

DETECTION OF QUALITY MONITORING USING DEEP LEARNING FOR DRAGON FRUITS

A Project Report Phase - II

Submitted in partial fulfillment of the requirements for the award of Bachelor of
Engineering Degree in Electronics and Communication Engineering

by

**R THANU SRI (39130470)
DODDI VAISHNAVI (39130121)**



**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
SCHOOL OF ELECTRICAL AND ELECTRONICS**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHISALAI,
CHENNAI - 600119**

APRIL - 2023



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with "A" grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **R THANU SRI (39130470)** and **DODDI VAISHNAVI (39130121)** who carried out the project entitled "**Detection of Quality monitoring using deep learning for Dragon fruit**" under supervision from November 2022 to April 2023.

Internal Guide

Dr. A. ARANGANATHAN, M Tech, Ph.D.

Head of the Department

Dr. T. RAVI, M.E., Ph.D.

Submitted for Viva voce Examination held on 26.04.23

Internal Examiner

External Examiner

DECLARATION

We, **R THANU SRI (39130470)** and **DODDI VAISHNAVI (39130447)** with this declare that the Project Report entitled “DETECTION OF QUALITY MONITORING USING DEEP LEARNING FOR DRAGON FRUIT” done by us under the guidance of **Dr. A. ARANGANATHAN MTech, Ph.D.**, Associate Professor, Department of Electronics and Communication Engineering, at SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY, CHENNAI is submitted in partial fulfilment of the requirements of the award of Bachelor of Engineering degree in ELECTRONICS AND COMMUNICATION ENGINEERING.

DATE: 26.04.23

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATES

1.



2.



ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. We are grateful to them.

We convey our thanks to **Dr. N.M. NANDHITHA M.E., Ph. D, Dean**, School of Electronics, **Dr. T. RAVI M.E., Ph.D.**, Head of the Department of Electronics and Communication Engineering for providing us necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to our Project Guide **Dr. A. ARANGANATHAN MTech, Ph.D.**, Associate Professor, Department of Electronics and Communication Engineering for his valuable guidance, suggestions, and constant encouragement paved way for the successful completion of our phase-2 project work.

We wish to express our thanks to all Teaching and Non-teaching staff members of the **Department of Electronics and Communication Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

The agriculture being a main source of income in many developing countries such as India, Indonesia, etc. The economic development of these countries depends on the GDP (Gross Domestic Progress) rate of the agricultural products. However due to miscalculations in the maturity of the fruits and vegetables leads to the wastage of foods. In general, many measure were taken to minimize the food spoilage and by tracking each stage of the vegetables and fruits carefully, but resulted in a hefty human labor, and weariness. Specifically, the non-climacteric fruit such as the dragon fruit requires much attention as it has to be harvested after it is ripened and cannot be ripened after harvesting using the hastening ripening process such as the ethylene, carbide, and CO₂ etc. So, the paper has put forth the application to identify the mellowness in the dragon fruit using the RESNET 152 a deep learning convolution neural network to identify the dragon fruits mellowness and it's time to harvest. The model was trained using the python and the tensor flow. The developed structure was trained using the pictures of the dragon fruit in the different stages of its mellowness and was tested using the region of convergence and the confusion matrix with 100 new data. The testing was carried with the different number of epochs ranging from 10 to 500. The results obtained were more accurate compared to the VGG16 /19 in the terms of accuracy and loss in training and testing.

TABLE OF CONTENTS

Chapter No.	TITLE	Page No.
	ABSTRACT	v
	LIST OF FIGURES	viii
	LIST OF TABLES	
1	INTRODUCTION	1
2	LITERATURE SURVEY	
	2.1 Inferences from Literature Survey	4
	2.2 Open problems in Existing System	8
3	REQUIREMENTS ANALYSIS	
	3.1 Feasibility Studies/Risk Analysis of the Project	9
	3.2 Software Requirements Specification Document	10
4	DESCRIPTION OF PROPOSED SYSTEM	
	4.1 Aim	16
	4.2 Architecture / Overall Design of Proposed System	34
	4.3 Description of Software for Implementation and Testing plan of the Proposed Model/System	38
	4.4 Project Management Plan	43
5	IMPLEMENTATION DETAILS	
	5.1 Development and Deployment Setup	47
	5.2 Algorithms	48
	5.3 Testing	52
6	RESULTS AND DISCUSSION	54

7	CONCLUSION	
	7.1 Conclusion	56
	7.2 Future work	56
	7.3 Implementation Issues	57
	REFERENCES	59

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	Page No.
3.2.1	Visual Studio Interference	14
4.1.1	Block diagram of proposed model	16
4.2.1	System Architecture of the proposed model	34
4.2.2	Small Image and Kernels	36
5.2.1	CNN layers	49
5.2.2	Simple CNN Architecture	50
5.2.3	Comparison of 56 - layer and 20 – layer architecture	51
5.2.4	RESNET 34 - Architecture	51
5.3.1	Dataset of Dragon fruit	53
6.1	Final Output	54
6.2	Final Graph Output	55

CHAPTER 1

INTRODUCTION

Agriculture is very important and essential for humans because they directly depend on it for food production. Especially, fruits are typically bought by every household and rich in nourishment; thus, it is required a continuous supply and production to satisfy the demand of the growing world population. For this reason, the entire agri-food sector chain experiences increasing challenges, which require to apply new innovative technologies in order to improve its productivity. Among other applications, computational technologies have been adopted for fruits recognition tasks and the effective detection of specific defects of its, both in wholesale and retail markets. Computer vision is one of the most used technological tools in the agro-industrial field, both in automatic fruit harvesting, fruit sorting machines, and fruit scanning in supermarkets.

All vision systems typically include different types of data generated by sensors or cameras. This data can be RGB images, RGB depth images (RGB-D), hyperspectral images, among many other types. So that, due to different computational methods and algorithms, required features must be extracted and processed to perform the corresponding task to the fruit industry sector. For example, in supermarkets, a fruit recognition process is required or in an orchard for harvest, the accurate detection of fruit.

Nowadays, artificial intelligence (AI) is a field with several practical applications in a wide range of industries and active research topics. The main challenge for AI is to solve the tasks that people intuitively solve, but hard to implement computationally. Therefore, AI systems must have the ability to acquire their knowledge, extracting raw data patterns, which is known as machine learning. Thus, AI-based techniques are very useful to solve complex problems where traditional methods would not be efficient. Machine learning (ML) allows researchers and developers to computationally address problems related to the knowledge of the real world.

ML endows computers with the ability to act without being explicitly programmed, building algorithms to recognize patterns on the data and make predictions based on it. ML-based systems are applied in several areas, such as information analysis, agriculture, ecology, mining, urban planning, defense, space exploration, among others. Currently, deep learning (DL) is one of the most used ML-based methods. An important characteristic of DL is that it has high levels of abstraction and the ability to automatically learn patterns present in images.

Convolutional Neural Network (CNN) is the main DL architecture used for image processing. CNNs is a kind of artificial neural networks (ANNs) that use convolution operations in at least one of their layers. Since 2012, when Krizhevsky et al. won the ImageNet competition (ILSVRC), CNNs have gained great popularity as an efficient method for image classification in many fields. Specifically in agriculture, CNN-based approaches have been used for fruit classification and fruit detection. In order to define the study areas of our review, we identify fruit classification task as the determination of the class according to their specific type. Besides, fruit quality control is focused on the determination of internal and external damages, as well as its maturity degree. On the other hand, fruit detection is oriented to carry out an automatic harvest.

Dragon fruit, known as pitaya, is a tropical fruit that is widely grown in about 20 countries and territories in Asia, the Middle East and America, with a large concentration in the Asia-Pacific region, especially in Vietnam, Thailand, Indonesia, Philippines, Mainland China, and Taiwan, because of its appealing taste and rich nutritional properties.

Dragon fruit contains many nutrients beneficial to human health, such as vitamin A, vitamin C, moisture, protein, etc. The quality of dragon fruit is evaluated largely by the shape and external defects of the fruit. In Vietnam, the annual output of dragon fruit is increasing.

After harvesting, dragon fruit are classified according to the standards of each different country in order to export them. The current classification is carried out by humans. This process is time-consuming and its accuracy is not high. Dragon fruit is divided into different groups, depending on the weight, volume, and external defects, according to the import standards of individual countries

Climate change-induced environmental stresses and limited agricultural land demanding intensification of sustainable agriculture over degraded land via crop diversification strategies. Dragon fruit is one of the potential options and popularizing in resource-poor degraded lands apart from its several nutraceutical advantages. Hence, understanding of facts related to its consumer acceptability and maintaining high quality for marketing and processing is highly essential.

Therefore, in this study, we have developed grading and sorting techniques for dragonfruit using machine learning algorithms CNN based on a thorough review of techniques or algorithms available to detect and classify fruit quality using various features of fruits and vegetables. Working of these algorithms is based on the, shape, size, weight, color, and diseases of dragon fruits. Raspberry functionality counts the total number of fruits that are available in the fruit bucket and these are separated by their maturity level using machine learning algorithms.

CHAPTER 2

LITERATURE SURVEY

2.1 INTERFERENCES FROM THE LITERATURE SURVEY

The paper titled "Attempting to estimate the unseen-correction for occluded fruit in tree fruit load estimation by machine vision with deep learning" by K.B. Walsh, K. B. Koirala, and Z. Wang was published in the journal *Agronomy* in 2021 [1]. The study proposes a deep learning-based approach to estimate the number of occluded fruits in tree fruit load estimation using machine vision. The authors aimed to improve the accuracy of fruit load estimation by developing a method to account for the occluded fruit, which are often overlooked by existing machine vision systems. The study presents promising results in accurately estimating the fruit load in tree crops, which has significant implications for improving the efficiency of fruit yield estimation and harvest management.

The paper titled "Yolov3: an incremental improvement" by A. Farhadi and J. Redmon was published in the Computer Vision and Pattern Recognition conference in 2018 [2]. The paper introduces a state-of-the-art object detection model, called YOLOv3, which stands for "You Only Look Once version 3". The YOLOv3 model builds on the success of its predecessors, YOLOv1 and YOLOv2, and improves the detection accuracy by incorporating several new techniques, such as feature pyramid networks, anchor boxes, and multiscale predictions. The paper presents a comprehensive evaluation of the YOLOv3 model on various benchmark datasets and demonstrates its superior performance compared to other popular object detection methods. The YOLOv3 model has since become widely adopted in the computer vision community and has been utilized in a wide range of applications, including autonomous vehicles, surveillance systems, and robotics.

The paper titled "Mellowness Detection of Dragon Fruit Using Deep Learning Strategy" was introduced by Dr. T. Vijayakumar and Mr. R. Vinothkanna in 2022 [3]. The study proposes a deep learning-based approach for detecting the mellowness of dragon fruit, which is an important quality indicator for determining its ripeness and marketability. The authors utilized a convolutional neural network (CNN) to extract features from the images of dragon fruit and classify them into different mellowness categories. The proposed model achieved high accuracy in predicting the mellowness of dragon fruit, which can significantly benefit the farmers and traders in optimizing the harvesting time and reducing the post-harvest losses. The paper presents a novel application of deep learning in the agricultural sector and has the potential to pave the way for developing similar models for detecting the quality of other fruits and vegetables.

The paper titled "Polynomial-based smooth trajectory planning for fruit-picking robot manipulator" by H. Wang, Q. Zhao, H. Li, and R. Zhao was published in the journal Information Processing in Agriculture in 2022 [4]. The study proposes a new trajectory planning method based on polynomial functions for the fruit-picking robot manipulator. The proposed method aims to generate smooth and efficient trajectories for the robot manipulator to follow while picking fruits from trees. The authors evaluate the performance of the proposed method by comparing it to other trajectory planning methods, and demonstrate its superiority in terms of trajectory smoothness, picking accuracy, and picking speed. The study has significant implications for the development of advanced fruit-picking robots, which can improve the efficiency and productivity of the agriculture industry while reducing labor costs and enhancing fruit quality.

In their paper titled "Review of Convolutional Neural Network Applied to Fruit Image Processing", José Naranjo-Torres, Marco Mora, Ruber Hernández-García, Ricardo J. Barrientos, Claudio Fredes, and Andres Valenzuela provide an overview of the use of convolutional neural networks (CNNs) in fruit image processing [5]. The paper provides a comprehensive review of the state-of-the-art in this field, including

the use of CNNs for fruit detection, recognition, and classification. The authors also discuss the challenges associated with fruit image processing, such as variations in fruit size, shape, and color, as well as the presence of occlusions and deformations. The paper concludes with a discussion of future research directions in this area, highlighting the potential for using CNNs in applications such as precision agriculture, fruit quality assessment, and automated fruit sorting. The paper provides valuable insights for researchers and practitioners in the field of computer vision and agriculture.

In their paper titled "Efficient Data Labeling Method for Multi-shape, Cross-Species Fruit Detection," Wenli Zhang, Kaizhen Chen, Chao Zheng, Yuxin Liu, and Wei Guo propose a novel approach to labeling data for fruit detection [10]. The authors note that existing methods for labeling fruit detection data are often time-consuming and labor-intensive, particularly when dealing with multi-shaped and cross-species fruits. To address this issue, the authors propose an efficient labeling method that leverages existing bounding box annotations and a modified k-means clustering algorithm. Their proposed method significantly reduces the time and effort required for data labeling while maintaining high accuracy in fruit detection. The authors demonstrate the effectiveness of their approach through experiments on various datasets, showcasing its potential for real-world applications in agriculture and food processing industries.

The "SSD: Single Shot Multi-Box Detector" paper was published in the proceedings of the European Conference on Computer Vision in 2016 by W. Liu, D. Anguelov, D. Erhan, et al [11]. The paper proposed a real-time object detection algorithm that can detect objects of various scales and aspect ratios with a single network architecture. The algorithm uses a set of convolutional feature maps to predict object locations and class probabilities, and it integrates prior knowledge of object sizes and shapes to improve detection accuracy. The SSD algorithm achieved state-of-the-art performance on several benchmark datasets, demonstrating its effectiveness and efficiency for real-time object detection applications. The paper has

become a cornerstone in the field of computer vision, influencing many subsequent research efforts and serving as a foundation for many modern object detection algorithms

The article titled "Symmetry-based 3d shape completion for fruit localization for harvesting robots," authored by Y. Ge, Y. Xiong, and P. J. From, was published in the Biosystems Engineering journal in 2020 [12]. The paper describes a novel approach to improving fruit localization for harvesting robots by using symmetry-based 3D shape completion. The authors propose a methodology that incorporates both geometric and color features to generate a 3D model of fruits, which enables accurate localization and harvesting by robots. The research presented in this article provides a significant contribution to the field of robotics and agricultural automation, which can ultimately lead to more efficient and sustainable farming practices.

The article titled "Research on the application of rigid-flexible compound driven fruit picking robot design in realizing fruit picking" by Z. Yang, published in the Journal of Physics: Conference Series, presents a novel approach to designing fruit picking robots [13]. The author proposes a rigid-flexible compound driven robot design that can adapt to complex and changing environments, making it ideal for fruit picking. The study describes the mechanical structure and motion control system of the robot, and presents experimental results demonstrating its effectiveness in fruit picking tasks. Overall, this research contributes to the development of efficient and reliable robotics solutions for fruit harvesting, which could ultimately improve productivity and reduce labor costs for farmers.

The paper titled "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks" was authored by S. Ren, K. He, R. Girshick, and J. Sun, and was published in the Advances in Neural Information Processing Systems journal in 2015 [8]. The paper proposes a novel method for object detection that uses a region proposal network (RPN) to generate potential object locations in an image. This is combined with a Fast R-CNN detector to classify and refine the proposed

regions. The proposed method achieves state-of-the-art results on several benchmark datasets, while also being significantly faster than previous approaches. The paper has since become a highly cited and influential work in the field of computer vision and object detection.

2.2 OPEN PROBLEMS IN EXISTING SYSTEM

1. Lack of diverse and representative data sets

Deep learning algorithms require large and diverse data sets to be trained effectively. However, there is a limited amount of publicly available data on dragon fruit, which makes it difficult to train deep learning models accurately. Additionally, dragon fruit has a unique appearance and texture that can vary significantly based on factors such as ripeness, lighting conditions, and camera angles. Therefore, there is a need to develop robust deep learning models that can accurately detect dragon fruit under different conditions.

2. Improve the interpretability of deep learning models

Another challenge is the need to improve the interpretability of deep learning models for dragon fruit detection. Deep learning models are often considered as "black boxes," and it can be challenging to understand how they make predictions. Developing transparent and interpretable deep learning models can increase user trust in the technology and provide insights into how the models work.

3. Requirement of high - end computer computing sources is expensive

The existing systems for detecting dragon fruit using deep learning require high-end computing resources, which can be expensive and limit their practical applications. There is a need to develop efficient deep learning models that can run on low-end devices such as smartphones or embedded systems, making the technology more accessible and cost-effective.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDIES/RISK ANALYSIS OF THE PROJECT

The first step would be to clearly define the problem to be solved. In this case, we want to use deep learning to detect dragon fruit. We will need to consider factors such as the types of dragon fruits that we are trying to detect, the environment in which the detection will take place, and the specific goals of the project.

Once the problem is defined, data is gathered to train the deep learning model. This involves taking pictures of different types of dragon fruit from various angles and lighting conditions, and labeling each image with the correct classification (e. g. "dragon fruit" or "not dragon fruit"). After gathering data, we build and train the deep learning model. This will likely involve using a convolutional neural network (CNN), a type of deep learning architecture commonly used for image classification. We then divide the collected data into training, validation, and test sets, and use an appropriate loss function and optimizer to train the model. After this we evaluate its performance on the test set. This will involve calculating metrics such as accuracy, precision, and recall, and comparing the results to a baseline model or other benchmarks. With the model's performance metrics in hand, we will have to conduct a feasibility analysis to determine whether the detection of dragon fruit using deep learning is feasible. This might involve considering factors such as the cost and availability of the required hardware and software, the feasibility of deploying the model in the field, and any legal or ethical considerations.

Finally, we conduct a risk analysis to identify any potential risks associated with the project. This might involve considering factors such as the accuracy and reliability of the model, the potential for false positives or false negatives, and any safety or security risks associated with the deployment of the model in the field.

3.2 SOFTWARE AND HARDWARE REQUIREMENTS SPECIFICATION DOCUMENTS

3.2.1 Hardware Requirements

For this Project a PC with the following specifications is required. They are:

System - Pentium i3 Processor or any next updated Generation

Hard disk – 500GB or above

Monitor – 15inch LED

Input Devices – Keyboard,

RAM – 4GB or above

3.2.2 Software Requirements

1. Operating System

Windows 10 is a Microsoft operating system for personal computers, tablets, embedded devices, and internet of things devices. The requirements of Windows 10 OS are as follows:

Processor: 1 gigahertz (GHz) or faster processor or system-on-a-chip (SoC)

RAM: 1 gigabyte (GB) for 32-bit or 2 GB for 64-bit

Hard disk space: 16 GB for 32-bit OS 20 GB for 64-bit OS

Graphics card: DirectX 9 or later with Windows Display Driver Model 1.0

Display: 800x600

2. Coding Language

The coding language we used in this project is Python. Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation via the off-side rule.

Python is an old Programming Language (1991) compared to many other Programming Languages like C# (2000), Swift (2014), Java (1995), PHP (1995).

- Python is highly extendable due to its high number of free available Python Packaged and Libraries
- Python can be used on all platforms (Windows, macOS and Linux).
- Python is multi-purpose and can be used for to program Web Applications, Enterprise Applications and Embedded Applications, and within Data.
- Science and Engineering Applications.
- The popularity of Python is growing fast.

The growing Python community makes it easy to Find documentation, code examples and get help when needed

- Introduction to Python: Python is an open source and cross-platform programming language, that has become increasingly popular over the last ten years. It was First released in 1991. Latest version is 3.7.0. Python is the reference implementation of the Python programming language. Written in C, Python is the default and most widely-used implementation of the language.
- Python is a multi-purpose programming language (due to its many extensions), examples are scientific computing and calculations, simulations, web development (using, e.g., the Django Web framework), etc. Python is an object-oriented programming language (OOP), but you can use Python in basic application without the need to know about or use the object- oriented features in Python.
- One of the most significant advantages of Python is its open-source nature, which means that it is free to use and can be modified and distributed by anyone. This has led to a large community of developers who contribute to the development of the language and its ecosystem.
- Python also has a broad range of applications in the field of artificial intelligence and machine learning. With the help of popular libraries such as NumPy, Pandas, and TensorFlow, developers can create powerful machine learning models and perform data analysis tasks efficiently.

- Python has a straightforward installation process, and there are various integrated development environments (IDEs) and code editors available to choose from. Some of the popular IDEs include PyCharm, Spyder, and Visual Studio Code.
- Overall, Python is an accessible and versatile language that has numerous applications and a thriving community of developers. It is an excellent language for beginners to learn and a powerful tool for experienced developers to create complex and innovative solutions.

3. Flask

Flask is a web application framework written in Python developed by Armin Ronacher, who led a team of international Python enthusiasts called Pocoo. Flask allows developers to build web applications quickly and easily. It is a lightweight framework that is designed to be flexible and extensible, allowing developers to customize their applications to fit their specific needs. One of the main advantages of Flask is its simplicity. Flask is built with a minimalistic approach, which means that it provides only the necessary tools and libraries to get started quickly. This makes it an excellent choice for developers who want to build web applications without the added complexity of larger frameworks.

Flask is also highly customizable. Developers can add their own extensions and modules to the framework, allowing them to add functionality to their applications easily. Flask also supports a wide range of plugins, making it easy to integrate with other popular Python libraries and frameworks.

Another significant advantage of Flask is its built-in support for unit testing. Flask provides a built-in testing framework that allows developers to write tests for their applications easily. This makes it easy to test and debug code as it is being developed, ensuring that applications are of high quality and free of bugs.

Flask is a powerful and flexible framework that provides developers with the necessary tools to build web applications quickly and easily. Its simplicity and customization options make it an excellent choice for developers who want to build applications quickly, while its built-in testing framework ensures that applications are of high quality and free of bugs. Flask is based on the Werkzeug WSGI toolkit and the Jinja2 template engine. Both are Pocco projects.

WSGI

The Web Server Gateway Interface (Web Server Gateway Interface, WSGI) has been used as a standard for Python web application development. WSGI is the specification of a common interface between web servers and web applications.

Werkzeug

Werkzeug is a WSGI toolkit that implements requests, response objects, and utility functions. This enables a web frame to be built on it. The Flask framework uses Werkzeug as one of its bases.

jinja2

jinja2 is a popular template engine for Python. A web template system combines a template with a specific data source to render a dynamic web page.

Flask provides configuration and conventions, with sensible defaults, to get started. This section of the documentation explains the different parts of the Flask framework and how they can be used, customized, and extended. Beyond Flask itself, look for community-maintained extensions to add even more functionality.

A Web Application Framework or a simply a Web Framework represents a collection of libraries and modules that enable web application developers to write applications without worrying about low-level details such as protocol, thread management, and so on.

4. Visual Studio

Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs including websites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code. Visual Studio includes a code supporting IntelliSense (the code completion component) as well as code refactoring. The integrated debugger works as both a source-level debugger and as a machine-level debugger.

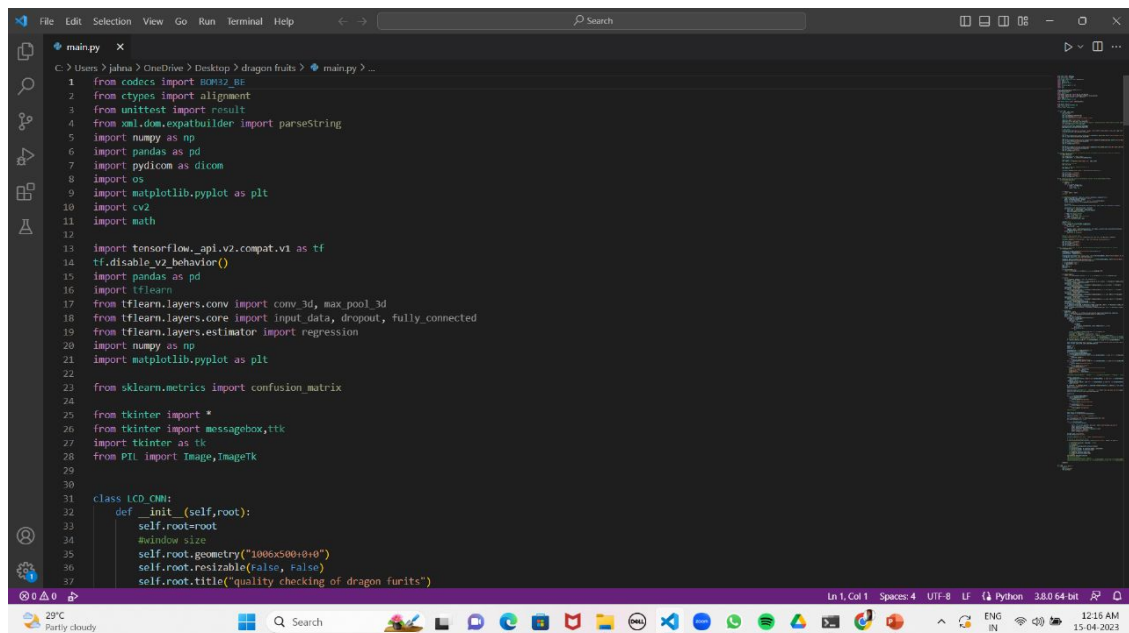


Fig 3.2.1: Visual Studio Interference

Other built-in tools include a code profiler, designer for building GUI applications, web designer, class designer, and database schema designer. It accepts plug-ins that expand the functionality at almost every level—including adding support for source control systems (like Subversion and Git) and adding new toolsets.

Visual Studio also provides excellent support for cloud development, allowing developers to build, deploy, and manage applications on cloud platforms such as Microsoft Azure. This is particularly useful for developers who are building web applications or mobile applications that require cloud integration.

Overall, Visual Studio is a powerful and versatile IDE that provides developers with a wide range of features and tools to build applications quickly and efficiently. Its extensibility and support for cloud development make it an excellent choice for developers who work with multiple languages and need to build applications for a range of platforms.

CHAPTER 4

DESCRIPTION OF PROPOSED SYSTEM

4.1 SELECTED METHODOLOGY OR PROCESS MODEL

The work utilizes the deep convolutional neural network to categorize and identify the dragon fruit that reached its mellow stages and that is ready for gleaning. The method utilizes the Residual Network that is shortly termed as the RESNET to perform the task. This was introduced by Microsoft as the remedy for the problems aroused in the deep networks due to the process of training.

The proposed method uses the RESNET 152-layer using more than three blocks per layer. Though the depth of the RESNET is increased it still has a lower complexity than the VGG16/19. The training process is done with the images that are manually categorized as fruit with mellow or ready for harvesting and not ready for harvesting, this was done with the help of experts. Every image used in training was labelled according to the experts' categorization, further the feature extracted was used in training the RESNET 15.

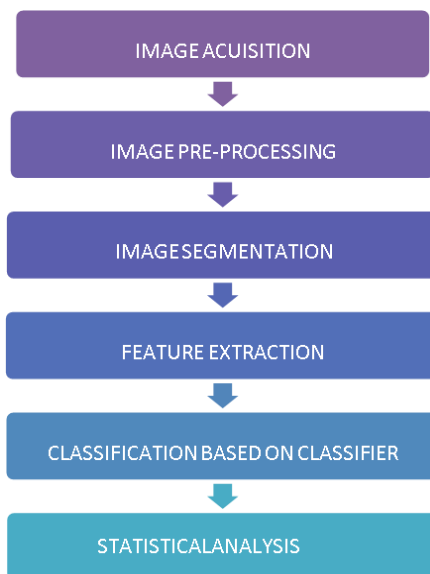


Fig 4.1.1: Block diagram of proposed model

1. IMAGE ACQUISITION

Image acquisition in image processing refers to the process of capturing or digitizing an image from the physical world and converting it into a digital form that can be processed and manipulated by a computer.

Cameras are the most common imaging devices used for image acquisition. They can be categorized into two types: digital and analog. Analog cameras use film to capture images, while digital cameras use image sensors to convert light into digital signals. In addition to cameras, scanners are also commonly used for image acquisition. Scanners are devices that use light sensors to capture images of printed documents or photographs.

Microscopes are another type of imaging device used for image acquisition. They are commonly used in the fields of biology and medicine to observe and capture images of microscopic objects. The quality of the acquired image depends on various factors such as the resolution and sensitivity of the imaging device, lighting conditions, and the stability of the object being imaged. Once the image has been acquired, it can be processed and analyzed using various image processing techniques such as filtering, segmentation, and feature extraction. These techniques are used to enhance the image, extract meaningful information from it, or prepare it for further analysis or manipulation. Once an image is acquired, it can be stored in various formats such as JPEG, PNG, or TIFF. These formats differ in terms of file size, compression, and quality.

Image acquisition is an important step in various applications such as medical imaging, scientific research, and industrial inspection. It is critical to obtain high-quality images to ensure accurate and reliable analysis of the acquired data. Image acquisition is an essential first step in many applications of image processing, such as medical imaging, surveillance, and robotics.

2. IMAGE PREPROCESSING

Image preprocessing is a set of techniques used to improve the quality and suitability of an image for further analysis or processing. It involves various operations performed on the raw image data to remove any noise, artifacts, or distortions, and enhance its visual quality, contrast, and sharpness. The purpose of image preprocessing is to make the image more suitable for a specific image processing application or to improve its interpretability. Some of the common image preprocessing techniques include:

Image resizing and cropping: This involves resizing the image to a specific resolution or aspect ratio and cropping out any unwanted parts of the image. Resizing an image can be done by resampling the original image, which involves either discarding or adding pixels to the image. Resampling can result in a loss of detail or image quality, particularly when resizing an image to a larger size. Image cropping involves selecting a portion of an image and removing the rest. This technique can be useful for removing unwanted portions of an image, focusing on a particular subject or aspect of the image, or adjusting the composition of the image. Cropping an image involves selecting a rectangular or non-rectangular region of interest (ROI) and removing the pixels outside the ROI. This process can be performed manually using image editing software or automatically using algorithms that detect the ROI based on various criteria such as color, texture, or edge information.

Image filtering: This involves removing any noise or unwanted frequencies from the image using filters such as median filters, Gaussian filters, or edge detection filters. Image filtering is a technique used in digital image processing to enhance or modify an image by applying a specific filter or mask to the image pixels. The filter modifies the pixel values of the image based on its neighboring pixels, thereby altering the overall appearance of the image. Image filtering is a broad term that encompasses various techniques, including spatial filtering, frequency filtering, and morphological filtering. Spatial filtering involves applying a filter mask to each pixel in an image to

modify its value based on the values of its neighboring pixels. The filter mask is a small matrix or kernel that moves over the image pixel by pixel, and its values are used to compute the new value of each pixel. Common spatial filters include mean filters, median filters, and Gaussian filters. Frequency filtering involves transforming an image from the spatial domain to the frequency domain, where it can be modified by filtering specific frequency components of the image. This technique is commonly used in applications such as image sharpening, noise reduction, and edge detection. Common frequency filters include high-pass filters, low-pass filters, and band-pass filters. Morphological filtering is a technique used to process binary or grayscale images by applying specific operations, such as dilation, erosion, opening, and closing. These operations modify the shape and structure of the image objects by expanding or contracting them based on specific criteria. Image filtering is a powerful technique that can be used to enhance image quality, remove noise, extract features, and segment objects. However, the choice of filter and its parameters depends on the specific application and the characteristics of the input image. It is important to choose appropriate filters and optimize their parameters to obtain the desired results.

Image normalization: This involves adjusting the brightness and contrast of the image to improve its visual quality and make it easier to analyze. Image normalization is a technique used in digital image processing to adjust the pixel values of an image to a specific range or distribution. The process involves scaling and shifting the pixel values of the image to ensure that they are within a desired range, which can help improve the visual appearance and facilitate subsequent image processing tasks. There are different methods of image normalization, including linear normalization, histogram equalization, and contrast stretching. Linear normalization involves scaling the pixel values of an image to a new range using a linear transformation function. The function maps the original pixel values to new values that range between 0 and 255 or between -1 and 1. This technique can help enhance the overall contrast and brightness of the image, making it more visually appealing. Histogram equalization is a technique used to adjust the distribution of pixel values in an image to a uniform distribution. This technique can help enhance the contrast and improve the visual

appearance of the image. The process involves computing the cumulative distribution function (CDF) of the pixel values and then mapping the original pixel values to new values based on the CDF. Contrast stretching is a technique used to expand the range of pixel values in an image by compressing the low and high pixel values and expanding the mid-range values. This technique can help enhance the contrast and make the image more visually appealing. Image normalization is an important technique in digital image processing that can help improve the quality and appearance of an image, as well as facilitate subsequent image processing tasks such as object detection, segmentation, and classification.

Image registration: This involves aligning two or more images of the same scene to correct any distortions or misalignments caused by camera movement or changes in viewpoint. Alignment can be looked at as a simple coordinate transform. Match features from the image to be aligned, to the reference image and store the coordinates of the corresponding key points. Key points are simply the selected few points that are used to compute the transform (generally points that stand out), and descriptors are histograms of the image gradients to characterize the appearance of a key point. In this post, we use ORB (Oriented FAST and Rotated BRIEF) implementation in the OpenCV library, which provides us with both key points as well as their associated descriptors.

Image preprocessing is a crucial step in image processing as it can significantly impact the quality and accuracy of the results obtained from further analysis.

3. IMAGE SEGMENTATION

Image segmentation is a process in image processing that involves dividing an image into multiple segments or regions based on certain criteria such as color, texture, intensity, or shape. The purpose of image segmentation is to extract meaningful information from an image by separating the objects or regions of interest from the background or other irrelevant areas. There are several techniques for image segmentation, depending on the type of image and the specific application.

Some of the commonly used techniques include:

Thresholding: This involves setting a threshold value and assigning all pixels in the image above that threshold to one class and all pixels below it to another class. Thresholding is a common technique used in image segmentation to separate regions of an image based on their intensity values. It involves setting a threshold value, which separates the pixels with intensity values above the threshold from those below it. This results in a binary image where pixels above the threshold are assigned one value (usually white) and pixels below it are assigned another value (usually black). This is a simple and efficient method for image segmentation, but it may not be suitable for complex images or images with varying lighting conditions.

Region-based segmentation:

This involves grouping pixels together based on their similarity in color, texture, or intensity, and forming a region or segment. Region-based segmentation is a technique used in image segmentation to divide an image into meaningful regions or objects based on their similarity in terms of texture, color, or other features. This approach involves grouping pixels together based on some predefined criteria to form homogeneous regions or objects. The most common method used in region-based segmentation is called region growing, where seeds or starting points are selected and then neighboring pixels with similar characteristics are iteratively added to the region until a stopping criterion is met. Another popular method is called split and merge, where the image is recursively split into smaller regions and then merged back together based on a similarity criterion. Region-based segmentation is effective in segmenting images with uniform regions or objects, but it can struggle with images with complex structures or significant variations in texture and color.

Edge-based segmentation:

This involves identifying the edges or boundaries of objects in the image by detecting changes in intensity or color and then grouping these edges together to form a segment.

Clustering-based segmentation: This involves clustering similar pixels together based on certain criteria such as color or texture, and then forming segments based on the resulting clusters.

Image segmentation is a fundamental step in many image processing applications such as object recognition, tracking, and analysis. By segmenting an image into meaningful regions or objects, it becomes easier to extract features, identify patterns, and make decisions based on the information extracted from the image.

4. FEATURE EXTRACTION

Feature extraction is an essential step in image processing, which involves extracting important and relevant information from the image to be used in subsequent analysis or classification tasks. Feature extraction is a crucial step in image processing, where relevant information is extracted from the raw image data. The aim of feature extraction is to identify and extract key features from an image that can be used to differentiate objects or regions of interest from the background. The extracted features can then be used for various applications such as object recognition, classification, and segmentation. Common techniques used for feature extraction in image processing include edge detection, corner detection, texture analysis, color histograms, and scale-invariant feature transform (SIFT). The choice of feature extraction technique depends on the specific image processing task, the type of images, and the characteristics of the features of interest. Feature extraction is an important and ongoing area of research in computer vision and image processing, with new techniques being developed to improve the accuracy and efficiency of feature extraction algorithms. The following are some commonly used techniques for feature extraction in image preprocessing:

Color Spaces Conversion: Convert the image to different color spaces like grayscale, HSV, LAB, RGB, etc. These color spaces have different channels, and each channel contains different information that can be used as features. Color space conversion

involves transforming the original image from one color space to another. This can be done to enhance certain color features, to separate color components for further analysis, or to standardize the color representation across different devices or environments. For example, in the HSV color space, the hue channel contains information about the color, while the saturation and value channels contain information about the intensity of the color. In feature extraction, color space conversion can be used to extract specific color features for classification or segmentation tasks. For instance, color histograms can be computed in different color spaces to capture the distribution of colors in an image. Color moments, which are statistical measures of the color distribution, can also be computed in different color spaces to capture the color properties of an image.

Edge Detection: Detecting edges in an image is a widely used technique for feature extraction. Edge detection techniques like Canny, Sobel, and Laplacian operators can be used to identify edges in the image, which can then be used as features. The most common edge detection techniques used in feature extraction include the Sobel, Canny, and Roberts operators. The Sobel operator works by convolving the image with a small filter that calculates the gradient of the image in the x and y directions. The Canny edge detector is a more sophisticated method that involves multiple steps, including smoothing the image, calculating the gradient, non-maximum suppression, and hysteresis thresholding. The Roberts operator works by calculating the difference between adjacent pixels in the image. Edge detection can be used in feature extraction for object detection, segmentation, and recognition. By detecting edges, it is possible to isolate regions of an image that correspond to specific objects or features. Edges can also be used to extract texture and shape information, which can be used for object recognition tasks

Feature Descriptors: Feature descriptors like SIFT, SURF, and ORB can be used to identify features in an image. These descriptors work by identifying distinctive points in the image and then extracting features from the surrounding area. Feature descriptors are compact representations of image features that can be used for

various image processing tasks, such as object recognition, matching, and retrieval. In feature extraction, feature descriptors are used to describe and represent the distinctive characteristics of image features, such as edges, corners, and textures, in a way that is invariant to changes in image scale, orientation, and illumination. Common feature descriptors used in feature extraction include Scale-Invariant Feature Transform (SIFT), Speeded Up Robust Features (SURF), and Local Binary Patterns (LBP). SIFT is a popular method for detecting and describing local features in an image, which involves identifying key points in an image and describing them using gradient information. SURF is a similar method that is faster and more robust to image transformations. LBP is a texture descriptor that captures the local texture patterns of an image by comparing the intensity values of neighboring pixels. Feature descriptors are important in feature extraction because they allow for efficient and accurate matching of features across different images. They also enable the representation of complex features in a way that is robust to image variations and noise. Feature descriptors can be used in various image processing applications, such as object recognition, image retrieval, and facial recognition.

Texture Analysis: Texture analysis involves analyzing the patterns and structures in an image. Texture features like Hara lick features, Gabor filters, and Local Binary Patterns (LBP) can be used to extract information about the texture of an image. Texture analysis is a technique used in feature extraction to identify and characterize patterns and structures in an image that can be used to differentiate between different objects or regions of interest. Texture features are often used in conjunction with other types of features, such as color and shape, to provide a more complete description of image content. Common texture analysis techniques used in feature extraction include Gray Level Co-occurrence Matrix (GLCM), Local Binary Patterns (LBP), and Gabor filters. GLCM calculates the probability of specific combinations of pixel values occurring within a given distance and direction, providing information about the texture properties of an image. LBP calculates the local texture patterns of an image by comparing the intensity values of neighboring pixels, and Gabor filters use a set of complex filters to capture texture information at different scales and

orientations. Texture analysis can be used in feature extraction for various applications, such as image segmentation, object recognition, and medical image analysis. For example, in medical imaging, texture analysis can be used to detect and classify abnormalities in tissue texture, providing valuable information for diagnosis and treatment planning.

Histogram of Oriented Gradients (HOG): HOG is a popular feature extraction technique that involves computing the gradient magnitude and orientation in the image. The gradients are then grouped into cells, and a histogram of the gradient orientations is computed for each cell. These histograms can then be used as features. HOG works by dividing an image into small overlapping cells and calculating the gradient orientation and magnitude within each cell. The gradient orientations are then binned into a histogram, with each bin representing a range of gradient orientations. The histogram of gradient orientations is then normalized to account for variations in illumination and contrast across different images. The resulting HOG feature descriptor provides a compact representation of the gradient orientations within an image or region of interest. The HOG descriptor is invariant to changes in image scale and orientation, making it useful for object detection tasks. HOG has been used successfully in a variety of computer vision applications, such as pedestrian detection, face recognition, and animal classification. One of the advantages of HOG is its ability to capture both local and global features of an image. By dividing an image into smaller cells, HOG can capture the fine details of an object, while the use of overlapping cells and block normalization allows for the extraction of global features that describe the overall shape and structure of the object.

Convolutional Neural Networks (CNNs): CNNs can be used to automatically learn features from an image. These networks consist of multiple layers of convolutional and pooling layers that learn increasingly complex features from the image. Convolutional Neural Networks (CNNs) are a type of artificial neural network that are commonly used in image recognition and processing. CNNs are designed to process and classify images by learning features and patterns directly from the pixel values of

the input images. The basic structure of a CNN consists of multiple layers of convolutional and pooling operations, followed by one or more fully connected layers. The convolutional layers use a set of learnable filters to convolve over the input image, extracting various features and patterns. The pooling layers then down sample the feature maps to reduce the size of the data and increase computational efficiency. Finally, the fully connected layers perform the classification based on the features extracted from the input image. One of the key advantages of CNNs is their ability to learn and extract features directly from raw pixel values, without the need for hand-crafted features. This makes CNNs very effective at tasks such as object detection, image segmentation, and face recognition. CNNs have also been shown to be very effective at transfer learning, where a pre-trained CNN model can be fine-tuned on a new dataset for a specific task. This allows for the efficient use of large, pre-trained models, saving both time and computational resources.

In summary, feature extraction is a critical step in image preprocessing, and the techniques used can vary depending on the type of image and the task at hand.

5. CLASSIFICATION BASED ON CLASSIFIERS

In image processing, classifiers are algorithms that are trained to automatically recognize and classify objects or features within an image. The classification methods based on classifiers can be broadly categorized into the following types:

Supervised Classification: This type of classification is based on a pre-defined set of labeled images that the classifier is trained on. The classifier learns to recognize the features of interest and assign a label to new images based on the features it has learned. Examples of supervised classification algorithms include support vector machines (SVM), neural networks, and decision trees. Supervised classification is a technique used in image processing to assign pixels in an image to predefined classes based on their spectral characteristics. It involves training a classifier using a set of labeled training samples, where each sample is associated with a known class label. The classifier uses the spectral information extracted from each pixel in the

image and compares it to the spectral signatures of the training samples to assign a class label to the pixel. Common supervised classification algorithms include maximum likelihood, support vector machines, and random forests. Supervised classification is useful for applications such as land-cover mapping, urban planning, and environmental monitoring, where accurate and consistent classification results are essential. However, it requires a representative set of training samples and may not perform well on images with high intra-class variability or complex spectral characteristics.

Unsupervised Classification: Unlike supervised classification, unsupervised classification does not require labeled training data. Instead, it uses statistical techniques to group pixels with similar properties together and create clusters. These clusters can then be used to identify objects or features within an image. Examples of unsupervised classification algorithms include k-means clustering and hierarchical clustering. Unsupervised classification is a technique used in image processing to group pixels in an image into clusters based on their spectral similarity without prior knowledge of the number or nature of the classes. It involves clustering the spectral information extracted from each pixel in the image into groups or classes based on some similarity measure such as Euclidean distance or correlation. Common unsupervised classification algorithms include k-means clustering, hierarchical clustering, and self-organizing maps. Unsupervised classification is useful for exploratory data analysis and identifying unknown patterns or features in the data. However, it may produce inconsistent or inaccurate results if the clusters are not representative of the underlying classes or if the spectral variability within a class is high. Unsupervised classification can be used as a pre-processing step to generate initial class assignments for subsequent refinement using supervised classification or other image processing techniques.

Semi-Supervised Classification: This type of classification is a combination of supervised and unsupervised methods. The classifier is trained on a small set of labeled images and then uses unsupervised methods to identify additional features in

the unlabeled images. This can improve the accuracy of the classification by incorporating additional information. Semi-supervised classification is a technique used in image processing that combines the benefits of supervised and unsupervised classification. It involves using a combination of labeled and unlabeled data to train a classifier to assign class labels to pixels in an image. The labeled data is used to train a classifier using supervised learning techniques, while the unlabeled data is used to refine the classification results by identifying patterns and structures in the data using unsupervised learning techniques. The semi-supervised classification approach is particularly useful when obtaining many labeled data is difficult, time-consuming, or expensive, and when the underlying structure of the data is complex and not fully understood. Semi-supervised classification can improve the classification accuracy and reduce the computational cost of classification compared to supervised classification while providing more consistent and reliable results than unsupervised classification.

Object Detection: is a type of classification used to identify the location of objects within an image. Object detection algorithms use a combination of classification and localization techniques to identify the objects and their locations within the image. Examples of object detection algorithms include Faster R-CNN, YOLO, and SSD. Object detection is a technique used in image processing to identify and locate objects of interest within an image or video. It involves detecting the presence of objects in an image and drawing a bounding box around each object to indicate its location. Object detection typically involves two stages: object localization and object classification. Object localization involves identifying the location of the object within the image, while object classification involves determining the type or class of the object. Object detection algorithms can be divided into two broad categories: feature-based and deep learning-based. Feature-based algorithms rely on hand-crafted features and machine learning techniques such as support vector machines, while deep learning-based algorithms use convolutional neural networks to learn features directly from the data. Object detection is used in a wide range of applications, including security and surveillance, autonomous vehicles, and medical imaging.

Segmentation: is a type of classification that involves dividing an image into multiple regions based on the similarity of the pixels. This can be used to separate objects or features from the background or to identify different regions of interest within an image. Segmentation is a fundamental technique used in image processing to divide an image into meaningful regions or objects based on their characteristics such as texture, color, and intensity. Segmentation can be used to identify and extract specific objects or features of interest within an image, which is essential for many applications such as object recognition, medical image analysis, and computer vision. There are various methods for image segmentation, including thresholding, region-based segmentation, edge-based segmentation, and clustering-based segmentation. These techniques can be supervised or unsupervised and may involve the use of machine learning algorithms or heuristics. The choice of segmentation technique depends on the nature of the image data and the specific requirements of the application. Segmentation is a challenging task, particularly for complex images with high levels of noise, variability, and occlusion, and remains an active area of research in image processing and computer vision. Examples of segmentation algorithms include watershed segmentation, region growing, and graph-cut segmentation.

6. STATISTICAL ANALYSIS

- a) Statistical analysis involves the use of mathematical models to analyse and interpret data, with the goal of identifying patterns and relationships between variables.
- b) In the context of quality monitoring for dragon fruits, statistical analysis can be used to identify key characteristics and features that are associated with high-quality fruits.
- c) Deep learning is a type of machine learning that involves the use of artificial neural networks to analyse large and complex datasets.

- d) Deep learning can be used to develop models that are capable of accurately identifying and classifying different qualities of dragon fruits based on a range of factors, such as colour, texture, and size.
- e) To perform statistical analysis on dragon fruit data, it is important to collect and record accurate and consistent measurements of key quality indicators, such as sugar content, acidity, and firmness.
- f) Statistical methods such as regression analysis, factor analysis, and cluster analysis can be used to identify correlations and patterns in the data, and to develop models that can be used to predict fruit quality based on these variables.
- g) Deep learning algorithms such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) can be used to analyse images of dragon fruits and extract key features that are associated with fruit quality.
- h) By combining statistical analysis and deep learning techniques, it is possible to develop highly accurate and reliable models for quality monitoring of dragon fruits, which can help farmers and producers to improve their yields and profits.

Statistical analysis is a powerful tool in image processing that can be used to extract meaningful information from images. Some common statistical analysis techniques used in image processing include:

Histogram analysis: involves creating a histogram of the pixel intensity values in an image. This can be useful for visualizing the distribution of pixel intensities and for identifying image features such as brightness and contrast. Histogram analysis is a statistical technique used to represent the distribution of a dataset. It involves constructing a histogram, which is a graphical representation of the distribution of data values. The histogram is formed by dividing the range of data into a set of

intervals, called bins, and counting the number of data points that fall into each bin. The resulting histogram provides insight into the shape of the distribution, including the location and spread of the data, as well as any skewness or multimodality. Histogram analysis is useful in a wide range of applications, including quality control, image processing, finance, and social sciences. It can be used to identify outliers, to assess the normality of a distribution, and to compare the distribution of two or more datasets. Histogram analysis is a fundamental technique in statistical analysis, and many more advanced statistical methods, such as hypothesis testing and regression analysis, rely on an understanding of the underlying distribution of data.

Mean and variance: The mean and variance of pixel intensity values can be used to characterize the overall brightness and contrast of an image. In statistical analysis, mean and variance are two fundamental measures of central tendency and variability, respectively, that are commonly used to describe a distribution of data. The mean is a measure of the central location of the data and is calculated by summing all the data points and dividing by the total number of data points. The variance, on the other hand, is a measure of the spread or dispersion of the data around the mean and is calculated as the average of the squared differences between each data point and the mean. The square root of the variance is called the standard deviation, which is a more interpretable measure of dispersion that has the same units as the original data. Mean and variance are used in a wide range of statistical analyses, including hypothesis testing, regression analysis, and quality control. They can be used to compare different datasets, to assess the normality of a distribution, and to identify outliers. Mean and variance are particularly useful when dealing with continuous data, where the distribution can be characterized by a probability density function, such as the normal distribution.

Correlation analysis: Correlation analysis involves measuring the degree to which two images or image features are related. This can be useful for identifying similarities or differences between images or for detecting patterns in the data. Correlation analysis is a statistical technique used to quantify the degree of association between two

variables. It measures the strength and direction of the linear relationship between two variables, which can be positive, negative, or zero. Correlation coefficients range from -1 to +1, with values closer to -1 or +1 indicating a stronger correlation and values closer to zero indicating a weaker correlation. Correlation analysis can be used to explore relationships between variables, to identify patterns and trends in data, and to test hypotheses about the underlying structure of data. It is commonly used in fields such as finance, economics, social sciences, and engineering to examine the relationship between variables, such as the relationship between stock prices and interest rates or the relationship between education and income. Correlation analysis should be interpreted with caution, however, as correlation does not imply causation. Other factors, such as confounding variables or selection bias, may influence the observed correlation, and further analysis is needed to establish a causal relationship between variables.

Principal Component Analysis (PCA): PCA is a technique for identifying the underlying structure of a dataset by finding the principal components that explain the most variance in the data. In image processing, PCA can be used to identify patterns in the pixel intensity values or to reduce the dimensionality of the data. Principal component analysis (PCA) is a statistical technique used to transform a set of correlated variables into a smaller set of uncorrelated variables, called principal components, while retaining as much of the original variability as possible. PCA is commonly used in data analysis to reduce the dimensionality of datasets and to identify patterns and relationships among variables. The principal components are obtained by finding the linear combinations of the original variables that account for the most variation in the data. The first principal component is the linear combination that explains the largest amount of variance, and each subsequent principal component explains the remaining variance in decreasing order. PCA is useful in many fields, including finance, biology, and engineering, where high-dimensional datasets need to be analyzed and visualized. PCA can be used to identify underlying factors that contribute to observed patterns in data and to explore relationships between variables. PCA has a range of applications, including image processing,

signal processing, and machine learning, where it is used to extract features and reduce the complexity of models.

Fourier analysis: Fourier analysis involves decomposing an image into its frequency components. This can be useful for identifying periodic patterns or for removing noise from an image. Fourier analysis is a mathematical technique used to represent a time series or signal as a sum of sinusoidal components with different frequencies and amplitudes. It is commonly used in signal processing, audio and image analysis, and many other fields where data can be represented as a function of time or space. The Fourier transform is a mathematical operation that converts a time-domain signal into its frequency-domain representation, which shows the relative contribution of each frequency component to the overall signal. The inverse Fourier transform can then be used to reconstruct the original signal from its frequency components. Fourier analysis is useful in many applications, including audio and image compression, noise reduction, and pattern recognition. It can also be used to extract features from time series data, such as the dominant frequency or the spectral density, which can be used in further analysis or modeling. Fourier analysis is a powerful tool in statistical analysis and is widely used in many fields of science and engineering.

Wavelet analysis: Wavelet analysis involves decomposing an image into its wavelet coefficients, which can be used to extract information at different scales and resolutions. Wavelet analysis is a mathematical technique used to analyze signals and data in both the time and frequency domains. It is based on the use of wavelets, which are mathematical functions that can be scaled and shifted to analyze a signal at different resolutions. Wavelet analysis is particularly useful for non-stationary signals, where the frequency content changes over time. The wavelet transform decomposes a signal into different frequency components at different resolutions, providing information about the time and frequency localization of different features in the signal. This allows for the identification of short-lived or transient features in a signal that may be missed by traditional Fourier analysis. Wavelet analysis has a wide range of applications, including image processing, signal

compression, noise reduction, and feature extraction. It is used in many fields, including engineering, physics, biology, and finance, to analyze and interpret complex data.

4.2 ARCHITECTURE / OVERALL DESIGN OF PROPOSED MODEL

4.2.1 Image Processing

Image processing is a method to convert an image into digital form and perform some operations on it, in order to get an enhanced image or to extract some useful information from it.

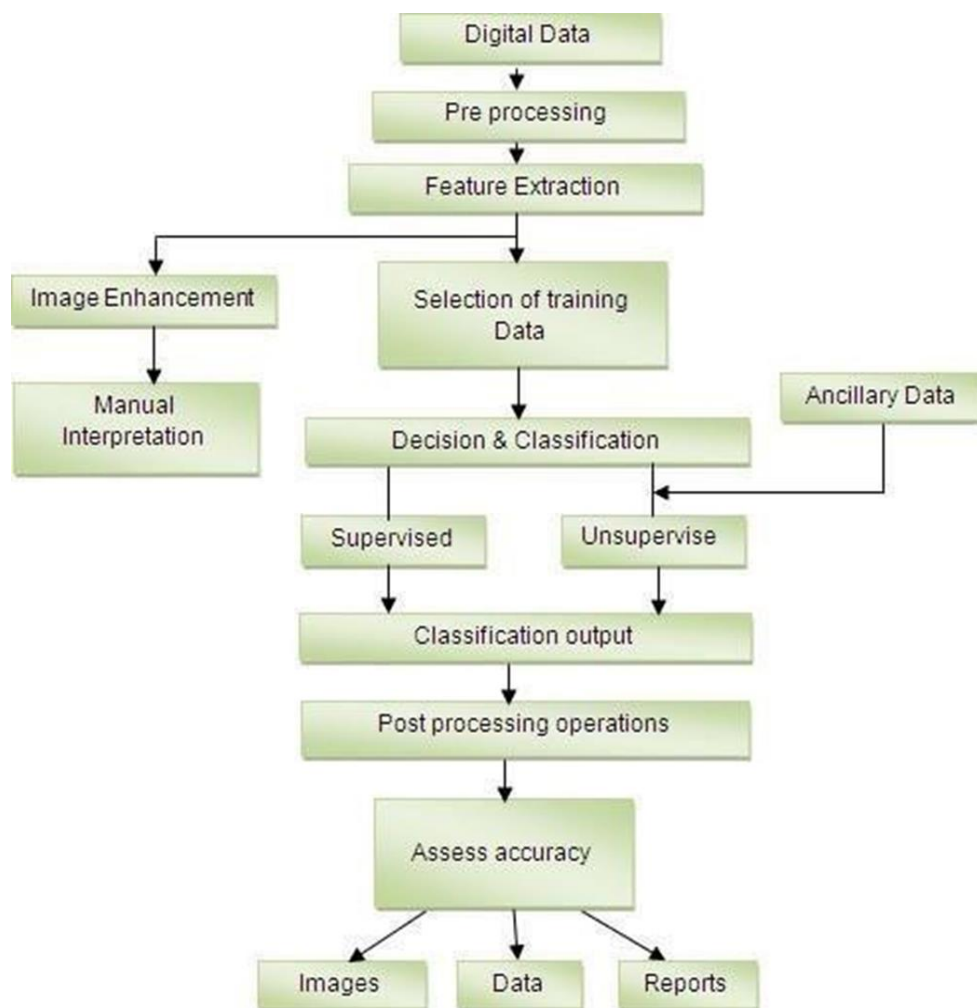


Fig 4.2.1: System Architecture of the proposed model

It is a type of signal dispensation in which input is image, like video frame or photograph and output may be image or characteristics associated with that image. Usually, Image Processing system includes treating images as two-dimensional signals while applying already set signal processing methods to them. It is among rapidly growing technologies today, with its applications in various aspects of a business. Image Processing forms core research area within engineering and computer science disciplines too.

Image processing basically includes the following three steps:

- Importing the image with optical scanner or by digital photography.
- Analyzing and manipulating the image which includes data compression and image enhancement and spotting patterns that are not to human eyes like satellite photographs.
- Output is the last stage in which result can be altered image or report that is based on imageanalysis.

The overall design of the proposed model is as shown in above fig., and involves different stages of processing techniques. They are:

Purpose of Image processing

The purpose of image processing is divided into 5 groups. They are:

1. Visualization - Observe the objects that are not visible.
2. Image sharpening and restoration - To create a better image.
3. Image retrieval - Seek for the image of interest.
4. Measurement of pattern – Measures various objects in an image.
5. Image Recognition – Distinguish the objects in an image.

Types:

The two types of methods used for Image Processing are Analog and Digital Image Processing Analog or visual techniques of image processing can be used for the hard copies like printouts and photographs. Image analysts use various

fundamentals of interpretation while using these visual techniques. The image processing is not just confined to area that must be studied but on knowledge of analyst. Association is another important tool in image processing through visual techniques. So, analysts apply a combination of personal knowledge and collateral data to image processing. Digital Processing techniques help in manipulation of the digital images by using computers. As raw data from imaging sensors from satellite platform contains deficiencies. To get over such flaws and to get originality of information, it must undergo various phases of processing. The three general phases that all types of data must undergo while using digital technique are Pre-processing, enhancement and display, information extraction.

Image Processing Concepts

1. Convolution

Convolution is a simple mathematical operation which is fundamental to many common image processing operators. Convolution provides a way of multiplying together two arrays of numbers, generally of different sizes, but of the same dimensionality, to produce a third array of numbers of the same dimensionality. This can be used in image processing to implement operators whose output pixel values are simple linear combinations of certain input pixel values.

In an image processing context, one of the input arrays is normally just a gray level image.

I₁₁	I₁₂	I₁₃	I₁₄	I₁₅	I₁₆	I₁₇	I₁₈	I₁₉
I₂₁	I₂₂	I₂₃	I₂₄	I₂₅	I₂₆	I₂₇	I₂₈	I₂₉
I₃₁	I₃₂	I₃₃	I₃₄	I₃₅	I₃₆	I₃₇	I₃₈	I₃₉
I₄₁	I₄₂	I₄₃	I₄₄	I₄₅	I₄₆	I₄₇	I₄₈	I₄₉
I₅₁	I₅₂	I₅₃	I₅₄	I₅₅	I₅₆	I₅₇	I₅₈	I₅₉
I₆₁	I₆₂	I₆₃	I₆₄	I₆₅	I₆₆	I₆₇	I₆₈	I₆₉

K₁₁	K₁₂	K₁₃
K₂₁	K₂₂	K₂₃

Fig 4.2.2: Small image and kernels

The second array is usually much smaller, and is also two-dimensional (although it may be just a single pixel thick), and is known as the kernel. The figure above shows an example image and kernel that we will use to illustrate convolution.

The above figure, an example small image (left) and kernel (right) to illustrate convolution. The labels within each grid square are used to identify each square. The convolution is performed by sliding the kernel over the image, generally starting at the topleft corner, to move the kernel through all the positions where the kernel fits entirely within the boundaries of the image. (Note that implementations differ in what they do at the edges of images, as explained below.) Each kernel position corresponds to a single output pixel, the value of which is calculated by multiplying together the kernel value and the underlying image pixel value for each of the cells in the kernel, and then adding all these numbers together.

So, in our example, the value of the bottom right pixel in the output image will be given by:

$$O_{57} = I_{57}K_{11} + I_{58}K_{12} + I_{59}K_{13} + I_{67}K_{21} + I_{68}K_{22} + I_{69}K_{23}$$

If the image has M rows and N columns, and the kernel has m rows and n columns, then the size of the output image will have M - m + 1 rows, and N - n + 1 columns.

Mathematically we can write the convolution as:

$$O(i, j) = \sum_{k=1}^m \sum_{l=1}^n I(i + k - 1, j + l - 1)K(k, l)$$

where 'i' runs from 1 to M - m + 1 and j runs from 1 to N - n + 1.

Note that many implementations of convolution produce a larger output image than this because they relax the constraint that the kernel can only be moved to

positions where it fits entirely within the image. Instead, these implementations typically slide the kernel to all positions where just the top left corner of the kernel is within the image. Therefore, the kernel overlaps the image on the bottom and right edges. One advantage of this approach is that the output image is the same size as the input image. Unfortunately, in order to calculate the output pixel values for the bottom and right edges of the image, it is necessary to invent input pixel values for places where the kernel extends off the end of the image. Typically, pixel values of zero are chosen for regions outside the true image, but this can often distort the output image at these places. Therefore, in general if you are using a convolution implementation that does this, it is better to clip the image to remove these spurious regions. Removing $n - 1$ pixels from the right-hand side and $m - 1$ pixels from the bottom will fix things.

Convolution can be used to implement many different operators, particularly spatial filters, and feature detectors. Examples include Gaussian smoothing and the Sobel edge detector.

4.3 DESCRIPTION OF SOFTWARE IMPLEMENTATION AND TESTING PLAN OF THE PROPOSED SYSTEM

The methodology of the proposed system uses three machine learning techniques CNN which detect and classify the Dragon fruits based on the features of the dragon fruit. In this methodology, the dataset of Dragon fruit images and camera captured Dragon fruit images are performing image processing operations such as image acquisition, image pre-processing, image segmentation, feature extraction for training, and testing path of Dragon fruit images. Detecting dragon fruit using deep learning can be done through the following methodology:

Data Collection: Collect a large dataset of dragon fruit images. This dataset should have images of dragon fruit from different angles and in different lighting conditions. The dataset should also have images of other fruits to help the deep learning model

distinguish dragon fruit from other fruits. The first step in data collection is to determine the type of data that will be used for training. In the case of CNNs, this typically involves collecting images or videos related to the task at hand, such as classifying different objects or recognizing facial expressions. The images should be diverse and representative of the different variations that the model will encounter in the real world. Once the images are collected, they must be preprocessed to prepare them for training. This includes tasks such as resizing, normalization, and data augmentation. Resizing ensures that all images are the same size and format, while normalization ensures that pixel values fall within a specific range. Data augmentation is a technique used to generate new images by applying random transformations such as rotation, flipping, or cropping. After preprocessing, the images are split into training, validation, and test sets. The training set is used to train the model, while the validation set is used to evaluate the model's performance during training and adjust the hyperparameters. The test set is used to evaluate the model's final performance on unseen data

Data Preprocessing: Preprocess the dataset to ensure that the images are in the same format, size, and resolution. This step can include resizing, cropping, and normalization. Data processing is a crucial step in building Convolutional Neural Networks (CNN) that involves preparing the input data for the model. In CNNs, data processing typically involves several steps, including data normalization, data augmentation, and data splitting. Data normalization is the process of transforming the input data to have a standard scale. This helps to ensure that the model learns meaningful patterns from the data and makes the training process more stable. In CNNs, normalization is often performed by subtracting the mean pixel value of the dataset from each image and then dividing the result by the standard deviation of the pixel values. Data augmentation is a technique used to increase the size of the training dataset by generating new images from the existing ones. This helps to improve the generalization performance of the model and reduce overfitting. In CNNs, data augmentation is often performed by applying random transformations such as rotation, flipping, or zooming to the input images. Data splitting is the process of

dividing the dataset into training, validation, and test sets. The training set is used to train the model, while the validation set is used to monitor the model's performance during training and adjust the hyperparameters. The test set is used to evaluate the model's final performance on unseen data.

Data Augmentation: Augment the dataset by adding variations of the original images. This step can include flipping, rotating, adding noise, and changing the brightness and contrast of the images. Data augmentation is a technique used in Convolutional Neural Networks (CNN) to increase the size of the training dataset by generating new images from the existing ones. The goal of data augmentation is to increase the diversity and variability of the training data, which helps to improve the generalization performance of the model and reduce overfitting. Data augmentation can be applied in different ways, depending on the size and complexity of the dataset. In some cases, data augmentation can be performed online during training, where the network randomly generates new variations of the training data on the fly. In other cases, the augmented images can be precomputed and stored in a separate file or folder for use during training. Overall, data augmentation is a powerful technique that can improve the performance of CNNs, especially when the training data is limited or the network is prone to overfitting. However, it is important to use data augmentation judiciously and avoid overdoing it, as this can lead to a loss of quality and realism in the augmented images.

Model Selection: Select a deep learning model for the detection of dragon fruit. Popular models for image classification tasks include convolutional neural networks (CNNs) such as ResNet, VGG, and Inception. Model selection is a critical step in building Convolutional Neural Networks (CNNs) that involves choosing the appropriate architecture and hyperparameters for the model. The choice of the model architecture can significantly impact the performance of the model, and it is important to select the architecture that best fits the problem at hand.

There are several factors to consider when selecting a CNN architecture, including:

Input data size and complexity: The size and complexity of the input data can

influence the choice of CNN architecture. For example, smaller datasets may require simpler models, while larger and more complex datasets may require deeper and more complex models.

Available computational resources: Deeper and more complex models require more computational resources, such as memory and processing power. It is important to consider the available resources when selecting a CNN architecture.

Performance requirements: The performance requirements of the model, such as accuracy and speed, can influence the choice of architecture. For example, if speed is a critical factor, a simpler and faster model may be preferred over a deeper and more accurate one.

Existing research: Existing research and state-of-the-art models can provide guidance in selecting an appropriate architecture for the problem at hand.

Once the CNN architecture is chosen, the next step is to optimize the hyperparameters, such as learning rate, batch size, and number of epochs, to improve the model's performance. This is typically done by performing a grid search or random search over a range of hyperparameters to find the optimal values.

Model Training: Train the selected deep learning model using the preprocessed and augmented dataset. During training, the model should learn to distinguish dragon fruit from other fruits. Model training is the process of using training data to optimize the parameters of the Convolutional Neural Network (CNN) model. The goal of training is to find the best set of parameters that minimize the difference between the predicted output and the actual output for the training data. It is important to note that model training can be computationally intensive and can require significant resources, such as high-end GPUs or TPUs, to complete in a reasonable amount of time. Therefore, it is important to carefully choose the appropriate hyperparameters and monitor the model's performance during training to ensure optimal performance.

Model Evaluation: Evaluate the performance of the trained model on a separate test dataset. This step can include calculating metrics such as accuracy, precision, recall, and F1-score. Model evaluation is an essential step in the development of

Convolutional Neural Network (CNN) models, and it involves assessing the performance of the model on a set of test data. The goal of model evaluation is to estimate the model's ability to generalize to new, unseen data. In addition to these metrics, it is important to visualize the model's performance on the test data, by plotting the model's predictions against the actual labels. This can help to identify patterns and potential areas of improvement. It is also important to keep in mind that the performance of a CNN model can vary depending on the problem at hand and the test data used. Therefore, it is important to use multiple evaluation metrics and to carefully interpret the results in the context of the problem being solved.

Model Optimization: Optimize the model to improve its performance. This step can include adjusting hyperparameters, fine-tuning the model, or using transfer learning to improve the model's accuracy. Model optimization is the process of improving the performance of a Convolutional Neural Network (CNN) model by adjusting its hyperparameters and optimizing its architecture. Hyperparameters are adjustable parameters that are not learned during the training process, such as learning rate, batch size, and number of epochs. In addition to hyperparameter tuning and architecture optimization, it is important to consider other factors that can impact the performance of a CNN model, such as data preprocessing and data augmentation techniques.

Deployment: Deploy the optimized model for dragon fruit detection. This can be done as a mobile or web application, or as part of a larger system for fruit detection in a farm or market. Deploying a Convolutional Neural Network (CNN) model is the process of making the trained model available for use in a production environment. The deployment involves converting the trained model into a format that can be used by applications, integrating it with the application code, and deploying it to the target environment. The deployed model is then tested and validated to ensure that it is performing as expected and delivering accurate results. Additionally, monitoring and maintenance are crucial to ensure that the deployed model is always delivering accurate results, and any issues are addressed through updates. Overall, deploying a

CNN model requires careful consideration of the deployment environment, the format of the model, and the integration with the application code to deliver reliable results.

Continuous Improvement: Continuously improve the model's performance by collecting feedback and updating the dataset with new images. This step can help the model adapt to new lighting conditions, angles, and types of dragon fruit. Continuous improvement in Convolutional Neural Networks (CNN) is an ongoing process of enhancing the performance of the model over time. It involves analyzing the model's performance, identifying areas of improvement, and implementing changes to the model or the data processing pipeline to improve its accuracy and efficiency. One of the key components of continuous improvement is monitoring the performance of the model over time. This can be done by tracking metrics such as accuracy, precision, recall, and F1-score. By analyzing these metrics, it is possible to identify areas where the model is underperforming and take steps to address them. Another important aspect of continuous improvement is data analysis. By analyzing the data used to train the model, it is possible to identify patterns and trends that can be used to improve the accuracy of the model. This can involve techniques such as data augmentation, data cleaning, and data normalization. In addition to monitoring performance and analyzing data, continuous improvement also involves keeping up with the latest research and techniques in the field of CNNs. New techniques for training and optimization are constantly being developed, and it is important to stay up to date with these developments to ensure that the model is using the most effective techniques.

4.4 PROJECT MANAGEMENT PLAN

The first step in the project is we are going to implement the code in Visual Studio using python language program. Later we will select the dragon from the sample data set and we will implement the data in python language with the resource code. Next select the cmd and give python main.py then the code start executing and accuracy level will be implemented and the dragon fruit image will be display on the program, the dragon image gets the import data, Pre-process data and train data. Then click on

import data the data set will be take the reading then click on train data it will also take the data and finally click on pre-process data then it will also take the data of the image and the final accuracy also be come to 97% and the quality of checking of dragon fruit table be displaced in that table the actual dataset will be good quality and the detected dragon will be also good quality. The final output of graph will confuse matrix and actual matrix. the actual matrix frequency is 1 and the confusion matrixes zero.

CHAPTER 5

IMPLEMENTATION DETAILS

5.1 DEVELOPMENT AND DEPLOYMENT SETUP

The project was done using python programming using visual studio platform.

It was done in different stages. They are:

- Dataset
- Importing the necessary libraries
- Retrieving the images
- Splitting the dataset
- Building the model
- Apply the model and plot the graphs for accuracy and loss
- Accuracy on test set
- Saving the Trained Model

Dataset:

In the first module, we developed the system to get the input dataset for the training and testing purpose.

Importing the necessary libraries:

We will be using Python language for this. First, we will import the necessary libraries such as keras for building the main model, sklearn for splitting the training and test data, PIL for converting the images into array of numbers and other libraries such as pandas, NumPy, matplotlib and TensorFlow.

Retrieving the images:

We will retrieve the images and their labels. Then resize the images to (224,224) as all images should have same size for recognition. Then convert the images into NumPy array.

Splitting the dataset:

Split the dataset into train and test. 80% train data and 20% test data.

Building of the Model:

For building the we will use sequential model from keras library. Then we will add the layers to make convolutional neural network. In the first 2 Conv2D layers we have used 32 filters and the kernel size is (5,5).

In the MaxPool2D layer we have kept pool size (2,2) which means it will select the maximum value of every 2 x 2 area of the image. By doing this dimension of the image will reduce by factor of 2. In dropout layer we have kept dropout rate = 0.25 that means 25% of neurons are removed randomly.

We apply these 3 layers again with some change in parameters. Then we apply flatten layer to convert 2-D data to 1-D vector. This layer is followed by dense layer, dropout layer and dense layer again. The last dense layer outputs 2 nodes as the brain tumor or not. This layer uses the SoftMax activation function which gives probability value and predicts which of the 2 options has the highest probability.

Apply the model and plot the graphs for accuracy and loss:

We will compile the model and apply it using fit function. The batch size will be 2. Then we will plot the graphs for accuracy and loss. We got average validation accuracy of 100% and average training accuracy of 98.7%.

Accuracy on test set:

We got an accuracy of 100% on test set

Saving the Trained Model:

Once confident enough to take our trained and tested model into the production-ready environment, the first step is to save it into a .h5 or .pkl file using a library like pickle.

5.2 ALGORITHMS

The Algorithms used in this project are CNN and RESNET. The detailed information about the algorithms is as follows:

5.2.1 Convolutional Neural Network

A Convolutional Neural Network (CNN) is a type of Deep Learning neural network architecture commonly used in Computer Vision. Computer vision is a field of Artificial Intelligence that enables a computer to understand and interpret the image or visual data.

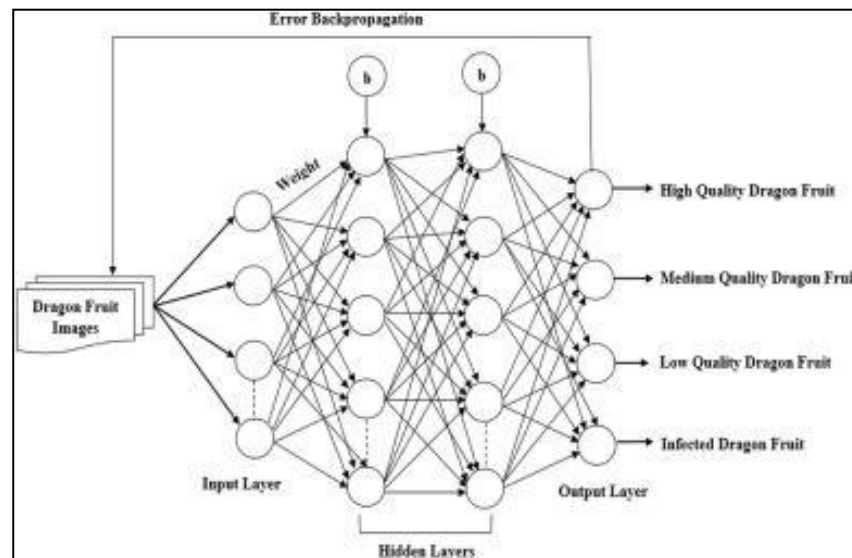


Fig: 5.2.1: CNN layers

When it comes to Machine Learning, Artificial Neural Networks, perform well. Neural Networks are used in various datasets like images, audio, and text. Different types of Neural Networks are used for different purposes, for example for predicting the sequence of words we use Recurrent Neural Network more precisely an LSTM, similarly for image classification we use Convolution Neural networks. In this blog, we are going to build a basic building block for CNN.

In a regular Neural Network, there are three types of layers:

- 1. Input Layers:** It is the layer in which we give input to our model. The number of neurons in this layer is equal to the total number of features in our data (number of pixels in the case of an image).
- 2. Hidden Layer:** The input from the Input layer is then feed into the hidden layer. There can be many hidden layers depending upon our model and data

size. Each hidden layer can have different numbers of neurons which are generally greater than the number of features. The output from each layer is computed by matrix multiplication of output of the previous layer with learnable weights of that layer and then by the addition of learnable biases followed by activation function which makes the network nonlinear.

3. Output Layer: The output from the hidden layer is then fed into a logistic function like sigmoid or SoftMax which converts the output of each class into the probability score of each class.

The data is fed into the model and output from each layer is obtained from the above step is called feedforward, we then calculate the error using an error function, some common error functions are cross-entropy, square loss error, etc. The error function measures how well the network is performing. After that, we backpropagate into the model by calculating the derivatives. This step is called Backpropagation which basically is used to minimize the loss.

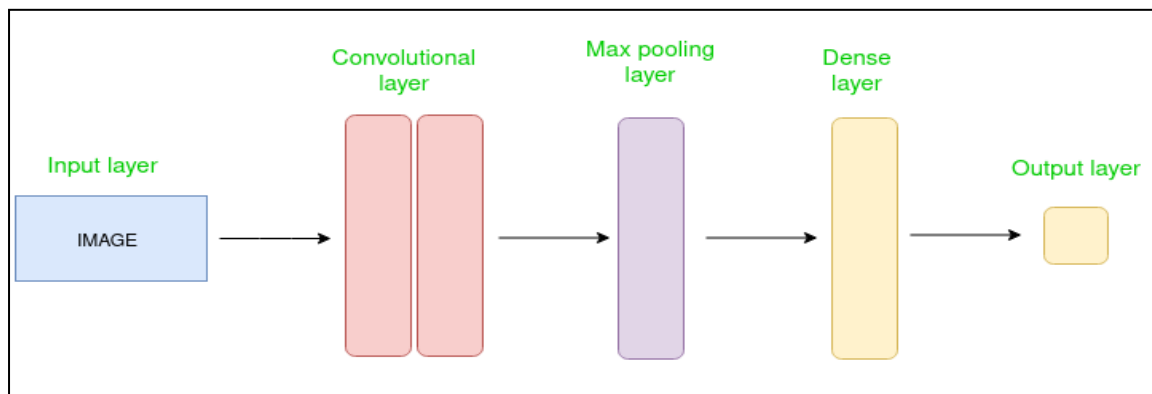


Fig: 5.2.2: Simple CNN Architecture

The Convolutional layer applies filters to the input image to extract features, the Pooling layer down samples the image to reduce computation, and the fully connected layer makes the final prediction. The network learns the optimal filters through backpropagation and gradient descent.

5.2.3 ResNet Algorithm

After the first CNN-based architecture (Alex Net) that win the ImageNet 2012 competition, every subsequent winning architecture uses more layers in a deep neural network to reduce the error rate. This works for a smaller number of layers, but when we increase the number of layers, there is a common problem in deep learning associated with that called the Vanishing/Exploding gradient. This causes the gradient to become 0 or too large. Thus, when we increase number of layers, the training and test error rate also increases.

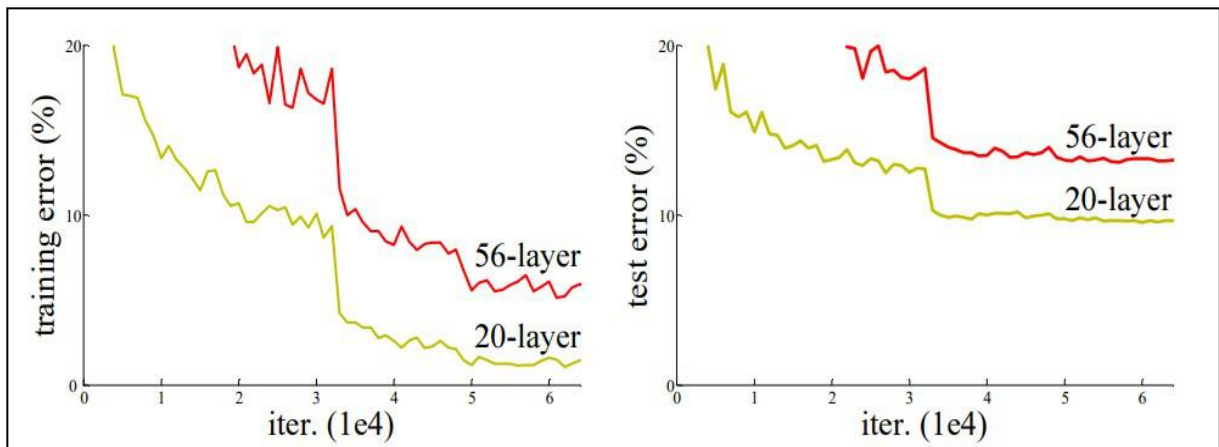


Fig 5.2.3: Comparison of 56 - layer and 20 – layer architecture

In the above plot, we can observe that a 56-layer CNN gives more error rate on both training and testing dataset than a 20-layer CNN architecture. ResNet, which was proposed in 2015 by researchers at Microsoft Research introduced a new architecture called Residual Network.

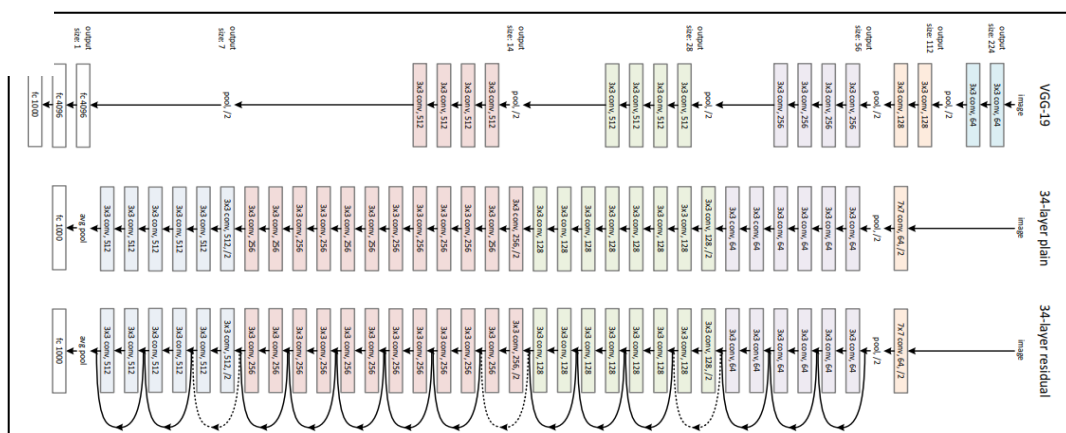


Fig: 5.2.4: RESNET 34 – Architecture

Residual Network: In order to solve the problem of the vanishing/exploding gradient, this architecture introduced the concept called Residual Blocks. In this network, we use a technique called skip connections. The skip connection connects activations of a layer to further layers by skipping some layers in between. This forms a residual block. Resnet are made by stacking these residual blocks together.

This network uses a 34-layer plain network architecture inspired by VGG-19 in which then the shortcut connection is added. These shortcut connections then convert the architecture into a residual network.

5.3 TESTING

5.3.1 Import Data

The selection of an appropriate dataset is crucial for the accurate detection of dragon fruit using deep learning techniques. A good dataset for this task should contain enough images of dragon fruit, covering different varieties, sizes, colors, and lighting conditions. The dataset should also include images of other fruits that might look like dragon fruit, such as pitaya or cactus fruit, to ensure that the model can distinguish between them accurately. The quality of the dataset is also important.

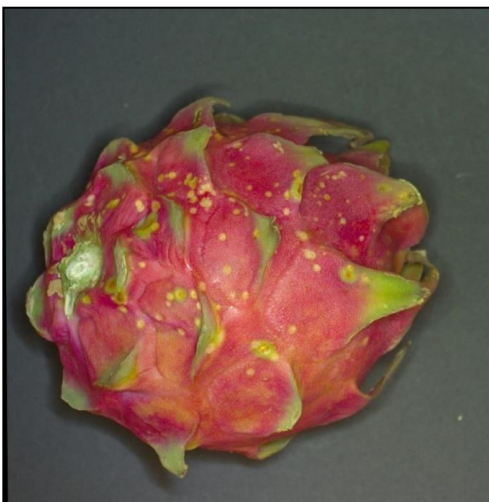




Fig 5.3.1: Dataset of Dragon fruit

The images should be high-resolution and well-lit, with a consistent background and minimal noise. The dataset should also be properly labeled, with each image labeled as either "dragon fruit" or "not dragon fruit" to enable the model to learn the characteristics of dragon fruit and differentiate it from other fruits. The test dataset should be like the training dataset in terms of the variety of images and objects. It is important to ensure that the test dataset does not contain any images that were used during training to prevent overfitting. Once we test our dataset, we used the trained deep learning model to make predictions on the images and compare the predicted labels with the true labels to evaluate the model's accuracy, precision, recall, and F1 score

Finally, the dataset should be diverse and representative of the real-world scenarios in which the model will be deployed. This means that the dataset should include images of dragon fruit in various stages of ripeness, as well as images captured from different angles and distances. In summary, the selection of an appropriate dataset is crucial for the accurate detection of dragon fruit using deep learning techniques. A good dataset should be diverse, high-quality, well-labeled, and representative of real-world scenarios.

CHAPTER 6

RESULTS AND DISCUSSION

Based on the results and analysis conducted, several conclusions were attained. The dragon fruit ripeness detection application is developed by firstly training the prepared dataset (training process) which produced labels and models that are used by the system to examine the test data (testing process) and label the output pictures.

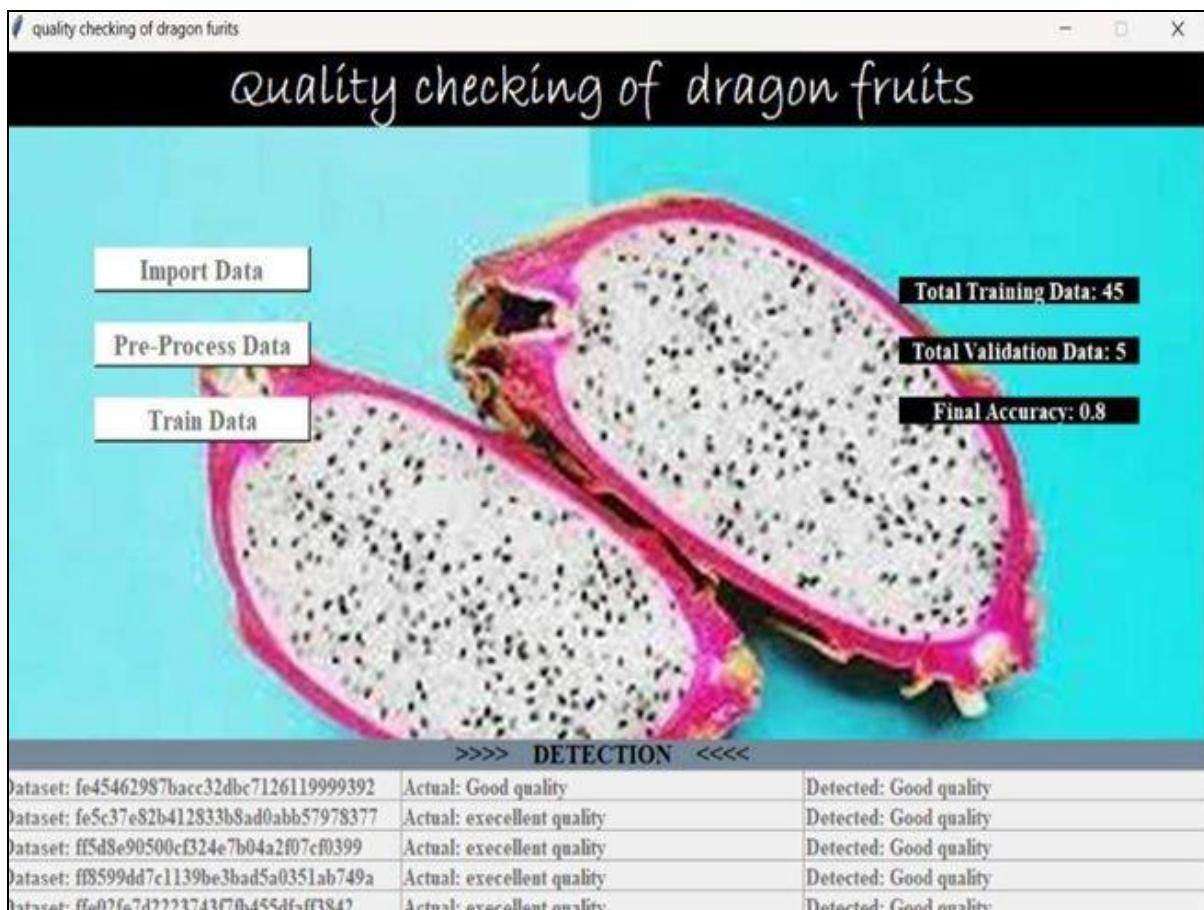


Fig: 6.1: Final Output

From the model training conducted with a model of maximum validation accuracy which is the training model with 50 epochs. This study utilizes new testing data, with a total of 100 pictures which were tested using the model formed.

The test results of the 50-epoch model show 91% accuracy level in classifying the ripeness of the dragon fruit. The test on the model in this study utilize the confusion matrix method, with a test condition in which the 100-test data must be new. To determine the false positive rate and true positive rate value of the epochs tested, the AUROC value attained from using the trapezoid method which is categorized as very good.

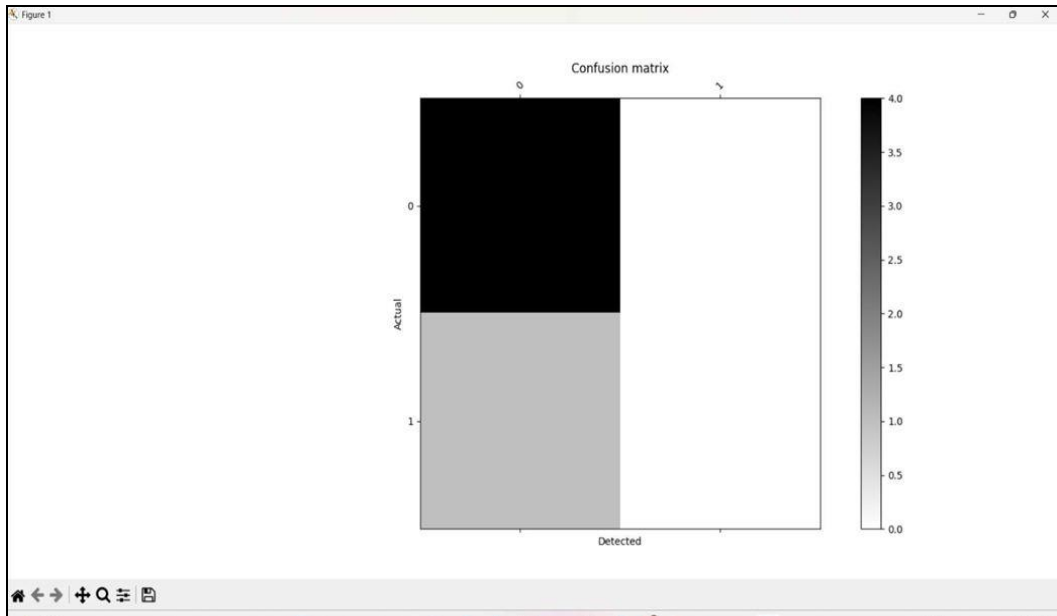


Fig: 6.2: Final graph output

To evaluate the model's performance across different classification thresholds, we have used the area under the receiver operating characteristic (AUROC) curve. The AUROC value provides a measure of how well the model can distinguish between positive and negative samples at different classification thresholds. A high AUROC value indicates a better-performing classifier. We used the trapezoid method to calculate the AUROC value, which is a commonly used method for computing the area under the curve.

Therefore, the study provides evidence that the deep learning model with 50 epochs is effective at classifying the ripeness of dragon fruit.

CHAPTER 7

CONCLUSION

7.1 CONCLUSION

The RESNET 152 deep CNN based model was developed to identify the mellowness of the dragon fruit and the training and the testing process was done using the live image gathered on the different stages of the dragon fruit. The results observed proved to have a better accuracy in training and testing even in increased number of epochs, unlike the VGGNET where the accuracy gets degraded as the network becomes deeper and the number of epochs increases. The AUROC based on the test results was observed to be 1.0 which proves that the proposed model does a better prediction than the prevailing.

The detection of dragon fruit using deep learning has shown promising results in accurately identifying and localizing dragon fruit in images. By using deep learning algorithms, such as convolutional neural networks (CNNs), it is possible to automatically extract relevant features from images and use them to classify whether an image contains a dragon fruit or not.

This technology can be applied in various fields such as agriculture, food processing, and marketing. For instance, it can help farmers to detect the ripeness of dragon fruits at an early stage, and it can help marketers to accurately identify the fruit in product packaging and labeling.

Overall, the application of deep learning in dragon fruit detection has shown great potential and is a promising field for future research and development.

7.2 FUTURE WORK

The detection of dragon fruit using deep learning has potential applications in the agricultural sector, specifically in the grading and sorting of fruits. Here are some potential future works for the detection of dragon fruit using deep learning:

1. *Dataset creation:* A large and diverse dataset of dragon fruit images needs to be created for training and validating the deep learning models. The dataset should contain images of dragon fruits at various stages of ripeness and with different levels of damage.
2. *Model selection:* Different deep learning models should be evaluated for their accuracy in detecting dragon fruit, such as convolutional neural networks (CNNs) and faster R-CNN. The selected model should be optimized for the specific task of detecting dragon fruit.
3. *Training and Validation:* The model should be trained using the created dataset, and the accuracy should be validated using a separate dataset. The model should also be tested on different dragon fruit varieties to ensure its generalizability.
4. *Deployment:* The model should be deployed in an agricultural setting, where it can be used to sort and grade dragon fruits. The model should also be integrated with an automated system that can sort the fruits based on their quality and ripeness.
5. *Further improvements:* The model should be continuously improved by collecting new data and updating the training dataset. Other deep learning techniques such as transfer learning and object tracking can also be explored to improve the accuracy and speed of the detection process.

7.3 IMPLEMENTATION ISSUES

1. The degradation caused in the convergence of the deeper networks, resulting in accuracy saturation followed by degradation
2. Error in training.
3. The optimization of the deeper networks was not much easy. The residual layer eluded the hoping of each small piled layers in order to straightforwardly fitting the sought-after fundamental mapping. The layers are explicitly let to fit into the residual mapping. The short links achieved through forward feed, help in jumping from one layer to the next and also in sorting out the mapping, further the outputs obtained are added to the piled layers output.

4. The benefits of the RESNET work are it is easy to optimize and has higher accuracy when the depth is increased unlike the previous network that results in degradation and saturated accuracy. The residual networks are similar to plain networks but have short links inserted to it, this makes the network into the counterpart residual network. The RESNET network is encompassed with much fewer filters and low complexity compared to the VGGNETs the residual network uses the short links directly if the inputs and the outputs are of similar dimensions and does the identity mapping and more zero entries to increase the dimensions as well as utilizes a projection shortcut to match the dimension if the input and the output has different dimensions. For the feature map with size two the operations the short links are performed in two strides.

REFERENCES

- [1] Koirala, K. B. Walsh, and Z. Wang, "Attempting to estimate the unseen-correction for occluded fruit in tree fruit load estimation by machine vision with deep learning," *Agronomy*, vol. 11, no. 2, p. 347, 2021.
- [2] Farhadi and J. Redmon, "Yolov3: an incremental improvement," *Computer Vision and Pattern Recognition*, vol. 1804, pp. 1–6, 2018.
- [3] Dr. T. Vijayakumar, Mr. R. Vinothkanna, (2022) Introduced "Mellowness Detection of Dragon Fruit Using Deep Learning Strategy".
- [4] H. Wang, Q. Zhao, H. Li, and R. Zhao, "Polynomial-based smooth trajectory planning for fruit-picking robot manipulator," *Information Processing in Agriculture*, vol. 9, no. 1, pp. 112 -122, 2022.
- [5] José Naranjo-Torres, Marco Mora, Ruber Hernández-García, Ricardo J. Barrientos, Claudio Fredes and Andres Valenzuela, gave a "Review of Convolutional Neural Network Applied to Fruit Image Processing"
- [6] L. Jian, Z. Mingrui, and G. Xifeng, "A fruit detection algorithm based on r-fcn in natural scene," in *2020 Chinese Control and Decision Conference (CCDC)*, pp. 487–492, Hefei, China, 2020.
- [7] N. T. Anderson, K. B. Walsh, and D. Wulfsohn, "Technologies for forecasting tree fruit load and harvest timing—from ground, sky and time," *Agronomy*, vol. 11, no. 7, p. 1409, 2021.
- [8] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *Advances in Neural Information Processing Systems*, vol. 28, 2015.
- [9] T. Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, Venice, Italy, 2017.
- [10] Wenli Zhang, Kaizhen Chen, Chao Zheng, Yuxin Liu proposed "Efficient Data Labeling Method for Multi-shape, Cross-Species Fruit Detection".

- [11] W. Liu, D. Anguelov, D. Erhan et al., "Ssd: single shot multibox detector," in Proc. European Conference on Computer Vision, Springer, 2016.
- [12] Y. Ge, Y. Xiong, and P. J. From, "Symmetry-based 3d shape completion for fruit localization for harvesting robots," Biosystems Engineering, vol. 197, pp. 188–202, 2020.
- [13] Z. Yang, "Research on the application of rigid-flexible compound driven fruit picking robot design in realizing fruit picking," Journal of Physics: Conference Series. IOP Publishing, vol. 1952, no. 2, article 022071, 2021.