one of the improvements for the final CNN models, the dataset for mango classification was refined and expanded to improve model robustness and reliability across both ripeness and bruise detection tasks for the training of the final CNN model where EfficientNetV2-B3 was used. The data was initially split into training (70%), validation (15%), and test (15%) sets, with augmentation applied only to the training set. However, after augmentation, the effective distribution shifted to 90% training, 5% validation, and 5% test. In addition, new Carabao mango images were incorporated across all classes to strengthen representation and improve generalization. As such, to train the final CNN models, the training set for ripeness category in Table 5.10b contained 4,900 images of green mangoes, 3,700 images of yellow mangoes, and 5,000 images of yellow_green mangoes. For validation in Table 5.10c, the set included 200 green mango images, 175 yellow mango images, and 210 yellow_green images. The test set in Table 5.10a consisted of 200 green mango images, 160 yellow mango images, and 220 yellow_green images. For the bruises category, the training set Table 5.9b contained 6,000 images of bruised mangoes and 7,000 images of not_bruised mangoes after augmentation. The validation set in Table 5.9c included 200

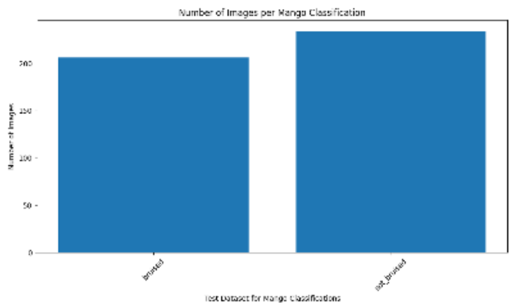
De La Salle University

bruised mango images and 225 not_bruised mango images, while the test set as seen in Table 5.9a contained 200 bruised mango images and 225 not_bruised mango images. This setup provided a balanced evaluation framework for the binary classification task, ensuring that both classes were consistently represented across training, validation, and testing.

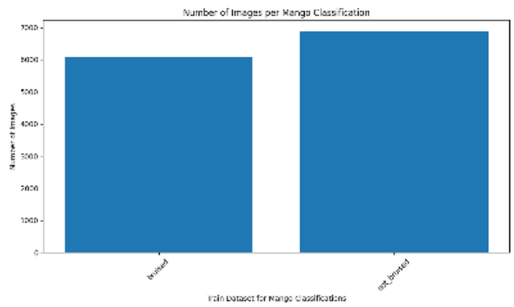
The dataset was also cleaned to remove sources of noise and ambiguity. Images with mixed ripeness features, such as mangoes with both large yellow and green portions, were placed under yellow_green instead, while ambiguous samples, such as yellow mangoes with residual greenish portions, were excluded to avoid confusing the model. Empty areas present in images were also removed to ensure that only the fruit itself was used for training.

Augmentation strategies were further refined to preserve class-defining features. For bruise classification, Gaussian blur was removed since it obscured critical bruise details. For ripeness classification, brightness and contrast adjustments were excluded, as these could shift mango colors between adjacent classes, such as yellow_green to yellow, introducing artificial mislabels. Other augmentations, such as rotation, flipping, scaling, and minor perspective transform, were retained to maintain variability without compromising class integrity.

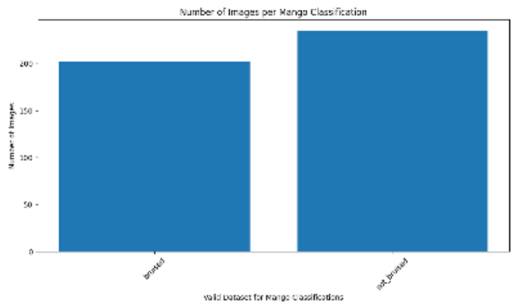
Through these improvements, expanded augmentation, inclusion of new Carabao mango samples, dataset cleaning, and task-specific augmentation refinements, the final dataset ensured that both CNN models were trained on high-quality, representative, and diverse data. This preparation supports fair evaluation on the validation and test sets while maximizing the models' ability to generalize to real-world mango classification scenarios.



(a) Test Image Size



(b) Train Image Size

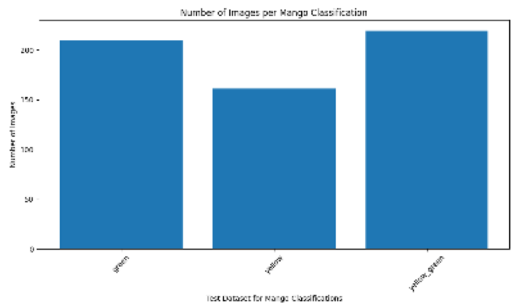


(c) Validation Image Size

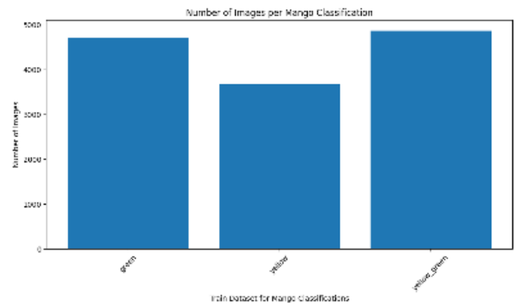
Fig. 5.9 Bruises Image Datasplit

5.6.4 Classification Report

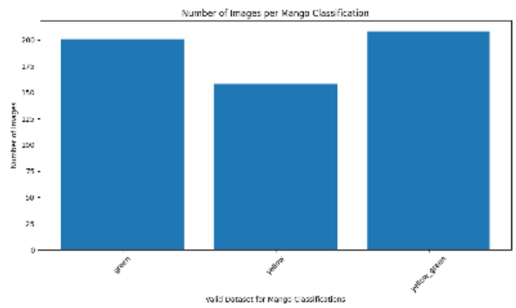
The classification report provides a detailed summary of the model's performance across all output classes by presenting key evaluation metrics such as precision, recall, F1-score, and support. Precision measures the accuracy of positive predictions, recall assesses the model's ability to identify all relevant instances, and the F1-score represents their harmonic mean, offering a balanced measure of performance. In this system, the classification report was used to evaluate how effectively the CNN models identified each mango category—both in ripeness and bruise detection. By analyzing these metrics, the report helps determine which class predictions are most accurate and where the model may require further improvement, ensuring a reliable and interpretable performance assessment for real-world



(a) Test Image Size



(b) Train Image Size



(c) Validation Image Size

Fig. 5.10 Ripeness Image Datasplit

mango classification.

5.6.4.1 Confusion Matrix

A confusion matrix is a table that visualizes the performance of a classification model. For a binary classification problem, it has four components:

	Predicted Positive	**Predicted Negative**
Actual Positive	TP	FN
Actual Negative	FP	TN

TABLE 5.2 CONFUSION MATRIX EXAMPLE

- True Positives (TP): Cases correctly predicted as positive
- True Negatives (TN): Cases correctly predicted as negative
- False Positives (FP): Cases incorrectly predicted as positive. (Type I error)
- False Negatives (FN): Cases incorrectly predicted as negative (Type II error)

5.6.4.2 Precision

$$\text{Precision} = \frac{TP}{TP + FP} \tag{5.1}$$

Precision measures how many of the predicted positives are actually positive. It answers the question: ”When the model predicts the positive class, how often is it correct?” High precision means low false positives.

5.6.4.3 Recall

$$\text{Recall} = \frac{TP}{TP + FN} \tag{5.2}$$

Recall, which is also called sensitivity, measures how many of the actual positives were correctly identified. It answers the question: ”Of all the actual positive cases, how many did the model catch?” High recall means low false negatives.

5.6.4.4 F1 Score

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5.3}$$

The F1 score is the harmonic mean of precision and recall. It provides a single metric that balances both concerns. This is particularly useful when you need to find a balance between precision and recall, as optimizing for one often decreases the other.

De La Salle University

5.6.4.5 Accuracy

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.4}$$

Accuracy measures the proportion of correct predictions (both true positives and true negatives) among the total cases. While intuitive, accuracy can be misleading with imbalanced datasets.

To test system performance, various measures of performance are used to evaluate. As seen on equation 5.4, accuracy score is used to measure the percentage of correctly classified mangoes to ensure the system maintains high precision levels. Precision as seen on equation 5.1 and recall as seen on equation 5.2 are used to measure consistency of classification to determine if the system classifies different ripeness levels and defects correctly. Furthermore, the F1 score formula as seen on equation 5.3 is used to evaluate the performance of the model's classification.

A confusion matrix is used to measure correct and incorrect classification to ensure the machine learning model is optimized and that minimum errors are achieved. Throughput analysis is also used to determine the rate and efficiency of sorting to ensure that the system maintains high capacity without bottlenecks to sort mangoes. Using these methods of testing, the system is constantly optimized to ensure high-quality and reliable mango classification.

5.6.5 Ripeness Training and Testing

For the testing of the ripeness classification, the Carabao mangoes are classified into three ripeness stages which are Green, green yellow, and yellow. Likewise, The green would represent the underripe mangoes while the green yellow would represent the semi ripe while the yellow would represent the ripe mangoes. In other words green is underripe, yellow is ripe, and yellow green is semi ripe mangoes. As reference, Figure 5.11 shows the different ripeness stages for Carabao/Pico mangoes Bureau of Agriculture and Fisheries Product Standards (2004).

Annex A

Stages of ripeness of 'carabao' and 'pico' mango fruits

Stage of ripeness	Peel color	Flesh color
Green	Completely light green	Yellowish white or light yellow green
Breaker	Traces of yellow	Middle area and fruit outline yellowish; other areas, white to yellowish white
Turning	More green than yellow	More yellow than white
Semi-ripe	More yellow than green	Yellow for 'carabao'; yellow orange for 'pico'
Ripe	80-100% yellow ('carabao') or yellow orange ('pico')	Middle area yellow for 'carabao'; yellow orange for 'pico'
Overripe	Yellow for 'carabao'; yellow orange for'pico'	100% yellow for 'carabao' and yellow orange for 'pico'

Fig. 5.11 Carabao Mango Ripeness Stages (Bureau of Agriculture and Fisheries Product Standards, 2004)

5.6.5.1 Green

The first classification the researchers selected is the Green stage where the mango's skin and cheek color is completely light green with no traces of yellow.

5.6.5.2 Yellow_Green

The second classification is the Yellow_Green or Green_Yellow. The main characteristics of this is that it follows the breaker, turning, and semi-ripe stage of the carabao mango. This means that if there is a trace of yellow and green on the skin and cheek of the mango then it is classified as Yellow_Green or Green_Yellow.

5.6.5.3 Yellow

The third and last classification is the Yellow stage where the mango is 80% to 100% yellow on the skin and cheek of the mango. Note that if the mango is overripe then it would be classified to be Yellow for ripeness.

5.6.6 Bruises Training and Testing

For the testing of the bruise classification of the Carabao mangoes, it would classified into two categories which are bruised and not bruised. To define what bruise and not bruise mangoes looked like Figure 5.12 is used as reference to categorize which mangoes are bruised and not bruised. This means that if the mango has any of these features are shown on the mango then it is considered as bruised.

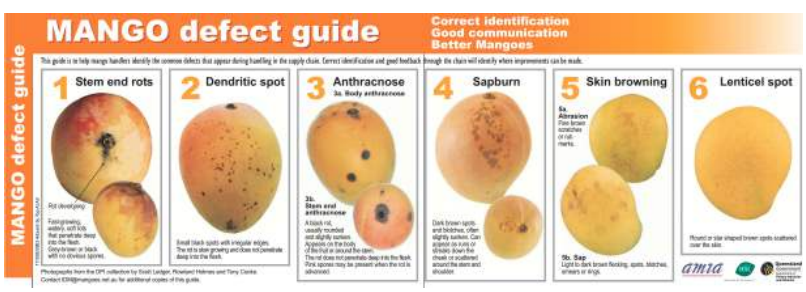


Fig. 5.12 Different Kinds of Mango Defects (Scott Ledger and Cooke, 2000)

5.6.6.1 Stem End Rots

They are characterized by fast-growing, watery, soft rots that penetrate deeply into the flesh. Likewise, they usually appear as grey-brown or black rots starting from the stem end, often without obvious spores, that can spread rapidly into the mango (de Souza-Pollo and de Goes, 2009; Kadam et al., 2002).

5.6.6.2 Dendritic Spot

They are small black spots with irregular edges scattered across the skin. Furthermore, they grow slowly and do not penetrate into the flesh, remaining largely superficial (Ltd, 2007).

5.6.6.3 Anthracnose

It appears in two forms. First form is body anthracnose. Body anthracnose presents as black rots on the fruit surface that are usually round, slightly sunken, and located on different parts of the mango. Likewise, the second form is stem end anthracnose, occurring around the stem and also presenting as black rots. While these rots do not penetrate deeply into the flesh, advanced cases may show pink spores (de Souza-Pollo and de Goes, 2009; Kadam et al., 2002).

5.6.6.4 Sapburn

They appear as dark brown spots or blotches that are often slightly sunken. Likewise, damage can occur as runs or streaks down the cheek or as scattered marks around the stem and shoulder, resulting from sap exposure (Paul, 1993).

5.6.6.5 Skin Browning

It may take two forms. The first form is abrasion while the second form is sap browning. Abrasion is recognized as fine brown scratches or rub marks, while sap-related browning appears as light to dark brown flecking, spots, blotches, smears, or rings. These types of browning are generally limited to the skin and do not penetrate deeply (Paul, 1993).

5.6.6.6 Lenticel Spot

They are another common defect, appearing as round or star-shaped brown spots scattered across the skin surface. Furthermore, these defects are usually cosmetic in nature and do not significantly affect the flesh (Nguyen, 2015).

5.6.7 Size Determination

To get the size of the mango, computer vision techniques such as Gaussian Blur and Thresholding are used to get the length and width of the mangoes. Refer to Figure 5.13 for the location of the length and width of mango.

5.6.7.1 Determining the Ranges for Mango Sizes Based on Area

A total of 42 Carabao mangoes, 27 from Batch 1 and 15 from Batch 2, were collected to serve as the dataset for size classification. Each mango will be manually measured using a caliper to obtain its length and width, ensuring consistent and accurate dimensional data. These measurements will then be used to compute the approximate area, which will serve as the primary feature for analysis. All recorded values will be compiled and converted into CSV format, allowing them to be used as a structured dataset for further

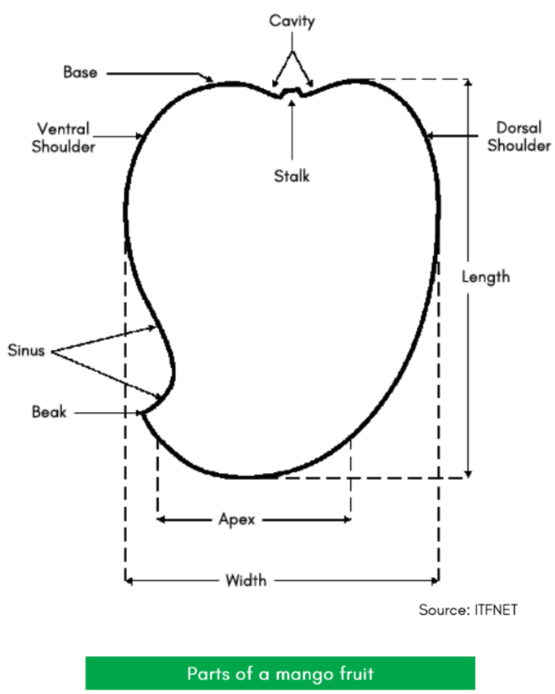


Fig. 5.13 Length and Width of Mango (Bureau of Agriculture and Fisheries Product Standards, 2006)

statistical processing. The dataset will then be analyzed using two methods namely K-means clustering, an unsupervised technique that will be applied to identify natural groupings in the area values, and a quartile-based classifier, which will categorize mangoes based on their statistical distribution. Both approaches will be applied to determine the ideal ranges for the size categories, small, medium, and large.

5.6.7.2 Estimating the Carabao Mango Size

Mango size will be estimated through an image-processing workflow implemented in Python using OpenCV. Each mango image will first be converted from the BGR to HSV color space to facilitate segmentation based on characteristic fruit colors, namely green, yellow, and yellow-green. Binary masks will be generated for each color range and combined to isolate the mango region. Morphological operations such as opening and closing will then be applied to remove noise and refine the mask. The largest contour will be extracted to represent the mango, and a bounding rectangle will be fitted around it. To convert pixel dimensions into real-world measurements, a scaling factor will be established using a reference, the conveyor belt gap which has a fixed size that can be measured in cm and its corresponding pixel count in the image. The bounding box dimensions will be multiplied by this scaling factor to obtain mango length and width in centimeters. The estimated area will be computed as the product of these dimensions, and classification thresholds will be applied based on the optimal area ranges determined by statistical means. Each mango will be measured twice, once from the top face and once from the bottom face, which represent its largest visible areas. The two measurements will then be averaged to obtain a more reliable estimate of size. The conveyor system will fix the mango's position during measurement, preventing slanting or unwanted orientations that could introduce error.

5.6.7.3 K-Means Classification

The K-Means clustering algorithm can be utilized to classify carabao mango data into three size categories of small, medium, and large by specifying the parameter n_clusters = 3, which will pertain to the number of size classes. Prior to clustering, the area of each mango will be computed from its length and width measurements, producing a single feature that will represent overall fruit size. The input data will therefore consist of area values organized in a simple dataset format. After the algorithm runs, it will compute the coordinates of three cluster centers, where each center will represent the mean area of a group. The process will then assign a cluster index to each mango observation (Pedregosa et al., 2011). Because K-Means is an unsupervised method, these numerical cluster indices will need to be interpreted externally and assigned the labels of 'small,' 'medium,' and 'large' based on the physical dimensions represented by their respective cluster centers.

5.6.7.4 Quantile-Based Classifier

Quantile-based classification approach will also be employed to categorize carabao mangoes into three distinct size categories, namely small, medium, and large in order to determine the ideal range for sizes. From the length and width data, the area of each mango will be computed to serve as a single feature representing overall fruit size. This transformation will ensure that classification is based on a unified measure of size rather than separate dimensions.The quantile-based classifier will then be applied to the computed area values. The method will be generalized for three populations (Π_1, Π_2, Π_3, representing g = 3 classes). A new observation, defined by its computed area, will be assigned to the population that yields the lowest quantile distance, expressed as $\Phi_k(z, \theta)$, where k denotes the population index. The classifier will rely on determining the quantile functions $q_k(\theta)$ for each class distribution. A crucial step in this procedure will involve selecting the optimal quantile percentage θ, which will minimize misclassification error in the training sample and define the empirically optimal quantile classifier. The median classifier will be considered as a special case of this rule, corresponding to $\theta = 0.5$ (Hennig and Viroli, 2013). Since the method is not scale equivariant, variable scaling will be performed prior to classification to ensure comparability across observations.

5.7 Mango Formula with User Priority

The linear equation used to calculate the Carabao mango grade is shown below. Likewise, the variables $B\,(P)$, $R\,(P)$, and $S\,(P)$ represent the user-defined priority weightings for bruising, ripeness, and size characteristics in the User Priority-Based Grading system. Additionally, $b\,(p)$, $r\,(p)$, and $s\,(p)$ correspond to the machine learning model's predicted values for the bruising, ripeness, and size attributes of the Carabao mango.

$$\text{Mango Grade} = b\,(P)\,B\,(P) + r\,(P)\,R\,(P) + s\,(P)\,S\,(P) \tag{5.5}$$

The machine learning predictions are assigned the following numerical values:

Ripeness Scores:

$$r(\text{yellow}) = 1.0 \tag{5.6}$$

$$r(\text{yellow green}) = 2.0 \tag{5.7}$$

$$r(\text{green}) = 3.0 \tag{5.8}$$

Bruises Scores:

$$b(\text{bruised}) = 1.0 \tag{5.9}$$

$$b(\text{not bruised}) = 2.0 \tag{5.10}$$

Size Scores:

$$s(\text{small}) = 1.0 \tag{5.11}$$

$$s(\text{medium}) = 2.0 \tag{5.12}$$

$$s(\text{large}) = 3.0 \tag{5.13}$$

Note that the scores value for each respective classification cannot be changed by the user without changing the code itself. This means that only the weight of either the ripeness, bruises, and size can be changed to either low, high, or remove it by setting it to zero. Furthermore, only real numbers are allowed to be inputted as a weight. This means that negative and imaginary numbers are not considered in Equation 5.5.

5.8 Expert Evaluation Methodology

The expert benchmark was established by Jerry Bravante, a farmer with 20 years of experience in mango species such as carabao, pico, indian, apple mango. Joined with 2 other professionals in the same farm as Sir Bravante their expertise was employed to provide a ground-truth classification for mango samples based on two key phenotypic traits:

- Skin Color: yellow, yellow-green, green
- Bruises: bruised, non-bruised

To ensure statistical significance and mitigate the potential for coincidental agreement, a substantial sample set was utilized. Each expert evaluated 26 individual mangoes. No other tools except the expert's knowledge and eyes were used to evaluate the mangoes to ensure that the evaluation is based solely on human sensory perception.

5.9 Summary

This chapter details the methodology for developing an automated Carabao mango grading and sorting system integrating machine learning and computer vision. The research employed an experimental approach managed via Scrum agile methodology to iteratively develop and test the hardware and software components. The hardware design features a conveyor belt system, an image acquisition setup with controlled lighting, and a RPi microcontroller coordinating DC motors and sorting actuators. The software, built with Python and PyTorch, utilizes a custom-trained CNN for classification. The core machine learning pipeline involved extensive comparative testing of architectures, including EfficientNet, VGGNet, and ResNet, with EfficientNetV2-B3 ultimately selected for its optimal balance of accuracy and efficiency.

A significant focus was placed on data collection and optimization. A custom dataset of Carabao mangoes was created by capturing video of individual fruits and extracting frames, which were then sorted into categories for ripeness (green, yellow-green, yellow) and bruises (bruised, not bruised). The dataset was split 70-15-15 for training, validation, and testing, with aggressive data augmentation (rotation, flipping, blur) applied only to the training set to improve model generalization. The training process incorporated several advanced optimizations: the AdamW optimizer for better generalization, mixed-precision training to accelerate computation, data loading and transfer optimizations to prevent bottlenecks, and regularization techniques like dropout and label smoothing to combat overfitting. A cosine annealing learning rate scheduler and early stopping were also implemented to ensure stable convergence.

For system evaluation, the methodology defined specific testing protocols for each attribute. Ripeness was classified into three visually distinct stages, while bruise detection was trained to identify defects like stem end rot and anthracnose based on a standard defect guide. Two methods for size determination were developed and compared: a traditional computer vision approach using foreground masking and thresholding, and a more robust object detection method using a Faster R-CNN model trained on 488 annotated mango images. A key innovation is the user-priority formula, a weighted equation that allows users to customize the importance of ripeness, bruises, and size in the final grade (A, B, or C).

De La Salle University

Listing 5.1: Datasplit Logs

```
Class Mapping:
------------------------------
green        -> ripeness/green
yellow       -> ripeness/yellow
yellow_green -> ripeness/yellow_green
bruised      -> bruises/bruised
unbruised    -> bruises/not_bruised
Splitting dataset into hierarchical structure...
Processing green -> ripeness/green
  Train: 1225, Val: 262, Test: 263
Processing yellow -> ripeness/yellow
  Train: 616, Val: 132, Test: 132
Processing yellow_green -> ripeness/yellow_green
  Train: 935, Val: 200, Test: 201
Processing bruised -> bruises/bruised
  Train: 1363, Val: 292, Test: 293
Processing unbruised -> bruises/not_bruised
  Train: 1143, Val: 245, Test: 246
Applying massive augmentation to generate 10000 additional images...
Total augmentation combinations available: 309
Original training images: 6832
Total augmented images created: 13664
Target was: 10000

Dataset Statistics:
============================================================

RIPENESS Category:
----------------------------------------
  green          - Train: 7830, Val:  488, Test:  478
  yellow         - Train: 4010, Val:  242, Test:  248
  yellow_green - Train: 6130, Val:  376, Test:  376
  Subtotal       - Train: 17970, Val: 1106, Test: 1102

BRUISES Category:
----------------------------------------
  bruised        - Train: 8820, Val:  526, Test:  538
  not_bruised  - Train: 7370, Val:  446, Test:  450
  Subtotal       - Train: 16190, Val:  972, Test:  988

============================================================
TOTAL        - Train: 34160, Val: 2078, Test: 2090
Ratios       - Train: 89.1%, Val: 5.4%, Test: 5.5%

Dataset processing complete! Output saved to: E:\dir

========================================
```

Chapter 6

RESULTS AND DISCUSSIONS

TABLE 6.1 SUMMARY OF METHODS FOR ACHIEVING THE OBJECTIVES

Objectives	**Methods**	**Locations**
GO: To develop a user-priority-based grading and sorting system for Carabao mangoes, using machine learning and computer vision techniques to assess ripeness, size, and bruises.	Results: 1. Successfully developed a user-priority-based grading and sorting system using machine learning and computer vision which can assess the mangoes' ripeness, size and bruises.	Sec. 6.8 on p. 148
SO1: To make an image acquisition system with a conveyor belt for automatic sorting and grading mangoes.	Results: 1. Successfully integrated a conveyor belt with the image acquisition in order to achieve efficient flow of automated sorting and grading of the mangoes. 2. Successfully integrated LED strips to provide optimal lighting for image capturing of the mangoes. 3. Successfully fixed the hardware components in place	Sec. 6.6 on p. 141
SO2: To get the precision, recall, F1 score, confusion matrix, and train and test accuracy metrics for classifying the ripeness and bruises with an accuracy score of at least 90%.	Results: 1. Successfully achieved 98% overall accuracy for ripeness classification of Carabao mangoes 2. Successfully achieved 99% overall accuracy for bruises classification of Carabao mangoes	Sec. 6.1 on p. 98

Continued on next page

Continued from previous page

Objectives	**Methods**	**Locations**
SO3: To create a microcontroller-based system to operate the image acquisition system, control the conveyor belt, and process the mango images through machine learning.	Results: 1. Successfully made a conveyor belt system to move the mangoes through the image acquisition system to the sorting system 2. Successfully mounted the image acquisition system on the prototype 3. Successfully made the frame for the conveyor belt and image acquisition system to sit on	Sec. 6.6 on p. 141
SO4: To grade mangoes based on user priorities for size, ripeness, and bruises.	Results: 1. Successfully grade mangoes based on the user priorities on the physical characteristics of the mango 2. Successfully verified with qualified individual the results 3. Successfully utilize the weighted equation to evaluate mango grade based on user priorities	Sec. 6.5 on p. 139
SO5: To classify mango ripeness based on image data using machine learning algorithms such as kNN, k-mean, and Naïve Bayes.	Results: 1. Successfully trained a CNN model using EfficientNetV2 and Adam Optimizer for ripeness 2. Achieved 98% accuracy on performance metrics using EfficientNetV2 3. Obtain performance metrics for KNN, K-Mean, and Naive Bayes methods for comparison and show the superior performance of using CNN 4. Successfully fine tuned the CNN model to achieve the highest accuracy possible, choosing the best performing model, and testing other CNN hyperparameters	Sec. 6.1.1 on p. 98

Continued on next page

Continued from previous page

Objectives	Methods	Locations
SO6: To classify mango size based on image data by getting its length and width using OpenCV, geometry, and image processing techniques.	Results: 1. OpenCV method demonstrated an accurate performance, with measured area percent difference of 4.8% to the manual measurement by getting its length and width, respectively.	Sec. 6.4 on p. 134
SO7: To classify mango bruises based on image data by employing machine learning algorithms.	Results: 1. Successfully trained a CNN model using EfficientNetV2 and Adam Optimizer for bruises 2. Achieved 99% accuracy on performance metrics 3. Successfully fine tuned the CNN model to achieve the highest accuracy possible, choosing the best performing, and testing other CNN hyperparameters	Sec. 6.2.4 on p. 125

6.1 Training and Testing Results of the Model

6.1.1 Ripeness Classification Results

6.1.1.1 Naive Bayes

Based on the evaluation metrics, the Naive Bayes model demonstrates a clear strength in identifying ripe, yellow mangoes but reveals a significant weakness in classifying those in the transitional yellow-green stage. The model's precision scores for the green and yellow classes are reasonably similar at around 79%. However, its performance drops considerably for the yellow-green class, where a precision of just 58% nearly half of its predictions for this category are incorrect. This pattern is reinforced by the recall scores. The model excels at finding true yellow mangoes, capturing 86% of them, which is its highest performance metric. Conversely, it struggles to identify yellow-green mangoes, with a recall of only 51%, meaning it misses almost half of all true instances of this class. The F1-score, which balances precision and recall, provides summary of this performance, yielding a strong score of 80% for yellow but a very poor score of 55% for yellow-green. This confirms that the transitional yellow-green stage is the model's primary source of confusion, likely due to its visual ambiguity, sharing features with both the green and ripe yellow classes.

	Precision	**Recall**	**F1**	**Support**
Green	0.78	0.79	0.78	132
Yellow	0.75	0.86	0.80	66
Yellow_Green	0.58	0.51	0.55	101
Accuracy			0.71	299
Macro Avg	0.70	0.72	0.71	299
Weighted Avg	0.71	0.71	0.71	299

TABLE 6.2 RIPENESS CLASSIFICATION REPORT USING NAIVE BAYES

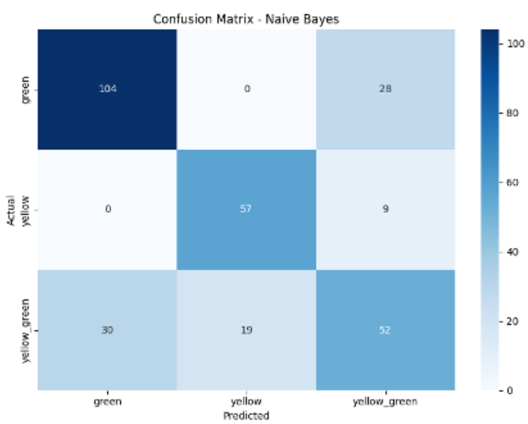


Fig. 6.1 Ripeness Confusion Matrix using Naive Bayes

6.1.1.2 KMeans

The KMeans model achieved a weak overall accuracy of 57%, with its performance characterized by a severe precision-recall trade-off across classes and a fundamental failure to identify the transitional stage. The model exhibited high recall for Green with score of 80% but low precision of 57%, which indicates that it captured most green mangoes but also frequently misclassified others as green. It was the opposite for Yellow, where high precision score of 83% and a low recall score of 52%, meaning its yellow predictions were reliable but it missed nearly half of them. Most critically, performance on the Yellow Green class was exceptionally poor with a F1 score of 34%, the model struggled both to correctly label them and to find them at all, this reveals that the clusters formed by KMeans are poorly separated for this specific ripeness classification task.

	Precision	**Recall**	**F1**	**Support**
Green	0.57	0.80	0.67	132
Yellow	0.83	0.52	0.64	66
Yellow_Green	0.41	0.30	0.34	101
Accuracy			0.57	299
Macro Avg	0.60	0.54	0.55	299
Weighted Avg	0.57	0.57	0.55	299

TABLE 6.3 RIPENESS CLASSIFICATION REPORT USING KMEANS

6.1.1.3 KNN

K-Nearest Neighbors (KNN) model demonstrates an improvement in performance, achieving an overall accuracy of 78%. Unlike previous models, KNN shows a strong and consistent balance between precision and recall across all three ripeness classes. The model excels at classifying the fully Green and Yellow stages, with high and well-balanced

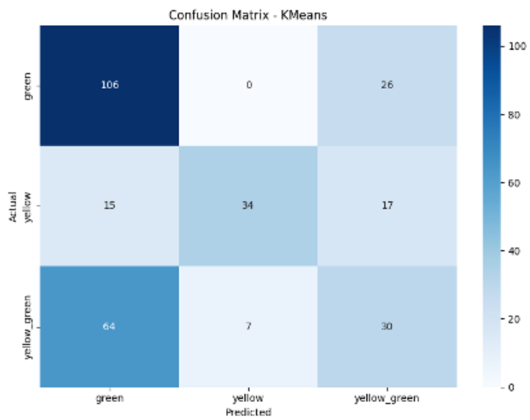


Fig. 6.2 Ripeness Confusion Matrix using KMeans

F1-scores of 0.85 and 0.81, respectively, indicating it is more reliable when making a prediction and effective at identifying all instances of these classes than previous models. KNN also shows improvement in handling the yellow-green class, achieving an F1-score of 68%. While this remains the most challenging class, the model's significantly higher scores compared to previous attempts confirm its ability to learn the distinguishing features between the stages.

	Precision	**Recall**	**F1**	**Support**
Green	0.85	0.85	0.85	132
Yellow	0.83	0.79	0.81	66
Yellow_Green	0.67	0.69	0.68	101
Accuracy			0.78	299
Macro Avg	0.78	0.78	0.78	299
Weighted Avg	0.78	0.78	0.78	299

TABLE 6.4 RIPENESS CLASSIFICATION REPORT USING KNN

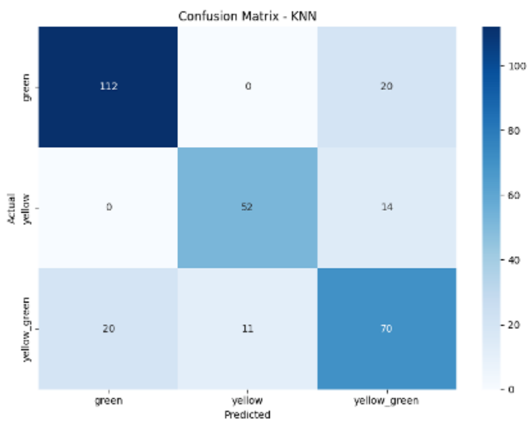


Fig. 6.3 Ripeness Confusion Matrix using KNN

6.2 Achieving the Highest Accuracy in CNN Models

6.2.1 Analyzing the Accuracy of Different CNN Network

For the classification of ripeness, the highest accuracy was obtained with EfficientNetV2-B0, which achieved 91%. This was followed by MobileNetV2, which achieved 90%, EfficientNet-B0 and GoogLeNet at 89%, DenseNet121 at 88%, and ResNet50 at 87%. In contrast, both VGGNet16 and AlexNet severely underperformed, each reaching only 43% accuracy. A closer inspection of their classification reports revealed that these two models predicted only the green class across all test samples, completely failing to recognize yellow and yellow_green. This explains why their accuracy plateaued at 43%, a value that directly corresponds to the proportion of green samples in the dataset. The collapse into a single-class prediction highlights the limitations of these older architectures: AlexNet and VGGNet16 lack the advanced feature extraction and efficient feature reuse mechanisms present in modern CNNs, making them less capable of capturing the subtle hue and texture variations that distinguish ripeness stages (Krizhevsky et al., 2012) (Simonyan and

Network	Prec	Rec	F1	Test Acc	Train Acc	Time	VRAM
VGG16	0.188	0.434	0.263	43	43.57	2h57m	7.0
ALEXNET	0.188	0.434	0.263	43	43.57	4h23m	2.3
RESNET50	0.870	0.869	0.868	87	89.22	7h13m	4.1
GOOGLENET	0.898	0.895	0.892	89	83.58	3h3m	2.9
MOBILENETV2	0.898	0.898	0.897	90	91.13	2h0m	3.6
DENSENET121	0.877	0.877	0.875	88	89.17	2h10m	5.5
EFFNET B0	0.890	0.888	0.887	89	91.24	2h14m	4.1
EFFNET B1	0.916	0.913	0.913	91	89.91	2h25m	5.3
EFFNET B2	0.906	0.902	0.900	90	89.46	2h26m	5.5
EFFNET B3	0.914	0.911	0.909	91	89.72	2h30m	6.8
EFFNET B4	0.899	0.898	0.896	90	92.34	2h50m	8.0
EFFNET B5	0.925	0.924	0.924	92	94.12	5h45m	11.6
EFFNET B6	0.934	0.933	0.933	93	96.03	7h12m	14.5
EFFNET B7	0.883	0.871	0.873	87	90.82	9h9m	18.8
EFFNETV2-B0	0.915	0.913	0.913	91	92.71	1h53m	3.0
EFFNETV2-B1	0.920	0.918	0.919	92	92.65	1h59m	3.7
EFFNETV2-B2	0.920	0.920	0.920	92	92.34	2h0m	3.8
EFFNETV2-B3	0.926	0.926	0.925	93	93.97	2h2m	4.5
EFFNETV2-S	0.894	0.893	0.891	89	90.47	2h17m	6.1
EFFNETV2-M	0.893	0.893	0.892	89	90.02	2h37m	9.9
EFFNETV2-L	0.875	0.871	0.870	87	89.93	13h39m	16.8
AVERAGE	**0.835**	**0.856**	**0.839**	**86**	**85.52**	**–**	**7.0**

TABLE 6.5 CNN TRAINING RESULTS FOR RIPENESS USING GPU

Zisserman, 2015). AlexNet, while revolutionary in 2012, was designed for large-scale but relatively coarse ImageNet classification and relies on shallow convolutional layers with large receptive fields, which limits its ability to capture fine-grained differences. Similarly, VGGNet16, though deeper, uses very uniform 3×3 convolutions without skip connections or dense connectivity, leading to redundancy and inefficient feature reuse, which modern architectures have since addressed. Furthermore, the training setup and hyperparameters, which favored faster convergence in lightweight and well-optimized

Network	**Prec**	**Rec**	**F1**	**Test Acc**	**Train Acc**	**Time**	**Mem**
VGG16	0.297	0.545	0.384	54	54.48	5h38m	6.5
ALEXNET	0.297	0.545	0.384	54	54.48	4h25m	3.3
RESNET50	0.858	0.844	0.844	84	83.92	8h24m	5.4
GOOGLENET	0.843	0.808	0.799	81	57.67	3h14m	4.0
MOBILENETV2	0.859	0.858	0.858	86	85.88	3h44m	4.8
DENSENET121	0.839	0.838	0.838	84	84.7	3h8m	6.7
EFFNET B0	0.873	0.870	0.870	87	90.04	2h37m	5.3
EFFNET B1	0.898	0.897	0.896	90	90.51	2h56m	6.7
EFFNET B2	0.901	0.901	0.901	90	91.21	3h8m	6.7
EFFNET B3	0.913	0.913	0.913	91	90.34	3h27m	8.0
EFFNET B4	0.897	0.897	0.897	90	92.16	4h17m	9.8
EFFNET B5	0.892	0.883	0.881	88	90.53	5h49m	12.2
EFFNET B6	0.884	0.883	0.882	88	90.43	7h51m	14.5
EFFNET B7	0.857	0.856	0.856	86	90.47	10h34m	18.0
EFFNETV2-B0	0.880	0.879	0.878	88	90.69	2h6m	4.4
EFFNETV2-B1	0.893	0.893	0.893	89	91.72	2h32m	5.1
EFFNETV2-B2	0.904	0.889	0.889	89	88.16	2h45m	5.4
EFFNETV2-B3	0.919	0.919	0.919	92	94.46	2h55m	6.2
EFFNETV2-S	0.859	0.858	0.858	86	86.58	2h58m	7.7
EFFNETV2-M	0.856	0.846	0.846	85	84.74	3h30m	9.3
EFFNETV2-L	0.849	0.836	0.836	84	85.05	14h58m	17.9
AVERAGE	**0.822**	**0.841**	**0.825**	**84.10**	**–**	**–**	**8.0**

TABLE 6.6 CNN BRUISES RESULTS FOR BRUISES USING CPU

	Accuracy	
Model	Ripeness	Bruises
EfficientNetB0	89%	87%
EfficientNetB2	92%	90%
VggNet16	43%	54%
AlexNet	43%	54%
Residual Network (ResNet)50	87%	84%
GoogleNet	89%	81%
MobileNetV2	90%	86%
DenseNet121	88%	84%

TABLE 6.7 ACCURACY OF DIFFERENT CNN MODELS

	Test Accuracy	
EfficientNet	Ripeness	Bruises
B0	89%	87%
B1	86%	90%
B2	92%	90%
B3	88%	91%
B4	90%	90%
B5	92%	88%
B6	93%	88%
V2B0	91%	88%
V2B1	92%	89%
V2B2	92%	89%
V2B3	93%	92%
V2-S	89%	86%
V2-M	89%	85%
V2-L	89%	84%

TABLE 6.8 TEST ACCURACY OF DIFFERENT EFFICIENTNET VERSION 1 AND 2

models such as MobileNetV2 and EfficientNet (Howard et al., 2017) (Tan and Le, 2019), did not provide the same benefit to AlexNet and VGGNet16 (Huang et al., 2017). Importantly, the train accuracy values further reinforce these findings where modern architectures such as EfficientNetV2-B3 (93% train, 93% test) and EfficientNet-B6 (96% train, 93% test) maintained close alignment between training and test performance, indicating strong generalization. In contrast, AlexNet and VGGNet16 stagnated at 43% for both training and test accuracy, indicating that they were underfitting and unable to capture the discriminative features necessary for ripeness classification. From a performance requirements perspective, the results also demonstrate that modern architectures not only achieved higher accuracy but did so with significantly lower training times and more efficient VRAM utilization. For instance, EfficientNetV2-B0 reached the highest accuracy in under two hours with an average VRAM usage of only 3 GB, while AlexNet required over four hours yet produced poor results, and VGGNet16 consumed the highest VRAM (7 GB) despite its low accuracy. This efficiency–accuracy balance makes modern CNNs far more suitable for practical deployment in ripeness classification tasks, where both computational cost and predictive reliability are critical.

For the classification of bruises, the highest accuracy was obtained with EfficientNetV2-B0, which achieved 88%. This was followed by EfficientNet-B0 at 87% and MobileNetV2 at 86%. ResNet and DenseNet121 both reached 84%, while GoogLeNet trailed slightly at 81%. In contrast, both VGG16 and AlexNet severely underperformed, each plateauing at only 54% accuracy. Similar to the results from training ripeness, VGG16 and AlexNet collapsed into underfitting, where both models produced very low precision (0.2965) and F1-scores (0.384), and their training accuracy stagnated at the same 54%, confirming their inability to learn discriminative features. By contrast, modern architectures such as EfficientNet and MobileNetV2 leverage depthwise separable convolutions, compound scaling, and optimized feature reuse, enabling them to achieve higher accuracy with fewer parameters and faster convergence. EfficientNetV2-B0 not only achieved the highest accuracy (88%) but also did so in just 2 hours and 6 minutes with an average VRAM usage of 4.4 GB, making it both the most accurate and the most computationally efficient. MobileNetV2, while slightly less accurate, also demonstrated excellent efficiency, completing training in under 4 hours with modest memory requirements. From a performance requirements perspective, these results highlight that modern CNNs are not only more accurate but also far more resource-efficient. VGG16, despite consuming the most VRAM (6.5 GB) and requiring over 5 hours of training, delivered poor results, while AlexNet trained for more than 4 hours yet plateaued at the same low accuracy. In contrast, EfficientNetV2-B0 and EfficientNet-B0 achieved state-of-the-art performance in a fraction of the time and memory.

Ultimately, choosing a CNN model from the EfficientNetV2 family represents the most practical and forward-looking decision for both ripeness and bruise classification tasks. These models consistently delivered the highest accuracy across experiments while maintaining shorter training times and lower memory footprints compared to other architectures. Their compound scaling strategy allows them to balance depth, width, and resolution more effectively than earlier CNNs, ensuring strong generalization without excessive computational cost Tan and Le (2019). This makes them not only state-of-the-art in predictive performance but also highly deployable in real-world agricultural settings, where efficiency, scalability, and reliability are critical. By combining accuracy, speed, and resource efficiency, the EfficientNetV2 family provides the best foundation for building robust and sustainable computer vision systems for fruit quality assessment.

6.2.2 Analysis of Table 6.6 and Table 6.5

For ripeness classification, among the EfficientNet V1 models as seen in Table 6.6 and Table 6.5 , B0 to B4 exhibited a performance plateau around 89–91% accuracy. This can be explained by the compound scaling principle where each successive variant increases depth, width, and input resolution in tandem (Tan and Le, 2019). However, for the benchmark, the input resolution was fixed at 224×224 for all models. Since B0–B4 are relatively shallow and narrow, their representational capacity is already well-matched to the available input information at 224×224. Scaling them further in depth and width without increasing resolution does not provide additional discriminative power, leading to plateau in accuracy. Notably, their training accuracies, ranging from 89.5% to 92.3%, closely mirrored their test accuracies, suggesting that these models were neither severely underfitting nor overfitting, but rather limited by the resolution bottleneck. In contrast, B5 and B6 showed measurable improvements (92–93% accuracy) even under the 224×224 constraint. This is because their increased depth and width allowed them to extract more abstract and hierarchical features, compensating for the lack of higher-resolution input. While they were not operating at their full theoretical potential, which would require larger input sizes like 456×456 or 528×528, their additional capacity still translated into better generalization for the 3-class ripeness classification task. Essentially, B5 and B6 reached a sweet spot where the added representational power was still beneficial, even though the input resolution bottleneck limited further gains. This is further supported by their training accuracies (94.1% for B5 and 96.0% for B6), which were slightly higher than their test accuracies, indicating strong learning capacity with only a modest generalization gap. By contrast, B7 crossed the threshold where additional scaling became counterproductive. With 18.8 GB of VRAM usage and a 9-hour training time, its extreme depth and parameter count, combined with the fixed low-resolution input, led to over-parameterization relative to the available information, optimization inefficiency, and degraded performance (87%). This increase in training time and memory usage is expected, as higher EfficientNet versions introduce significantly more parameters. For instance, B6 has over 43 million parameters compared to B5's 30 million, resulting in longer forward and backward passes and greater memory consumption per epoch. If the required memory exceeds available VRAM, the system resorts to RAM, which has slower access speeds, thereby significantly increasing training time. On the other hand, EfficientNetV2 models demonstrated superior efficiency and faster convergence. Variants B0–B3 consistently achieved 91–93% accuracy, with V2-B3 emerging as the top performer (precision 0.9258, recall 0.9256, F1-score 0.9253, accuracy 93%) while maintaining modest VRAM usage (4.5 GB) and a short training time (∼2 hours). Their training accuracies (92.3–94.0%) were well aligned with their test accuracies, confirming

De La Salle University

that these models generalized effectively without significant overfitting. In contrast, the larger variants (V2-S, V2-M, V2-L) all exhibited diminishing returns, as their increased depth and parameter counts did not translate into higher accuracy, instead plateauing at 87–89% while demanding substantially more computational resources, similar to the case with EfficientNetV1 series. Their longer training times and higher VRAM usage reflect the same scaling trade-offs observed in B7, where added complexity does not yield proportional performance gains under fixed input resolution (Tan and Le, 2021). This was also reflected in their training accuracies (90.0–90.5%), which showed little advantage over their test results, reinforcing that additional complexity did not yield meaningful gains. This performance limitation may also be attributed to the fixed input image size of 224×224, which constrained the representational capacity of deeper models , a phenomenon similarly observed with the EfficientNetV1-B7. This suggests that for a 3-class dataset of approximately 6,000 images, additional model complexity does not yield proportional performance gains and may even hinder optimization efficiency. Under these conditions, V2-B3 stands out as the most effective architecture, striking the best balance between accuracy, efficiency, and training time.assessment.

For bruise classification as seen in Table 6.6, mid-tier EfficientNet V1 models (B1–B3) delivered the strongest results, with B3 achieving the highest performance (precision = 0.913, recall = 0.913, F1-score = 0.9129, accuracy = 91%). Their training accuracies (∼90–91%) were closely aligned with their test results, indicating that these models generalized well without significant overfitting. In contrast, the larger V1 variants (B5–B7) required substantially more training time and memory yet plateaued at 86–88% accuracy, reflecting the same diminishing returns noted in ripeness classification. This was further supported by their training accuracies (∼90–90.5%), which were only marginally higher than their test scores, suggesting that additional depth and parameters did not translate into meaningful generalization gains. Among the V2 models, V2-B3 stood out with 92% accuracy and balanced precision/recall (0.919 each), surpassing the best V1 models while maintaining shorter training times and lower memory usage. Meanwhile, the larger V2 variants (S, M, L) mirrored the inefficiencies of their V1 counterparts, consuming more resources without corresponding accuracy gains. Their training accuracies (∼85–86%) were nearly identical to their test results, confirming that these models were underutilizing their added capacity under the fixed 224×224 input constraint. Across both families, GPU-based training consistently achieved shorter training times than CPU-only runs, even though the bruise classification task involved only two classes and used the same dataset.

Overall, EfficientNetV2-B3 emerged as the most practical and effective model for both ripeness and bruise classification, combining high accuracy (93% and 92%, respectively) with modest VRAM requirements and short training times (∼2–3 hours). Its balance of performance and efficiency makes it particularly well-suited for deployment in real-world agricultural applications, where computational resources may be limited but reliable, high-accuracy classification is essential. Complementing this, training with GPUs proved consistently advantageous across both tasks, as their massively parallel architecture is optimized for the matrix multiplications and convolution operations central to deep learning. This allowed models to converge significantly faster than on CPUs, reducing training times from several hours to just a fraction of that. The efficiency gains were especially evident in deeper networks, where CPU-only training often became impractically slow. Notably, bruise classification, despite involving only two classes and the same dataset size, still trained more slowly on CPU than ripeness classification did on GPU, underscoring the decisive role of hardware acceleration in practical deep learning workflows.

6.2.3 Analysis of Confusion Matrix together with Validation Loss and Accuracy

In this section, the performance of the top three models for both ripeness and bruise classification is examined in greater detail through their validation loss and accuracy curves, as well as their corresponding confusion matrices. These analyses provide deeper insight into how each model converged during training, the stability of their learning process, and their ability to generalize beyond the training set. The confusion matrices, in particular, highlight the distribution of correct and incorrect predictions across classes, allowing for a clearer understanding of where misclassifications occur.

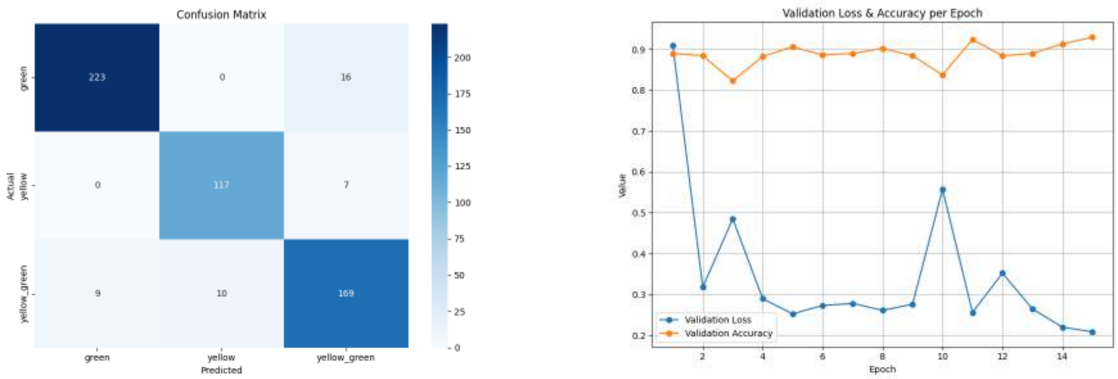
6.2.3.1 Ripeness Classification

To start off, for ripeness classification, The EfficientNet-B5 model achieved strong overall performance, with a precision of 0.9246, recall of 0.9238, and an F1-score of 0.924, corresponding to an overall accuracy of 92%, being the 3rd best model for the task. These values indicate that the model is highly effective at distinguishing between the three ripeness classes, with balanced precision and recall suggesting that it does not disproportionately favor one class over another. Training required approximately 5 hours and 45 minutes, with an average VRAM usage of 11.6 GB, reflecting the computational demands of a high-capacity architecture such as EfficientNet-B5.

Based on the confusion matrix in Figure 6.4, the model classified the majority of samples correctly across all categories, with particularly strong results for the green and yellow classes. For instance, 223 out of 239 green samples were correctly identified, with only 16 misclassified as yellow_green. Similarly, the yellow class showed minimal confusion, with 117 correct predictions and only 7 misclassified as yellow_green. The greatest overlap occurred in the yellow_green class, where 169 samples were correctly predicted, but 19 were misclassified as either green or yellow. This pattern suggests that the transitional nature of the yellow_green class poses the greatest challenge, as its visual features overlap with both neighboring categories. Nonetheless, the relatively low misclassification rates confirm that the model captures the key discriminative features of each ripeness stage.

The validation loss and accuracy curves in Figure fig:effnetb5 further illustrate the model's behavior during training. Validation accuracy remained consistently high, stabilizing above 0.90 across all epochs, which indicates that the model generalized well to unseen data. In contrast, validation loss exhibited noticeable fluctuations, with sharp drops and occasional peaks at specific epochs. This divergence between stable accuracy and variable loss suggests that while the model consistently predicted the correct class, it sometimes assigned lower confidence to its predictions. This behavior is common in multi-class classification tasks where class boundaries are less distinct, as in the case of the yellow_green category. Importantly, the absence of a downward trend in accuracy despite the oscillations in loss indicates that the model did not suffer from severe overfitting.

From a performance requirements perspective, the EfficientNet-B5 model demonstrates a favorable balance between accuracy and computational cost. Achieving over 92% accuracy with an F1-score of 0.924 while maintaining an average VRAM usage of 11.6 GB indicates that the model is both reliable and feasible for deployment on high-end GPUs commonly available in research and industrial settings. The total training time of 5 hours and 45 minutes is reasonable given the model's depth and parameter count, suggesting that retraining or fine-tuning for new datasets is practical within typical project timelines. Importantly, the stability of validation accuracy across epochs implies that the model converges efficiently without requiring excessive epochs, further reducing computational overhead. These results highlight that EfficientNet-B5 not only meets accuracy benchmarks but also aligns with resource efficiency considerations, making it a strong candidate for real-world applications where both predictive performance and hardware constraints must be balanced.



(a) Confusion Matrix (b) Validation and Accuracy per Epoch

Fig. 6.4 Ripeness Training and Testing of EfficientNet-B5

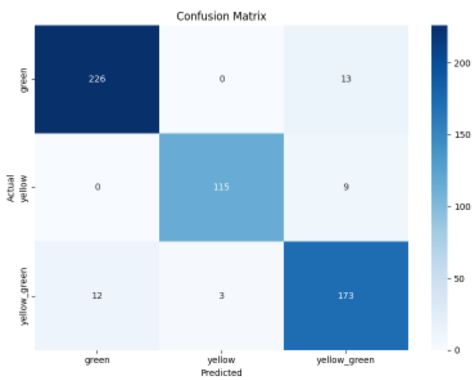
The second-best model for ripeness classification is EfficientNet-B6, achieving a precision of 0.9339, recall of 0.9328, and an F1-score of 0.9331, corresponding to an overall accuracy of 93%. Like EfficientNet-B5, it demonstrated strong and balanced performance across all three ripeness categories, but with slightly higher accuracy. Training required approximately 7 hours and 12 minutes, with an average VRAM usage of 14.5 GB, which is substantially more demanding than B5, reflecting the deeper architecture and larger parameter count.

The confusion matrix in Figure 6.4 shows that the green class was classified with high reliability, with 226 correct predictions and only 13 misclassified as yellow_green. The yellow class also performed well, with 115 correct predictions and 9 misclassified as yellow_green. As with B5, the yellow_green class posed the greatest challenge due to its transitional characteristics, with 173 correct predictions but 15 misclassified as either green or yellow. This reinforces the earlier observation that intermediate ripeness stages are inherently more ambiguous, though overall misclassification rates remained low.

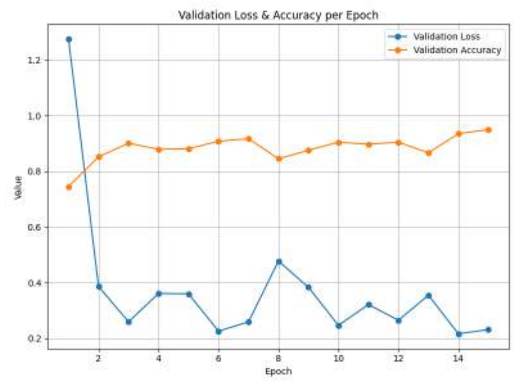
The validation curves in Figure 6.4 further illustrate the model's training dynamics. Validation loss decreased sharply after the first epoch and stabilized between 0.2 and 0.4, while validation accuracy steadily increased, reaching approximately 0.97 by the final epoch. This consistent improvement indicates effective convergence without signs of severe overfitting. Compared to B5, B6 leveraged its higher representational capacity to refine feature extraction further, leading to more confident predictions.

From a performance standpoint, EfficientNet-B6 clearly delivers superior accuracy compared to B5, but at the cost of significantly higher resource consumption. While its 93% accuracy and F1-score above 0.93 make it highly reliable for practical applications, the 14.5 GB VRAM requirement and extended training time of over 7 hours highlight the trade-off between accuracy gains and efficiency. As with B5, this makes B6 well-suited for research and industrial environments with high-end GPUs, but less practical for real-time or edge deployment without model compression or optimization.

The best-performing model for ripeness classification was EfficientNetV2-B3, achieving a precision of 0.9258, recall of 0.9256, F1-score of 0.9253, and an overall accuracy of 93%. These results confirm that the model is highly effective at distinguishing between the three ripeness categories, with balanced precision and recall indicating consistent performance across classes. Training required only 2 hours and 2 minutes with an average VRAM usage of 4.5 GB, making it far more efficient than deeper variants such as B5 and B6 while still achieving comparable accuracy.



(a) Confusion Matrix



(b) Validation and Accuracy per Epoch

Fig. 6.5 Ripeness Training and Testing of EfficientNet-B6

The confusion matrix in Figure 6.5 provides further insight into class-level performance. The green class was classified with high reliability, with 231 correct predictions and only 8 misclassified as yellow_green. The yellow class also performed strongly, with 115 correct predictions and 9 misclassified as yellow_green. As with the other models, the yellow_green class posed the greatest challenge, with 164 correct predictions but 24 misclassified as either green or yellow. This reflects the inherent ambiguity of the transitional stage, where visual features overlap with both neighboring categories. Despite this, overall misclassification rates remained low, confirming that the model effectively captured the discriminative features of each ripeness stage.

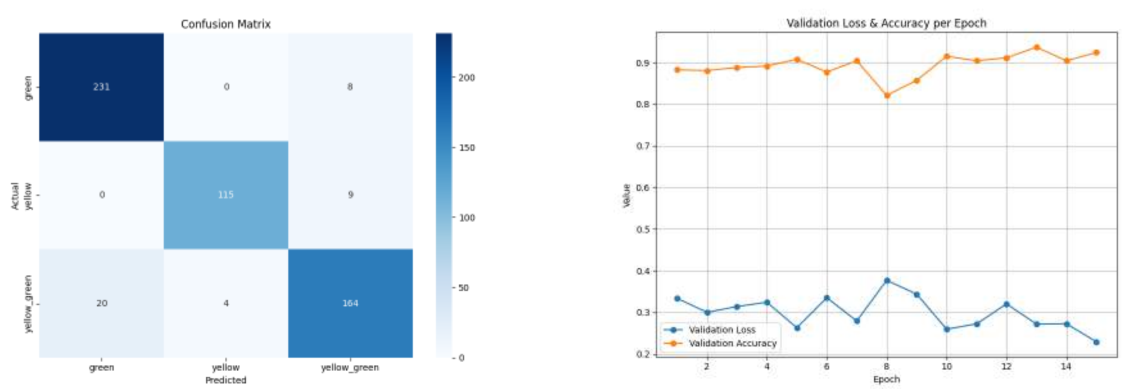
The validation curves in Figure 6.5 further illustrate the model's training dynamics. Validation accuracy remained consistently high, stabilizing between 0.85 and 0.92 across epochs, while validation loss fluctuated between 0.2 and 0.4. The stability of accuracy, despite minor oscillations in loss, suggests that the model generalized well to unseen data and avoided severe overfitting. The fluctuations in loss likely reflect varying confidence in predictions for the ambiguous yellow_green class, but the consistently high accuracy

De La Salle University

demonstrates that the model still assigned correct labels in most cases.

From a performance standpoint, EfficientNetV2-B3 offers the best balance between accuracy and computational efficiency. Achieving 93% accuracy with an F1-score above 0.92 while requiring only a fraction of the training time and memory of B5 or B6 highlights its practicality for deployment. While B6 achieved slightly higher precision and recall, its steep computational demands, over 7 hours of training and 14.5 GB of VRAM, make it less suitable for iterative experimentation or resource-constrained environments. Similarly, B5 delivered strong accuracy but required nearly 6 hours of training and 11.6 GB of VRAM, reflecting a high resource cost for only marginal gains. In contrast, V2-B3 enables faster experimentation cycles, more accessible deployment, and robust classification of both ripeness extremes and transitional classes.

Ultimately, EfficientNetV2-B3 provides the optimal trade-off between high- quality classification and manageable computational requirements, making it the best candidate for mango ripeness classification.



(a) Confusion Matrix

(b) Validation and Accuracy per Epoch

Fig. 6.6 Ripeness Training and Testing of EfficientNetV2-B3

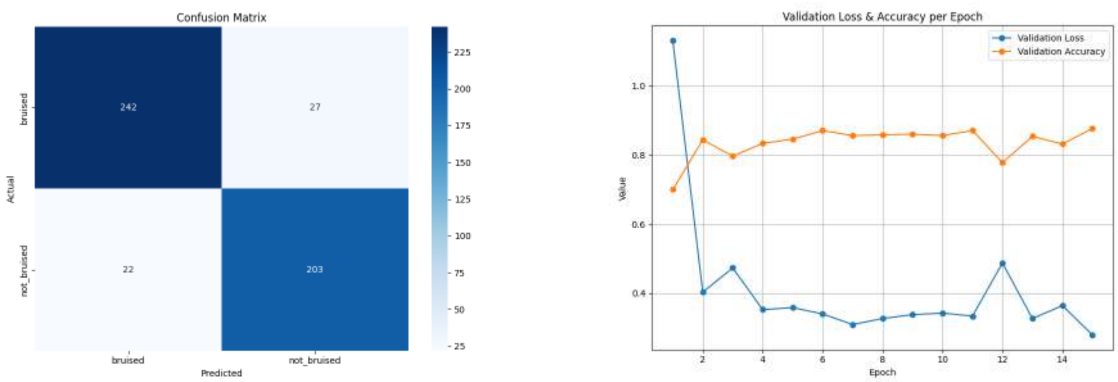
6.2.3.2 Bruises Classification

Moving forward with bruise classification, the EfficientNet-B2 model achieved strong performance overall. It reached a precision of 0.9012, recall of 0.9008, and F1-score of 0.9009. The overall accuracy was 90%, ranking as the third-best model tested. These results show a well-balanced model with minimal trade-offs in detection. It effectively identifies both bruised and not-bruised cases with reliable accuracy. Training lasted approximately 3 hours and 8 minutes under stable GPU performance. Average VRAM usage was about 6.7 GB during the entire training session. This computational demand remains manageable for most modern GPU-based research setups.

The confusion matrix in Figure 6.7 reveals the class-level distribution clearly. The model correctly identified 242 bruised and 203 not-bruised fruit samples. However, it misclassified 27 bruised items as not bruised, indicating false negatives. Additionally, 22 not-bruised items were misclassified as bruised, producing false positives. This pattern suggests a slight tendency to under-detect bruised mango samples. False negatives are critical in quality control because they allow defects through. Despite these errors, the model maintains strong reliability in classification results overall.

The validation curves in Figure 6.7 illustrate training stability and convergence well. Validation loss dropped sharply after the first epoch and continued declining steadily. Validation accuracy increased quickly, stabilizing around 0.85 after several epochs completed. These trends indicate efficient learning and absence of severe overfitting during training. Minor oscillations in loss and accuracy reflect normal exploration of local minima. Such fluctuations are typical in deep learning models seeking optimal decision boundaries.

From a performance perspective, EfficientNet-B2 satisfies practical requirements for bruise detection systems. With 90% accuracy and balanced precision-recall metrics, it ensures consistent defect detection. The model offers reliability without imposing excessive computational or memory resource demands. Its three-hour training time supports scalability for mid-range GPU deployment setups. However, false negatives remain a primary issue affecting industrial screening reliability. Reducing them may involve threshold adjustments or using cost-sensitive learning approaches. Ensemble methods could further improve robustness and minimize undetected bruised cases effectively.



(a) Confusion Matrix

(b) Validation and Accuracy per Epoch

Fig. 6.7 Bruises Training and Testing of EfficientNet-B2

The EfficientNet-B3 model demonstrated strong classification performance across all evaluation metrics. It achieved a precision of 0.913, recall of 0.913, and F1-score of 0.9129. Overall accuracy reached 91%, ranking as the second-best model for bruise classification. These values reflect high consistency in identifying both bruised and not-bruised samples. The trade-offs between false positives and false negatives remained minimal overall. Training required approximately 3 hours and 27 minutes using stable GPU resources. Average memory usage was 8 GB, slightly higher than EfficientNet-B2's requirements. Despite this, resource demands remained feasible for most modern GPU systems.

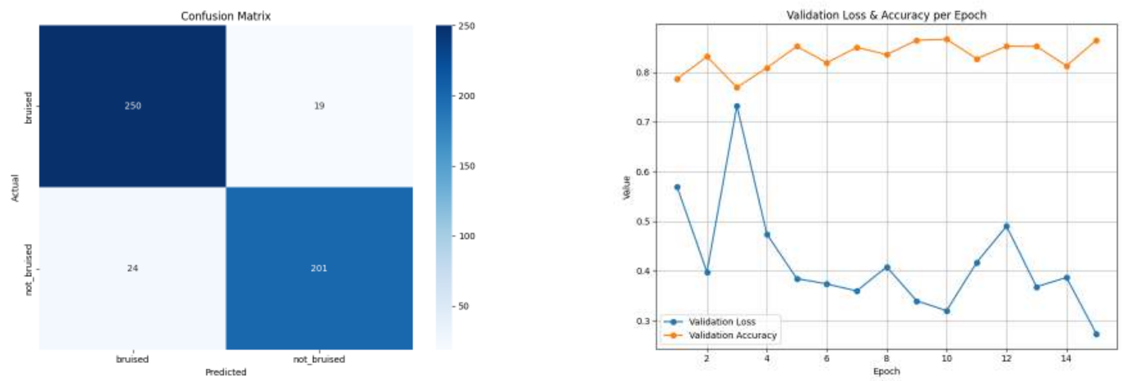
De La Salle University

The confusion matrix in Figure 6.8 presents the model's classification outcomes clearly. The network correctly identified 250 bruised and 201 not-bruised fruit samples. It misclassified 19 bruised items as not bruised, representing false negatives. Additionally, 24 not-bruised items were misclassified as bruised, forming false positives. Compared to EfficientNet-B2, this model reduced false negatives significantly overall. This reduction decreases the likelihood of defective mangoes passing inspection unnoticed. Such improvement is crucial in quality control, where undetected bruising is costly. False alarms are less concerning than missed detections in industrial screening tasks.

The validation curves in Figure 6.8 depict stable training and convergence performance. Validation loss decreased steadily across epochs, showing consistent learning throughout training. Validation accuracy stabilized between 0.80 and 0.85 with minimal oscillations. This parallel pattern of low loss and stable accuracy suggests good generalization. The relatively flat accuracy curve after early epochs indicates efficient convergence overall. No signs of instability or severe overfitting were observed during final training.

From a performance perspective, EfficientNet-B3 offers improved reliability over EfficientNet-B2. It balances classification accuracy and computational efficiency more effectively for bruise detection. Although training time and memory usage slightly increased, accuracy gains justify the cost. The reduced false negatives strengthen model dependability for automated quality control. This characteristic ensures fewer defective fruits are misclassified as acceptable products. Overall, EfficientNet-B3 represents a dependable and scalable choice for industrial bruise inspection.

The EfficientNetV2-B3 model achieved the best overall performance for bruise classification. It reached precision, recall, and F1-score values all equal to 0.919. The overall accuracy was 92%, demonstrating strong and balanced predictive capability. These metrics



(a) Confusion Matrix

(b) Validation and Accuracy per Epoch

Fig. 6.8 Bruises Training and Testing of EfficientNet-B3

confirm consistent performance across both bruised and not-bruised mango classes. Neither precision nor recall dominated at the expense of the other. Training was notably efficient, finishing in just 2 hours and 55 minutes. Average VRAM usage measured only 6.2 GB throughout the training process. This requirement was lower than both EfficientNet-B2 and EfficientNet-B3 models. Despite lower computational demand, the model still achieved superior classification accuracy. This efficiency-accuracy balance makes V2-B3 practical for constrained computing environments.

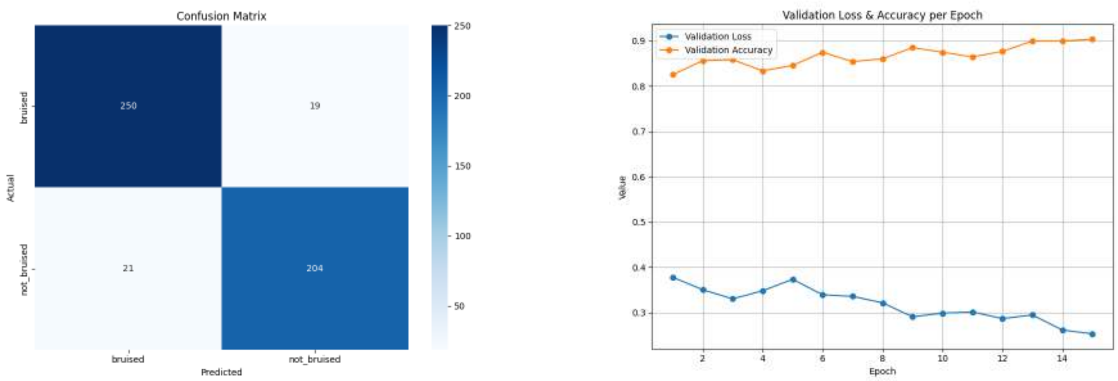
The confusion matrix in Figure 6.9 illustrates the model's predictive distribution. The network correctly classified 245 bruised and 202 not-bruised fruit samples. It misclassified 24 bruised items as not bruised, forming false negatives. Meanwhile, 23 not-bruised items were incorrectly labeled as bruised, forming false positives. Compared to previous models, V2-B3 exhibited a more balanced error profile. EfficientNet-B2 and B3 showed slightly higher false negatives or false positives respectively. From a practical perspective, false negatives pose greater risks in production. Undetected bruised fruit directly threaten overall product quality and customer satisfaction. Although the number of missed detections was relatively small, optimization remains beneficial. Techniques such as threshold tuning or cost-sensitive loss functions may further reduce them.

The validation curves in 5.9 show consistent training convergence behavior. Validation accuracy steadily increased and stabilized close to 0.9 after several epochs. Validation loss fluctuated slightly but showed a clear downward trend overall. This parallel pattern of stable accuracy and decreasing loss indicates effective generalization. Minor oscillations in loss reflect expected variations due to batch differences. Such fluctuations were also observed in EfficientNet-B2 and EfficientNet-B3 models. However, V2-B3 maintained consistently higher accuracy across the entire training process. No severe overfitting or instability was observed during model development or validation.

In summary, EfficientNetV2-B3 outperformed both EfficientNet-B2 and EfficientNet-B3 comprehensively. It delivered superior predictive accuracy while reducing training time and memory consumption. The model also demonstrated smoother convergence and improved stability during optimization. This balance of precision, efficiency, and robustness highlights its deployment suitability. EfficientNetV2-B3 stands as the most effective network for automated bruise detection. It provides a scalable, reliable, and resource-efficient solution for industrial quality control.

6.2.3.3 CNN

The final CNN model for ripeness and bruise classification utilized EfficientNetV2-B3. Collected experimental data confirmed that it achieved the best performance-to-efficiency ratio. It consistently outperformed other architectures tested during benchmarking and optimization stages. For the final ripeness classification, the complete dataset contained around 14,000 images. The model achieved a test accuracy of 98%, with precision, recall,



(a) Confusion Matrix

(b) Validation and Accuracy per Epoch

Fig. 6.9 Bruises Training and Testing of EfficientNetV2-B3

and F1-score near 0.985. This consistency across metrics demonstrates both high accuracy and class-balanced reliability. It performed uniformly across all ripeness categories without favoring any particular class. Validation accuracy of 98.41% closely matched the test accuracy, confirming excellent generalization. The slightly higher training accuracy of 99.37% indicated minimal overfitting occurrence. The narrow gap between training, validation, and test results reflected stable learning. These findings confirm that dataset refinement and optimization prevented memorization effectively. They also promoted genuine feature learning across ripeness categories and lighting variations.

The confusion matrix in Figure 6.10 further supports these conclusions clearly. Misclassifications were minimal and occurred mostly between adjacent ripeness categories. Errors were concentrated between transitional stages such as yellow-green and yellow mangoes. This pattern matches the biological ambiguity seen during mango ripening transitions. Even human evaluators sometimes disagree on borderline ripeness due to visual overlap. The model's strong accuracy in these ambiguous cases reflects superior discriminative ability. It demonstrates practical reliability for deployment in real-world mango grading systems.

Several major modifications to the training pipeline improved overall model effectiveness significantly. Mixed-precision training using GradScaler and autocast reduced GPU memory consumption substantially. This optimization increased batch size from 32 to 56, enhancing training stability. Larger batch sizes improved gradient estimation and smoothed convergence across training epochs. Input resolution was corrected to 300×300, matching EfficientNetV2-B3's native architecture. This adjustment improved feature extraction and ensured compatibility with pretrained weights. The optimizer was changed from Adam to Adam with Decoupled Weight Decay (AdamW) for stronger regularization. Learning rate was set to 3e-4, and weight decay to 1e-4. These parameters decoupled regularization from gradient updates, ensuring stable convergence behavior. A cosine annealing warm-restart scheduler with T0 = 5 and Tmult = 2 was applied. It included three warm-up epochs to escape sharp minima effectively during training.

Additional refinements further improved training robustness and model generalization performance. CrossEntropy loss with label smoothing of 0.05 reduced overconfident predictions. This adjustment improved resilience to ambiguous ripeness categories and noisy image labels. Early stopping with a patience of five epochs prevented redundant computation cycles. Checkpointing saved the best weights once performance improvements plateaued consistently. Data loading was optimized with workers set to half of available CPU cores. Pin_memory and non_blocking transfers accelerated CPU-to-GPU data streaming throughput. These optimizations minimized data bottlenecks and reduced idle GPU computation time. Regularization through dropout = 0.25 and drop-path = 0.15 improved network robustness. These techniques prevented neuron co-adaptation and encouraged diverse feature representations.

The validation curves in Figure 6.11 confirm stable convergence throughout training.

Validation accuracy increased steadily before plateauing at a consistently high level. Validation loss showed minor oscillations but followed an overall downward trajectory. This inverse relationship between loss and accuracy indicates strong discriminative learning ability. Accuracy stability despite small loss fluctuations shows resistance to overfitting. These patterns confirm that optimizations such as label smoothing and annealing worked effectively. The model maintained robustness and generalization even in complex visual conditions. Its smooth convergence underscores training stability and computational efficiency across all epochs.

Lastly, dataset enhancements contributed substantially to achieving these superior results overall. The dataset expanded from approximately 6,000 to 14,000 well-curated mango images. New Carabao mango samples were added, improving variety and biological representativeness. Ambiguous or noisy samples were removed to reduce label uncertainty significantly. Augmentation strategies were refined to introduce meaningful color, rotation, and lighting diversity. These augmentations enhanced robustness by exposing the network to realistic visual variations. As a result, the final model generalized strongly and maintained stable performance. Across all dataset splits, it demonstrated consistent accuracy and balanced classification reliability.

	Precision	**Recall**	**F1**	**Support**
Green	0.98	0.99	0.99	210
Yellow	0.99	0.99	0.99	161
Yellow_Green	0.98	0.98	0.98	219
Accuracy			0.98	590
Macro Avg	0.99	0.99	0.99	590
Weighted Avg	0.98	0.98	0.98	590

TABLE 6.9 EFFICIENTNETV2-B3 RIPENESS CLASSIFICATION REPORT WITH PRECISION: 0.9848, RECALL: 0.9847, F1 SCORE: 0.9847

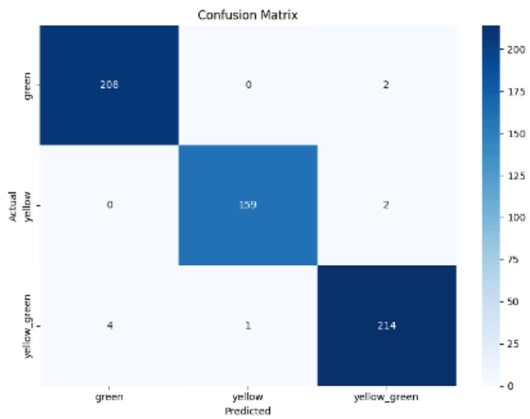


Fig. 6.10 EfficientNetV2-B3 Ripeness Confusion Matrix

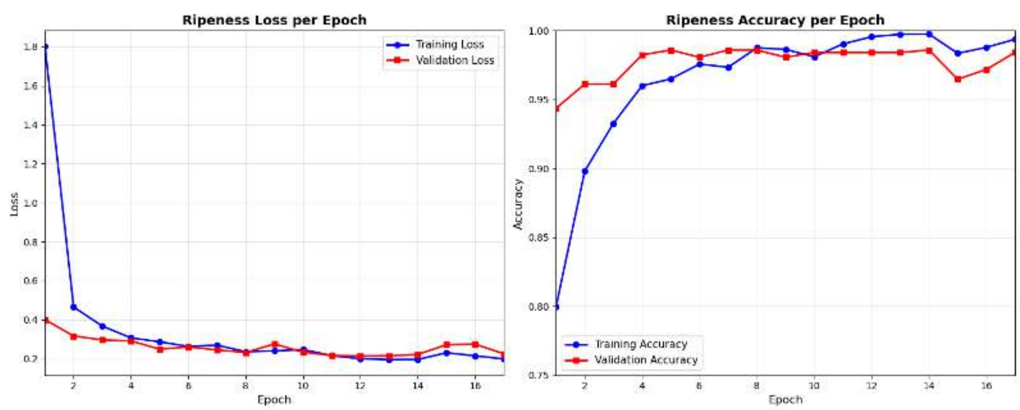


Fig. 6.11 EfficientNetV2-B3 Ripeness Accuracy and Loss Graph

6.2.4 Bruises Classification Results

6.2.4.1 CNN

For bruise classification, the final EfficientNetV2-B3 model also performed excellently. It achieved a test accuracy of 99%, with precision, recall, and F1-score near 0.989. The validation accuracy of 99.31% and training accuracy of 99.86% confirmed stability. These results demonstrate exceptional reliability and consistent performance across all dataset

De La Salle University

splits. The training configuration was refined to improve both computational efficiency and robustness. Batch size was increased to 60, fully utilizing available GPU memory capacity. This adjustment enhanced gradient stability and accelerated convergence across training epochs effectively. Regularization parameters were tuned with a dropout rate of 0.2 overall. A drop-path rate of 0.1 was also applied to further control overfitting. Together, these settings balanced high predictive accuracy with improved model generalization capability. Early stopping with a patience of 10 epochs was employed during training. This ensured meaningful improvement capture while avoiding unnecessary computation after convergence detection.

The confusion matrix in Figure 6.12 reinforces these excellent quantitative results clearly. The model correctly identified nearly all samples across both bruise categories tested. Only four false negatives and one false positive occurred in total predictions. This minimal error distribution illustrates a well-balanced and highly reliable classification profile. The model demonstrated strong sensitivity to bruised fruit and high specificity otherwise. Low false negatives are particularly important in postharvest quality control applications. Undetected bruises pose a major risk to maintaining consistent product quality standards. The low occurrence of such cases underscores the model's robustness and precision. These characteristics make EfficientNetV2-B3 ideal for deployment in real-time inspection systems.

The validation curves in Figure 6.13 further illustrate stable training convergence behavior. Validation accuracy rose rapidly during initial epochs and stabilized near 0.99 overall. Meanwhile, validation loss decreased sharply early on and then gradually leveled off. Minor fluctuations in loss reflect typical batch-level variations during optimization cycles. Despite these oscillations, accuracy remained consistently high and stable throughout training. This indicates that the network maintained strong confidence in its classification predictions. The inverse correlation between loss and accuracy confirms effective learning of features. These patterns demonstrate robust generalization and the absence of significant overfitting problems. Together, the curves validate that all applied optimizations improved convergence stability efficiently. EfficientNetV2-B3 thus combines exceptional accuracy, reliability, and computational efficiency effectively. This performance level establishes it as the optimal model for bruise classification. Its predictive precision makes it suitable for industrial-grade automated quality control systems.

	Precision	**Recall**	**F1**	**Support**
Bruised	1.00	0.98	0.99	206
Not Bruised	0.98	1.00	0.99	234
Accuracy			0.99	440
Macro Avg	0.99	0.99	0.99	440
Weighted Avg	0.99	0.99	0.99	440

TABLE 6.10 EFFICIENTNETV2-B3 BRUISES CLASSIFICATION REPORT WITH PRECISION: 0.9887, RECALL: 0.9886, F1 SCORE: 0.9886

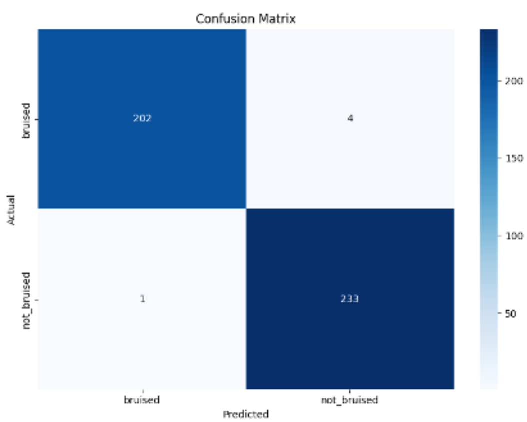


Fig. 6.12 EfficientNetV2-B3 Bruises Confusion Matrix

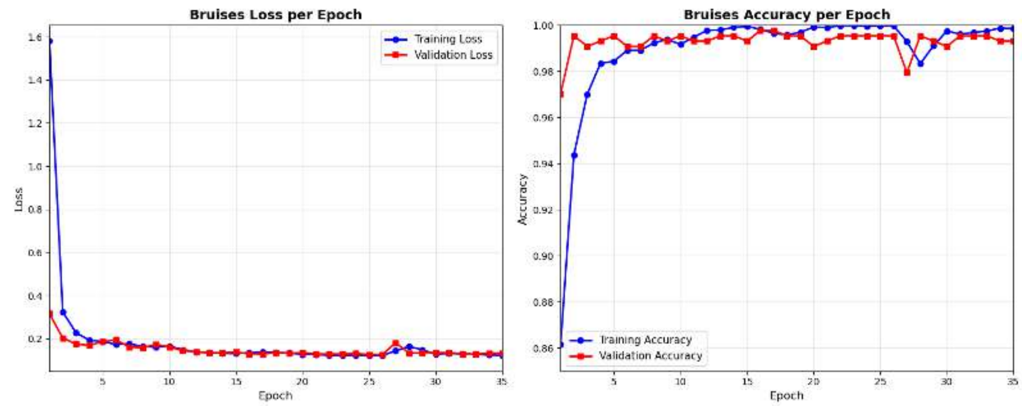


Fig. 6.13 EfficientNetV2-B3 Bruises Accuracy and Loss Graph

6.3 Comparative Analysis: Model Performance vs. Expert Benchmark

To establish a robust benchmark for model performance, a comparative analysis was conducted against the expert assessment of a qualified horticulturist. This section outlines the methodology for the expert evaluation and presents a comparative summary of the results.

6.3.1 Comparative Results

The expert's classifications for the 26 images randomly sampled from the dataset are presented in Table 6.13. These results serve as the validated ground truth against which the predictive accuracy of the computational models was measured. Note that terms g, yg, and y refer to the mango color categories: green, yellow-green, and yellow, respectively. Likewise, b and nb indicate bruised and non-bruised mango surfaces.

De La Salle University

TABLE 6.11 EXPERT CLASSIFICATION RESULTS FOR MANGO PHENOTYPIC TRAITS FROM PROFESSIONAL 1

Mango ID	Color Category		Bruising Status		Result
	Expert	Model	Expert	Model	
001	y	y	b	b	1
002	yg	yg	b	b	1
003	g	g	nb	nb	1
004	y	y	b	b	1
005	g	g	nb	nb	1
006	yg	yg	b	b	1
007	yg	y	nb	nb	0.5
008	g	g	nb	nb	1
009	y	y	b	b	1
010	yg	yg	nb	nb	1
011	yg	yg	b	b	1
012	yg	yg	b	b	1
013	g	g	nb	nb	1
014	yg	yg	b	b	1
015	g	g	b	b	1
016	yg	yg	nb	nb	1

Continued on next page

Table 6.11 – continued from previous page

Mango ID	Color Category		Bruising Status		Result
	Expert	Model	Expert	Model	
017	yg	yg	nb	nb	1
018	yg	yg	b	nb	0.5
019	yg	yg	nb	nb	1
020	g	g	nb	nb	1
021	yg	yg	b	b	1
022	g	g	nb	nb	1
023	y	y	b	b	1
024	yg	yg	b	b	1
025	yg	yg	b	b	1
026	g	g	nb	nb	1

TABLE 6.12 EXPERT CLASSIFICATION RESULTS FOR MANGO PHENOTYPIC TRAITS FROM PROFESSIONAL 2

Mango ID	Color Category		Bruising Status		Result
	Expert	Model	Expert	Model	
001	y	y	b	b	1
002	yg	yg	b	b	1

Continued on next page

Table 6.12 – continued from previous page

Mango ID	Color Category		Bruising Status		Result
	Expert	Model	Expert	Model	
003	g	g	nb	nb	1
004	y	y	b	b	1
005	g	g	nb	nb	1
006	y	yg	b	b	0.5
007	yg	y	nb	nb	0.5
008	g	g	nb	nb	1
009	y	y	b	b	1
010	yg	yg	nb	nb	1
011	yg	yg	b	b	1
012	yg	yg	nb	b	0.5
013	g	g	nb	nb	1
014	yg	yg	b	b	1
015	g	g	b	b	1
016	yg	yg	nb	nb	1
017	yg	yg	nb	nb	1
018	yg	yg	b	nb	0.5
019	yg	yg	nb	nb	1
020	g	g	nb	nb	1
021	yg	yg	b	b	1

Continued on next page

Table 6.12 – continued from previous page

Mango ID	Color Category		Bruising Status		Result
	Expert	Model	Expert	Model	
022	g	g	nb	nb	1
023	y	y	b	b	1
024	y	yg	b	b	0.5
025	yg	yg	b	b	1
026	yg	g	nb	nb	0.5

TABLE 6.13 EXPERT CLASSIFICATION RESULTS FOR MANGO PHENOTYPIC TRAITS FROM PROFESSIONAL 3

Mango ID	Color Category		Bruising Status		Result
	Expert	Model	Expert	Model	
001	y	y	b	b	1
002	y	yg	b	b	0.5
003	yg	g	nb	nb	0.5
004	y	y	b	b	1
005	g	g	nb	nb	1
006	y	yg	b	b	0.5
007	y	y	nb	nb	1

Continued on next page

Table 6.13 – continued from previous page

Mango ID	Color Category		Bruising Status		Result
	Expert	Model	Expert	Model	
008	g	g	nb	nb	1
009	y	y	b	b	1
010	yg	yg	nb	nb	1
011	yg	yg	b	b	1
012	yg	yg	b	b	1
013	g	g	nb	nb	1
014	yg	yg	b	b	1
015	g	g	b	b	1
016	yg	yg	nb	nb	1
017	yg	yg	nb	nb	1
018	yg	yg	b	nb	0.5
019	yg	yg	nb	nb	1
020	g	g	nb	nb	1
021	yg	yg	b	b	1
022	g	g	nb	nb	1
023	y	y	b	b	1
024	y	yg	b	b	0.5
025	yg	yg	b	b	1
026	yg	g	nb	nb	0.5

After compiling the scores, the model achieved an overall score of 71 out of 78. This translates to a 91.02% accuracy rate, meaning the model's answers were correct 91.02% of the time when compared to the mango expert's benchmark.

It is important to note that the expert's grading was conducted independently and consecutively, without external guidance or tools to aid their judgment. This purely human evaluation, while authoritative, inevitably introduces a degree of inherent human error.

6.4 Size Determination Results

6.4.1 Actual and Estimated Length

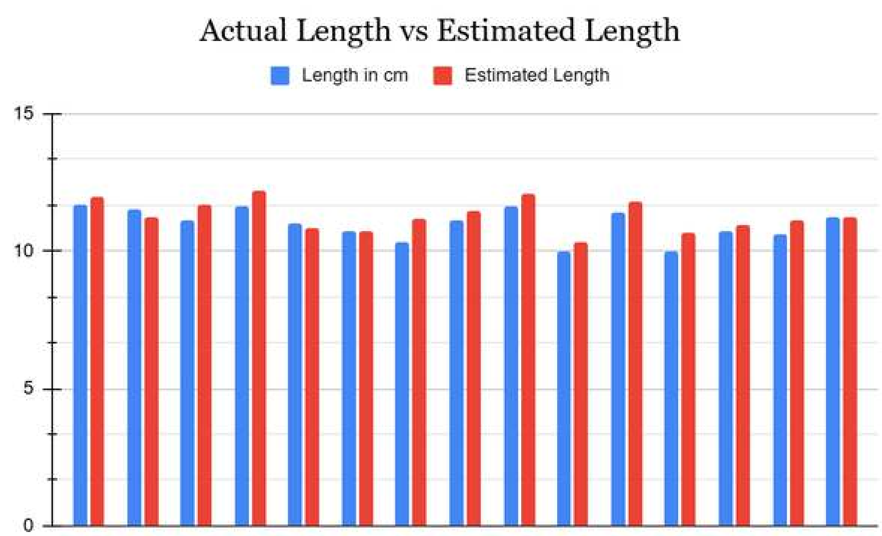


Fig. 6.14 Bar Graph of Actual vs Estimated Length

Starting off for size determination, the method for measuring length achieved an average error of 3.41% with a median of 3.15% and a standard deviation of 0.02, showing that length estimation was highly consistent and tightly clustered around the mean. Most mangoes exhibited differences below 5%, with only a few samples such as Mango 3 and Mango 4 exceeding this threshold as seen on Figure 6.14. These deviations were primarily due to bounding box approximation, where slight misalignment of contours led to overestimation. The low variability demonstrates that the code reliably captures mango length, and the small errors are unlikely to affect classification outcomes

6.4.2 Actual and Estimated Width

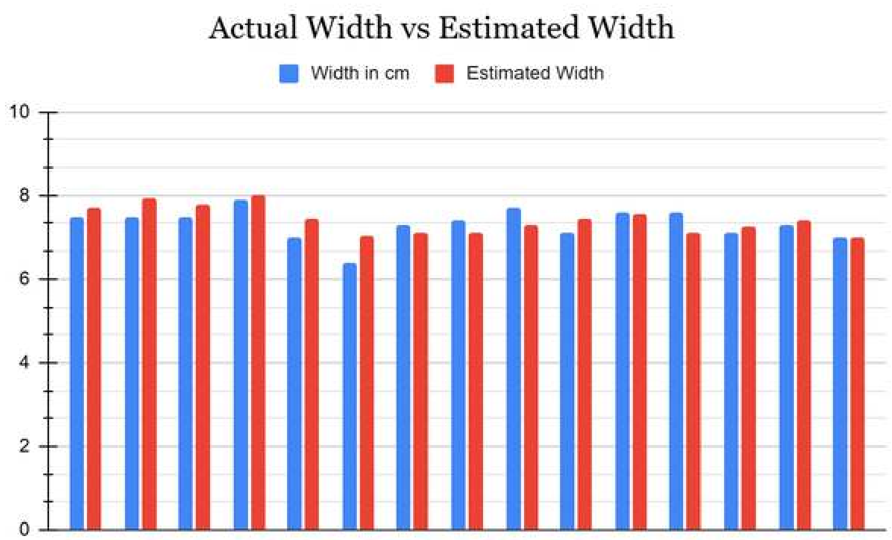


Fig. 6.15 Bar Graph of Actual vs Estimated Width

For width, the average error was 3.81%, the median was 3.92%, and the standard deviation was 0.03, reflecting slightly higher variability compared to length but still within a stable range as seen on Figure 6.15. Most mangoes showed differences between 2–6%, though Mango 6 was a clear outlier with a width error of 9.67%, which inflated the overall variability. This error was likely caused by segmentation inconsistencies at the fruit edges, where the HSV mask occasionally included background pixels or missed portions of the mango contour. Despite this, the majority of samples demonstrated stable width estimation, confirming that the method is effective but sensitive to segmentation accuracy.

6.4.3 Calculated Area and Estimated Area

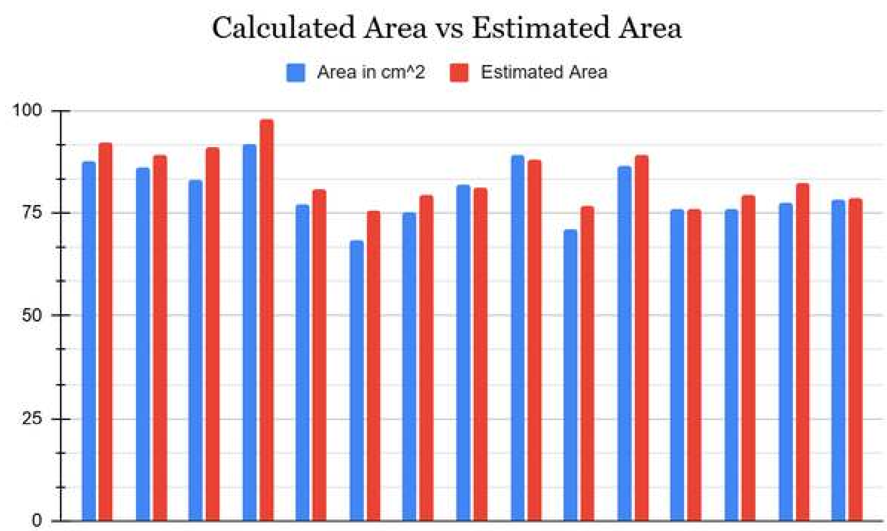


Fig. 6.16 Bar Graph of Actual vs Estimated Area

For area, which is the most critical parameter for size classification, the code produced an average error of 4.51%, a median of 4.83%, and a standard deviation of 0.03, indicating consistent performance across the dataset. Most mangoes were measured within a 2–6% difference, with Mango 15 showing nearly perfect agreement at 0.14% error as seen on Figure 6.16. Larger deviations were observed in Mango 3 and Mango 6, where area errors reached 8–9%, primarily due to compounding effects of length and width misestimation. These results highlight that while area estimation is generally reliable, boundary cases near classification thresholds may be more prone to misclassification. Nonetheless, the overall accuracy demonstrates that the code is effective for non-destructive mango grading, with error margins well within acceptable tolerance.

6.4.4 Summarized Size Results

Mango Index	Length in cm	Width in cm	Area in cm^2	Weight in g	Estimated Length	Estimated Width	Estimated Area	Length % Difference	Width % Difference	Area % Difference
1	11.7	7.5	87.75	295.1	11.96	7.7	92.092	2.20%	2.63%	4.83%
2	11.5	7.5	86.25	296.2	11.24	7.93	89.1332	2.29%	5.57%	3.29%
3	11.1	7.5	83.25	286.2	11.66	7.8	90.948	4.92%	3.92%	8.84%
4	11.6	7.9	91.64	268.2	12.21	8.01	97.8021	5.12%	1.38%	6.51%
5	11	7	77	270.5	10.85	7.45	80.8325	1.37%	6.23%	4.86%
6	10.7	6.4	68.48	231.1	10.72	7.05	75.576	0.19%	9.67%	9.85%
7	10.3	7.3	75.19	231.1	11.16	7.11	79.3476	8.01%	2.64%	5.38%
8	11.1	7.4	82.14	236.9	11.45	7.11	81.4095	3.10%	4.00%	0.89%
9	11.6	7.7	89.32	245.6	12.08	7.3	88.184	4.05%	5.33%	1.28%
10	10	7.1	71	237.2	10.32	7.45	76.884	3.15%	4.81%	7.96%
11	11.4	7.6	86.64	303.1	11.77	7.57	89.0989	3.19%	0.40%	2.80%
12	10	7.6	76	232.2	10.66	7.11	75.7926	6.39%	6.66%	0.27%
13	10.7	7.1	75.97	243	10.93	7.26	79.3518	2.13%	2.23%	4.35%
14	10.6	7.3	77.38	236.1	11.14	7.41	82.5474	4.97%	1.50%	6.46%
15	11.2	7	78.4	235.3	11.2	7.01	78.512	0.00%	0.14%	0.14%
Average	10.97	7.33	80.43	256.52	11.29	7.42	83.83	3.41%	3.81%	4.51%
SD	0.57	0.37	6.89	27.07	0.56	0.33	6.84	0.02	0.03	0.03
Median	11.1	7.4	78.4	243	11.2	7.41	81.4095	3.15%	3.92%	4.83%

Fig. 6.17 List of Size Results

Overall, based on Figure 6.17, the data shows that the mango size determination code produced results that were consistently close to manual caliper measurements across the 15-sample dataset. The average error margins of 3.41 % for length, 3.81% for width, and 4.51% for area, combined with very low standard deviations of 0.02, 0.03, and 0.03 respectively, indicate that the system maintained stable accuracy with minimal variability. Most mangoes fell within a 2–6% difference, which is acceptable for practical grading, while only a few outliers exceeded 8–9% error. Likewise, the small, medium, and large classification with a 3cm^2 is shown in Figure 6.18 and the more than 40 mangoes can be found on Figure 6.19. These findings confirm that the methodology is effective for non-destructive mango sizing and classification, with errors generally small and consistent across samples.

Size Classification	Area per cm^3
Large	Area > 101
Medium	88 < Area < 98
Small	Area < 85

Fig. 6.18 Size Area Classification with cm^2 Gap

Weight (g)	Length	Width
260.8	11.8	7.8
299.4	12.8	7.8
238.4	11.4	7.6
335.6	13.8	10.5
272.4	12.9	8.5
267.9	13.1	8.2
274	12.6	8.2
272.3	13.3	8
281.6	13	8
286.2	13.8	8
284.6	12.6	9
265.7	13.3	8
278.1	13	7.6
263.8	12.9	7.5
222	12.1	7.8
240.1	13.5	8.2
290.7	13.5	8.5
260.1	12.8	8
253.6	12.9	7.5
225.9	12	7.5
301.2	11.8	7.8
291.4	11.3	7
239.1	10.8	6.5
277	10.8	6.4
260.1	10.1	6.7
272.3	11	7
304.3	10.8	7.1
295.1	11.7	7.5
296.2	11.5	7.5
286.2	11.1	7.5
268.2	11.6	7.9
270.5	11	7
231.1	10.7	6.4
231.1	10.3	7.3
236.9	11.1	7.4
245.6	11.6	7.7
237.2	10	7.1
303.1	11.4	7.6
232.2	10	7.6
243	10.7	7.1
236.1	10.6	7.3
235.3	11.2	7

Fig. 6.19 Tested 42 Mangoes

6.5 Formula with User Priority

The Figures 6.20, 6.21 and 6.22 are explained in this section where the inputted weight values are all real number since negative and imaginary number are not allowed. The purpose of this section is to demonstrate the different possible cases of using the zero value in the user priority.

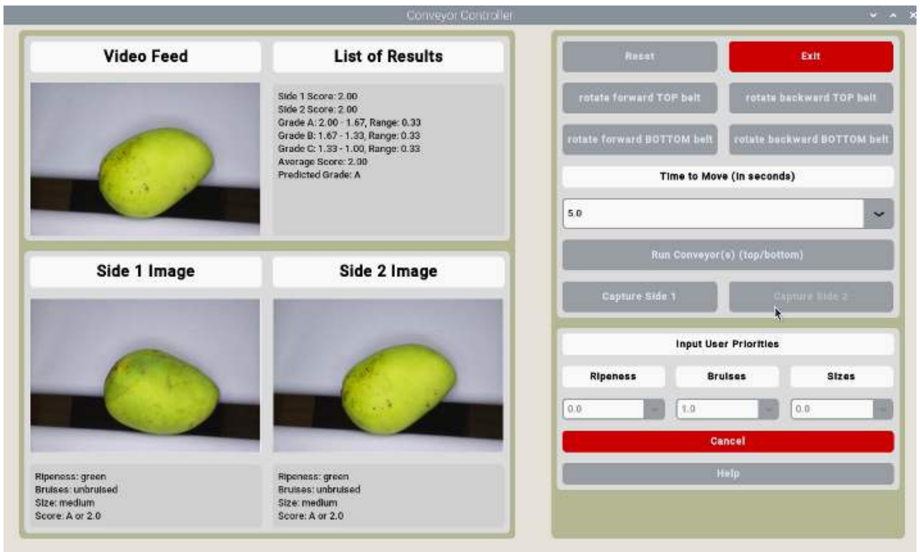


Fig. 6.20 Only Bruises as a None Zero Value

An example of where the user only prioritizes bruises is shown on Figure 6.20. This implies that the user disregards the ripeness and the size of the Carabao mangoes by setting the input priority value to zero.

Another example shown on Figure 6.21 shows where the user only prioritized two mango characteristics which are the bruises and the ripeness. This is because the user set the size to zero. As such when grading the mangoes, it would still show the prediction of the size however when grading the Carabao mango it would disregard the size in its calculation.

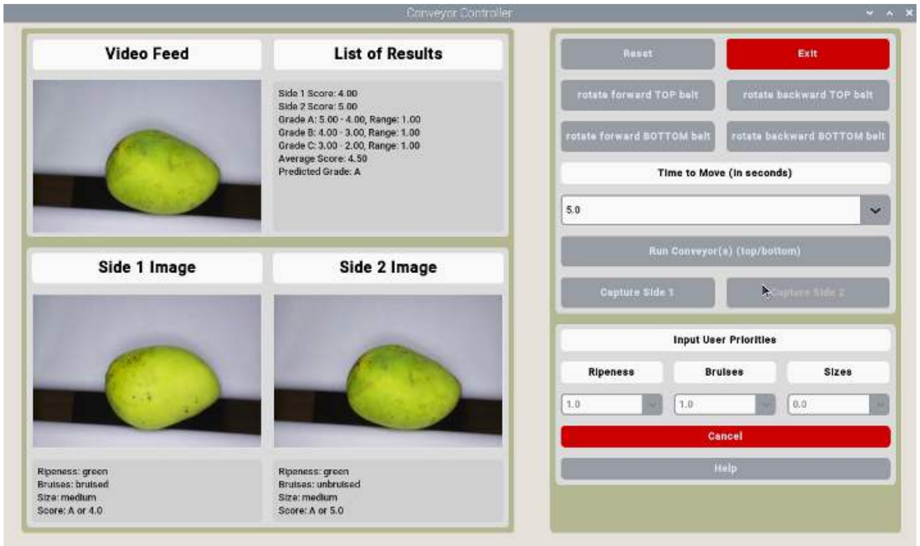


Fig. 6.21 Only Ripeness and Bruises as a None Zero Value

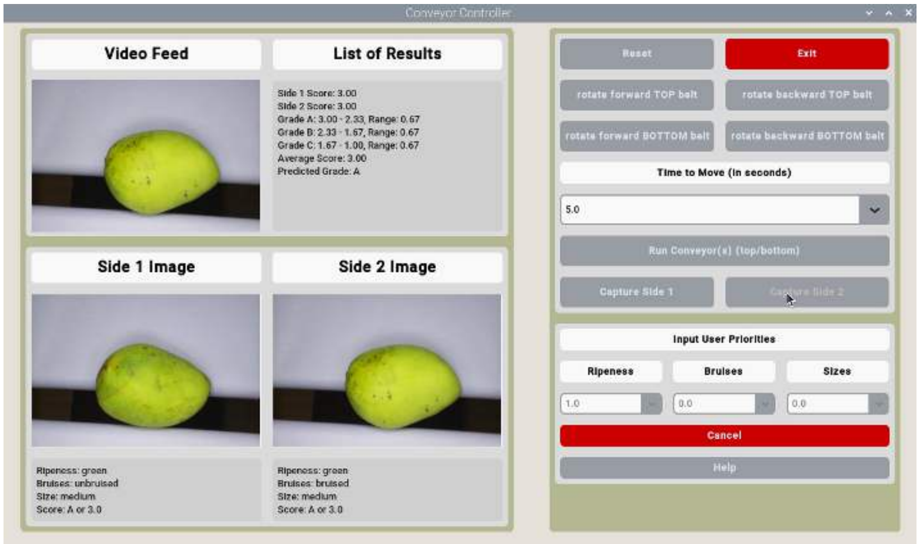


Fig. 6.22 Only Ripeness as a None Zero Value

Another similar user priority input to Figure 6.20 is Figure 6.22 where it only prioritizes one parameter which is the ripeness. Furthermore, notice the range of values for each grade has a maximum of 3.00 and a minimum of 1.00. This is because the input weight of the ripeness is 1.0 meaning that the possible values are 1.00, 2.00, and 3.00.

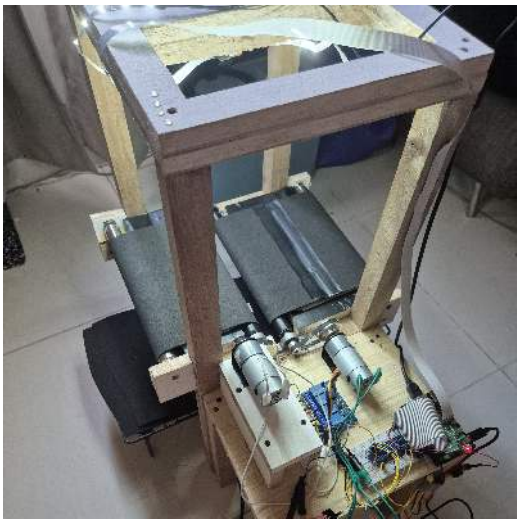
6.6 Physical Prototype

6.6.1 Version 1: Barebone with Black Conveyor Sheets

For the physical prototype, there are two main parts which are the image acquisition system and the conveyor belt. Both of these parts are being controlled by a RPi through a python script. Note that the DC motors, 4 channel relay, and camera can be seen on Figure 6.24. For the first version of the prototype, Figure 6.23 shows three images which are the top view, entrance view of the Carabao mangoes and the side view of the prototype. Notice that it is a barebone prototype made out of plywood with four rollers and black matte sheets for moving the Carabao mangoes. There are two DC motors controlling each conveyor belt. As seen on the side of the prototype on Figure 6.23, the black sheet is not flexible and too stiff to be able to move it with the mangoes. This means that the conveyor belt would not be able to rotate and move the Carabao mangoes consistently.

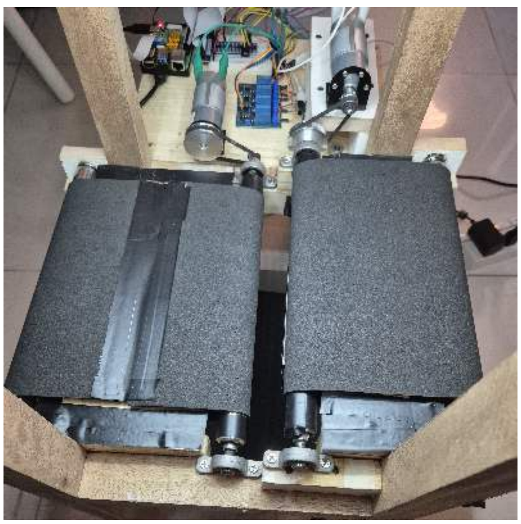
6.6.2 Version 2: Enclosed with White Conveyor Sheets and Physical Sorter

For the second version of the prototype as seen on Figure 6.25, improvements such as replacing the black sheet to a white sheet which improved the efficiency and reduced the



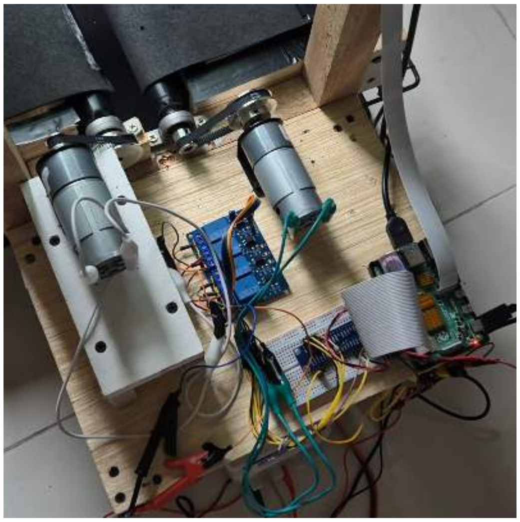
(a) Prototype Top View

(b) Entrance Conveyor Belt View

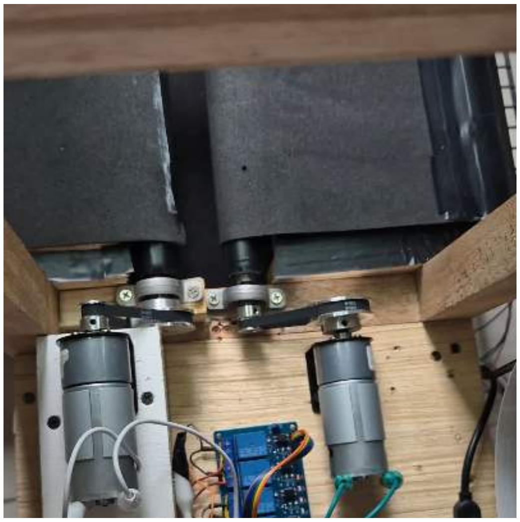


(c) Side Conveyor Belt View

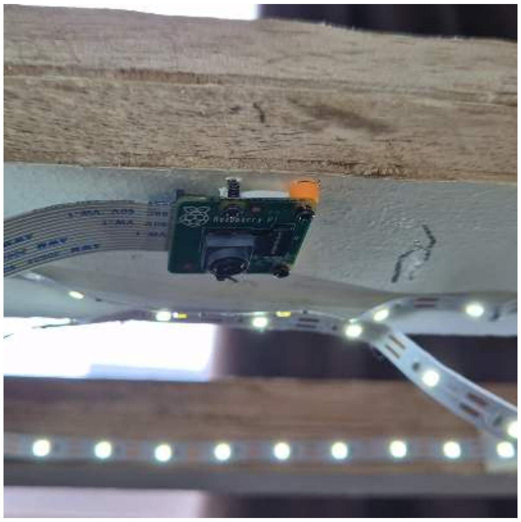
Fig. 6.23 Version 1 of the Prototype



(a) Prototype Main Hardware



(b) DC Motor and Pulley



(c) LED Lights and Camera Module

Fig. 6.24 Hardware View

frequency of requiring maintenance. Another improvement for this version is enclosing the electronic devices in a container. This helps protect it from unwanted liquid spills. For the sorting of mangoes, the conveyors would sort it into three grades which are Grade A, B, and C. It would first go through the longest conveyor and the shorter conveyor depending on the grade. This is because if the Grade is A (which is the highest), then it would exit to the east of the prototype and not go through the shorter conveyor belt. For Grade B, it would go through the west side and then north of the prototype. Finally for grade C, it would go through west side and then south of the prototype. The code for this can be seen on Listing 6.1.

Listing 6.1: Sorting the Mangoes

```
if ave_letter.upper()  == 'A':
    button_state_array = [0, 1, 0, 0]
    print(button_state_array)
    self.sort.set_motors(button_state_array)
elif ave_letter.upper() == 'B':
    button_state_array = [1, 0, 1, 0]
    print(button_state_array)
    self.sort.set_motors(button_state_array)
elif ave_letter.upper() == 'C':
    button_state_array = [1, 0, 0, 1]
    print(button_state_array)
    self.sort.set_motors(button_state_array)
```

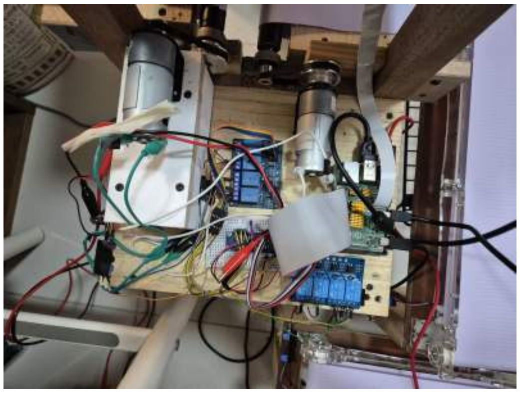
6.7 Software Application

6.7.1 Version 1: Progress Bar with Black Conveyor Sheets

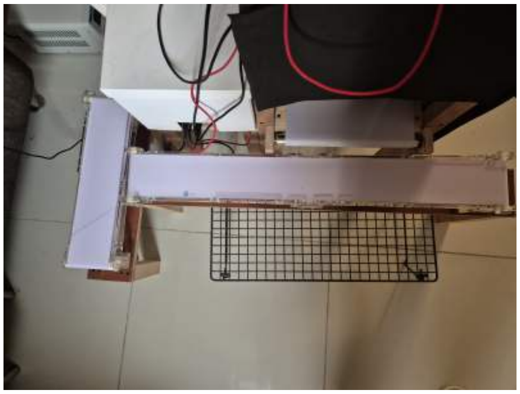
For the software application inside the RPi, CustomTkinter is used as the main GUI for the python application. For the versions, there are two main versions. The first version which involves a fully automated capturing of both sides of the Carabao mango and the second

(a) Side View of Improved Prototype

(b) Top View of Improved Prototype



(c) Inside Hardware View

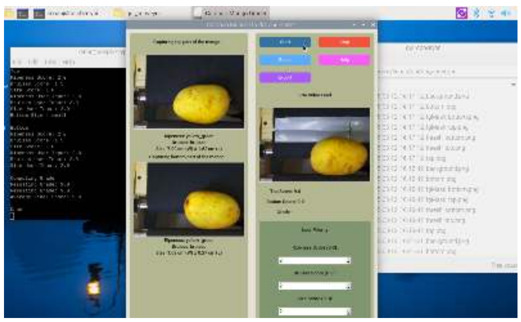


(d) Sorting Mangoes Using Two Conveyor Belts

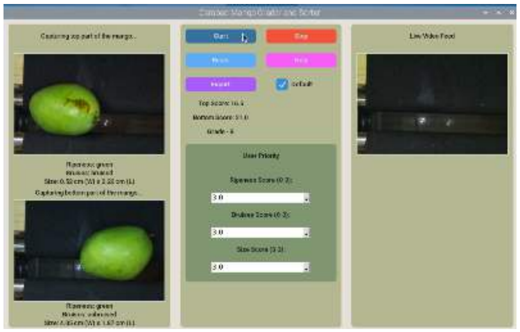
Fig. 6.25 Version 2: Improved Prototype

version which uses a part by part picturing and moving of mangoes.

For this version, some of the initial UI design are shown on Figure 6.26. There are two three main columns which are the live video feed with a progress bar, two sides of the mango cheek, and the control panel with the different buttons such as the user priority, and reset, stop, export, and help. The approach to this one involves fully automatically moving and grading the mango which caused the grading to be inconsistent because it was not able to fully rotate the mango at most cases.



(a) Version 1.1



(b) Version 1.2

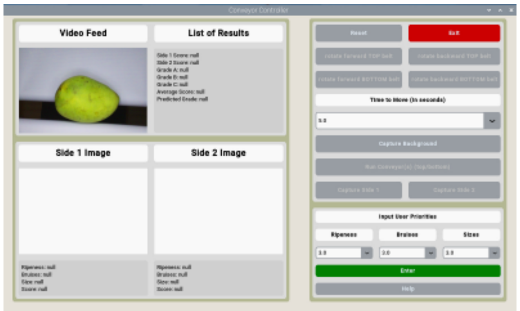
(c) Version 1.3

Fig. 6.26 Version 1 of the RPi's User Interface

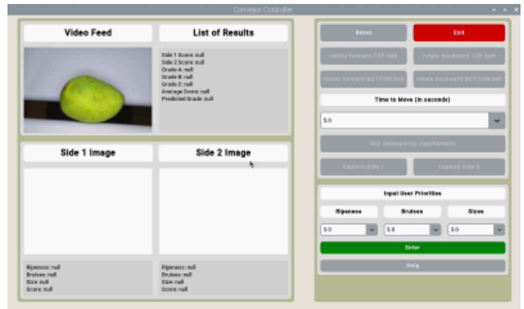
6.7.2 Version 2: Improved UI without Progress Bar

For the second version of the software as seen on Figure 6.27, an overhaul of the UI design was done with the hopes that it would be cleaner and intuitive. Some features such as the progress bar was removed because this method uses a step by step approach for rotating the mango where the user would rotate it using the buttons and how long they want to move the conveyors. Likewise, the stop buttons for all the conveyors are added.

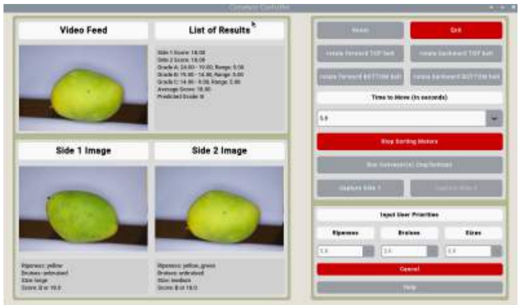
De La Salle University



(a) Version 2.1 with Background Image



(b) Version 2.2 without Background Image



(c) Version 2.3 with Stop Sorting Button

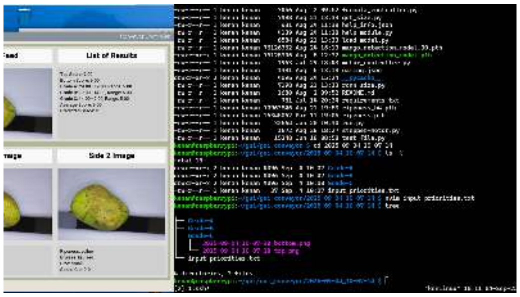
Fig. 6.27 Version 2 of the RPi's User Interface

6.7.3 Mango Image Sorting

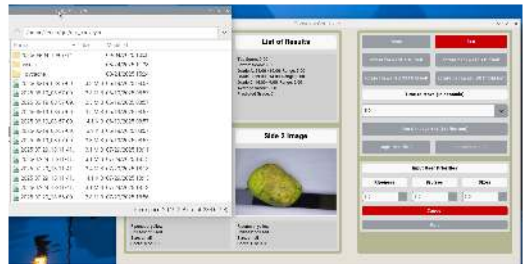
Figure 6.28 shows the method sorting the mango images through a directory containing the year, date, and time. Likewise, inside that directory, is the three possible grades from A to C and the input priorities of the user.

6.7.4 Error Handling

Figure 6.29 shows the three possible error messages when the user inputs all zero in the user priority, presses all and none of the buttons when moving the conveyor. In the case the user inputs a a letter or negative value, then the not number error message would pop up as



(a) Folder Tree Directory of Each Grade



(b) Directory with Year, Date, and Time

(c) Saved Input Priority/Weights

Fig. 6.28 Mango Image Data Sorting

shown in Figure 6.30.

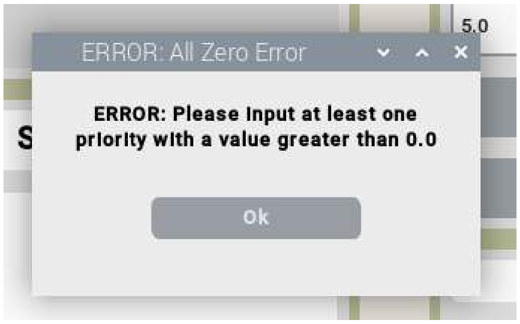
6.7.5 Sample UI Outputs

Figure 6.31 shows the help page containing information about the button and their purpose to assist the user navigate and utilizing the application. Furthermore, Figure 6.32 shows an example output for each possible case of green, yellow_green, and yellow ripeness classification together with bruise and not bruised and small and medium size mangoes.

6.8 Summary

This chapter shows its successful integration of software intelligence, hardware functionality, and user-centric design. The core of the system's success lies in its high-precision deep learning models, with the final EfficientNetV2-B3 architecture achieving exceptional accuracies of 98% for ripeness classification and 99% for bruise detection. Through extensive

De La Salle University



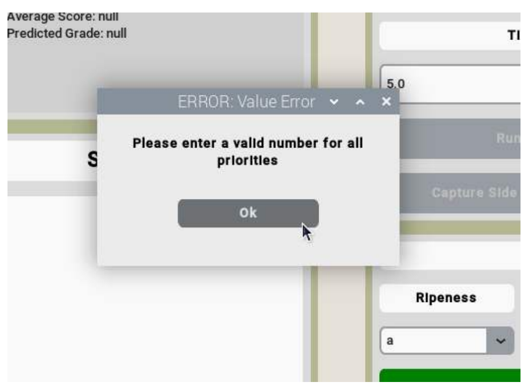
(a) All Zero Error

(b) Input Error

(c) Null Button Error

Fig. 6.29 Error Messages

(a) Not Number at Conveyor Time



(b) Not Number at Priority

Fig. 6.30 Error message for Letter as Input

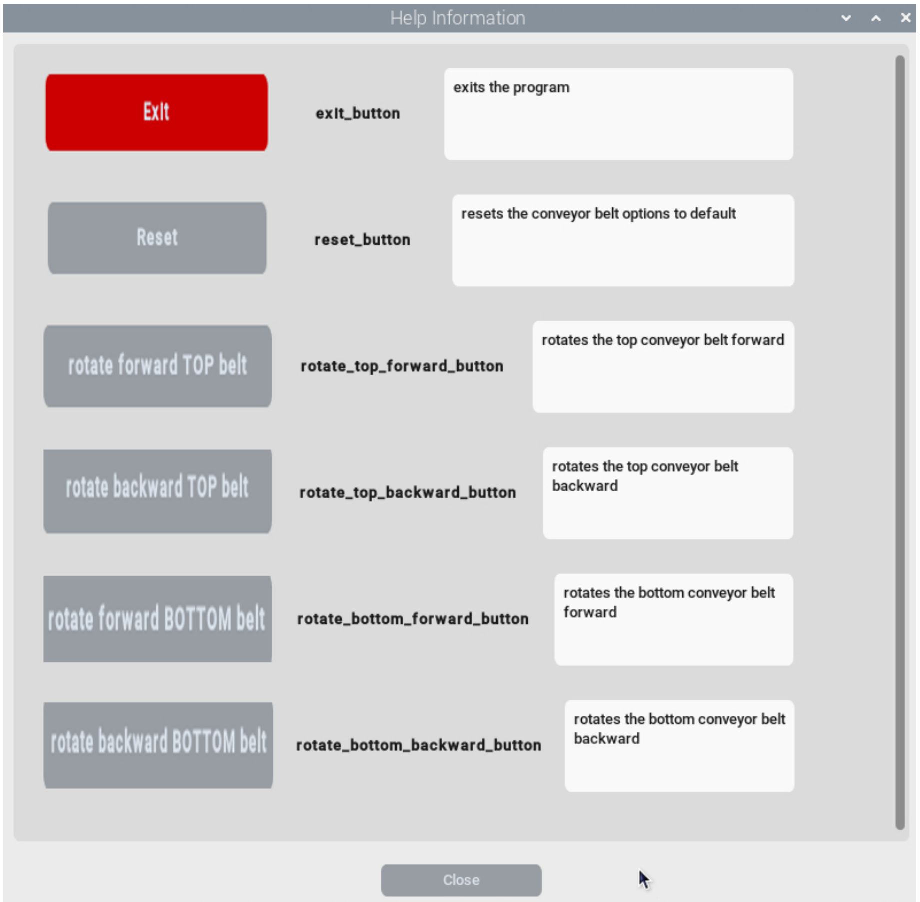
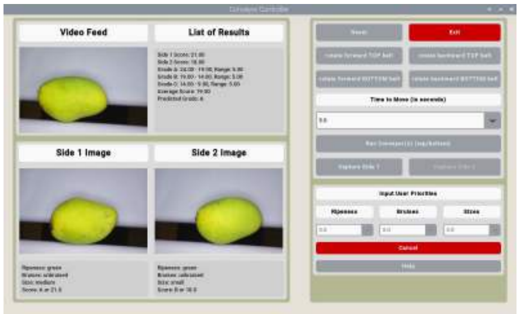
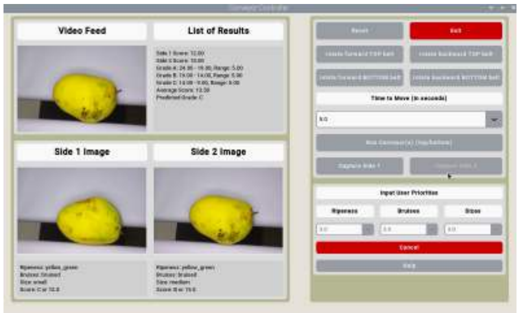


Fig. 6.31 Help Page UI

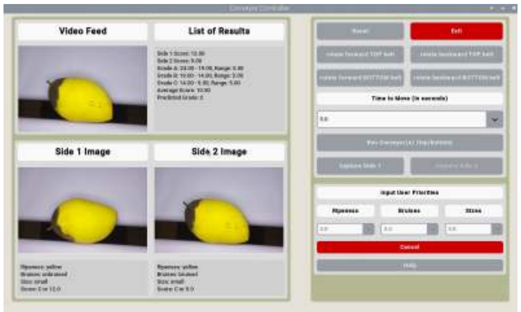
benchmarking, modern CNNs like EfficientNet were proven superior, offering an optimal balance of accuracy and computational efficiency. The system is able to get an overall percent difference to measured area of 4.8 for the size. The system's practical validity was further confirmed through a comparative analysis with a human expert, achieving a 79% agreement rate, which accounts for the inherent subjectivity of manual grading. This robust software is embodied in a functional physical prototype that evolved into a refined version with an efficient conveyor system and a fully enclosed, three-way sorting mechanism that accurately directs mangoes into designated grades. Controlling this hardware is an



(a) Green and Unbruised



(b) Yellow_Green and Bruised



(c) Yellow and Bruised

Fig. 6.32 Sample Ripeness and Bruises Results

intuitive software application on the Raspberry Pi, featuring a user-friendly interface that allows for custom priority weighting of mango characteristics and includes comprehensive error handling and data logging. Overall, the results conclusively show that the research has successfully bridged the gap between theoretical model development and a practical, deployable system capable of automatically and accurately grading Carabao mangoes based on customizable, user-defined standards.

Chapter 7

CONCLUSIONS, RECOMMENDATIONS, AND FUTURE DIRECTIVES

De La Salle University

7.1 Concluding Remarks

In this Thesis, the prototype is successful in grading and sorting Carabao mangoes based on the user priority and machine learning algorithm. More specifically, the prototype is successful in classifying Carabao mangoes based on ripeness (Green, Green Yellow, and Yellow), size (Large, Medium, Small), and bruises (bruised and not bruised).

7.1.1 Objectives Achieved

7.1.1.1 GO: To develop a user-priority-based grading and sorting system for Carabao mangoes, using machine learning and computer vision techniques to assess ripeness, size, and bruises.

For GO, the study successfully developed a user-priority-based grading and sorting system for Carabao mangoes by integrating machine learning and computer vision techniques to assess ripeness, size, and bruises. The system achieved high accuracy and reliability while maintaining a non-destructive process through its hardware and software integration using a Raspberry Pi platform.

7.1.1.2 SO1: To make an image acquisition system with a conveyor belt for automatic sorting and grading mangoes.

For SO1, the researchers designed and implemented an automated image acquisition system consisting of a Raspberry Pi 4, camera module, LED lighting, and a conveyor belt, which ensured consistent lighting and image alignment necessary for precise visual analysis and classification.

7.1.1.3 SO2: To get the precision, recall, F1 score, confusion matrix, and train and test accuracy metrics for classifying the ripeness and bruises with an accuracy score of at least 90%.

For SO2, multiple models were trained and evaluated, with EfficientNetV2 achieving precision, recall, and F1 scores of approximately 0.98 and accuracy above 98%, which surpassed the target performance threshold and validating the effectiveness of the selected machine learning architecture.

7.1.1.4 SO3: To create a microcontroller-based system to operate the image acquisition system, control the conveyor belt, and process the mango images through machine learning.

For SO3, a microcontroller-driven setup using the Raspberry Pi was developed to synchronize conveyor movement, image capture, and data processing, demonstrating a fully automated and self-contained embedded system capable of real-time classification.

7.1.1.5 SO4: To grade mangoes based on user priorities for size, ripeness, and bruises.

For SO4, the grading module incorporated a linear weighting formula that allowed users to assign priority values to ripeness, bruises, and size, effectively producing customizable grading outcomes that reflected user-defined criteria and market standards.

7.1.1.6 SO5: To classify mango ripeness based on image data using machine learning algorithms such as kNN, k-mean, and Naïve Bayes.

For SO5, various algorithms were implemented and tested, with CNN-based EfficientNetV2 outperforming traditional classifiers, achieving 98% accuracy in categorizing mango ripeness into green, yellow-green, and yellow stages based on color and texture features.

7.1.1.7 SO6: To classify mango size based on image data by getting its length and width using OpenCV, geometry, and image processing techniques.

For SO6, the system utilized OpenCV with an average percent difference of 4.8% in area measurement.

7.1.1.8 SO7: To classify mango bruises based on image data by employing machine learning algorithms.

For SO7, the implemented CNN models effectively detected and classified visible surface bruises, achieving a 99% accuracy rate and demonstrating robustness in identifying varying bruise intensities under controlled lighting conditions.

7.2 Contributions

The contributions of each group member are as follows:

- BANAL Kenan A.: Scrum Master (Project manager in charge of the hardware and software integration, assisted in mango size determination, incharge of dataset collection and data augmentation, assisted in mango size determination and estimation)

- BAUTISTA Francis Robert Miguel F.: Front End Engineer (UI/UX Designer in charge of software interface and hardware assistant of the Scrum Master, assisted in dataset splitting, categorization and collectio, assisted in mango size determination and estimation)

- HERMOSURA Don Humphrey L. : Back End Engineer (in charge of mango size determination, assisted in machine learning algorithm)

- SALAZAR Daniel G.: Product Engineer (Software Engineer in charge of training and testing of the machine learning algorithm, assisted in dataset collection and data augmentation, assisted in mango size determination and estimation)

7.3 Recommendations

The researchers recommend that the prototype be enhanced through the optimization of the machine learning algorithm and the refinement of the hardware design, with further testing in the actual grading and sorting of Carabao mangoes in the market. To strengthen its performance and reliability, the improvements should include expanding the dataset by incorporating a larger number of mango samples, utilizing the latest available machine learning models at the time of implementation, and updating the size determination method by applying recent advancements in machine learning and computer vision. In addition, the prototype should integrate weight determination through the use of weight sensors, implement a non-destructive technique for sweetness detection, and employ industry-grade equipment to ensure robust implementation and automation.

7.4 Future Prospects

Future researchers may consider the following recommendations for future work:

1. User testing of the prototype in the actual grading and sorting of Carabao mangoes in the Philippine market.

2. Additional of weight measurement to the prototype to improve the grading and sorting of Carabao mangoes.

3. Integration of a custom PCB to improve the hardware design of the prototype.

REFERENCES

Abbas, Q., Niazi, S., Iqbal, M., and Noureen, M. (2018). Mango Classification Using Texture & Shape Features. *IJCSNS International Journal of Computer Science and Network Security*, 18(8).

Abu, M., Olympio, N. S., and Darko, J. O. (2021). Determination of Harvest Maturity for Mango Mangifera indica L. Fruit by Non-Destructive Criteria. *Agricultural Sciences*, 12(10):1103–1118.

Adam, J. A. P., Dato, K. S., Impelido, M. C. D., Tobias, R. G., and Pilueta, N. U. (2022). Non-destructive microcontroller-based carabao mango ripeness grader. Far Eastern University (FEU) College of Engineering.

Alejandro, A. B., Gonzales, J. P., Yap, J. P. C., and Linsangan, N. B. (2018). Grading and sorting of Carabao mangoes using probabilistic neural network. Bandung, Indonesia.

Amna, M. W. A., Guiqiang, L., and Muhammad Zuhaib AKRAM, M. F. (2023). Machine vision-based automatic fruit quality detection and grading. *Frontiers of Agricultural Science and Engineering*.

Badali, A. P., Zhang, Y., Carr, P., Thomas, P. J., and Hornsey, R. I. (2005). Scale factor in digital cameras. *Vision Sensor Laboratory, York University / Topaz Technology Inc. (via ResearchGate)*. Discusses the relation between object size and image size via scale factor.

Bayogan, E. R. and Secretaria, L. (2019). Enhancing mango fruit quality in asian mango chains: Case study – the philippines. Technical report, Australian Centre for International Agricultural Research (ACIAR), Canberra, Australia.

Bureau of Agriculture and Fisheries Product Standards (2004). Philippine national standard: Fresh fruits – mangoes – specification. Technical Report PNS/BAFPS 13:2004, Department of Agriculture, Manila, Philippines. Philippine National Standard for Fresh Fruits – Mangoes (Mangifera indica Linn.).

Bureau of Agriculture and Fisheries Product Standards (2006). Illustrative guide for the philippine national standard (pns/bafps 13:2004) – mangoes. Technical report, Department of Agriculture, Manila, Philippines. Supplementary visual guide for PNS/BAFPS 13:2004, detailing classification, defects, and grading criteria for fresh mangoes.

Centino, M. F., Castano, M. C. N., and Ebo, J. B. F. (2020). The Current Status of Philippine Mango in the Global Value Chain.

Chakraborty, S. K., Subeesh, A., Dubey, K., Jat, D., Chandel, N. S., Potdar, R., Rao, N. G., and Kumar, D. (2023). Development of an optimally designed real time automatic citrus fruit grading-sorting machine leveraging computer vision-based adaptive deep learning model. *Engineering Applications of Artificial Intelligence*, 120:105826.

D'Adamo, G. (2018). The determinants of export quality in the euro area. *Quarterly Report on the Euro Area (QREA)*, 17(1):23–31. Publisher: Directorate General Economic and Financial Affairs

(DG ECFIN), European Commission.

de Souza-Pollo, A. and de Goes, A. (2009). Fruit diseases. In Litz, R. E., editor, *The Mango: Botany, Production and Uses*, pages 169–190. CABI Publishing. Covers descriptions of mango diseases including anthracnose, stem-end rot, etc.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Guillergan, G., Sabay, R., Madrigal, D., and Bual, J. (2024). Naive Bayes Classifier in Grading Carabao Mangoes. *Technium: Romanian Journal of Applied Sciences and Technology*, 22:14–32.

Guillermo, M. C. S., Naciongayo, D. S., and Galela, M. G. C. (2019). Determining 'Carabao' Mango Ripeness Stages Using Three Image Processing Algorithms.

Guo, C. and et al. (2024). Cross entropy versus label smoothing: A neural collapse perspective. *arXiv preprint*.

Hennig, C. and Viroli, C. (2013). Quantile-based classifiers. *Biometrika*.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. (2017). Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269.

Huang, G., Sun, Y., Liu, Z., Sedra, D., and Weinberger, K. Q. (2016). Deep networks with stochastic depth. In *European conference on computer vision*, pages 646–661. Springer.

Hussein, B. M. and Shareef, S. M. (2024). An empirical study on the correlation between early stopping patience and epochs in deep learning. *ITM Web of Conferences*, 64(12):01003.

Kadam, J., Shimpi, S., Biradar, R., and Chate, S. (2002). Review on post harvest diseases and management of mango fruits. *J. Plant Pathol. Microbiol*, 13:1000635.

Knight, R. J. J., Campbell, R. J., and Maguire, I. (2009). *Important Mango Cultivars and their Descriptors*. CAB International, Wallingford, UK.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS 2012)*, pages 1097–1105. Curran Associates, Inc.

Lacap, A., Bayogan, E. R., Secretaria, L., Joyce, D., Ekman, J., and Goldwater, A. (2021). Bruise Injury and Its Effect on 'Carabao' Mango Fruit Quality. *Philippine Journal of Science*, 150(6B).

Lee, S., Moon, G., Lee, C., Kim, H., An, D., and Kang, D. (2024). Check-qzp: A lightweight checkpoint mechanism for deep learning frameworks. *Applied Sciences*, 14(19):8848.

Loshchilov, I. and Hutter, F. (2016). Sgdr: stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*.

Loshchilov, I. and Hutter, F. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Ltd, H. A. (2007). Status of mango postharvest disease management: Options and solutions for the australian mango industry. Technical report. Includes description of Dendritic Spot symptoms, epidemiology, and post-harvest behavior.

Markidis, S. and et al. (2018). Nvidia tensor core programmability, performance & precision. *IEEE International Symposium on High Performance Computer Architecture (HPCA) Proceedings*.

Migacz, S. (2020). Performance tuning guide. PyTorch Tutorials Documentation. Accessed [Date of access would normally be provided, but must be excluded per source constraints.].

Moens, V. (2024). A guide on good usage of non_blocking and pin_memory() in pytorch. PyTorch Tutorials Documentation. Accessed [Date of access would normally be provided, but must be excluded per source constraints.].

Nguyen, T. M. (2015). Lenticel damage on 'b74' mango fruit. PhD thesis, UQ eSpace. Details on lenticel discoloration symptoms and post-harvest effects.

Patel, K. K., Khan, M. A., Kumar, Y., and Yadav, A. K. (2019). Novel Techniques in Post Harvest Management of Mango- An Overview. *South Asian Journal of Food Technology and Environment*, 05(02):821–835.

Paul, R. E. (1993). Postharvest physiology of mango fruit. *University of Hawaii Free Publications*. Mentions sapburn injury as a major post-harvest disorder.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Perez, L. and Wang, J. (2017). The effectiveness of data augmentation in image classification using deep learning.

Philippine Statistics Authority (2023). Major fruit crops quarterly bulletin, april-june 2023. https://psa.gov.ph/content/major-fruit-crops-quarterly-bulletin-april-june-2023-0.

Richter, M. L., Shenk, J., Byttner, W., Arpteg, A., and Huss, M. (2020). Feature space saturation during training. *arXiv preprint arXiv:2006.08679*. Published in proceedings of ICANN 2021.

Rizwan Iqbal, H. M. and Hakim, A. (2022). Classification and Grading of Harvested Mangoes Using Convolutional Neural Network. *International Journal of Fruit Science*, 22(1):95–109.

Samaniego Jr., L., De Jesus, L. C., Apostol, J., Betonio, D., Medalla, J. D., Peruda Jr., S., Brucal, S. G., and Yong, E. (2023). Carabao mango export quality checker using matlab image processing. *International Journal of Computing Sciences Research*, 7:2080–2094.

Schulze, K., Nagle, M., Spreer, W., Mahayothee, B., and Müller, J. (2015). Development and assessment of different modeling approaches for size-mass estimation of mango fruits (mangifera indica l., cv. 'nam dokmai'). *Computers and Electronics in Agriculture*, 114:269–276.

Scott Ledger, R. H. and Cooke, T. (2000). Mango defect guide. Infographic. From Queensland Government, Department of Primary Industries and Fisheries . Published in collaboration with the Australian Mango Industry Association (AMIA) and Horticulture Australia Limited (HAL). Licensed under CC BY 4.0.

Shorten, C. and Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(60):1–48.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition.

Supekar, A. D. and Wakode, M. (2020). Multi-Parameter Based Mango Grading Using Image Processing and Machine Learning Techniques. *INFOCOMP Journal of Computer Science*, 19(2):175–187. Number: 2.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826.

Tai, N. D., Lin, W. C., Trieu, N. M., and Thinh, N. T. (2024). Development of a mango-grading and -sorting system based on external features, using machine learning algorithms. *Agronomy*, 14(4).

Tan, M. and Le, Q. (2019). EfficientNet: Rethinking model scaling for convolutional neural networks. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR.

Tan, M. and Le, Q. V. (2021). Efficientnetv2: Smaller models and faster training. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *PMLR*, pages 10096–10106. PMLR.

Tomas, M. C., Celino, J. P. L., Escalambre, I. E., and Secreto, B. P. (2022). Detection of Overall Fruit Maturity of Local Fruits Using Support Vector Machine through Image Processing. In *2022 12th International Conference on Software Technology and Engineering (ICSTE)*, pages 96–102.

Veling, P. S. (2019). Mango Disease Detection by using Image Processing. *International Journal for Research in Applied Science and Engineering Technology*, 7(4):3717–3726.

Zheng, B. and Huang, T. (2021). Mango Grading System Based on Optimized Convolutional Neural Network. *Mathematical Problems in Engineering*.

Appendix A
STUDENT RESEARCH ETHICS CLEARANCE

De La Salle University

2583

RESEARCH ETHICS CLEARANCE FORM[1] For Thesis Proposals
Names of Student Researcher(s): BANAL, Kenan A. BAUTISTA, Francis Robert Miguel F. HERMOSURA, Don Humphrey L. SALAZAR, Daniel G
College: GCOE
Department: ECE
Course: Computer Engineering
Expected Duration of the Project: from: January 4 2025 to: January 4 2026
Ethical considerations *(The Ethics Checklists may be used as guides in determining areas for ethical concern/consideration)*
To the [redacted] **the ethical issues listed above have been addressed in the research.** Dr. Reggie C. Gustilo ______________________________ **Name and Signature of Adviser/Mentor:** **Date:** February 5, 2025
Noted [redacted] Dr. Argel Bandala ______________________________ **Name and Signature of the Department Chairperson:** **Date:** February 6, 2025

[1] The same form can be used for the reports of completed projects. The appropriate heading need only be used.

Appendix B
REVISIONS TO THE PROPOSAL

De La Salle University

2586

PRO1 Panel Comments and Revisions – Appendix Z

PRO1 Panel Comments and Revisions

Zoom Recording:
https://zoom.us/rec/share/mrn9zBtPz3bJ5laVcy2E8-iBno8A6fBRgOCacMrhmzLPCNO0lDxXBHiK_xzdicEb.MzbHGzrD7rL3tVgJ?startTime=1731326444000
Passcode: +?qL6DZE

Panelist's Comments and Revisions	Action Taken	Page Number
Capture both two sides of the mango and not just one to remove error	The image capturing system would only capture the two sides of the mango which are the two largest surface areas of the skin.	18
How will you get large dataset with sweetness and how will you classify it?	Remove Sweetness in the SO	13
Size and weight are not the same.	Remove Weight in objectives but retained size in the SO4 and SO6	
Specify in the specific objectives that it will be automatic sorting	SO1: To make an image acquisition system with a conveyor belt for automatic sorting and grading mangoes.	13
Add what process will be used to get the size classification	SO6: To classify mango size by gettings its length and width using OpenCV, geometry, and image processing techniques	13
Add what process the ripeness classification will be	SO5: To classify mango ripeness using kNN or nearest neighbors algorithm	13
Get rid of texture in the general objectives	Texture is removed in the SOs	13
Get rid of CNN in general objectives and replace with machine learning	CNN is removed and replaced with machine learning GO: To develop a user-priority-based grading and sorting system for Carabao mangoes, using machine learning to assess ripeness, size, and bruises.	13
Remove Raspberry Pi on the SO's and generalize to "to create a microcontroller based application"	SO3: To create a microcontroller application to operate and control the prototype.	13
Remove SO4. No need for user testing	Removed user test and the new SO4 is SO4: To grade mangoes based on user priorities for size, ripeness, and bruises.	13
Fix IPO to the correct input and output	Input: Two side image of the Carabao Mango and the User Priority Attributes Process: Machine Learning Algorithm, Grading Formula, and CNN model using a microcontroller Output: Size, Ripeness, and Bruises	20

2587

PRO1 Panel Comments and Revisions – Appendix Z

	Classification with its Overall Grade	
Define bruises	The black or brown area of the mango that is visible on the skin of the mango.	6
Dataset should use at least 10,000 images	Added to expected deliverables SO2: To use a publicly available dataset of at least 10,000 mango images for classification of ripeness, and bruises.	14
Add to specific objectives the percentage accuracy	SO2: To get the precision, recall, F1 score, confusion matrix, and train and test accuracy metrics for classifying the ripeness and bruises with an accuracy score of at least 90%.	14
Weight sensor just adds complexity	removed all mention of load sensor, load cell. removed load cell methodology	39,40,41,42,43,44 previously

De La Salle University

2588

PRO1 Panel Comments and Revisions – Appendix Z

PRO1 Panel Comments and Revisions

Zoom Recording:
https://zoom.us/rec/share/mrn9zBtPz3bJ5laVcy2E8-iBno8A6fBRgOCacMrhmzLPCNO0lDxXBHiK_xzdicEb.MzbHGzrD7rL3tVgJ?startTime=1731326444000
Passcode: +?qL6DZE

Summary:
- Specific Objectives
  - Add:
    - what process will be used to get the sweetness classification
    - what process the ripeness classification will be
    - what process will be used to get the size classification
    - Specify in the specific objectives that it will be automatic sorting
  - Remove:
    - get rid of texture in the general objectives
    - get rid of cnn in general objectives and replace with machine learning
    - remove Raspberry Pi on the SO's and generalize to "to create a microcontroller based application"
    - remove SO4. No need for user testing

Comments:
*[00-00] time stamps from recording
- [15:00] Why only the top side of the mango? Isn't the point of automation to reduce human error? Then what about the bottom side wouldn't that just introduce another error if the mango happens to have defects on the bottom?
- [16:09] What is the load cell for? Size is not the same as weight. if size is taken from the weight wouldn't size be also taken from the image. if size then adding a load cell would just introduce more complexity, if weight then load cell is fine. reminder that size is not the same as weight.
- [17:36] When computer vision, state input and output parameters. Output parameters in this case would be sweetness, ripeness, size and bruising. Input parameters would be images.
- [18:12] No mention of how the dataset would be gathered. Would you be gather your own dataset or using a publicly available dataset
- [21:38] Fix IPO based on mention input and output parameters.
- [21:50] Dataset is lacking. Usually in machine learning at least 10,000 images. can take more than one image per mango. after taking an image of mango can make more out of the image using data augmentations.
- [22:48] Add to specific Objectives the mentioned 80%
- [23:09] Consultant that would grade the mangoes as a third party to remove biases. For both the testing and the training
- [24:55] How do you detect the sweetness of mangoes? Add these to the specific objectives. What are the categories of sweetness? Add these to specific objectives. How do

2589

PRO1 Panel Comments and Revisions – Appendix Z

you detect the correct categorization of sweetness? How to automate the classification of the sweetness.

○ [33:10] Why is the dataset destructive but the testing non destructive? Clarify this further to avoid confusion.

○ [35:09] What is the basis of sweetness using images? Clarify this further.

○ [35:35] How would you know if the classifier is correct or not? What is your ground truth (for the sweetness)?

○ [38:55] When can you say you are getting the top side of the mango? How would you know if the mango images showing the top side or the bottom side of both cheeks of the mango can be captured? If it doesn't matter then any side can be captured so why is it in the limitations that only the top side can be captured. Clarify the limitations.

○ [48:10] What classifier would you use here? What features would you extract from the images?

○ [52:07] Does it explain what process will be used to get the sweetness classification? Add it to the specific objectives

○ [54:00] How will ripeness be classified? Will it use the same dataset as the sweetness classification did? How was ground truth obtained?

○ [55:44] Why not the nearest neighbor? it is more fit in this scenario. Do not specify CNN in the objectives. The embedded systems as well, do not specify the Raspberry pi unless truly sure

○ [57:30] Table is just image processing. Is there a specific objective that would describe how ripeness classification will be done? Add this to the specific objectives.

○ [59:10] How is the weight obtained? Add it to the specific objectives. Remember that size is not proportional to weight. Size could be obtained from the image as the camera is from a fixed distance. Add to specific objectives how to get the size

○ [1:00:00] get rid of texture in the general objectives. get rid of cnn in general objectives and replace with machine learning. as each parameter will use a different method.

○ [1:04:00] remove Raspberry Pi on the SO's and generalize to "to create a microcontroller based application"

○ [1:04:37] remove SO4. no more user testing

○ [1:05:00] The formula used for grading the mangoes, is this used as industry standard? How do they measure the export quality of mango

○ [1:07:00] Specify in the specific objectives that it will be automatic sorting

Here are my comments on my end :)

1. Ensure seamless integration between hardware (sensors, motors, etc.) and software (CNNs, Raspberry Pi). You can consider using a modular approach for easier troubleshooting.

2. How do you gather a comprehensive and diverse dataset for training your CNN. This will enhance the model's robustness and accuracy.

3. Make sure that the weight sensors are calibrated correctly to avoid measurement errors.

De La Salle University

PRO1 Panel Comments and Revisions – Appendix Z

4. Implement data augmentation techniques to enhance your image dataset, which can improve model generalization and accuracy.

5. Design an intuitive user interface for the Raspberry Pi application.

6. Besides precision, recall, and F1 score, consider incorporating confusion matrices to better understand model performance and error types.

7. Conduct user testing of the application to gather feedback on usability and functionality. This can lead to improvements in design and user experience. Consider how the system can be scaled or adapted for different fruits or larger processing volumes in the future.

Noted by:

[redacted]

Dr. Donabel de Veas Abuan
Chair of Panel

Date: November 11 2024

Note: Keep a copy of this Appendix. It is a requirement that has to be submitted in order to qualify for PRO3 Defense.

Appendix C
REVISION TO THE FINAL

2593

Thesis Revisions Form – Appendix P

De La Salle University
Gokongwei College of Engineering
Department of Electronics & Computer Engineering

PANEL RECOMMENDATIONS PRIOR TO APPROVAL

TITLE: Non-Destructive Carabao Mango Sorter and Grader based on Physical Characteristics using Machine Learning

Time & Date of Defense: November 8, 2025 Venue of Defense: AG1103

Revisions:

Area of Thesis	Comments from Panel	Required Changes / Additions
Objective & Ground Truth	Panel noted confusion on the *basis of mango size classification* (small/medium/large). Ground truth was unclear.	Clearly define the ground truth reference for mango sizing. State whether classification is based on area, pixel count, bounding box dimensions, or physical calibration (e.g., coin reference).
Size Categorization	Ambiguity in how small, medium, and large are determined. Boundaries between categories not well defined, leading to possible misclassification.	Provide numerical thresholds or ranges for each category (e.g., area in cm² or pixel count). Justify with official references or calibration experiments.
Bounding Box vs. Actual Area	Panel highlighted errors when bounding box area was used (includes background pixels, not just mango).	Revise methodology to use segmented mango area instead of bounding box area. Explain error margins and how segmentation reduces misclassification.
Calibration Method	Use of "piso" (coin) as reference was questioned—panel asked what its connection is to mango sizing.	Clarify calibration method. If using coin reference, explain rationale and accuracy. Otherwise, replace with standardized calibration object or direct measurement.
Consistency of Measurement	Inconsistencies noted in how pixel/area measurements were applied.	Ensure consistent measurement approach across all samples. Document error analysis and tolerance levels.

De La Salle University

2594

Thesis Revisions Form – Appendix P

AI vs. Traditional Methods	Panel stressed that AI (YOLO, CNN) is only for detection/tracking, not for actual size measurement.	Revise methodology section: separate AI detection (classification) from size measurement (OpenCV/area computation). Remove claims that CNN/YOLO directly measure size.
Reference to Prior Work	Panel mentioned earlier works as more accurate.	Add a related works section comparing your method with prior studies. Highlight improvements and justify differences.
Color Space & Image Processing	RGB-only processing criticized; suggested conversion to other color spaces (HSV, HSB, etc.) for better segmentation.	Add experiments using HSV/HSB color space for mango segmentation. Document improvements in accuracy.
Error Analysis	Panel emphasized large errors at category boundaries (small ↔ medium, medium ↔ large).	Include error analysis section: quantify misclassification rates at boundaries, propose tolerance margins.
Methodology Documentation	Panel noted missing or unclear steps in methodology (bounding box drawing, pixel extraction, calibration).	Rewrite methodology with step-by-step workflow: detection → segmentation → area measurement → classification. Include diagrams or flowcharts.
Mechanical/Practical Considerations	Mention of conveyor movement and mechanical variation affecting classification.	Add discussion on how the conveyors and sorter position the mangoes.
Final Recommendation	Panel said AI part is acceptable, but sizing concept is the core issue.	Strengthen sizing methodology section. AI classification can remain, but emphasize accurate sizing as the thesis' main contribution.

[redacted]

Dr. Donabel de Veas Abuan, Ph.D. ECE
Chair of the Panel of Examiners

Appendix D
QUESTIONNAIRE TO THE EXPERT

De La Salle University

2597

Comparative Analysis: Expert's Assessment

Please fill up the following information.

Full Name: ______________________________

Years of Experience: ______________________________

Current Role/Position: ______________________________

Address of Farm: ______________ Hectares: ______________

Mango Varieties Familiar With: ______________________________

Experience with Quality Standards: ______________________________

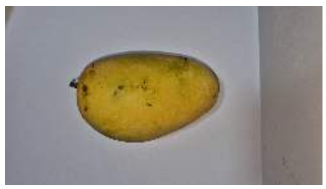
Date of Analysis: ______________________________

Instructions: Your task is to categorize the mangoes based on its color and bruising. Each image will have checkboxes pertaining to the category. More specifically categorize the mango's color into yellow, yellow-green, and green. And the bruises category into bruised and non-bruised.

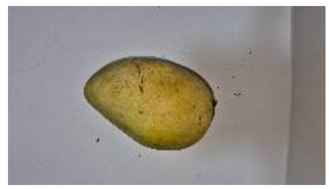
____
Name & Signature

De La Salle University

2598



Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised



Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised

Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised

Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised

Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised

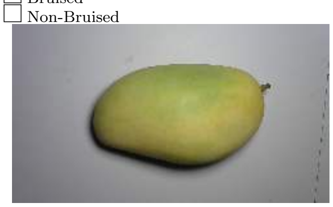
Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised

2599

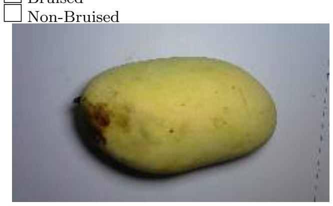
Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised

Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised

Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised



Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised



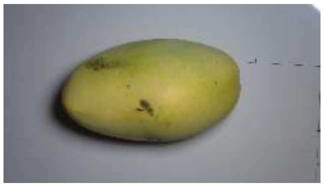
Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised

Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised

De La Salle University

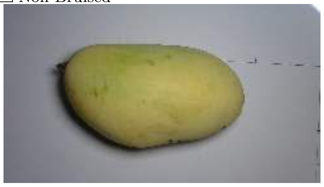
2600

Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised



Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised

Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised



Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised

Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised

Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised

2601

Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised

Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised

Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised

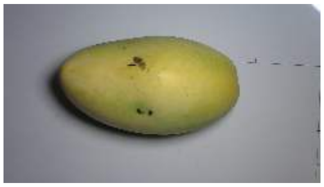
Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised

Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised

Skin Color/Ripeness:
☐ Yellow
☐ Yellow-green
☐ Green
Bruising:
☐ Bruised
☐ Non-Bruised

De La Salle University

2602



Skin Color/Ripeness:
- [ ] Yellow
- [ ] Yellow-green
- [ ] Green

Bruising:
- [ ] Bruised
- [ ] Non-Bruised

Skin Color/Ripeness:
- [ ] Yellow
- [ ] Yellow-green
- [ ] Green

Bruising:
- [ ] Bruised
- [ ] Non-Bruised

6

Appendix E
CERTIFICATE FROM FARMERS

2605

Comparative Analysis: Expert's Assessment
Please fill up the following information.

Full Name: Jesus Redome
Years of Experience: 20
Current Role/Position: farmer
Address of Farm: Ibaan Batangas Hectares: 4
Mango Varieties Familiar With: Piko, Kalabaw, indian
Experience with Quality Standards: 10
Date of Analysis: nov 4, 2024

Instructions: Your task is to categorize the mangoes based on its color and bruising. Each image will have checkboxes pertaining to the category. More specifically categorize the mango's color into yellow, yellow-green, and green. And the bruises category into bruised and non-bruised.

[redacted]
Name & Signature

1

2606

Name: Jesus Redome Date: Nov 4, 2025

Position/Role: Farmer

CERTIFICATION OF CARABAO MANGO SORTING AND DATASET VERIFICATION

This is to certify that the dataset of Carabao Mangoes used in the thesis project entitled **"Non-Destructive Carabao Mango Sorter and Grader based on Physical Characteristics using Machine Learning"** conducted by **AISL-1-2425-C5** of Department of Electronics and Computer Engineering, De La Salle University, **has been reviewed and verified.**

The mangoes represented in this dataset has been properly sorted based on the standards defined by experts. This verification confirms the dataset's integrity for academic and technical use.

Issued this ______________, for documentation and thesis validation purposes.

Si[redacted]

Jesus Redome

Name & Signature

1

De La Salle University

2607

Comparative Analysis: Expert's Assessment

Please fill up the following information.

Full Name: Ivan Joseph Palma

Years of Experience: 10

Current Role/Position: Farmer, Helper

Address of Farm: Ibaan, Batangas Hectares: 4

Mango Varieties Familiar With: Carabao, Pico

Experience with Quality Standards: 5

Date of Analysis: Nov 4, 2025

Instructions: Your task is to categorize the mangoes based on its color and bruising. Each image will have checkboxes pertaining to the category. More specifically categorize the mango's color into yellow, yellow-green, and green. And the bruises category into bruised and non-bruised.

[redacted]

Iva[redacted]oseph Palma

Na[redacted] Signature

1

De La Salle University

2608

Name: Ivan Joseph Palma Date: Nov 4, 2025

Position/Role: Farmer Helper

CERTIFICATION OF CARABAO MANGO SORTING AND DATASET VERIFICATION

This is to certify that the dataset of Carabao Mangoes used in the thesis project entitled "Non-Destructive Carabao Mango Sorter and Grader based on Physical Characteristics using Machine Learning" conducted by AISL-1-2425-C5 of Department of Electronics and Computer Engineering, De La Salle University, has been reviewed and verified.

The mangoes represented in this dataset has been properly sorted based on the standards defined by experts. This verification confirms the dataset's integrity for academic and technical use.

Issued this ________________, for documentation and thesis validation purposes.

Sincerely,

Ivan [redacted] Palma

Name & Signat[redacted]

1

2609

Comparative Analysis: Expert's Assessment

Please fill up the following information.

Full Name: Ai len Q Redome

Years of Experience: 10

Current Role/Position: Farmer Helper

Address of Farm: Ibaan Batangas Hectares: 4

Mango Varieties Familiar With: Piko, Indian, Kalabaw

Experience with Quality Standards: 7

Date of Analysis: Nov 4, 2025

Instructions: Your task is to categorize the mangoes based on its color and bruising. Each image will have checkboxes pertaining to the category. More specifically categorize the mango's color into yellow, yellow-green, and green. And the bruises category into bruised and non-bruised.

[redacted]

Name & Signature

1

De La Salle University

Name: Ailen Q Redome Date: Nov 4, 2025

Position/Role: Farmer Helper

CERTIFICATION OF CARABAO MANGO SORTING AND DATASET VERIFICATION

This is to certify that the dataset of Carabao Mangoes used in the thesis project entitled **"Non-Destructive Carabao Mango Sorter and Grader based on Physical Characteristics using Machine Learning"** conducted by **AISL-1-2425-C5** of Department of Electronics and Computer Engineering, De La Salle University, **has been reviewed and verified.**

The mangoes represented in this dataset has been properly sorted based on the standards defined by experts. This verification confirms the dataset's integrity for academic and technical use.

Issued this ________________, for documentation and thesis validation purposes.

Sincerely, [redacted]

Ailen Q. Redome

Name & Signature

1

Comparative Analysis: Expert's Assessment

Please fill up the following information.

Full Name: JERRY BRAVANTE
Years of Experience: 50 YRS
Current Role/Position: FARMER
Address of Farm: IBAAN, BATANGAS Hectares: 4
Mango Varieties Familiar With: CARABAO, PICO, INDIAN, APPLE MANGO
Experience with Quality Standards: 20 YRS
Date of Analysis: Sept 26 2025

Instructions: Your task is to categorize the mangoes based on its color and bruising. Each image will have checkboxes pertaining to the category. More specifically categorize the mango's color into yellow, yellow-green, and green. And the bruises category into bruised and non-bruised.

1

Appendix F
DATASET VALIDATION

De La Salle University

2614

Name: ______________________ **Date:** ______________________

Position/Role: ____

CERTIFICATION OF CARABAO MANGO SORTING AND DATASET VERIFICATION

This is to certify that the dataset of Carabao Mangoes used in the thesis project entitled **"Non-Destructive Carabao Mango Sorter and Grader based on Physical Characteristics using Machine Learning"** conducted by **AISL-1-2425-C5** of Department of Electronics and Computer Engineering, De La Salle University, **has been reviewed and verified**.

The mangoes represented in this dataset has been properly sorted based on the standards defined by experts. This verification confirms the dataset's integrity for academic and technical use.

Issued this ____________________, for documentation and thesis validation purposes.

Sincerely,

____

Name & Signature

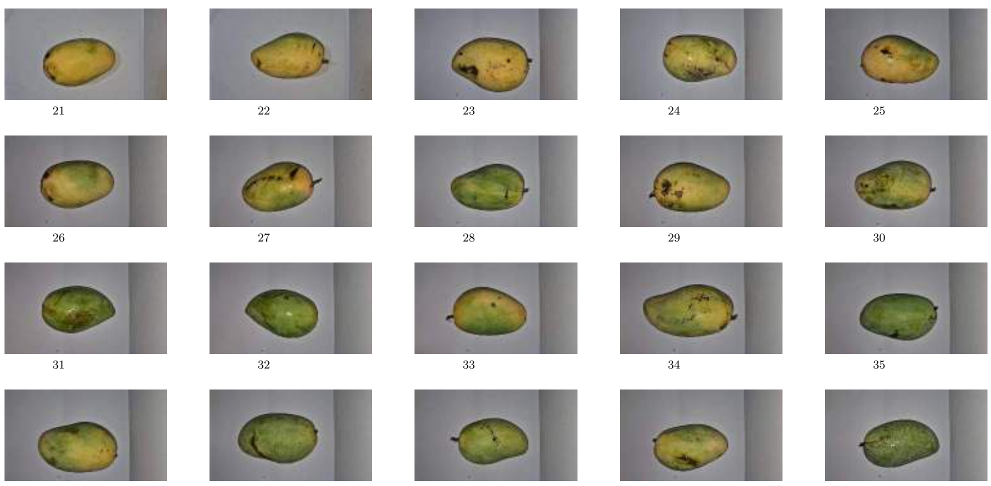
1

De La Salle University

Bruised Images (1-20)

2616

Bruised Images (21-40)



3

Bruised Images (41-60)


41


42


43


44


45


46


47


48


49


50


51


52


53


54


55


56


57


58


59


60

2618

Bruised Images (61-80)


61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

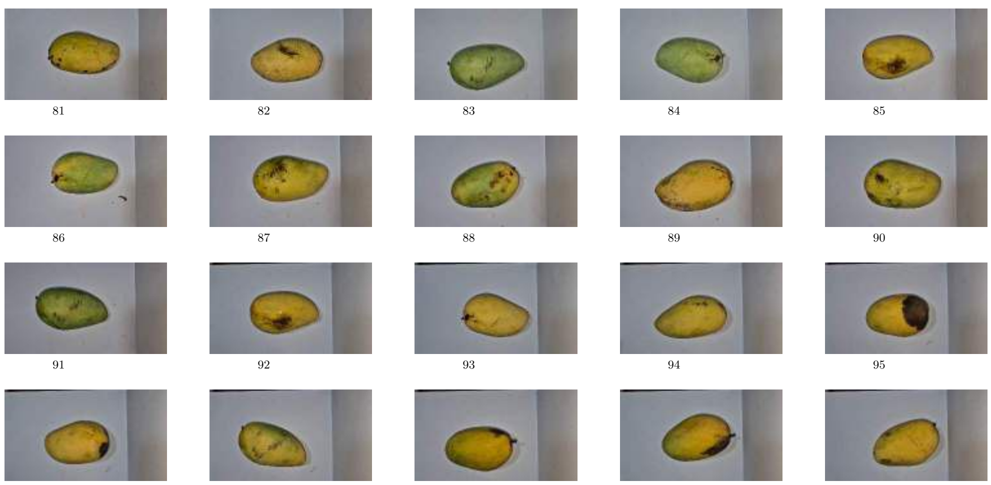
78

79

80

De La Salle University

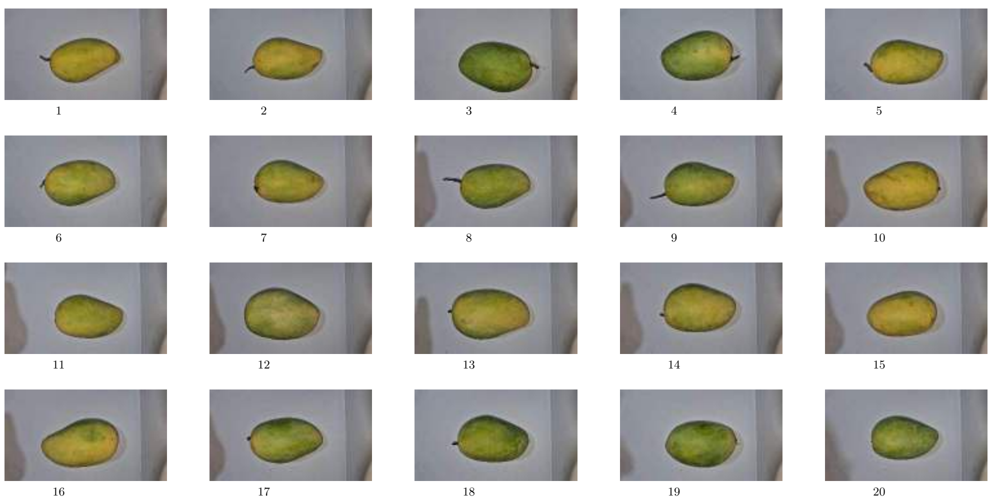
Bruised Images (81-100)



81 82 83 84 85
86 87 88 89 90
91 92 93 94 95
96 97 98 99 100

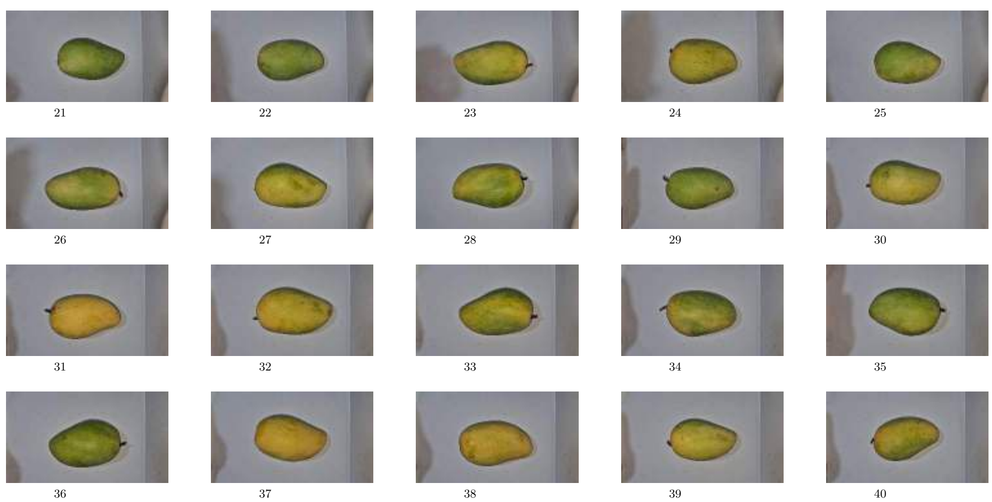
2620

Non-Bruised Images (1-20)



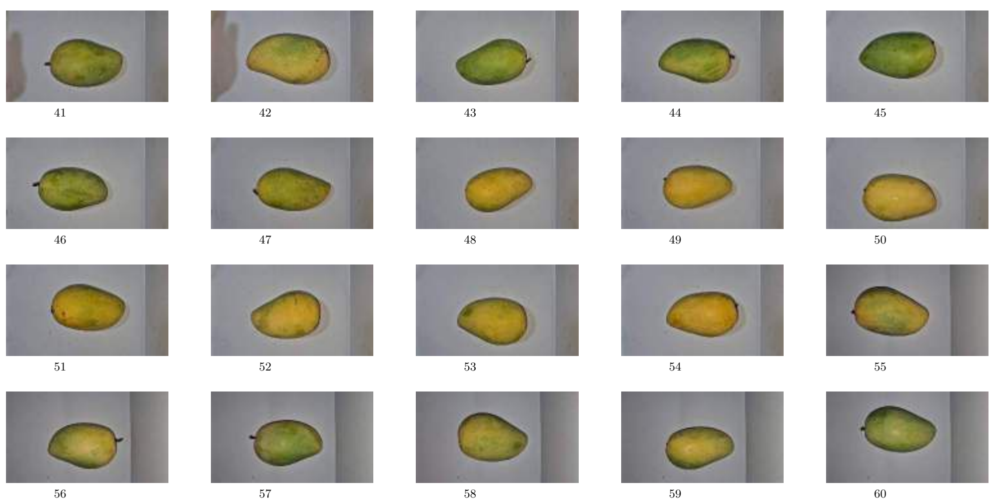
2621

Non-Bruised Images (21-40)



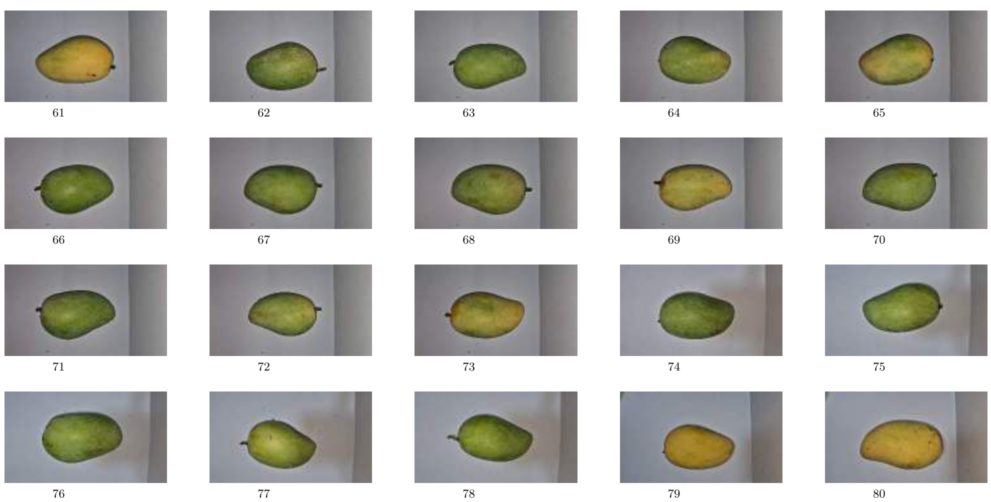
De La Salle University

Non-Bruised Images (41-60)



2623

Non-Bruised Images (61-80)



10

De La Salle University

2624

Non-Bruised Images (81-100)


81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

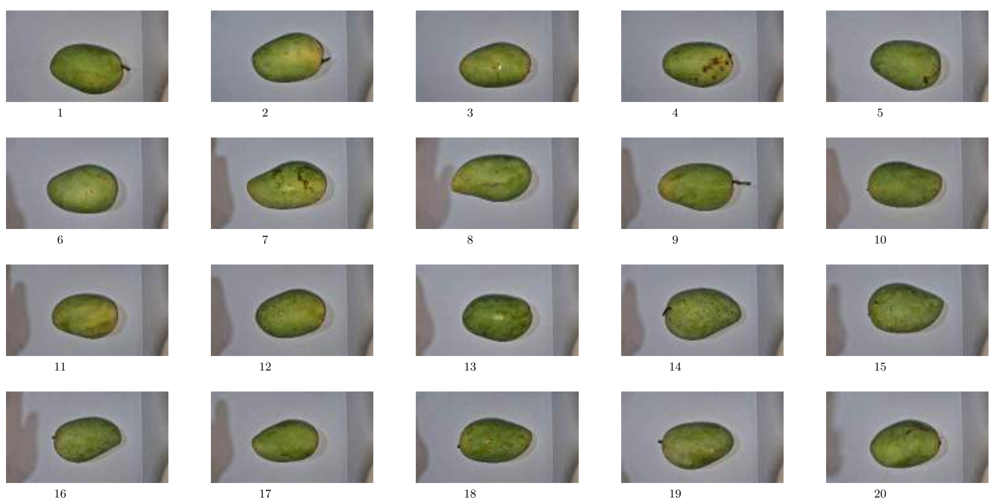
98

99

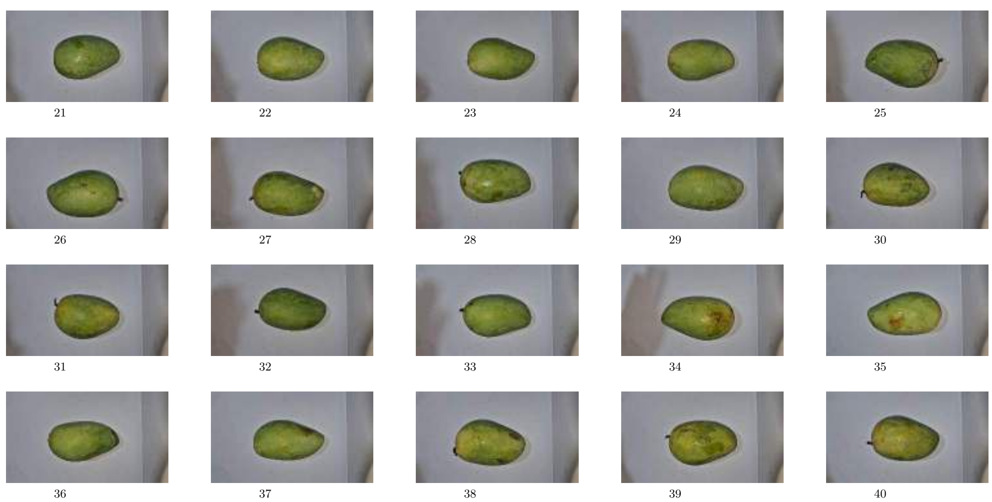
100

11

Green Images (1-20)



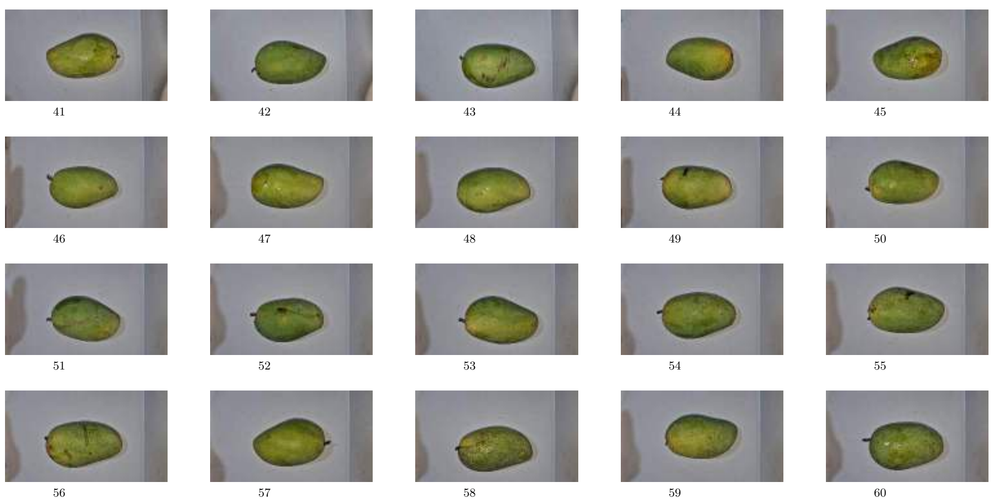
Green Images (21-40)



13

2627

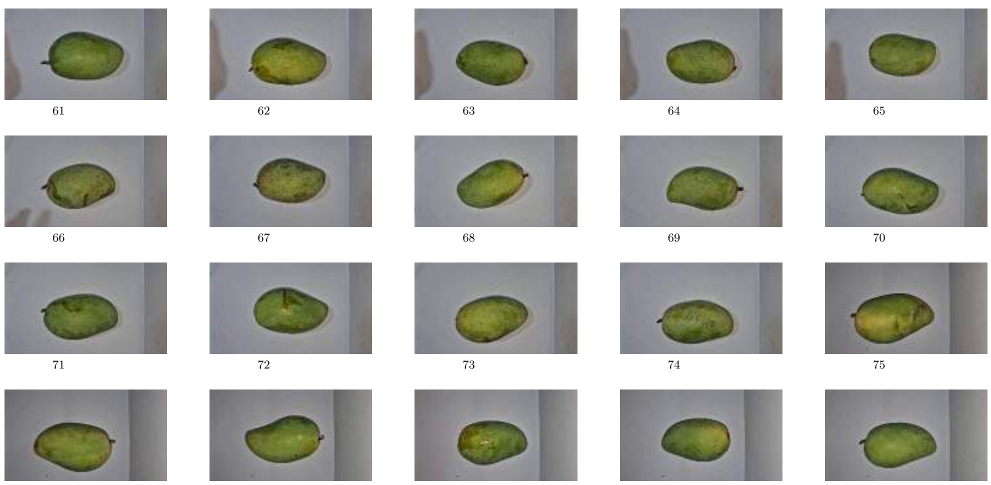
Green Images (41-60)



14

2628

Green Images (61-80)



61 62 63 64 65

66 67 68 69 70

71 72 73 74 75

76 77 78 79 80

15

De La Salle University

2629

Green Images (81-100)


81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100

Yellow Images (1-20)


1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

De La Salle University

Yellow Images (21-40)


21

22

23

24

25


26

27

28

29

30


31

32

33

34

35


36

37

38

39

40

De La Salle University

2632

Yellow Images (41-60)


41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

19

2633

Yellow Images (61-80)


61


62


63


64


65


66


67


68


69


70


71


72


73


74


75


76


77


78

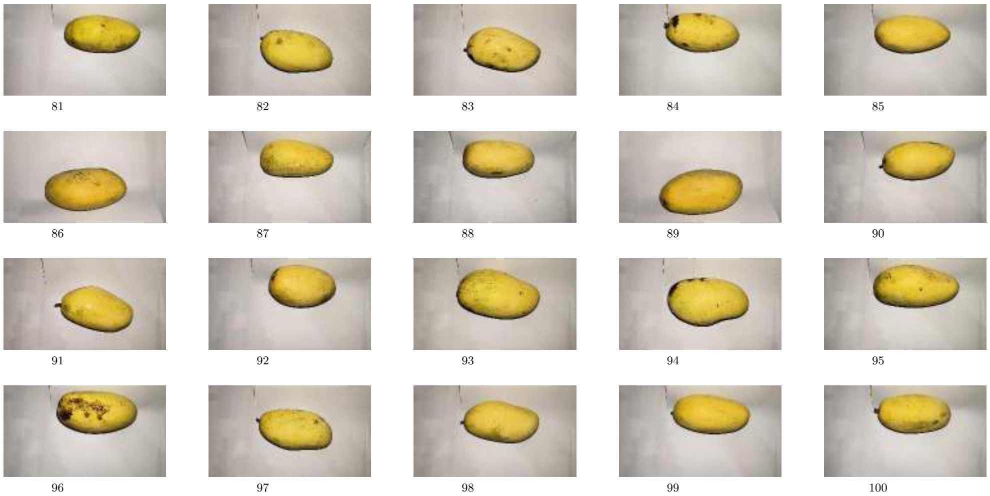

79


80

20

De La Salle University

Yellow Images (81-100)



2635

Yellow-Green Images (1-20)


1


2


3


4


5


6


7


8


9


10

22


11


12


13


14


15


16


17

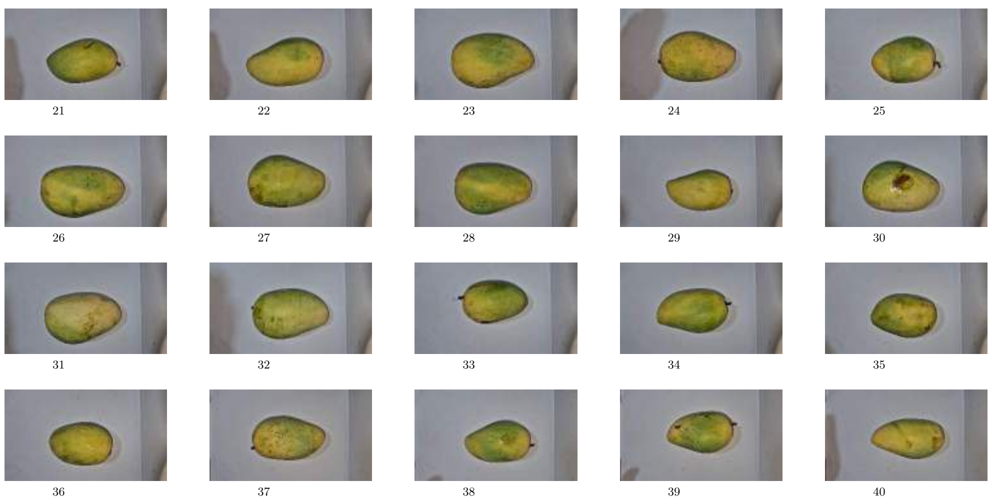

18


19


20

2636

Yellow-Green Images (21-40)



23

2637

Yellow-Green Images (41-60)


41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

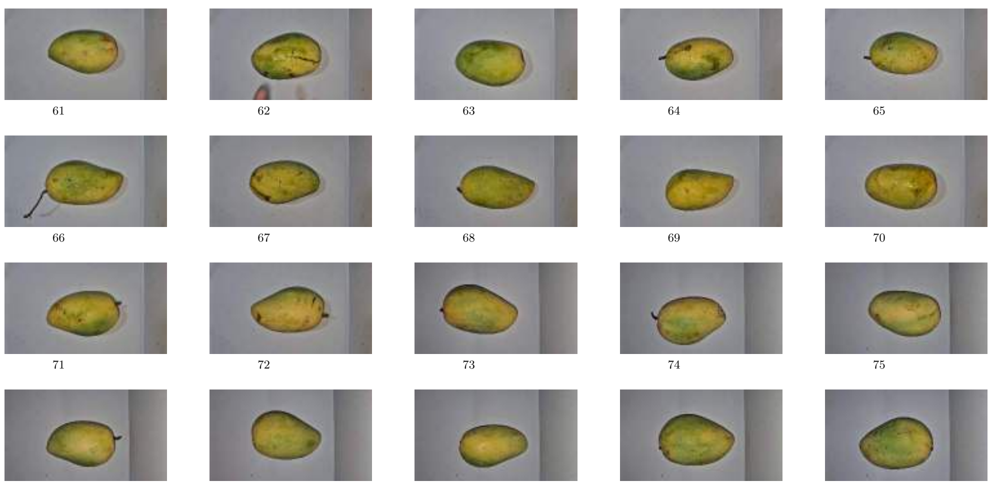
59

60

24

2638

Yellow-Green Images (61-80)



61 62 63 64 65

66 67 68 69 70

71 72 73 74 75

76 77 78 79 80

2639

Yellow-Green Images (81-100)


81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100