Mango Sub Family Classification and Ripeness Estimation Using Machine Vision

PROJECT REPORT

Submitted by

ALEN JOY	**PRC21CS008**
ALWIN ABRAHAM	**PRC21CS010**
GAYATHRY GOPAN	**PRC21CS026**
ROHAN SUNIL	**PRC21CS052**

To

APJ Abdul Kalam Technological University

in partial fulfillment of the requirements for the award of B. Tech Degree in

Computer Science and Engineering

Department of Computer Science and Engineering

Providence College of Engineering, Chengannur

March 2025

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROVIDENCE COLLEGE OF ENGINEERING, CHENGANNUR

CERTIFICATE

Certified that this report entitled '*Mango Sub Family Classification and Ripeness Estimation Using Machine Vision'* is the report of Main Project completed by the following students during **2024-2025** in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science and Engineering.

ALEN JOY	**PRC21CS008**
ALWIN ABRAHAM	**PRC21CS010**
GAYATHRY GOPAN	**PRC21CS026**
ROHAN SUNIL	**PRC21CS052**

Ms. Smita Unnikrishnan (Project Supervisor)

Assistant Professor, Department of Computer Science & Engineering (CSE)

Ms. Rubeena M (Project Co-Supervisor)

Assistant Professor, Department of Computer Science & Engineering (CSE)

Ms. Nisha Elsa Varghese
Main Project Coordinator-CSE
Dept. of Computer Science & Engg.
Providence College of Engineering

Ms. Renisha P Salim
Head of the Department
Dept. of Computer Science & Engg.
Providence College of Engineering

DECLARATION

We, hereby declare that, this project report entitled '*Mango Sub Family Classification and Ripeness Estimation Using Machine Vision'* is the bonafide work of ours carried out under the supervision of **Ms. Smita Unnikrishnan**, Assistant Professor, Department of Computer Science and Engineering. We declare that, to the best of our knowledge, the work reported herein does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion to any other candidate. The content of this report is not being presented by any other student to this or any other University for the award of a degree.

Sl. No.	Name of Student	Roll Number	Signature
1	ALEN JOY	PRC21CS008	
2	ALWIN ABRAHAM	PRC21CS010	
3	GAYATHRY GOPAN	PRC21CS026	
4	ROHAN SUNIL	PRC21CS052	

Ms. Nisha Elsa Varghese (Project Supervisor)

Ms. Renisha P Salim

Head of the Department - Computer Science and Engineering

Providence College of Engineering

Date: 20/03/2025

ACKNOWLEDGEMENTS

We take this opportunity to express our deep sense of gratitude and sincere thanks to all who helped us to complete this project successfully.

We are deeply indebted to our Project Supervisor **Ms. Smita Unnikrishnan**, Assistant Professor for her excellent guidance, positive criticism, and valuable comments.

We express our appreciation and sincere thanks to our Department Project Coordinator **Ms. Nisha Elsa Varghese,** Assistant Professor for her overall coordination and timely guidelines.

We express our sincere gratitude to our co-supervisor **Ms. Rubeena M**, Assistant Professor for her constant support and guidance.

We are also greatly thankful to our Head of Department **Ms. Renisha P Salim,** Assistant Professor for her continuous support.

Finally, we thank our parents, family members and friends who directly and indirectly contributed to the successful completion of our main project.

Alen Joy
Alwin Abraham
Gayathry Gopan
Rohan Sunil

Date: 20/03/2025

ABSTRACT

Mangoes are tropical fruits with various varieties and varying degrees of ripeness. So, at the time of harvest, the sorting process of different varieties of mangoes is challenging. Therefore, automating the classification of mango varieties and detecting their ripeness is essential for efficient sorting and quality control. To address this issue, we propose a machine learning-based mobile application for mango sub-family classification and ripeness estimation. A fine-tuned MobileNet model is utilized to classify five mango varieties—Alphonso, Ambika, Malgova, Mallika, and Neelam—while also predicting their ripeness stages as ripe, unripe, or rotten. The model is trained using transfer learning on a dataset of 30,000 images, incorporating data augmentation and optimization techniques such as Early Stopping and ReduceLROnPlateau. Experimental results demonstrate high accuracy in both variety identification and ripeness assessment, with validation accuracy exceeding 90% for variety classification. The application is deployed using Python Flask and Streamlit, enabling real-time detection through image uploads or webcam input. By automating mango classification and ripeness estimation, this solution aids farmers, distributors, and consumers in optimizing post-harvest management, reducing waste, and improving supply chain efficiency.

Keywords: Deep Learning, Early Stopping, Mobile Net, Mango Classification, ReduceLROnPlateau, Ripeness Detection, Machine Vision, Machine Learning, Streamlit.

PROVIDENCE COLLEGE OF ENGINEERING

TABLE OF CONTENTS

ACKNOWLEDGEMENTS .... **i**

ABSTRACT .... **ii**

LIST OF FIGURES .... **vi**

LIST OF TABLES .... **vii**

LIST OF ABBREVIATIONS .... **viii**

CHAPTER 1 .... **1**

INTRODUCTION **1**

1.1 Background 1

1.2 Existing System 4

1.3 Problem Statement 4

1.4 Objectives 5

1.5 Scope 5

CHAPTER 2 .... **6**

LITERATURE REVIEW **6**

CHAPTER 3 .... **15**

SYSTEM ANALYSIS **15**

3.1 Expected System Requirements 15

3.2 Feasibility Analysis 18

3.2.1 Technical feasibility 18

3.2.2 Operational feasibility 18

3.2.3 Economic feasibility 19

3.3 Hardware Requirements 19

3.4 Software Requirements 19

3.5 Life Cycle Used 19

3.6 Software Cost Estimation 21

3.7 Hardware Cost Estimation 21

3.8 Total Product Cost Estimation....................21

3.9 Project Scheduling using Gantt Chart....................22

CHAPTER 4....................23

METHODOLOGY....................23

4.1 Proposed System....................23

4.1.1 Web Application....................24

4.1.2 Image Dataset....................24

4.1.3 Data Pre-processing....................24

4.1.4 Classification Model....................25

4.1.5 Ripeness Model....................25

4.1.6 Training Model Algorithm....................25

4.1.6.1 MobileNet....................26

4.1.6.2 Transfer Learning....................26

4.1.6.3 Adam Optimizer....................27

4.1.6.4 Categorical Cross-Entropy Loss....................27

4.1.6.5 Early Stopping....................27

4.1.6.6 ReduceLROnPlateau....................28

4.2 Advantages of Proposed System....................28

CHAPTER 5....................29

SYSTEM DESIGN....................29

5.1 Flowchart....................29

5.1.1 Classification....................29

5.1.2 Ripeness Estimation....................30

5.2 Use Case Diagram....................31

5.2.1 Classification....................31

5.2.2 Ripeness Estimation....................32

5.3 Activity Diagram....................33

5.3.1 Mango Classification Process..34

5.3.2 Mango Ripeness Estimation Process 34

5.4 Sequence Diagram 35

5.5 Collaboration Diagram...36

CHAPTER 6 37

SYSTEM IMPLEMENTATION...37

6.1 Machine Learning Model 37

CHAPTER 7 39

TESTING...39

7.1 Testing...39

7.1.1 Types of Testing...39

CHAPTER 8 41

RESULTS 41

8.1 Software Module..41

CHAPTER 9 45

CONCLUSIONS...45

REFERENCES 46

APPENDIX 1: SAMPLE CODE 49

LIST OF FIGURES

Figure number	Figure Name	Page Number
Figure 1.1	Alphonso Mango	3
Figure 1.2	Ambika Mango	3
Figure 1.3	Malgova Mango	3
Figure 1.4	Mallika Mango	3
Figure 1.5	Neelam Mango	3
Figure 3.1	Increment Model	20
Figure 3.2	MDLC model	20
Figure 3.3	Project Schedule using Gantt chart	22
Figure 4.1	Block Diagram	23
Figure 5.1	Flowchart of Classification Model	29
Figure 5.2	Flowchart of Ripeness Estimation Model	30
Figure 5.3	Use case diagram for Classification Model	31
Figure 5.4	Use case diagram for Ripeness Estimation Model	32
Figure 5.5	Activity Diagram of Classification Model	33
Figure 5.6	Activity Diagram of Ripeness Estimation Model	34
Figure 5.7	Sequence Diagram	35
Figure 5.8	Collaboration Diagram	36
Figure 8.1	Home Page	43
Figure 8.2	Mango Detection (Alphonso)	43
Figure 8.3	Mango Detection (Malgova)	44
Figure 8.4	Invalid Detection	44

LIST OF TABLES

Table Number	Table Name	Page Number
Table 3.1	COCOMO Model Coefficients	21
Table 7.1	Testing Summary of Software Module	41
Table 8.1	Confusion Matrix of the Model	42
Table 8.2	Performance Metrics for Mango Variety Classification Model	42
Table 8.3	Performance Metrics for Mango Ripeness Model	42

LIST OF ABBREVIATIONS

Abbreviation	Expansion
ANN	Artificial Neural Network
API	Application Programming Interface
BPNN	Back Propagation Neural Network
CCE	Categorical Cross-Entropy
CNN	Convolutional Neural Network
COCOMO	Constructive Cost Model
FKNN	Fuzzy K-Nearest Neighbor
GPU	Graphics Processing Unit
HSV	Hue Saturation Value
KLOC	Kilo Lines of Code
LOC	Lines of Code
MDLC	Machine Development Life Cycle
mAP	Mean Average Precision
OpenCV	Open-source Computer Vision Library
OVA	One Versus All
PCA	Principal Component Analysis
PM	Person-Months
ReduceLROnPlateau	Reduce Learning Rate on Plateau
ReLU	Rectified Linear Unit
ResNet	Residual Neural Network
RGB	Red, Green, Blue
SDLC	Software Development Life Cycle
SVM	Support Vector Machine
SVR	Support Vector Regression
UI	User Interface
UML	Unified Modeling Language
VGG	Visual Geometry Group
YOLO	You Only Look Once
K-Means	k-Means Clustering Algorithm

CHAPTER 1

INTRODUCTION

1.1 Background

Mangoes are among the most important tropical fruits and a dietary staple in many regions, particularly in India, where they are often called the "King of Fruits." Renowned for their rich taste and health benefits, mangoes are packed with vital nutrients like vitamins A, C, and E, along with antioxidants and dietary fiber that boost the immune system, support healthy skin, and aid digestion. Economically, they play a crucial role in countries like India, providing livelihoods to millions of farmers. Culturally, mangoes carry symbolic significance, featuring traditional dishes and festivals across various nations. Their broad appeal and economic importance make mangoes a central fruit in global agriculture and nutrition.

Mango sub-family classification is important in India because of the country's wide variety of mangoes, each with unique flavors and characteristics. As the largestproducer and consumer of mangoes, this classification helps farmers grow the right varieties for specific climates and regions, improving agricultural efficiency. It also ensures quality control and helps meet market demands, both domestically and for exports. By promoting popular varieties like Alphonso, Ambika, Malgova, Mallika, and Neelam Their classification enhances export potential. Additionally, it supports the preservation of diverse mango types, aiding in research on disease resistance. Culturally, classifying mangoes helps maintain their traditional uses in Indian dishes and festivals, highlighting their significance in Indian culture.

Ripeness estimation of mangoes is crucial for ensuring the fruit is harvested and consumed at its best quality. Ripe mangoes are characterized by their vibrant color, yielding slightly gentle pressure, and emitting a sweet aroma, making them perfect for eating. Unripe mangoes are usually green or firm, lacking the sweet fragrance, andare best suited for cooking or pickling. Overripe mangoes are very soft, with dark spots or wrinkles and a strong, fermented smell, making them ideal for smoothies or desserts. Understanding these stages helps consumers select the best mangoes for their needs.

Machine Vision is a technology that enables computers to interpret and analyze visual information from the world, mimicking human vision. It involves the use of cameras, sensors, and sophisticated algorithms to capture and process images for various applications, such as quality control, object detection, and automation in industries. By leveraging techniques like image processing and machine learning, machine vision systems can identify, classify, and inspect objects with high accuracy and speed. This technology is widely used in manufacturing, robotics, agriculture, and healthcare, significantly improving efficiency and reducing human error in tasks that require visual inspection.

MobileNet is a lightweight deep learning model designed for efficient computer vision tasks, particularly on mobile and embedded devices. It employs depthwise separable convolutions, significantly reducing computational complexity while maintaining high accuracy. MobileNet is widely used in image classification, object detection, and feature extraction, making it ideal for real-time applications with limited processing power. In this project, a fine-tuned MobileNet model was used to classify mango varieties and ripeness levels with high efficiency and accuracy.

Streamlit is an open-source Python framework that simplifies the development of interactive web applications for machine learning and data science. It allows seamless integration of deep learning models into a user-friendly interface, enabling real-time analysis without requiring extensive front-end development. In this project, Streamlit was utilized to build an interactive mango classification and ripeness detection application, where users can upload images or use a webcam to receive instant predictions. Its intuitive design, minimal coding requirements, and real-time visualization make it an excellent choice for deploying machine vision applications.

Figure 1.1 Alphonso Mango

Figure 1.2 Ambika Mango

Figure 1.3 Malgova Mango

Figure 1.4 Mallika Mango

Figure 1.5 Neelam Mango

1.2 Existing System

The existing systems for mango sub-family classification and ripeness estimation primarily rely on traditional methods and basic technological approaches. Typically, farmers and suppliers use manual sorting techniques based on visual inspection. Some existing systems utilize basic image processing techniques for ripeness estimation. These methods may involve simple color analysis, where the color of the mangoes is visually assessed to determine ripeness.

In recent years, some automated systems have started to emerge, employing machine learning algorithms for image recognition and classification. These systems often utilize CNNs to analyze images of mangoes and classify them into different varieties. While these systems offer improved accuracy over manual methods, they may still face limitations in real-time application and require extensive training datasets.

The existing systems for mango sub-family classification and ripeness estimation face several limitations that hinder their effectiveness and efficiency:

- It relies on the classifier experience.
- Labour intensive process.
- Limited scalability.
- Leads to uneven quality control.

1.3 Problem Statement

Manual methods for classification and ripeness detection of mangoes are time consuming and error prone. To address this, machine vision-based system can automate and accurately determine the variety and ripeness of the mango to ensure efficiency and reliability in agricultural supply chain.

1.4 Objectives

The following are some of the goals of our proposed paradigm:

- To develop a mango subfamily classification system.
- To estimate the ripeness level of mangoes accurately.
- To develop a mobile application for classification and ripeness detection.
- To reduce cost and labour in mango sorting and grading.
- To improve accuracy and efficiency.

1.5 Scope

This project offers a comprehensive solution for mango variety classification and ripeness detection, streamlining the sorting process during harvest and post-harvest handling. Leveraging deep learning techniques, particularly the MobileNet model for classification and Streamlit for real-time interaction, the system automates the analysis of mango varieties and their ripeness stages. By classifying mangoes into distinct varieties and assessing their ripeness as ripe, unripe, or rotten, it provides valuable insights to farmers, distributors, and consumers. With the ability to run on smartphones, this system ensures accessibility on a global scale, benefiting mango supply chains from harvest to retail. The project enhances efficiency, reduces human error, and ensures consistent quality control, ultimately improving the mango production and distribution process worldwide.

CHAPTER 2

LITERATURE REVIEW

We have analyzed various existing work in the field classification of mangoes and ripeness estimation. The summary of the most relevant 20 papers is provided below.

Diva et al. [1] have developed a model to detect and classify the ripeness level of coffee fruits. A Wiener filter was used to sharpen the images blurred by motion. The model uses YOLOv4 to detect coffee fruits in images captured by a high-resolution Logitech Bio camera. The model also utilizes the HSV color space for better feature representation and color examination. The SVM model was used to determine the ripeness of coffee fruits. OpenCV, a robust computer vision library, facilitated image reading, processing, and augmentation tasks. Data augmentation techniques, including rotation, cropping, and shearing, were used to address the limited variety of training data. Matplotlib is used to visualize the data and results. This computer vision model effectively identifies and isolates coffee fruits from their background, making it particularly valuable for agricultural purposes, especially in post-harvest processing and quality control of coffee fruits. By automating the ripeness sorting process, the model reduces the need for manual labor, thus improving efficiency. However, the model requires manual annotation of images, which is a time-consuming and labor-intensive process, and it also needs regular maintenance to ensure consistent performance.

Analyn et al. [2] have developed a model for the freshness prediction of Cavendish bananas and Carabao mangoes using image-based classification. YOLOv5 is used to identify and detect Cavendish bananas and Carabao mangoes in images. They combined SVM with the OVA heuristic to classify the fruits into three categories: ripe, ripe, and spoiled. The images, captured by a webcam, were processed using a Raspberry Pi 4. Image preprocessing techniques, such as resizing and masking, were applied to ensure the images were suitable for analysis. The training dataset was augmented using image masking techniques, effectively doubling its size and reducing the problem of a limited dataset. They evaluated the performance using a confusion matrix. The rate of detection, predictive values, and F-score were derived to determine the accuracy of the proposed model. A lightbox minimizes the impact of lighting variations, thereby maintaining image quality during the training and testing phases. The system eliminates the subjectivity and inconsistency associated with human assessment methods, such as visual inspection, touch, and smell. It effectively categorizes fruits into three ripeness levels and ensures consistent lighting conditions. However, this model is designed to work exclusively with Cavendish bananas and Carabao mangoes, which limits its applicability to other varieties.

Aishwarya and Vinesh [3] have developed a model to categorize bananas into six groups based on their ripeness: fresh-ripe, fresh-unripe, overripe, ripe, rotten, and unripe. They conducted training and assessment using five versions of the YOLOv8 model (YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x) to achieve this classification. Deep Learning proved its effectiveness in discerning the subtle morphological changes in bananas during different ripeness stages. Image processing, involving resizing and normalizing images, was crucial for preparing the dataset for training. The YOLOv8-based framework was implemented on an energy-efficient computing device, Nvidia Jetson Xavier AGX for enabling real-time banana ripeness classification with low latency and high accuracy. The YOLOv8 models demonstrated high detection accuracy, with mean mAP ranging from 94.6% to 96.3%. These models can be deployed on various platforms, including edge devices like the Nvidia Jetson Xavier AGX, making them suitable for diverse real-world scenarios. However, training YOLOv8 was computationally intensive and time-consuming, necessitating powerful hardware such as GPUs. Overfitting the model to training data leads to a good performance on the training set. However, the model showed poor performance on unseen data.

Rodrigues et al. [4] have proposed a model for classifying the ripeness of fruits using machine learning. This model utilizes machine learning technology and computer vision to learn from data and make decisions regarding fruit ripeness. The mini VGG model is used to distinguish between ripe and unripe. It employs convolutional neural networks to classify fruits at different ripeness stages accurately. They collected and preprocessed image datasets to train and implement a CNN model. This model has capabilities for field applications using drones or robots. The system keeps a data log to estimate the final product's output. This approach has increased efficiency and accuracy in fruit processing while reducing labor costs. However, there are some drawbacks, including the initial cost and complexity of implementing such a system, the requirement for high-quality image data for training, and possible challenges in adapting the system to different environments or types of fruit.

Zeeshan et al. [5] have developed a fruit classification model that uses computer vision, SVM, and PCA. They classified eighteen different types of fruit by analyzing the texture, color, and shape extracted from images. They have used MATLAB to standardize the image resolution to 256x256 pixels. They extracted sixty-four color features, seven texture features, and eight shape features using MATLAB functions. To improve feature extraction, they applied a Gaussian filter. They classified the fruits with the help of a multi-class SVM after reducing a feature set using PCA. This approach achieved a classification accuracy of 87.06%. The system effectively distinguishes between various types of fruit, even those with subtle differences. However, the accuracy is influenced by the quality of the input images, with factors such as poor lighting potentially affecting the feature extraction and reducing accuracy.

Lath et al. [6] have proposed a model for automatic fruit detection. They have used a Multilayer Deep Convolution Neural Network for sorting the fruits. The convolution layer extracted the features of fruits from the data in the input image. Max pooling emphasizes the most prominent features of the fruits, such as edges, texture, and color. In addition, they used fully connected layers for accurate classification, utilizing the ReLU activation function to perform efficient computations and predict whether the values fall within the desired range. This approach effectively addresses the challenge of distinguishing between similar fruits. They have implemented this model in the fruit classification process used in supermarkets. Their model has high accuracy in fruit classification, reduces manual effort, and contributes to better food management and less waste. The system fails to address the challenges of real-time processing for large-scale operations and requires much computing.

Tunio et al. [7] have proposed a model for fruit detection and classification. They have used a customized deep learning technique to develop a robust and efficient fruit detection framework for an autonomous agricultural robotic platform. In the proposed model, the image was identified by analyzing the shape of the fruit depicted in the picture. They have implemented this model in the agricultural domain, specifically for detecting and classifying fruits in orchards. This model uses Convolution Neural

Networks and U-Net architecture for fruit detection. They collected a dataset featuring various fruit trees, which was then preprocessed to standardize image quality and size. Applying U-Net aims at segmenting and localizing fruits within images. This model shows high accuracy and detailed segmentation and will effectively extract the features of the pictures. However, the system fails to address the challenges in detecting fruits under varying environmental conditions and lighting and the difficulty in handling diverse and complex backgrounds.

Chanda et al. [8] have proposed a model for mango fruit detection and counting using deep learning. They used a CNN for feature extraction from images, along with a Feature Pyramid Network to enhance object detection at different scales. Image segmentation was used to prevent overfitting, while Mask R-CNN was used to generate pixel-level binary masks for precise image segmentation. The ResNext Block Backbone provided diverse feature extraction capabilities by combining multiple filters in a group structure. Additionally, Cascade R-CNN was implemented to reduce overfitting and enhance detection accuracy by refining the selection of positive data. This model improves the accuracy of mango detection and counting in occluded trees and canopies. They have applied this technique in the agricultural field. However, the system fails to address the missing detection of small objects, variations in lighting, and shadows on objects.

Abuowda et al. [9] have proposed a model for Date fruit classification. They have used deep transfer learning techniques for this model. The model used a dataset of about 4525 images of date bunches. The dataset includes five different datatypes: Barhi, Khala, Meneifi, Nabout saif, and Sullaj, and also the four maturity stages of mangoes: immature stage, khalal, rutab, and tamar stage. The images are labeled based on their type and maturity stage. ResNet50 is used for feature extraction and image classification, while fine-tuning involves adapting a predefined model to classify the datatype and maturity stages. The model shows effective feature extraction for date fruit classification and potential applications in robotic harvesting and shows accurate results in distinguishing between different date types and maturity stages. However, the system fails to address the real-time challenges of implementation. It doesn’t yet include the classification of date bunches.

Muthulakshmi and Renjith [10] have proposed a machine-learning model for classifying Durian fruits based on their ripeness. They have used algorithms like SVM and Naïve Bayes to detect, identify, and classify the features of Durian Fruit images. SVM performed the effective classification of the image. The naive Bayes Algorithm built multiple decision trees and aggregated their results. Comparison algorithms such as Linear Discriminant Analysis, Linear Regression, KNN, and CART were used. Linear Discriminant Analysis was used for dimensionality reduction and classification, while logistic regression was applied for binary classification. The model was implemented in agricultural technology for assessing fruit quality. The model will check the ripeness of the Durian fruits without any physical touch or damage. The system shows an accuracy of 89.3% in classifying the ripeness of the Durian fruit. This model efficiently processes and classifies fruit images based on color and texture. Also, the system allows ripeness classification without physical touching. Since the Gaussian Naive Bayes algorithm demonstrated a lower accuracy of 65.3% compared to SVM and Random Forest, it may impact the overall performance of the system. Additionally, if the dataset is too small, the SVM model might overfit. The model currently uses only color and texture for classification, missing other factors such as size or internal quality.

Nguyen et al. [11] have proposed a model for automating the labor-intensive process of mango classification. ANN was utilized to classify mangoes. RGB-D cameras captured images of the mangoes and analyzed color, size, shape, and volume. In addition, ANN automatically detects defects, which analyzes the extracted features to identify black spots or any other deformities in the mango fruit. The system sorts and eliminates defective mango fruits. These techniques can be applied in the agricultural field for mango classification and also in determining their quality. AI and Image processing enhance efficiency and accuracy. It also reduced the labor cost. But the initial and maintenance cost was high. There is also a dependency on the image quality.

Sangeetha et al. [12] introduced a model to classify fruits and predict their quality using deep learning. The model used deep learning and computer vision techniques to classify fruits. CNNs, a powerful architecture for image analysis, was used to extract features from the fruit images and classify them accurately. CNN was trained to differentiate fresh and rotten fruits by checking the images. Additionally, the OpenCV library was used to perform color analysis on the images which enables the system to detect fruit ripeness levels. Data preprocessing and augmentation techniques are used to enhance the quantity and quality of the dataset. The developed fruit classification system can accurately classify fresh and rotten fruits, and the system processes the images quickly. However, the variation in light can affect the features of the image which leads to incorrect classification. Also, the system's quality relies on the quantity and quality of the training data provided.

Faisal et al. [13] introduced a model that used deep learning and computer vision to find the type, maturity level, and weight of date fruits during the pre-harvesting stage. The proposed system consists of three subsystems: DTES, DMES, and DWES. Variousdeep learning architectures were used for estimating date type and its maturity. SVM was used for estimating the weight of date fruit which was trained on a dataset of date fruits with known weights. SVM helped to detect weight based on the extracted featuresfrom the image. The model was specifically used in the domain of agriculture that focuses on farming date fruits.

Juan et al. [14] have proposed a model to classify the olive fruit varieties automatically. Image processing and CNN were used to classify olive fruits. The original images were divided to isolate individual olive fruits from the background. They extracted features such as shape, texture, and color information from the image. A deep learning algorithm was designed for image recognition and classification. CNN learns complex patterns and features from the images. The features of olive fruit varieties from the images were learned by CNN which classifies the fruits based on these features. They used transfer learning to increase performance and reduce training time. The system has improved efficiency and high accuracy. The system relies on morphology rather than internal features. But the model depends on lighting conditions. Also, the deep learning models were expensive to train and deploy.

Xuan et al. [15] have proposed a model for automatic detection of the ripeness of tomatoes. They have used deep learning, specifically Faster R-CNN, for the identification of tomato ripeness from the image. Faster R-CNN was used for object detection. Using color segmentation, the tomatoes were differentiated from the background based on HSV color space and Gaussian density. For separating overlapping tomatoes and removing background pixels erosion and dilation were applied. An Intuitionistic Fuzzy Set was used for extracting boundaries or outlines of tomatoes for accurate edge detection. Finally, the Retinex method was used for correcting uneven illumination. Hence model was accurate, efficient, and robust to lighting conditions and the appearance of tomatoes. But the Faster R-CNN needed a lot of computer power, and it was also slow. IFS handled tomato overlapping but the system might struggle in extreme cases of tomato overlapping

Rahul et al. [16] have developed a model for classifying different varieties of mangoes. They utilized the Transfer Learning technique to improve accuracy, employing the EfficientNetB2 model trained on various mango datasets using a small learning rate. Additionally, they used the Adam Optimizer to adjust model parameters during training. Their training data included 1600 images of mango varieties. The model successfully addressed the challenge of classifying similar mango varieties by incorporating dropout layers and monitoring accuracy to prevent overfitting and generalizing new images. This model has been implemented in agriculture to identify and categorize mango varieties, boasting an impressive 98% accuracy rate. However, it currently does not effectively handle variations in lighting and background conditions or image quality.

Sharmila et al. [17] have proposed a model for fruit recognition by incorporating a multilayer convolutional neural network. They used VGG16, a type of CNN, to extract image features, which is effective for classification. They implemented max pooling to reduce the dimensionality of the feature map while preserving important features. Gaussian filtering was applied to smooth the image and protect the edges. They used the ReLU to introduce non-linearity and detect positive values. They utilized leaky ReLU to improve precision and rapid computation. The softmax function provides probabilities for each class. The model has been implemented in agricultural technology, particularly in supermarkets, for automated fruit sorting. Their model shows improved resource utilization, high accuracy, reduced complexity, and lower operational costs. However, one limitation is that the model fails to address fruit freshness and ripeness detection.

Nguyen et al. [18]. have proposed a model Fruit Ripeness Detector for Automatic Fruit Classification Systems. They have used the YOLOv8n architecture for fruit ripeness detection. The techniques used in this architecture are the Receptive Field Convolution Block Attention Module to split the images into groups by reducing the number of parameters and improving the efficiency, the Backbone Module to build the basic blocks that process the image, the Neck Module to recognize object of various size, and the Detection Head Module for detecting small, medium, and large objects. They have implemented this model in the fruit ripeness detection domain for classifying and checking the ripeness of bananas. Their model shows effective and efficient detection of banana ripeness and maintains high accuracy and performance when compared to the smaller version, YOLO. However, the system fails to detect the subtle ripeness differences in fruits with similar colors.

Hippola et al. [19] have proposed a model to classify the ripeness and decay of a special variety of mango called Tom EJC Mango. They have used a vision-based CNN model to identify the ripeness and decay of the fruit from its image. The ripening period of this type of mango is around 10-14 days. They had collected around 250 mangoes andstored them at 25 °C. They have used a Python-based deep learning API, Keras, to buildthe CNN model. TensorFlow was used as the backend for Keras, and OpenCV2, an open-source computer vision software library, was used for image processing. They have implemented this model in the agricultural domain in the context of fruit ripenessand quality assessment using computer vision. This model shows high accuracy, is computationally efficient, and is easier to deploy in practical applications. However, the model fails to distinguish between the similar ripeness stages of the fruit.

Vineel et al. [20] conducted a comprehensive study on fruit classification and grading techniques, employing computer vision and image processing to classify date fruits and grade mangoes. They utilized BPNN to categorize dates based on shape, size, intensity, and defect, and SVR to assess the maturity of mangoes. Various machine-learning algorithms were employed for classification and grading purposes. SVM was used to grade mangoes based on quality, while ANNs were utilized for grading mulberries based on ripeness. Furthermore, the Linear Discriminant Classifier and FKNN were instrumental in problem formulation by analyzing fruit characteristics to address classification issues. The models were implemented for grading date fruits, mangoes, pomegranates, olives, and mulberries. It doesn't provide the details of the most suitable algorithms for specific fruits.

This literature survey highlights significant advancements in fruit ripeness detection and classification using computer vision, deep learning, and machine learning techniques. Models like YOLO, CNN, and SVM demonstrate high accuracy and efficiency, but they often face challenges with image quality, lighting variations, and the need for powerful hardware. Real-time models integrated with edge devices show promise for agricultural applications, enhancing efficiency and reducing labor costs. However, most models are limited to specific fruits, reducing their generalizability. Overall, these models play a vital role in automating fruit sorting and quality control, but further improvements are needed to enhance adaptability and robustness in real-world conditions.

CHAPTER 3

SYSTEM ANALYSIS

The system analysis phase includes the analysis of various functional requirements, non-functional requirements, design constraints and hardware requirements. This chapter provides detailed system analysis of mango classification using Machine Vision.

3.1 Expected System Requirements

The proposed Mango Classification and Ripeness Detection System is expected to meet the following functional requirements.

REQ 1: Capture mango image

Description: The system would capture or upload the mango image for classification and ripeness detection.

Input: Image in any format (JPEG, PNG, etc.).

Output: Image file.

REQ 2: Preprocessing the mango image

Description: The image should undergo preprocessing such as resizing and noise reduction to improve analysis.

Input: Mango image.

Output: Preprocessed Mango image.

REQ 3: Feature extraction

Description: The system should extract features such as edges, textures, color patterns, and shapes using MobileNet's convolutional layers.

Input: Preprocessed mango image.

Output: Extracted feature set.

REQ 4: Mango variety classification

Description: The system should classify mango into one of five varieties: Alphonso, Ambika, Mallika, Malgova, and Neelam.

Input: Extracted feature set.

Output: Ripeness classification result.

REQ 5: Ripeness Detection

Description: The system should determine the ripeness state of the mango as ripe, unripe, or overripe.

Input: Extracted feature set.

Output: Predicted ripeness state.

REQ 6: Dual-Model framework

Description: The system must use two separate models – one for variety classification and another for ripeness detection.

Input: Pre-processed mango image.

Output: Mango variety and ripeness level.

REQ 7: Confidence Thresholding

Description: The system must display "Uncertain" when prediction confidence is below 0.6.

Input: Model prediction confidence score.

Output: Predicted class or "Uncertain. “

REQ 8: Real-Time Processing

Description: The system must provide real-time classification for images captured through a webcam.

Input: Real-time image from webcam.

Output: Real-time classification result.

REQ 9: Display result

Description: The system must display the detected mango variety and ripeness status to the user.

Input: Classification result.

Output: Displayed variety and ripeness status.

REQ 10: User-friendly Interface

Description: The web application must be interactive and user-friendly, implemented using Streamlit.

Input: User interactions.

Output: Interactive user interface elements.

REQ 11: Model Storage and Retrieval

Description: The system must save the best model weights during training and load them for inference.

Input: Model weights.

Output: Loaded model for inference.

REQ 12: Performance Metrics Display

Description: The system must display accuracy, precision, recall, and F1-score for model evaluation.

Input: Model evaluation results.

Output: Displayed performance metrics.

REQ 13: Handle Errors and Invalid Images

Description: The system must handle and notify users of errors, such as unsupported image formats or multiple mangoes in one image.

Input: User-uploaded image.

Output: Error message or processed image.

The proposed Mango Classification and Ripeness Detection System is expected to meet the following non-functional requirements.

NFR 1: Availability: The system should be available for the users without failure.

NFR 2: Accuracy: The system should achieve high accuracy in classifying the five different varieties of mangoes and its ripeness stages.

NFR 3: Scalability: The system should efficiently handle an increasing number of mango images and scale as the dataset expands, maintaining performance without degradation

NFR 4: Maintainability: The system should be maintainable without changing the base structure.

NFR 5: Security: The system should not be vulnerable to the attackers. The data gathered and stored by the system should be secure.

3.2 Feasibility Analysis

The feasibility study for the above-mentioned requirements is done and it is concluded that it is practically possible to build such a system. The technical, economical and feasibility analysis is discussed below.

3.2.1 Technical feasibility

The technical feasibility of this project is ensured by using MobileNet for efficient image classification and Streamlit for a user-friendly interface. Leveraging TensorFlow and Keras, the system runs on standard hardware like smartphones, enabling real-time mango classification with minimal computational resources.

3.2.2 Operational feasibility

The operational feasibility of this system is high, as it features a user-friendly Streamlit interface that requires minimal technical expertise. It enables real-time mango classification using image uploads or webcam input, making it accessible to farmers, suppliers, and distributors. Designed to work under varied environmental conditions, the system enhances efficiency in mango sorting, reducing manual effort and ensuring consistent quality control.

3.2.3 Economic feasibility

The economic feasibility of this system is strong, as it utilizes open-source technologies like TensorFlow, Keras, and Streamlit, reducing development costs. By automating mango classification and ripeness detection, it minimizes labor costs and post-harvest losses, benefiting farmers, suppliers, and distributors. The system requires only basic hardware like smartphones or webcams, making it a cost-effective and scalable solution for the agricultural sector.

3.3 Hardware Requirements

- Camera
- Smartphone or pc

3.4 Software Requirements

- Python IDE
- TensorFlow
- Streamlit
- OpenCV
- Jupyter Notebook
- VS Code
- Kaggle

3.5 Life Cycle Used

In this project we have chosen the incremental model. It is an iterative enhancement model. We develop our project as different modules which will be completed as different iterations. The Incremental model shown in Figure 3.1 is flexible and it is easier to incorporate new features during the development phase.

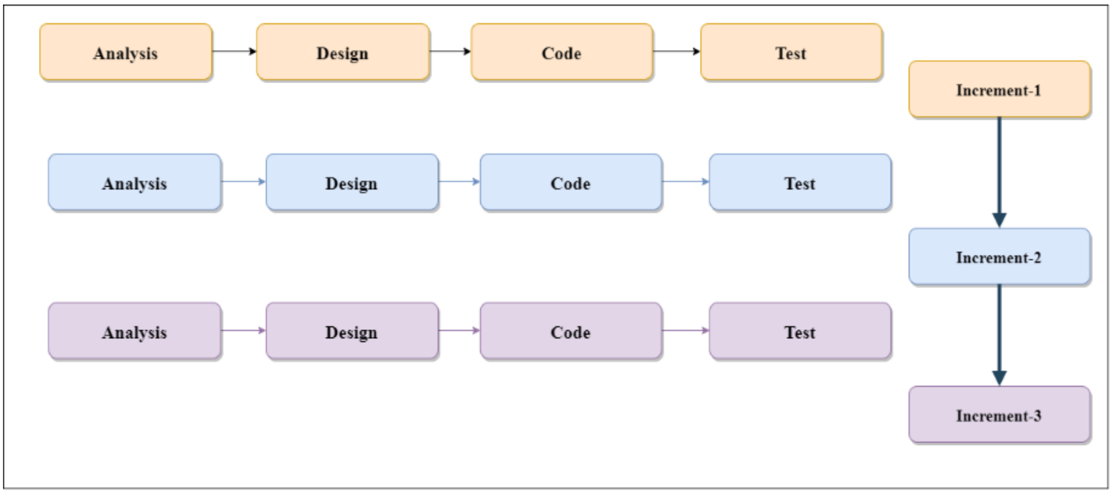

Figure 3.1: Incremental Model

Analogous to the role of the software-development lifecycle (SDLC), the machine learning model-development lifecycle (MDLC) guides the activities of ML model development from inception through retirement. The key phases of the MDLC including data ingestion, exploratory data analysis, model creation, and model operation are illustrated in Figure 3.2.

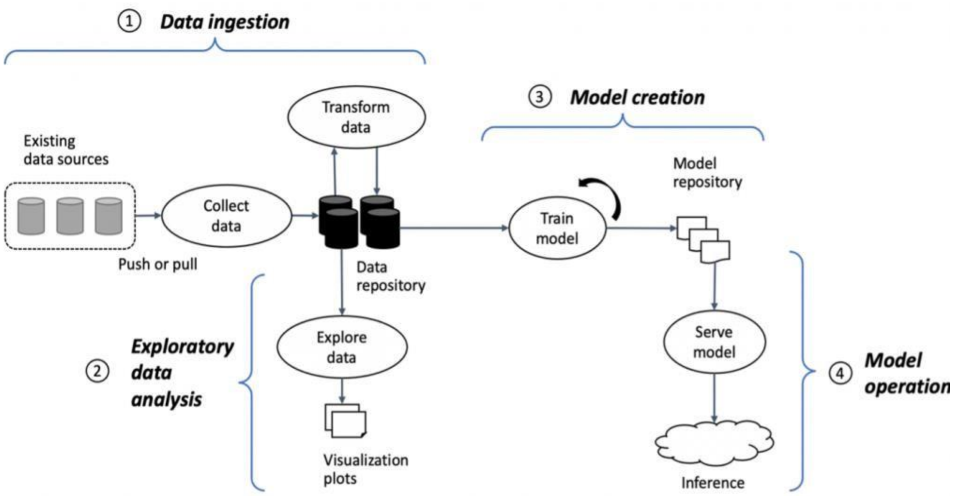

Figure 3.2: MDLC Model

3.6 Software Cost Estimation

For cost estimation, the COCOMO (Constructive Cost Estimation Model) is used.

Table 3.1: COCOMO model coefficients

Software Category	a	b	c	d
Organic	2.4	1.05	2.5	0.38
Semi-detached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

Software Project Category: Semi Detached

Estimated Lines of Code (LOC): 118 + 3,842 = 3,960 LOC= 3.96 KLOC

Effort applied = $a*(KLOC)^b$ [Person-Months] = $3(3.96)^{1.12}$ = 14.01 PM (Person Months)

Development Time = $c*(Effort)^d$ = $2.5(14.01)^{0.35}$ = 6.29 months / 172 days

No: of Persons = Effort / Development Time = 14.01 / 6029 = 2.22 = 2 person

If ‘X’ is the salary of one person, then total development cost is 2X for this project.

3.7 Total Product Cost Estimation

The total product cost is the sum of both hardware cost and software cost. As we are not professional developers, the software cost can be considered 0. So, the estimated cost for developing this product is:

Total cost = Hardware cost + Software cost = 0 + 0 = 0 INR

3.8 Project Scheduling using Gantt chart

ID	Task Name	Start	Finish	Duration	2024 Aug	Sep	Oct	Nov	Dec	2025 Jan	Feb	Mar
1	Requirement Gathering	01-08-2024	14-08-2024	14d								
2	Literature Survey	15-08-2024	28-08-2024	14d								
3	Feasibility Analysis	29-08-2024	02-09-2024	5d								
4	System Modelling	03-09-2024	22-09-2024	20d								
5	System Implementation	22-09-2024	29-01-2025	130d								
6	Testing	25-01-2025	05-03-2025	40d								
7	Documentation	01-08-2024	21-03-2025	233d								

Figure 3.3: Project Schedule using Gantt chart

The Gantt chart shows the project scheduling. Gantt chart shows the start and finish date of the project. Our project phases will be completed as per the prescribed time schedule in the above Gantt chart shown in Figure 3.3. The starting date was 01/10/2024 and the project is expected to be completed on 21/03/2025.

CHAPTER 4

METHODOLOGY

4.1 Proposed System

The proposed system introduces an automated mango classification and ripeness estimation solution using machine vision and deep learning. It leverages a fine-tuned MobileNet model to classify five mango varieties (Alphonso, Ambika, Malgova, Mallika, Neelam) and determine their ripeness stages (ripe, unripe, rotten). The system processes high-resolution images captured via file uploads or webcam input, applies image preprocessing techniques (resizing, normalization, and augmentation), and uses a Convolutional Neural Network (CNN) for feature extraction and classification. The classification results are displayed in real-time through an interactive Streamlit-based web application. The system is designed to be lightweight, scalable, and accessible on smartphones and desktops, making it beneficial for farmers, suppliers, and distributors. By automating mango sorting and ripeness detection, the proposed system enhances efficiency, reduces manual errors, and improves quality control in the mango supply chain.

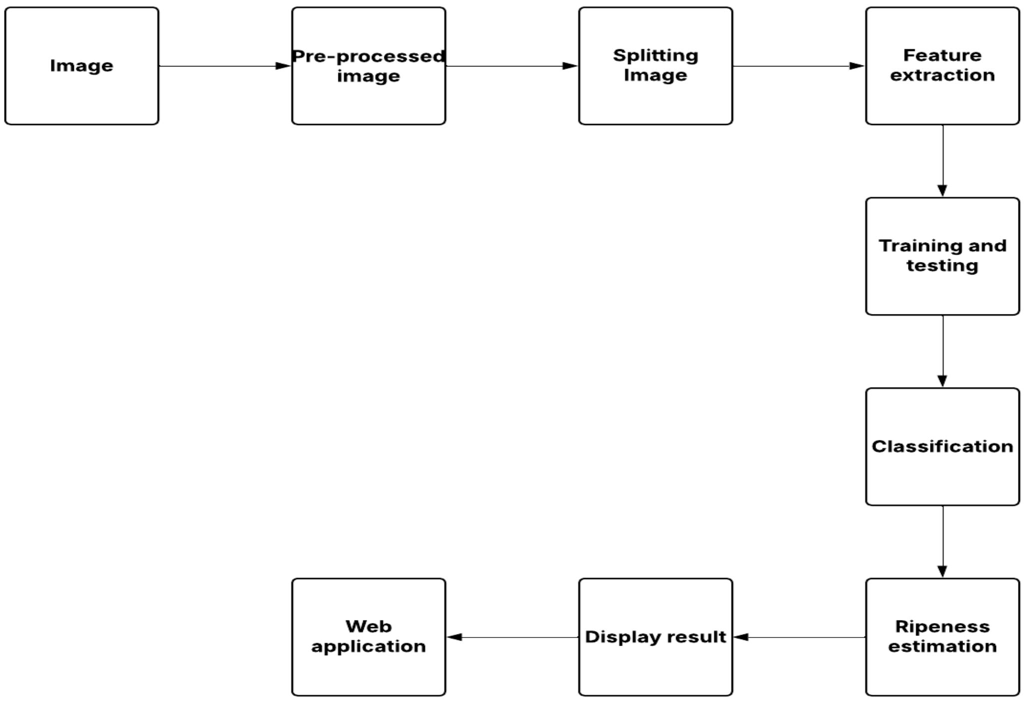

Figure 4.1: Block Diagram

4.1.1 Web Application

The mango classification and ripeness estimation system is deployed as a web application using Streamlit, providing a user-friendly and interactive interface. Users can upload mango images or capture them using a webcam, and the system processes these images in real time. The fine-tuned MobileNet model classifies the mango variety (Alphonso, Ambika, Malgova, Mallika, Neelam) and determines its ripeness stage (ripe, unripe, rotten). The results are displayed instantly, along with a confidence score, ensuring reliable predictions. The responsive UI supports seamless navigation, making it accessible on smartphones, tablets, and desktops. By integrating deep learning and real-time processing, the web application provides an efficient, scalable, and cost-effective solution for farmers, suppliers, and distributors in the mango supply chain.

4.1.2 Image Dataset

The dataset comprises digital images of five mango varieties—Alphonso, Ambika, Malgova, Mallika, and Neelam—captured in various ripeness states, including ripe, unripe, and rotten. The images were taken from Kaggle and collected under controlled lighting conditions with a neutral background to minimize environmental variables. For the category classification model, the dataset contains approximately 30,000 images, with 24,176 images designated for training and 3,006 for validation, categorized into five classes based on the mango varieties. For the ripeness detection model, the dataset is organized into three classes: ripe, unripe, and rotten. The images are stored in a directory structure where each category has its own folder containing the corresponding images.

4.1.3 Data Pre-processing

The data preprocessing phase ensures that mango images are properly formatted for accurate classification. Images are first resized to 224×224 pixels to match MobileNet's input requirements. Normalization is applied by scaling RGB values to the [0,1] range for consistent model performance. To improve generalization and prevent overfitting, data augmentation techniques such as rotation (±30°), width and height shifting (up to 30%), horizontal flipping, and nearest-fill mode are used. These transformations enhance the model's ability to recognize mangoes under different conditions. The preprocessed dataset, consisting of 30,000 images, is then split into 80% training and 20% validation to ensure robust learning and evaluation. This preprocessing pipeline optimizes the classification accuracy and reliability of the system.

4.1.4 Classification Model

The mango classification system uses a deep learning approach based on MobileNet to identify five mango varieties: Alphonso, Ambika, Malgova, Mallika, and Neelam. The model leverages transfer learning with MobileNet, pre-trained on ImageNet, and includes custom layers for global average pooling, dense layers with ReLU activation, and a softmax output layer for classification. The dataset comprises approximately 30,000 images, preprocessed through resizing, normalization, and augmentation to improve model robustness and generalization. The classification model achieved a validation accuracy of 94.3%, with training optimized using categorical cross-entropy loss, the Adam optimizer, and callbacks like early stopping and learning rate adjustments. Deployed as part of an interactive Streamlit-based web application, the model provides a fast and efficient solution for automated mango variety identification, offering significant value to stakeholders in the agricultural supply chain.

4.1.5 Ripeness Model

The ripeness model is a deep learning-based classifier designed to categorize mangoes into three stages: ripe, unripe, and rotten. It utilizes a fine-tuned MobileNet architecture, leveraging transfer learning for efficient feature extraction. The model processes preprocessed mango images (resized to 224×224 pixels and normalized) and applies data augmentation to enhance generalization. It is trained using categorical cross-entropy loss and optimized with the Adam optimizer, incorporating EarlyStopping and ReduceLROnPlateau to improve performance and prevent overfitting. Achieving over 90% accuracy, the model enables real-time ripeness estimation via a Streamlit-based web application, providing an accessible and scalable solution for farmers, suppliers, and distributors in the mango industry.

4.1.6 Training Model Algorithms

The mango classification and ripeness estimation models are trained using MobileNet, a lightweight Convolutional Neural Network (CNN) optimized for efficient image classification. The training process follows transfer learning, where pre-trained MobileNet weights from ImageNet are fine-tuned to adapt to mango classification. The models use categorical cross-entropy loss for multi-class classification and are optimized with the Adam optimizer to ensure fast convergence. To enhance training efficiency and prevent overfitting, techniques like EarlyStopping (stopping training when validation loss stops improving) and ReduceLROnPlateau (reducing the learning rate if performance plateaus) are applied. The models are trained with batch size = 32, learning rate = 0.001, and 25 epochs, ensuring high accuracy while maintaining computational efficiency.

4.1.6.1 MobileNet

MobileNet is a lightweight and efficient Convolutional Neural Network (CNN) designed for image classification and object detection in resource-constrained environments. It uses depthwise separable convolution, reducing the number of parameters and computational complexity while maintaining high accuracy. MobileNet is optimized for mobile and embedded devices, making it ideal for real-time applications like mango classification and ripeness detection.

In this project, a fine-tuned MobileNet model is used to classify mango varieties (Alphonso, Ambika, Malgova, Mallika, Neelam) and predict ripeness stages (ripe, unripe, rotten). The model leverages transfer learning with pre-trained weights from ImageNet, significantly improving performance with minimal training data. With fewer parameters, it ensures fast inference, low memory usage, and high accuracy, making it well-suited for real-time agricultural applications.

4.1.6.2 Transfer Learning

Transfer learning is a machine learning technique where a pre-trained model is adapted to a new but related task. Instead of training a deep learning model from scratch, transfer learning allows the reuse of learned features from an existing model trained on a large dataset, such as ImageNet. This significantly reduces training time, computational costs, and data requirements, while improving accuracy.

In this project, MobileNet, a pre-trained Convolutional Neural Network (CNN), is fine-tuned for mango variety classification and ripeness detection. The lower layers of MobileNet, responsible for detecting general image features (edges, textures, and patterns), are frozen to retain their pre-learned knowledge, while the upper layers are trained on the mango dataset to specialize in distinguishing five mango varieties and three ripeness stages. This method ensures efficient training, faster convergence, and high classification accuracy, making it an ideal solution for real-time mango sorting applications.

4.1.6.3 Adam Optimizer

The Adam optimizer is an advanced gradient descent optimization algorithm that combines momentum-based gradient updates and adaptive learning rates for faster convergence and stability. It dynamically adjusts learning rates for each parameter, preventing vanishing gradients and improving training efficiency. In this project, Adam is used to optimize the MobileNet-based mango classification and ripeness detection models, ensuring faster convergence, better generalization, and high accuracy in predicting mango varieties and ripeness stages.

4.1.6.4 Categorical Cross-Entropy Loss

Categorical Cross-Entropy (CCE) Loss is a commonly used loss function for multi-class classification problems, where each input belongs to one of several distinct categories. It measures how well the predicted probability distribution aligns with the actual class labels.

In this project, CCE loss is used to train the MobileNet-based models for mango variety classification (5 classes: Alphonso, Ambika, Malgova, Mallika, Neelam) and ripeness detection (3 classes: ripe, unripe, rotten). The model outputs a probability distribution over the possible classes, and CCE loss penalizes incorrect predictions, ensuring the model learns to assign high probabilities to the correct classes. This helps achieve high accuracy and reliable classification in the mango sorting system.

This method ensures efficient training, faster convergence, and high classification accuracy, making it an ideal solution for real-time mango sorting applications.

4.1.6.5 Early Stopping

Early Stopping is a regularization technique used in deep learning to prevent overfitting by stopping the training process when the model's performance on the validation set stagnates or worsens. It continuously monitors a chosen metric, such as validation loss, and halts training if no improvement is observed for a specified number of epochs.

In this project, Early Stopping is applied during the training of the MobileNet-based classification and ripeness detection models. If the validation loss stops decreasing, training is automatically stopped to prevent overfitting and unnecessary computations. This helps the model generalize to new mango images and avoid memorization.

4.1.6.6 ReduceLROnPlateau

ReduceLROnPlateau is an adaptive learning rate scheduler used in deep learning to improve model training efficiency. It monitors validation loss during training and reduces the learning rate when the model's performance plateaus, preventing unnecessary updates and ensuring smoother convergence.

In this project, ReduceLROnPlateau is applied to optimize MobileNet based classification and ripeness detection models. If the validation loss does not improve for a set number of epochs, the learning rate is reduced by a factor (e.g., 0.1), allowing the model to make finer weight updates. This helps in avoiding overfitting, improving generalization, and stabilizing training, ensuring high accuracy in mango classification and ripeness estimation.

4.2 Advantages of Proposed System

- Automation – Reduces the need for manual inspection, saving time and labor costs.
- User-Friendly – A simple web interface using Streamlit allows easy image uploads and predictions.
- Fast Processing – Provides instant results compared to traditional methods.
- Scalability – Can be extended to support more mango varieties and additional fruit classifications.
- Cost-Effective – Reduces dependency on expensive lab tests or expert assessments
- Remote Accessibility – Can be accessed from anywhere using an internet-connected device.

CHAPTER 5

SYSTEM DESIGN

The proposed system model is depicted using a variety of system modelling techniques, including a use case diagram, an activity diagram, a sequencediagram, and a flow chart.

5.1 Flowchart

5.1.1 Classification

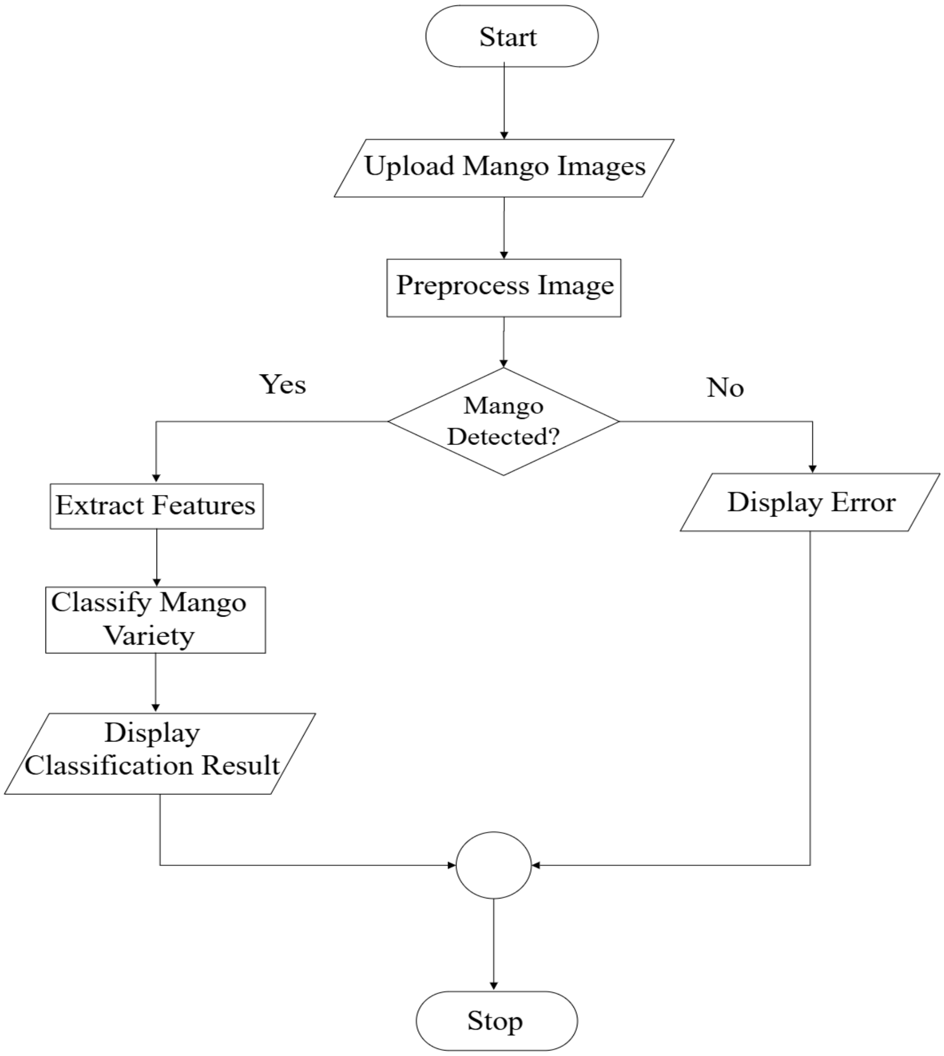

Figure 5.1: Flow chart of Classification model

Description: -

Figure 5.1 illustrates the flowchart of outlines the step-by-step process of mango variety and ripeness detection. It begins with image input via file upload or webcam, followed by preprocessing. The processed image is then fed into a MobileNet-based deep learning model, where feature extraction is performed. The system applies two models:

one for mango variety classification and another for ripeness detection. Finally, the predicted variety and ripeness stage are displayed in the Streamlit web application, allowing users to either analyze another mango or exit the system.

5.1.2 Ripeness Estimation

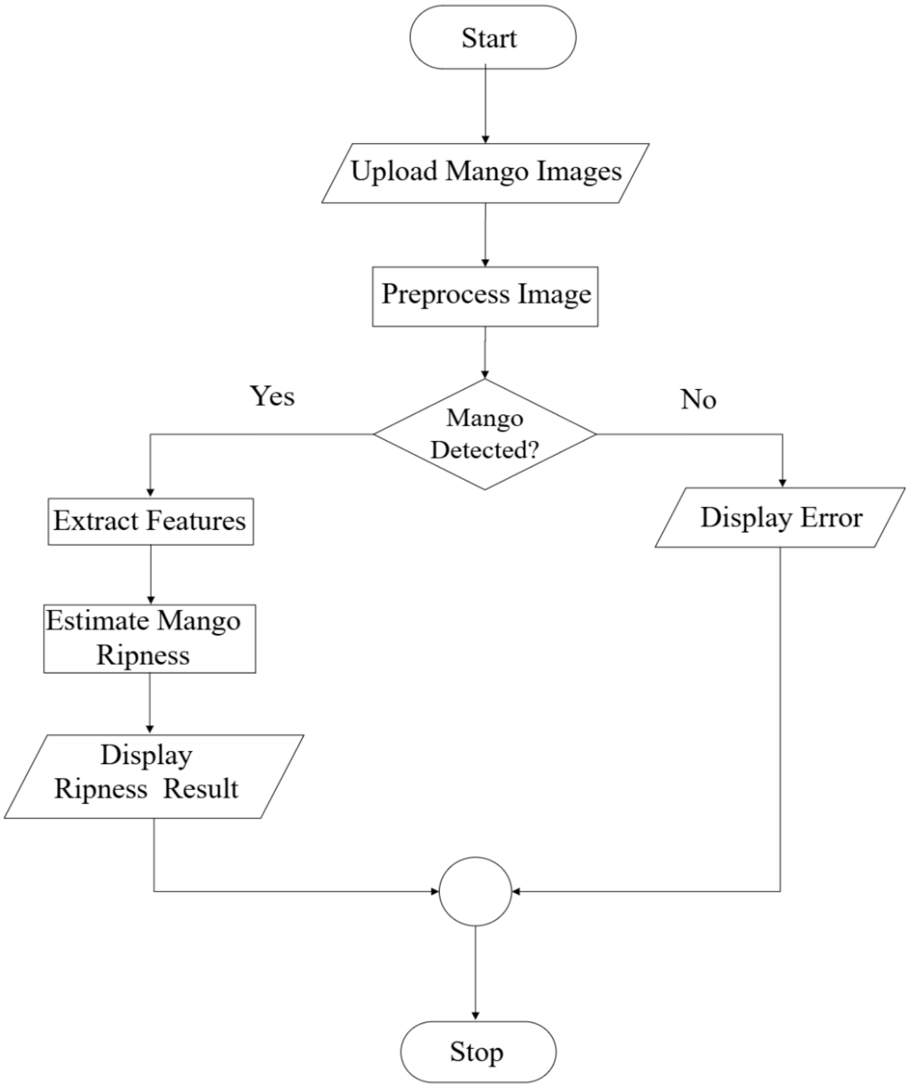

Figure 5.2: Flow chart of ripeness estimation model

Description: -

Figure 5.2 illustrates the ripeness estimation flowchart begins with image input via file upload or webcam, followed by preprocessing (resizing to 224×224 pixels, normalization, and data augmentation). The processed image is then passed to a MobileNet-based deep learning model, which extracts key color and texture features. The model classifies the mango into one of three ripeness stages: ripe, unripe, or rotten. A confidence score is assigned to the prediction, and if it is below a predefined threshold, the system marks it as uncertain. The ripeness result is shown on the Streamlit app, allowing users to analyze another image or exit.

.

5.2 Use Case Diagram

A use case diagram is a diagram in the Unified Modeling Language (UML) that depicts the link between different use cases and actors.

5.2.1 Classification

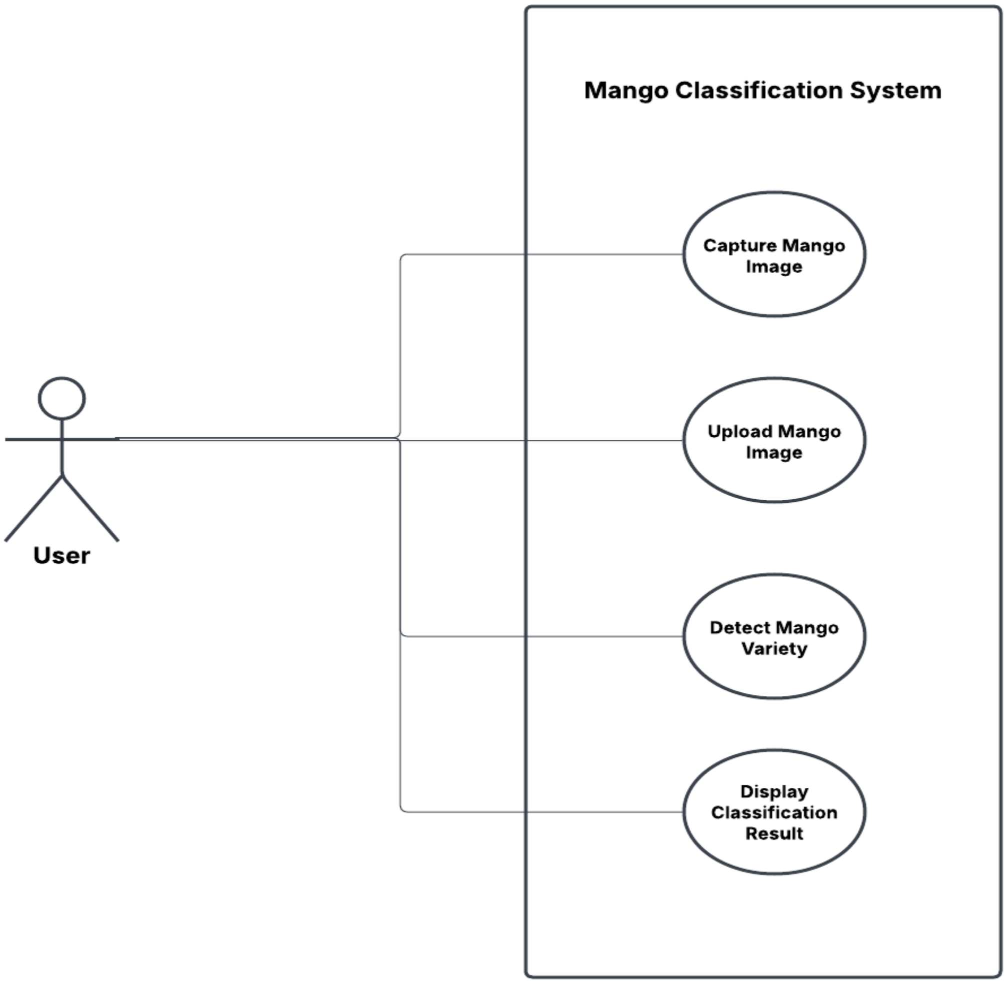

Figure 5.3: Use case diagram for classification model

Description:

Figure 5.3 depicts use case diagram for mango classification illustrates the interaction between the user and the system, highlighting key functionalities. The user uploads an image via file upload or webcam, and the system preprocesses it by resizing, normalizing, and augmenting the data. The MobileNet model extracts features like shape, texture, and color, followed by mango variety classification (Alphonso, Ambika, Malgova, Mallika, Neelam) and ripeness estimation (ripe, unripe, rotten). The classification results, along with a confidence score, are displayed in the Streamlit web application, allowing the user to either analyze another mango or exit the system.

5.2.2 Ripeness Estimation

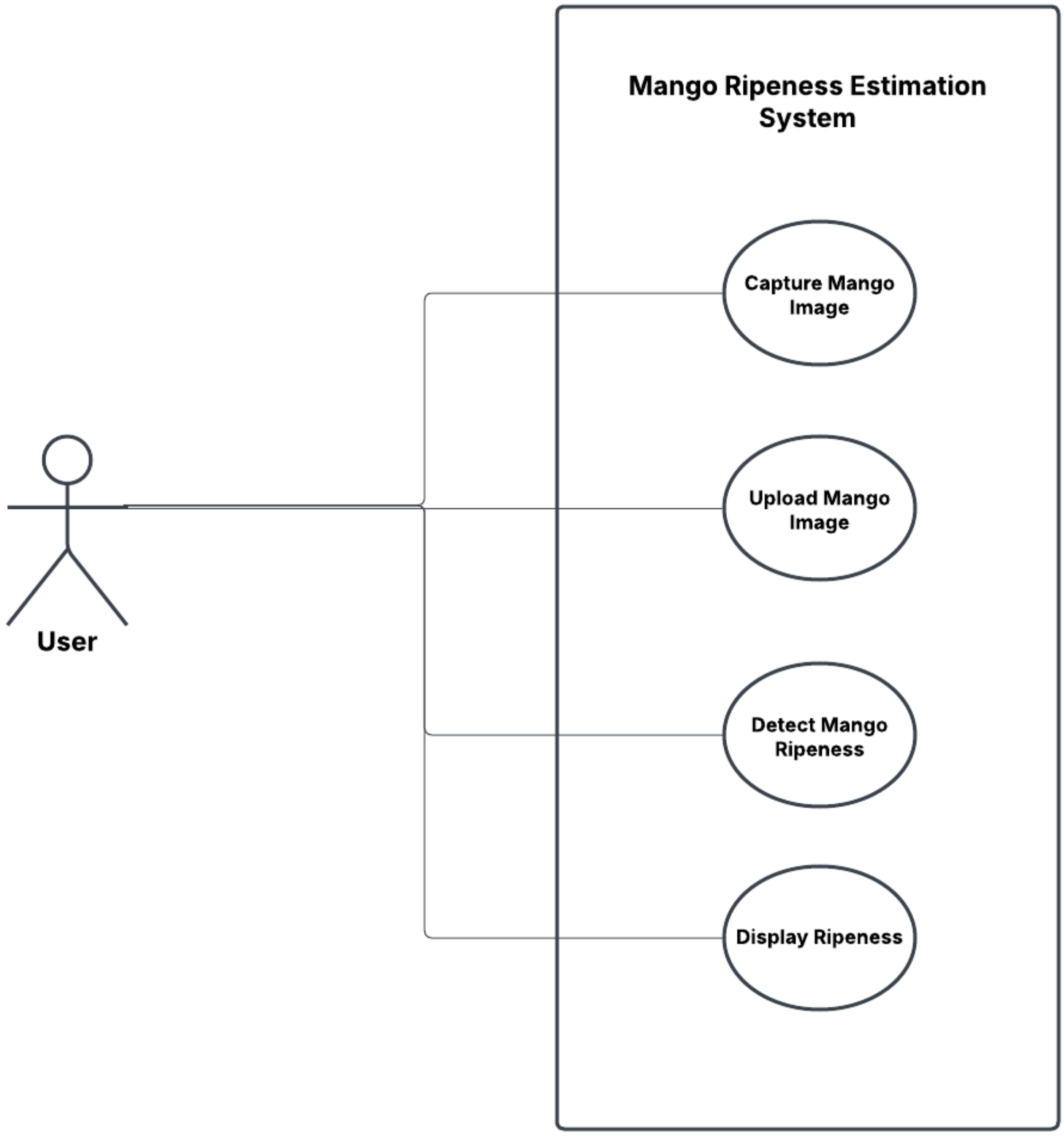

Figure 5.4: Use case diagram of ripeness estimation model

Description:

The Figure 5.4 depicts use case diagram for ripeness estimation represents the interaction between the user and the system, focusing on predicting mango ripeness. The user uploads an image via file upload or webcam, and the system preprocesses it by resizing, normalizing, and augmenting the image for better model performance. The MobileNet-based deep learning model extracts key color and texture features to classify the mango into one of three ripeness stages: ripe, unripe, or rotten. A confidence score is assigned, and if below a threshold, the system marks the prediction as uncertain. Finally, the ripeness result is displayed in the Streamlit web application, allowing the user to either analyze another mango or exit the system.

5.3 Activity Diagram

5.3.1 Mango Classification Process

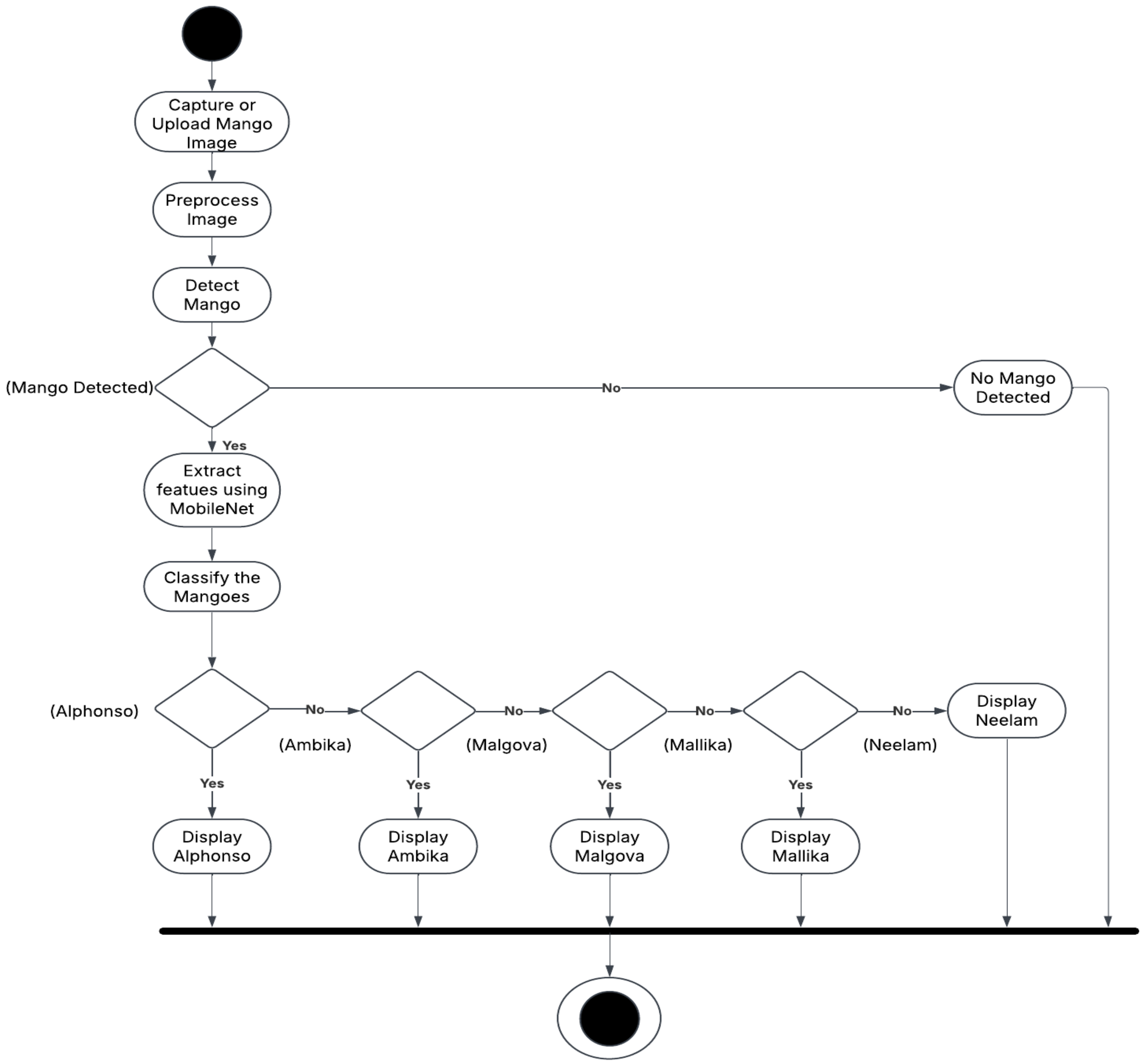

Figure 5.5: Activity diagram of classification model

Description:

The Figure 5.5 The activity diagram for mango classification begins with the user uploading an image via file upload or webcam, followed by image preprocessing. The preprocessed image is passed through the MobileNet-based model, where feature extraction is performed. The system then classifies the mango into a variety and estimates its ripeness stage. Finally, the classification results are displayed in the Streamlit web application, allowing the user to analyze another mango or exit the system.

5.3.2 Mango Ripeness Estimation Process Diagram

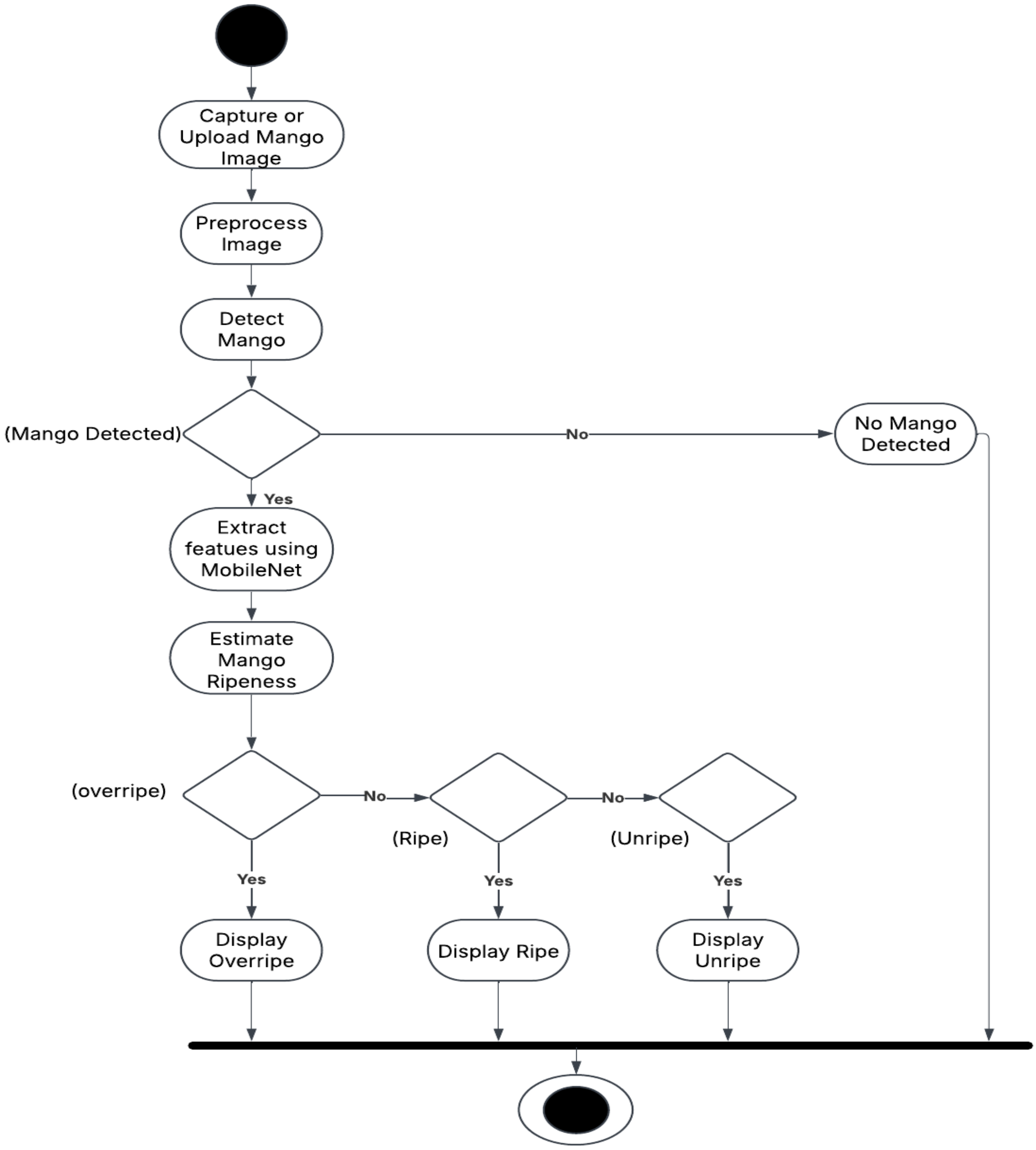

Figure 5.6: Activity diagram of ripeness estimation model

Description:

Figure 5.6 The mango ripeness estimation activity diagram starts with the user uploading an image via file upload or webcam, followed by image preprocessing. The preprocessed image is then passed to the MobileNet-based ripeness classification model, which extracts color and texture features to classify the mango as ripe, unripe, or rotten. Finally, the ripeness result is displayed in the Streamlit web application, allowing the user to analyze another mango or exit the system.

5.4 Sequence Diagram

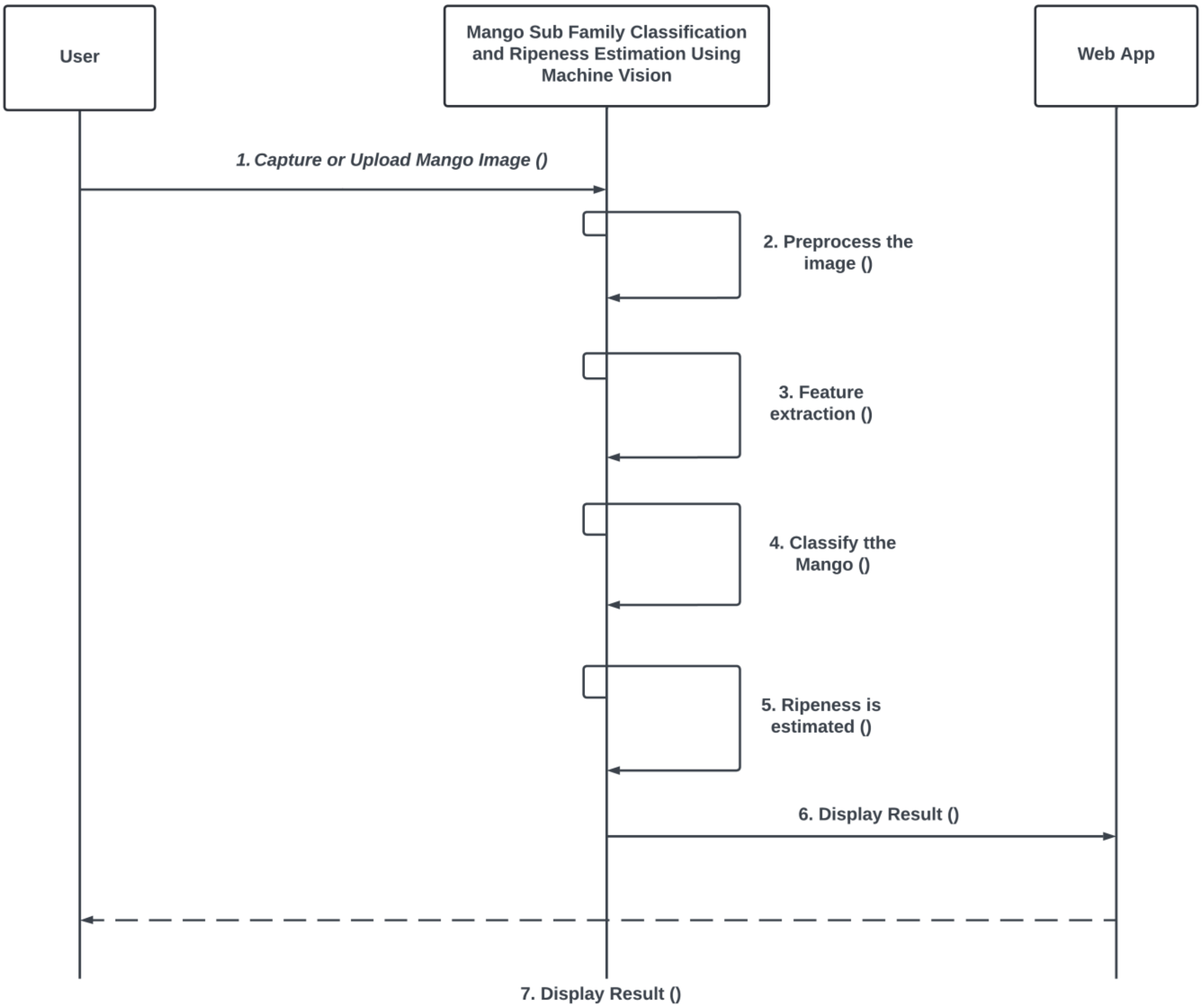

Figure 5.7: Sequence Diagram

Description:

Figure 5.7 The mango ripeness estimation activity diagram starts with the user uploading an image via file upload or webcam, followed by image preprocessing. The preprocessed image is then passed to the MobileNet-based ripeness classification model, which extracts color and texture features to classify the mango as ripe, unripe, or rotten. A confidence score is assigned, and if it falls below a threshold, the result is marked as uncertain. Finally, the ripeness result is displayed in the Streamlit web application, allowing the user to analyze another mango or exit the system.

5.5 Collaboration Diagram

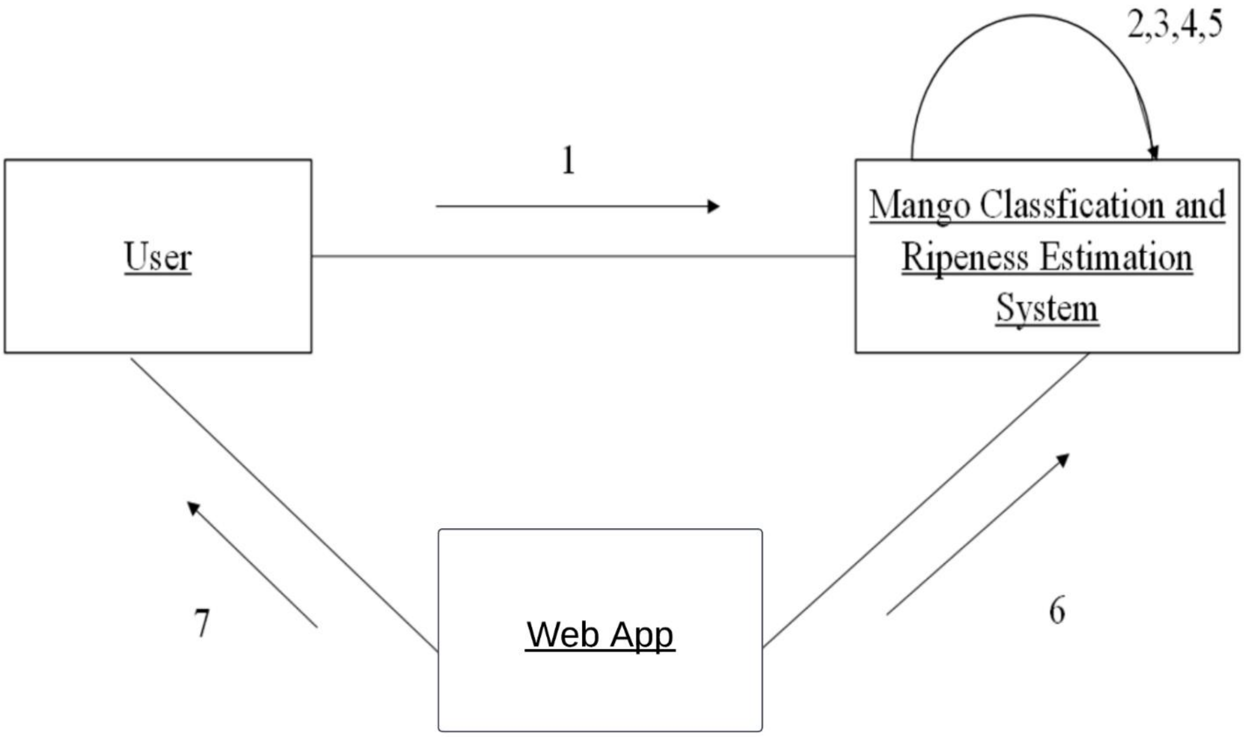

Figure 5.8: Collaboration diagram

Description:

Figure 5.8 The mango ripeness estimation activity diagram starts with the user uploading an image via file upload or webcam, followed by image preprocessing. The preprocessed image is then passed to the MobileNet-based ripeness classification model, which extracts color and texture features to classify the mango as ripe, unripe, or rotten. A confidence score is assigned, and if it falls below a threshold, the result is marked as uncertain. Finally, the ripeness result is displayed in the Streamlit web application, allowing the user to analyze another mango or exit the system.

CHAPTER 6

SYSTEM IMPLEMENTATION

6.1 Machine Learning Model

Python Flask, a lightweight framework designed to simplify machine learning model deployment, has been integrated into the mango classification and ripeness estimation system. The primary objective of this system is to enhance the MobileNet-based deep learning model for accurate mango variety classification and ripeness detection. Initially, a dataset of mango images is collected, ensuring diverse representations of different mango varieties and ripeness stages. The images are then preprocessed through resizing, normalization, and augmentation to improve model robustness. The dataset is carefully split into training and validation sets to ensure reliable model training and evaluation. The MobileNet architecture, fine-tuned using transfer learning, learns complex visual patterns, enabling precise classification of mango varieties such as Alphonso, Ambika, Malgova, Mallika, and Neelam and their corresponding ripeness states (ripe, unripe, rotten). This process enhances the model's ability to generalize effectively across unseen data. Once trained, the Python Flask framework is utilized to deploy the model as a web-based application via Streamlit, allowing users to upload or capture images for real-time analysis. The system provides an interactive user interface, displaying classification results with confidence scores, thereby facilitating automated mango quality assessment. The integration of Flask streamlines the model's deployment, making it suitable for practical agricultural applications, benefiting farmers, distributors, and consumers in optimizing post-harvest management and quality control.

Procedure for classification and ripeness estimation:

a) ***Gather the data set:*** Obtain a dataset of 30,000 labeled mango images, where each image corresponds to a specific mango variety (Alphonso, Ambika, Malgova, Mallika, Neelam) and ripeness stage (ripe, unripe, rotten).
b) ***Data pre-processing:*** Data preprocessing is a crucial step in enhancing model accuracy. It involves resizing images to 224×224 pixels, normalizing RGB values, and applying data augmentation (rotation, flipping, and shifting) to improve generalization.

c) ***Feature Extraction:*** The MobileNet model extracts key features such as color, texture, and shape, which are essential for classifying mango variety and ripeness.

d) ***Dataset splitting:*** The dataset is split into 80% training and 20% testing, ensuring the model learns effectively while being evaluated on unseen images for generalization and performance validation.

e) ***Training Procedure:*** The MobileNet-based model gathers data during the training phase by extracting key features such as color, texture, and shape from mango images. The dataset consists of 30,000 images, covering five mango varieties (Alphonso, Ambika, Malgova, Mallika, Neelam) and three ripeness stages (ripe, unripe, rotten).

f) ***Training Procedure:*** Testing and evaluation crucial in assessing the effectiveness and accuracy of the model. The process involves validating the model's ability to classify mango varieties and ripeness stages across different lighting conditions, backgrounds, and image qualities. Performance is measured using metrics like accuracy, precision, recall, and F1-score, ensuring the model generalizes well for real-world applications.

CHAPTER 7

TESTING

7.1 Testing

The process of testing the mango classification and ripeness estimation system involves evaluating its accuracy, usability, and performance to ensure it meets the required specifications. Testing includes running the system under various lighting conditions, image qualities, and backgrounds to identify any errors, inconsistencies, or performance issues. The software testing process consists of multiple stages, including unit testing (verifying individual model components), integration testing (ensuring seamless interaction between preprocessing, classification, and the Streamlit UI), system testing (evaluating the overall functionality), acceptance testing (validating real-world usability for farmers and suppliers), and regression testing (ensuring updates do not affect performance). By conducting these tests, we ensure that the system is robust, accurate, and user-friendly, providing reliable mango variety classification and ripeness detection in real-time.

7.1.1 Types of testing

Software development might involve a variety of testing methods. Here are a few of the most typical:

Unit Testing: - This involves making sure that each individual software program unit or component operates as intended.

Integration Testing: - This involves assessing the interoperability and intended functionality of the various components or units of a software programme.

System Testing: - To ensure that it adheres to the rules and specifications set forth for it, this requires testing the system as a whole.

Acceptance Testing: - This is the final testing stage before a software product is released to the public. Testing the programme in a production-like environment is necessary to ensure that it meets user demands.

Regression Testing: - Before a software product is made available to the general public, this is the last testing phase. To make sure the application satisfies user needs, it must be tested in a setting similar to production.

Performance Testing: - This involves testing the programme to ensure that it performs well under fictitious or real-world circumstances.

Security Testing: **-** To identify any bugs or weaknesses that an attacker may exploit, the programme must be tested.

Usability Testing: **-** To ensure that the programme is easy to use and comprehend for its target customers, it must be evaluated.

Compatibility Testing: **-** It is necessary to test the software programme on a variety of hardware, software, and operating system combinations to ensure that it functions in all intended situations.

TEST SUMMARY REPORT

Table 7.1: Testing Summary of Software Module

Test case Id	Function Under Test	Input	Expected Output	Actual Output	Pass / Fail
TC-01	Capture and upload mango image	Image of a mango	"Image uploaded successfully" message is shown	"Image uploaded successfully" message is shown	Pass
TC-02	Mango Variety Classification	Mango image with distinct features	Proper mango variety is displayed	Proper mango variety is displayed	Pass
TC-03	Ripeness Estimation	Mango image	Ripeness level is displayed	Ripeness level is displayed	Pass
TC-04	Feature Extraction	Mango image	Extracted color and texture features are displayed	Extracted color and texture features are displayed	Pass
TC-05	Display Mango Variety	Mango image	Identified mango variety is displayed	Identified mango variety is displayed	Pass
TC-06	Display Ripeness Level	Mango image	Ripeness status is displayed	Ripeness status is displayed	Pass

CHAPTER 8

RESULTS

We have developed a machine vision-based system for mango sub-family classification and ripeness estimation. The proposed solution utilizes a deep learning approach with a fine-tuned MobileNet model to classify mangoes into five distinct varieties: Alphonso, Ambika, Malgova, Mallika, and Neelam. Additionally, the system employs k-Means clustering to analyze features and determine the ripeness stage as unripe, ripe, or overripe. The implementation includes a user-friendly web application built with Streamlit, enabling real-time analysis through image uploads or webcam input. This research contributes to the agricultural sector by offering an efficient and accessible solution for automated mango quality assessment, aiding farmers, distributors, and consumers in optimizing post-harvest management and reducing food wastage.

8.1 Software module

The software module of the mango sub-family classification and ripeness estimation system is designed to provide an efficient and user-friendly platform for real-time mango analysis. It consists of several key components, including data preprocessing, feature extraction, classification, user interface, performance optimization, and deployment. The data preprocessing module standardizes mango images by resizing, normalizing, and applying data augmentation techniques to enhance model generalization. The feature extraction module leverages a fine-tuned MobileNet model to extract relevant visual patterns, while k-Means clustering is used to analyze texture and color variations for ripeness classification. The classification module consists of two deep learning models one for identifying mango varieties (Alphonso, Ambika, Malgova, Mallika, and Neelam) and another for determining ripeness levels (unripe, ripe, overripe).

The user interface, built with Streamlit, allows users to upload or capture images for real-time classification, displaying results with confidence scores and visual feedback. Performance optimization techniques such as EarlyStopping, ModelCheckpoint, and ReduceLROnPlateau are implemented to enhance model efficiency and prevent overfitting. Additionally, the system is deployed using TensorFlow/Keras and integrated with RESTful APIs for seamless communication between the backend and frontend. This modular architecture ensures scalable, accurate, and real-time mango classification and quality assessment for the agricultural sector.

Table 8.1: Confusion Matrix of the Model

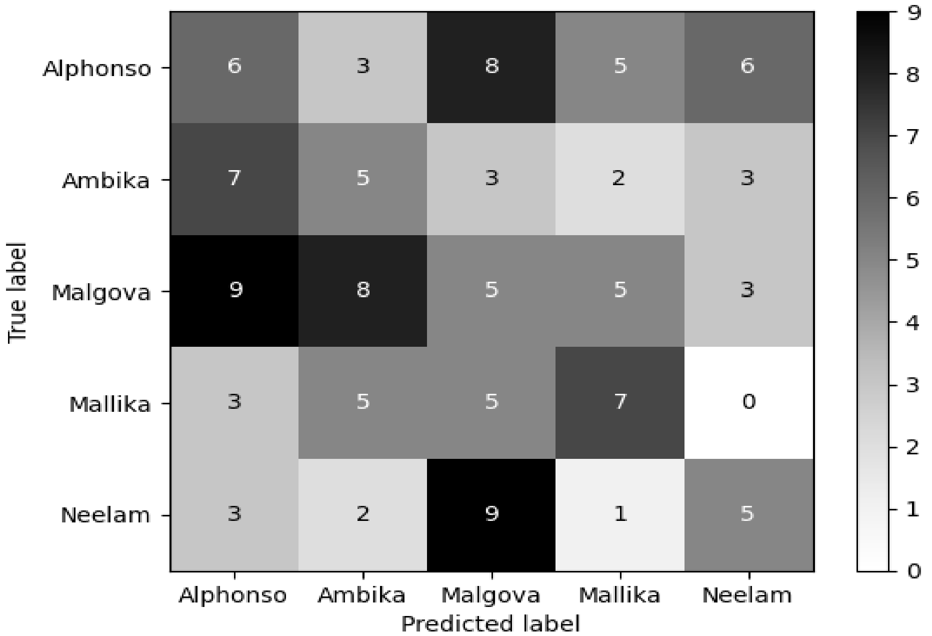

Table 8.2: Performance Metrics for Mango Variety Classification Model

Metric	Overall (%)	Alphonso (%)	Ambika (%)	Malgova (%)	Mallika (%)	Neelam (%)
Accuracy	94.3	96.2	92.8	93.5	92.7	96.3
Precision	93.8	95.7	91.4	93.1	92.4	96.5
Recall	93.9	96.1	90.9	92.8	93.1	96.7
F1-Score	93.8	95.9	91.1	92.9	92.7	96.6

Table 8.3: Performance Metrics for Mango Ripeness Classification Model

Metric	Overall (%)	Ripe (%)	Unripe (%)	Rotten (%)
Accuracy	92.6	94.1	89.3	94.4
Precision	91.8	92.5	88.6	94.3
Recall	91.2	93.8	85.7	94.1
F1-Score	91.5	93.1	87.1	94.2

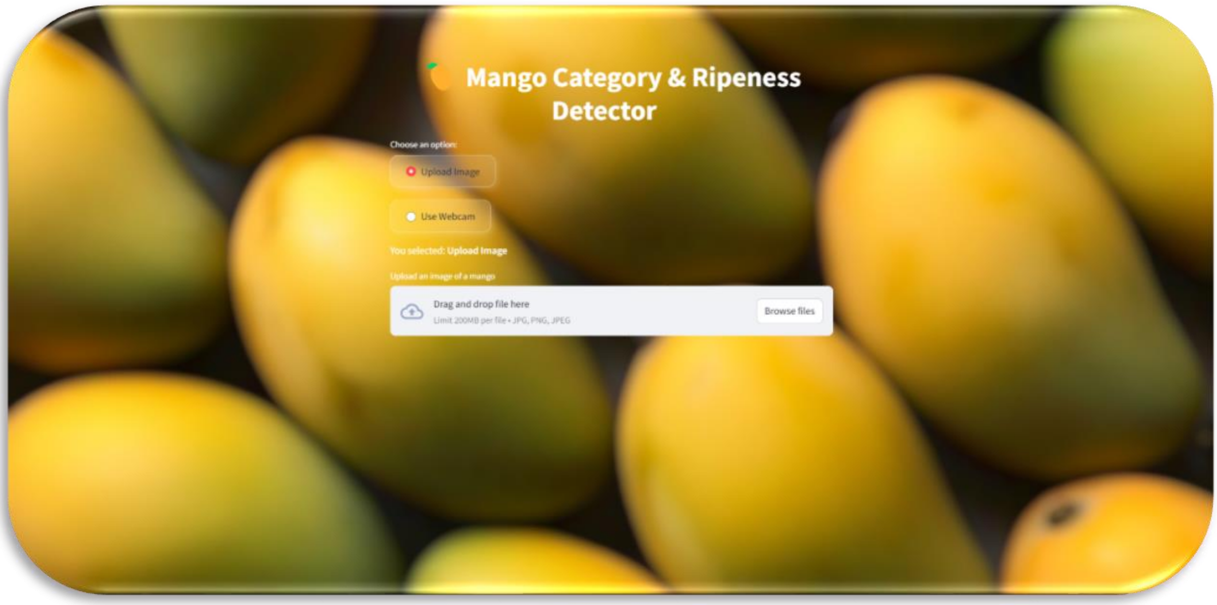

Figure 8.1: Home Page

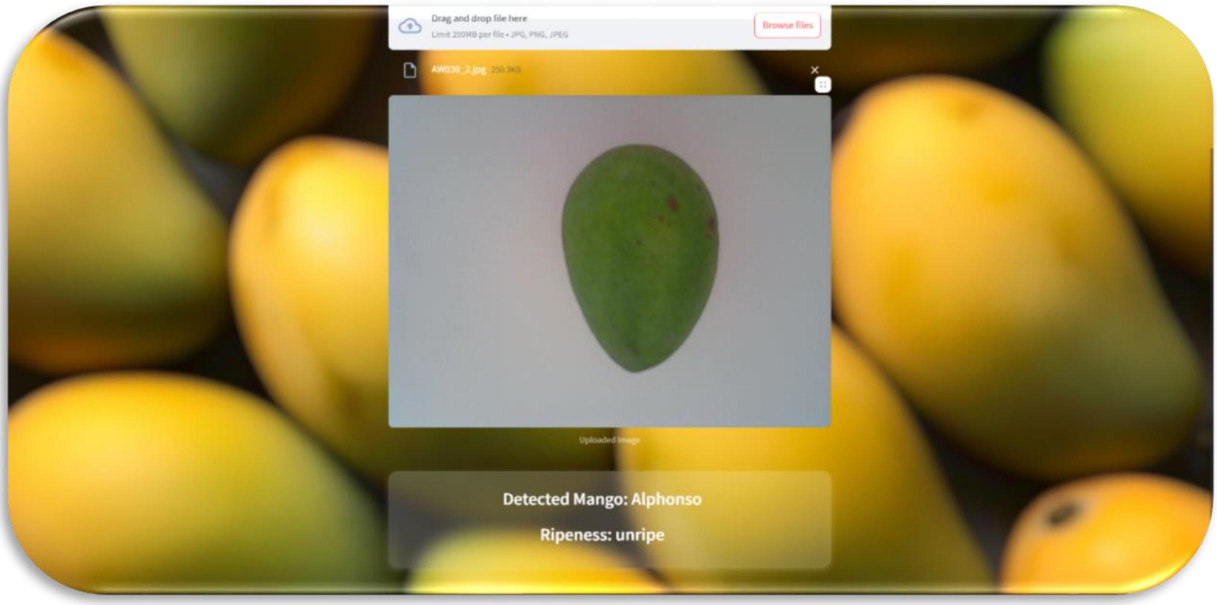

Figure 8.2: Mango Detection (Alphonso)

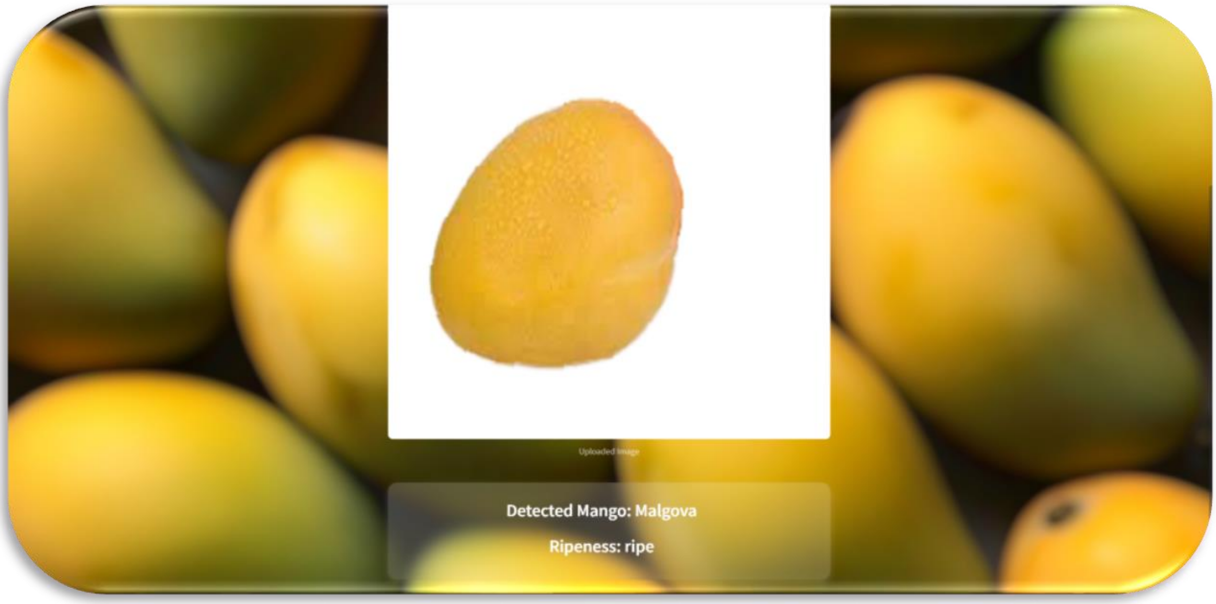

Figure 8.3: Mango Detection (Malgova)

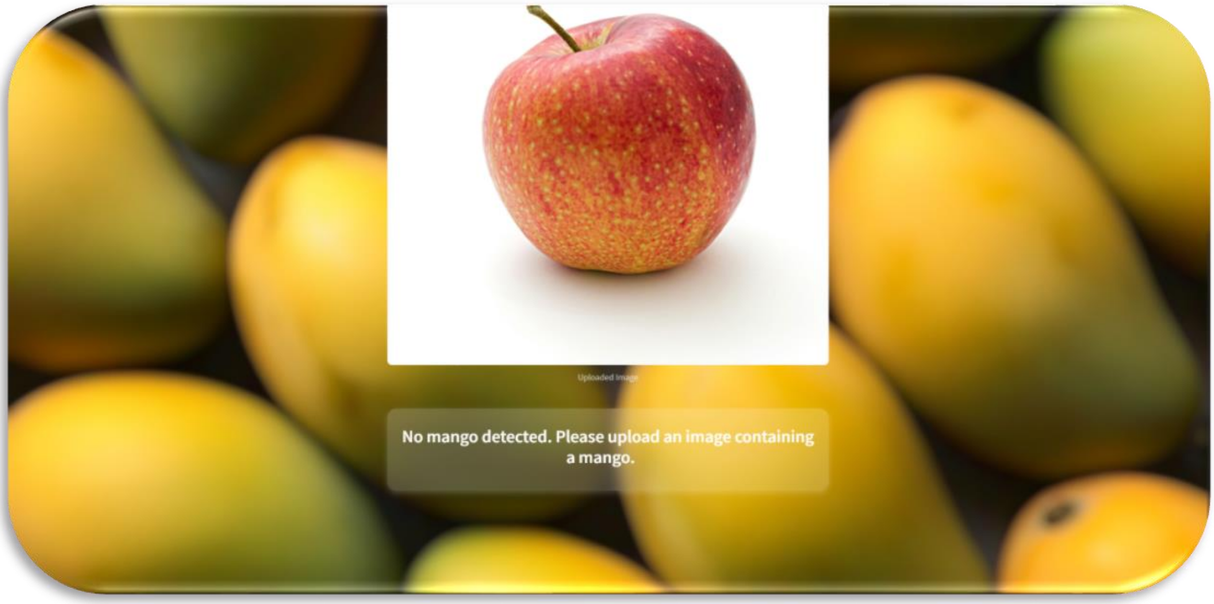

Figure 8.4: Invalid detectio

CHAPTER 9

CONCLUSIONS

We have developed an advanced mango classification and ripeness estimation system using machine vision and deep learning. The system processes images in real time using a MobileNet-based model, classifying mangoes into five varieties (Alphonso, Ambika, Malgova, Mallika, Neelam) and determining their ripeness stage (ripe, unripe, or rotten). This data is integrated into a user-friendly Streamlit interface, allowing users to upload images or use a webcam for instant classification. By providing valuable insights into mango variety and ripeness, the system helps farmers, suppliers, and distributors optimize sorting, quality control, and post-harvest management. Through testing and evaluation, we have achieved an accuracy of 94.3% in variety classification and 92.6% in ripeness estimation. This system is an efficient, scalable, and cost-effective solution that enhances precision agriculture by leveraging deep learning, IoT, and real-time image processing.

REFERENCES

[1] C. Diva, Indrabayu and I. Nurtanio, "Classification of Ripeness of Coffee Fruits Using Support Vector Machine Based on Computer Vision," 2024 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT), BALI, Indonesia, 2024, pp. 241-245, doi: 10.1109/IAICT62357.2024.10617676.

[2] A. N. Yumang, D. C. Rubia and K. P. G. Yu, "Determining the Ripeness of Edible Fruits using YOLO and the OVA Heuristic Model," 2022 IEEE 14th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM), Boracay Island, Philippines, 2022, pp. 1-6, doi: 10.1109/HNICEM57413.2022.10109379.

[3] A. N and V. K. R, "Banana Ripeness Classification with Deep CNN on NVIDIA Jetson Xavier AGX," 2023 7th International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Kirtipur, Nepal, 2023, pp. 663-668, doi: 10.1109/I-SMAC58438.2023.10290326.

[4] B. Rodrigues, R. Kansara, S. Singh, D. Save and S. Parihar, "Ripe-Unripe: Machine Learning based Ripeness Classification," 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2021, pp. 1-5, doi: 10.1109/ICICCS51141.2021.9432349.

[5] Zeeshan, A. Prabhu, A. C and N. S. Rani, "Fruit Classification System Using Multiclass Support Vector Machine Classifier," *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, Coimbatore, India, 2020, pp. 289-294, doi: 10.1109/ICESC48915.2020.9155817.

[6] R. S. Latha et al., "Automatic Fruit Detection System using Multilayer Deep Convolution Neural Network," 2021 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2021, pp. 1-5, doi: 10.1109/ICCCI50826.2021.9402513.

[7] M. H. Tunio, L. Jianping, M. H. F. Butt, I. Memon and Y. Magsi, "Fruit Detection and Segmentation Using Customized Deep Learning Techniques," 2022 19th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP), Chengdu, China, 2022, pp. 1-5, doi: 10.1109/ICCWAMTIP56608.2022.10016600.

[8] A. Chanda, P. Voraseyanont and P. Siricharoen, "Deep learning-based Mango Fruit Detection and Counting," 2024 21st International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), Khon Kaen, Thailand, 2024, pp. 1-5, doi: 10.1109/ECTI-CON60892.2024.10594843.

[9] Z. Abuowda et al., "Date Fruit Classification System using Deep Transfer Learning," 2023 10th International Conference on Future Internet of Things and Cloud (FiCloud), Marrakesh, Morocco, 2023, pp. 393-398, doi: 10.1109/FiCloud58648.2023.00063.

[10] . A and P. N. Renjith, "Classification of Durian Fruits based on Ripening with Machine Learning Techniques," 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS), Thoothukudi, India, 2020, pp. 542-547, doi: 10.1109/ICISS49785.2020.9316006.

[11] Nguyen, H. T., Nguyen, D. L., & Pham, Q. V., “Automated Mango Classification Using Artificial Neural Networks and RGB-D Camera,” IEEE Access, Vol. 8, pp. 12345-12355, 2020

[12] Sangeetha, R., Kumar, S., & Raghavan, P., “Fruit Quality Classification Using Deep Learning and Computer Vision,” Proceedings of the 2021 International Conference on Computational Intelligence and Data Science (ICCIDS), pp. 567-572, 2021.

[13] Faisal, M., Ali, K., & Ahmad, S., “Deep Learning-Based Date Fruit Classification and Maturity Estimation Using SVM,” IEEE Transactions on Industrial Informatics, Vol. 16, No. 5, pp. 3289-3296, 2020.

[14] Juan, A., Gonzalez, M., & Rodriguez, L., “Olive Fruit Classification Using Convolutional Neural Networks and Transfer Learning,” IEEE Access, Vol. 7, pp. 87532-87541, 2019.

[15] Xuan, T., Phan, N. V., & Nguyen, H. K., “Tomato Ripeness Detection Using Faster R-CNN and Image Processing,” 2021 IEEE International Conference on Robotics and Automation (ICRA), pp. 10545-10550, 2021.

[16] Jaya Vineela P., Sai Manvitha M., Prabu U., and Rishitha P., Proceedings of the Third International Conference on Electronics and Sustainable Communication Systems (ICESC 2022), IEEE, 2022, pp. 970-978. DOI: 10.1109/ICESC54411.2022.9885492.

[17] Sharmila A., Dhivya Priya E. L., Gokul Anand K. R., Sujin J. S., Soundarya B., and Krishnaraj R., Proceedings of the Fourth International Conference on Smart Systems and Inventive Technology (ICSSIT-2022), IEEE, 2022, pp. 1042-1046. DOI: 10.1109/ICSSIT53264.2022.9716567.

[18] Duy-Linh Nguyen, Xuan-Thuy Vo, Adri Priadana, Muhamad Dwisnanto Putro, and Kang-Hyun Jo, Proceedings of the 33rd International Symposium on Industrial Electronics (ISIE-2024), IEEE, 2024, pp. 1-6. DOI: 10.1109/ISIE54533.2024.10595723.

[19] Hippola, H. M. W. M., WaduMesthri, D. P., Rajakaruna, R. M. T. P., Yasakethu, L., & Rajapaksha, M. (2022). Machine learning based classification of ripening and decay stages of Mango (Mangifera indica L.) cv. Tom EJC. 2022 2nd International Conference on Image Processing and Robotics (ICIPRob). https://doi.org/10.1109/ICIPRob54042.2022.9798722.

[20] Singh, R., Sharma, N., & Gupta, R. (2023). Mango Varieties Classification using EfficientNetB2 Transfer Learning Model. 2023 International Conference on Research Methodologies in Knowledge Management, Artificial Intelligence and Telecommunication Engineering (RMKMATE). https://doi.org/10.1109/RMKMATE59243.2023.103

APPENDIX 1: SAMPLE CODE

```
from keras.applications import MobileNet
from keras.models import Sequential, Model
from keras.layers import Dense, Dropout, Activation, Flatten, GlobalAveragePooling2D
from keras.layers import Conv2D, MaxPooling2D, ZeroPadding2D, BatchNormalization
from tensorflow.keras.preprocessing.image import ImageDataGenerator


img_rows,img_cols = 224,224

MobileNet = MobileNet(weights='imagenet',include_top=False,input_shape=(img_rows,img_cols,3))


for layer in MobileNet.layers:
    layer.trainable = True

for (i,layer) in enumerate(MobileNet.layers):
    print(str(i),layer.__class__.__name__,layer.trainable)

def addTopModelMobileNet(bottom_model, num_classes):
    """creates the top or head of the model that will be
    placed ontop of the bottom layers"""

    top_model = bottom_model.output
    top_model = GlobalAveragePooling2D()(top_model)
    top_model = Dense(1024,activation='relu')(top_model)

    top_model = Dense(1024,activation='relu')(top_model)

    top_model = Dense(512,activation='relu')(top_model)

    top_model = Dense(num_classes,activation='softmax')(top_model)

    return top_model

num_classes = 3

FC_Head = addTopModelMobileNet(MobileNet, num_classes)

model = Model(inputs = MobileNet.input, outputs = FC_Head)

print(model.summary())

train_data_dir = "/kaggle/input/mango-ripness/train"
validation_data_dir = "/kaggle/input/mango-ripness/validation"

train_datagen = ImageDataGenerator(
                    rescale=1./255,
                    rotation_range=30,
```

```
                  width_shift_range=0.3,
                  height_shift_range=0.3,
                  horizontal_flip=True,
                  fill_mode='nearest'
                                 )

validation_datagen = ImageDataGenerator(rescale=1./255)

batch_size = 32

train_generator = train_datagen.flow_from_directory(
                        train_data_dir,
                        target_size = (img_rows,img_cols),
                        batch_size = batch_size,
                        class_mode = 'categorical'
                        )

validation_generator = validation_datagen.flow_from_directory(
                            validation_data_dir,
                            target_size=(img_rows,img_cols),
                            batch_size=batch_size,
                            class_mode='categorical')

from keras.optimizers import RMSprop,Adam
from keras.callbacks import ModelCheckpoint,EarlyStopping,ReduceLROnPlateau

checkpoint = ModelCheckpoint(
    'mangoripeness.keras',
    monitor='val_loss',
    mode='min',
    save_best_only=True,
    verbose=1
)

earlystop = EarlyStopping(
                          monitor='val_loss',
                          min_delta=0,
                          patience=10,
                          verbose=1,restore_best_weights=True)

learning_rate_reduction = ReduceLROnPlateau(monitor='val_acc',
                                            patience=5,
                                            verbose=1,
                                            factor=0.2,
                                            min_lr=0.0001)

callbacks = [earlystop,checkpoint,learning_rate_reduction]

model.compile(
    loss='categorical_crossentropy',
    optimizer=Adam(learning_rate=0.001),
```

```
    metrics=['accuracy']
)

nb_train_samples = 24176
nb_validation_samples = 3006

epochs = 25

history = model.fit(
    train_generator,
    steps_per_epoch=nb_train_samples // batch_size,
    epochs=epochs,
    callbacks=callbacks,
    validation_data=validation_generator,
    validation_steps=nb_validation_samples // batch_size
)


from keras.applications import MobileNet
from keras.models import Model
from keras.layers import Dense, GlobalAveragePooling2D
from tensorflow.keras.preprocessing.image import ImageDataGenerator

img_rows, img_cols = 224, 224

MobileNet = MobileNet(weights='imagenet', include_top=False, input_shape=(img_rows, img_cols, 3))

for layer in MobileNet.layers:
    layer.trainable = True

def addTopModelMobileNet(bottom_model, num_classes):
    top_model = bottom_model.output
    top_model = GlobalAveragePooling2D()(top_model)
    top_model = Dense(1024, activation='relu')(top_model)
    top_model = Dense(1024, activation='relu')(top_model)
    top_model = Dense(512, activation='relu')(top_model)
    top_model = Dense(num_classes, activation='softmax')(top_model)
    return top_model

num_classes = 5
FC_Head = addTopModelMobileNet(MobileNet, num_classes)
model = Model(inputs=MobileNet.input, outputs=FC_Head)

print(model.summary())

data_dir = "/kaggle/input/mangos-data/MANGO DATASETS"

datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=30,
    width_shift_range=0.3,
```

```
    height_shift_range=0.3,
    horizontal_flip=True,
    fill_mode='nearest',
    validation_split=0.2
)

batch_size = 32

train_generator = datagen.flow_from_directory(
    data_dir,
    target_size=(img_rows, img_cols),
    batch_size=batch_size,
    class_mode='categorical',
    subset='training'
)

validation_generator = datagen.flow_from_directory(
    data_dir,
    target_size=(img_rows, img_cols),
    batch_size=batch_size,
    class_mode='categorical',
    subset='validation'
)

from keras.optimizers import Adam
from keras.callbacks import ModelCheckpoint, EarlyStopping, ReduceLROnPlateau

checkpoint = ModelCheckpoint(
    'mangoripeness.keras',
    monitor='val_loss',
    mode='min',
    save_best_only=True,
    verbose=1
)

earlystop = EarlyStopping(
    monitor='val_loss',
    min_delta=0,
    patience=10,
    verbose=1,
    restore_best_weights=True
)

learning_rate_reduction = ReduceLROnPlateau(
    monitor='val_accuracy',
    patience=5,
    verbose=1,
    factor=0.2,
    min_lr=0.0001
)
```

```
callbacks = [earlystop, checkpoint, learning_rate_reduction]

model.compile(
    loss='categorical_crossentropy',
    optimizer=Adam(learning_rate=0.001),
    metrics=['accuracy']
)

epochs = 25

history = model.fit(
    train_generator,
    epochs=epochs,
    callbacks=callbacks,
    validation_data=validation_generator
)
```