A

Major Project Report

On

Detection and Calorie Estimation of Food Using Deep Learning

Submitted to **CMREC, HYDERABAD**

In Partial Fulfillment of the requirements for the Award of Degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING -DATA SCIENCE

Submitted By

B. RAMCHANDU	**(218R1A6770)**
Ch. SAI PRASAD	**(218R1A6776)**
K. LALITH	**(218R1A6797)**
P. UDAY KUMAR	**(218R1A67B7)**

Under the Esteemed guidance of

Mr. E. Laxman

Assistant Professor, Department of CSE (Data Science)

Department of Computer Science & Engineering -Data Science

CMR ENGINEERING COLLEGE

UGC AUTONOMOUS

(Approved by AICTE, NEW DELHI, Affiliated to JNTU, Hyderabad)
Kandlakoya, Medchal Road, Medchal Dist. Hyderabad 501401.

2024-2025

CMR ENGINEERING COLLEGE
UGC AUTONOMOUS

(Accredited by NBA Approved by AICTE NEW DELHI, Affiliated to JNTU,Hyderabad)
Kandlakoya, Medchal Road, Hyderabad-501 401

Department of Computer Science & Engineering -Data Science

<u>CERTIFICATE</u>

This is to certify that the project entitled **"Detection and Calorie Estimation of Food Using Deep Learning"** is a Bonafide work carried out by

B. RAMCHANDU	**(218R1A6770)**
Ch. SAI PRASAD	**(218R1A6776)**
K. LALITH	**(218R1A6797)**
P. UDAY KUMAR	**(218R1A67B7)**

in partial fulfillment of the requirement for the award of the degree of **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)** from CMR Engineering College, affiliated to JNTU, Hyderabad, under our guidance and supervision. The results presented in this Major project have been verified and are found to be satisfactory. The results embodied in this Major project have not been submitted to any other university for the award of any other degree or diploma.

Internal Guide	**Major Project Coordinator**	**Head of the Department**	**External Examiner**
Mr. E. Laxman	**Mr.B.Kumara Swamy**	**Dr. M. Laxmaiah**	
Assistant Professor	Assistant Professor	Professor & H.O.D	
CSE (Data Science),	CSE (Data Science),	CSE (Data Science),	
CMREC,Hyderabad	CMREC, Hyderabad	CMREC,Hyderabad	

DECLARATION

This is to certify that the work reported in the present Major project entitled "**Detection and Calorie Estimation of Food Using Deep Learning"** is a record of Bonafide work done by us in the Department of Computer Science and Engineering (Data Science), CMR Engineering College, UGC Autonomous, JNTU Hyderabad. The reports are based on the project work done entirely by us and not copied from any other source. We submit our project for further development by any interested students who share similar interests to improve the project in the future.

The results embodied in this Major project report have not been submitted to any other University or Institute for the award of any degree or diploma to the best of our knowledge and belief.

B.RAMCHANDU	**(218R1A6770)**
Ch.SAI PRASAD	**(218R1A6776)**
K.LALITH	**(218R1A6797)**
P.UDAY KUMAR	**(218R1A67B7)**

ACKNOWLEDGMENT

We are extremely grateful to **Dr. A. Srinivasula Reddy**, Principal and **Dr. M. Laxmaiah,** HOD**, Department of CSE (Data Science), CMR Engineering College** for their constant support.

We are extremely thankful to **Mr. E. Laxman**, Assistant Professor, Internal Guide, Department of CSE(DS), for his/ her constant guidance, encouragement and moral support throughout the project.

We will be failing in duty if We do not acknowledge with grateful thanks to the authors of the references and other literatures referred in this Project.

We thank **Mr. B .Kumara Swamy,** Assistant Professor ,CSE(DS) Department , Major Project Coordinator for his constant support in carrying out the project activities and reviews.

We express my thanks to all staff members and friends for all the help and co-ordination extended in bringing out this project successfully in time.

Finally, We are very much thankful to our parents who guided me for every step.

B.RAMCHANDU	**(218R1A6770)**
Ch.SAI PRASAD	**(218R1A6776)**
K.LALITH	**(218R1A6797)**
P.UDAY KUMAR	**(218R1A67B7)**

ABSTRACT

The modern world healthy body depends on the number of calories consumed, hence monitoring calorie intake is necessary to maintain good health. At the point when your Body Mass Index is somewhere in between from 25 to 29. It implies that you are conveying overabundance weight. Assuming your BMI is more than 30, it implies you have obesity. To get in shape or keep up the solid weight individuals needs to monitor the calorie they take. The existing system calorie estimation is to be happened manually. The proposed model is to provide unique solution for measuring calorie by using deep learning algorithm. The food calorie calculation is very important in medical field. Because this food calorie is provide good health condition. This measurement is taken from food image in different objects that is fruits and vegetables. This measurement is taken with the help of neural network. The tensor flow is one of the best methods to classify the machine learning method. This method is implementing to calculate the food calorie with the help of Convolutional Neural Network. The input of this calculated model is taken an image of food. The food calorie value is calculated the proposed CNN model with the help of food object detection. The primary parameter of the result is taken by volume error estimation and secondary parameter is calorie error estimation. The volume error estimation is gradually reduced by 20%. That indicates the proposed CNN model is providing higher accuracy level compare to existing model.

CONTENTS

LIST OF FIGURES

CHAPTER-1

INTRODUCTION

1.1 INTRODUCTION

Food is the key of human’s body. Nowadays more and more people care about the dietary intake since unhealthy diet leads to numerous diseases. A diet plan always needs to take into consideration the total number of calories to be consumed to maintain a fit and healthy life. Weight is an ailment and means you have an excess of muscle to fat ratio. Assuming your BMI is more than 30, it implies you have obesity. Weight can have different reasons. One of these reasons is burning-through a lot calorie. Devouring an excessive number of calories implies that measure of calories that you are taking is greater than measure of calories you consume. The body stores the abundance calorie as muscle to fat ratio. To get in shape or keep up the solid weight individuals needs to monitor the calorie they take. Yet, this interaction can be troublesome and tiring. Since individuals will in general dodge troublesome and tiring things, they regularly don't follow the amount they eat and this may prompt stoutness. Among these examinations, two fundamental variables of the precision change are object location calculation, volume and calorie assessment strategy. For instance, Support Vector Machine (SVM) is utilized for object discovery and characterization. For volume and calorie, the assessment reference point is the thing that has the effect. Utilizing diverse reference points impacts the reasonableness of the application. Reason for this investigation is to make this following simpler. For this, we concoct a Machine Learning Base methodology. With just two pictures (one is from the side and one is from the highest point) of the food and a solitary coin, individuals will actually want to know the calorie of the food that they are eating. In this investigation we discover and characterize the food and make an expectation about the volume of the food. At last, we figure the calorie of the food dependent on the volume that models have anticipated. Nonetheless, we found that assessing the calories straightforwardly was giving us much precise outcomes. But, in most cases, unfortunately people face difficulties in estimating and measuring the amount of food intake due to the mainly lack of nutritional information, which includes manual process of writing down this information, and other reasons. As such, a system to record and measure the number of calories consumed in a meal is of a great help. Hence accurate prediction of food calorie is equally important in such cases. In the last three years, object classification and detection capabilities have dramatically improved due to advances in deep learning and convolutional networks. Harnessing this technology to accurately classify and detect food objects is significantly essential for a healthy and fit life. But to always refer to the nutritional content in each food item is an extremely tedious task. In this project, we use a deep learning-based fruit image recognition algorithm to improve the accuracy of dietary assessment and analyze each of the network architecture.

1. Objective

In today's world, maintaining a healthy diet is crucial as unhealthy eating habits lead to numerous diseases, including obesity. Obesity occurs when calorie intake exceeds calorie expenditure, causing excess fat accumulation. Tracking daily calorie intake is essential for weight management, but manual methods are often tedious and inaccurate. To address this issue, we propose a Machine Learning-based approach to estimate food calories using deep learning techniques.

Our system utilizes two images of a food item—one from the side and one from the top—along with a reference object (such as a coin) to estimate the food's volume. The model first detects and classifies the food using deep learning techniques such as Support Vector Machines (SVM) or Convolutional Neural Networks (CNNs). Then, it predicts the food's volume and calculates its calorie content based on pre-existing nutritional data. This eliminates the need for manual tracking and enhances accuracy.

Recent advancements in deep learning have significantly improved object detection and classification. By leveraging this technology, our project aims to develop an automated dietary assessment system, making calorie tracking more accessible. The system simplifies nutritional analysis, helping individuals maintain a balanced diet with ease and accuracy.

2. Motivation

In today's fast-paced world, unhealthy eating habits have led to a significant rise in diet-related diseases such as obesity, diabetes, and cardiovascular disorders. One of the key challenges in maintaining a healthy lifestyle is accurately tracking daily calorie intake. Traditional methods, such as manually logging food consumption or estimating portion sizes, are often inaccurate, time-consuming, and inconvenient. As a result, many individuals fail to monitor their dietary habits effectively.

With recent advancements in deep learning and computer vision, automated food recognition and calorie estimation have become feasible. These technologies can help individuals make informed dietary choices without the burden of manual tracking. Developing a system that accurately identifies food items, estimates their volume, and calculates their calorie content can greatly assist people in maintaining a balanced diet.

The motivation behind this project is to bridge the gap between nutritional awareness and technological advancements by creating an easy-to-use, accurate, and efficient food calorie estimation system. By leveraging deep learning techniques, our approach aims to simplify dietary tracking, promote healthier eating habits, and ultimately contribute to improved public health outcomes.

1.3 Problem Statement

Obesity and other diet-related health issues have become a growing concern due to excessive calorie consumption and poor dietary habits. Accurate tracking of food intake is essential for maintaining a healthy lifestyle, but manual methods, such as writing down nutritional information, are tedious and prone to human error. Most individuals struggle with estimating portion sizes and calorie content due to a lack of precise nutritional knowledge.

Existing food calorie estimation systems either require complex manual inputs or lack accuracy in volume estimation. With recent advancements in deep learning and computer vision, automated food recognition and calorie estimation can significantly improve dietary assessment. However, current approaches often fail to provide precise measurements due to variations in food size, shape, and presentation.

This project aims to develop a machine learning-based system that accurately detects food items, estimates their volume using two images (one from the top and one from the side) with a reference object, and calculates calorie content. By leveraging deep learning techniques, this system seeks to simplify and enhance food tracking, providing an easy-to-use solution for individuals to monitor their dietary intake effectively.

4. Applications

1. **Personal Health & Fitness Tracking**

- Helps individuals monitor their daily calorie intake effortlessly.
- Assists fitness enthusiasts in maintaining a balanced diet based on their fitness goals.

2. **Obesity & Weight Management**

- Provides an automated tool for people trying to lose or maintain weight.
- Helps dietitians and nutritionists in creating personalized diet plans for clients.

3. **Medical & Clinical Use**

- Supports healthcare professionals in managing patients with dietary-related conditions such as diabetes, hypertension, and obesity.
- Aids in monitoring and regulating the nutritional intake of hospital patients.

4. **Dietary Research & Nutrition Studies**

- Enhances research on dietary patterns and their impact on health.
- Provides accurate data collection for large-scale nutritional studies.

5. **Smart Restaurants & Food Industry**

- Can be integrated into restaurant systems to provide customers with real-time calorie information.
- Helps in developing smart menus with automated calorie tracking.

6. **Food & Wellness Apps**

- Can be incorporated into mobile applications for seamless diet tracking.
- Provides an AI-powered solution for individuals looking to adopt healthier eating habits.

7. **Assistance for Visually Impaired Individuals**

- Helps visually impaired individuals identify food items and estimate their nutritional values through image-based recognition.

8. Educational & Awareness Programs

- Can be used in schools and universities to educate students about healthy eating and nutrition.
- Promotes awareness of calorie intake and portion control among the general public.

CHAPTER -2
LITERATURE SURVEY

1. Effect of high calorie diet on intestinal flora in LPS-induced pneumonia rats.:

https://www.nature.com/articles/s41598-020-58632-0

ABSTRACT: Intestinal flora plays an important role in inflammatory response to systemic or local organs of its host. High calorie diet has been shown to aggravate the condition of pneumonia and delay recovery, especially in children. However, the underlying mechanisms remain unclear. This study placed SPF rats in a conventional environment, high calorie diet or LPS atomization was performed respectively or combined. Analysis of high-throughput sequencing of intestinal content combined with animal weight, organ index, serum inflammatory factors indicators and bioinformatics found that after pulmonary infection combined with a high-calorie diet, rats showed significant changes such as weight loss and increased lung weight index, and their lung and intestinal tissues showed more obvious inflammatory changes. And its gut flora structure suggests, the abundance of Leuconostocaceae in significantly reduced; abundance of Staphylococcus, Planococcaceae, Staphylococcus, Staphylococcaceae, Bacillales, Gemellales and Aerococcus significant increased. The study showed that high calorie diet and LPS atomization synergistically promoted pneumonia process in rat pups, which is related to changes in structure of intestinal flora. It is worth noting that pneumonia rats fed by convention diet also causing intestinal flora imbalance.

2. Real-world application of machine learning and deep learning:

https://ieeexplore.ieee.org/document/8987844

ABSTRACT: The world today is running on the latest computer technologies and one of those is machine learning. The real life example that most of us know is speech recognition. Google Assistant is the common example for this Speech recognition. This google assistant is not only limited till `Ok Google', but it responds to all your questions in a smart way. It can manage all your calls or can book appointments. Imagine you fell down while de-boarding a bus. So, Next time you take care so that you don't fall that is something that your brain has interpreted from your past experience. This is what exactly deep learning is, it imitates human brain works. Deep learning is sub-branch of machine learning. It is able to build all new things based on its previous experiences. Many of us have heard about driverless cars and medical diagnosis. Recently google has developed a new technology where all your cardiovascular events can be predicted by eye scan so, that doctors can get a clear view of what is inside the body of a patient. These all are developed using machine learning. It has a capability to change the human world into a complete robotic world. Anyways, it also has its own disadvantages. This article discusses about those, Scope of machine learning, its Market potential, financial growth and Current applications of machine learning.

3. Cyber Secure Man-in-the-Middle Attack Intrusion Detection Using Machine Learning Algorithms:

https://www.researchgate.net/publication/338303164_Cyber_Secure_Man-in-the-Middle_Attack_Intrusion_Detection_Using_Machine_Learning_Algorithms

ABSTRACT: The main objective of this chapter is to enhance security system in network communication by using machine learning algorithm. Cyber security network attack issues and possible machine learning solutions are also elaborated. The basic network communication component and working principle are also addressed. Cyber security and data analytics are two major pillars in modern technology. Data attackers try to attack network data in the name of man-in-the-middle attack. Machine learning algorithm is providing numerous solutions for this cyber-attack. Application of machine learning algorithm is also discussed in this chapter. The proposed method is to solve man-in-the-middle attack problem by using reinforcement machine learning algorithm. The reinforcement learning is to create virtual agent that should predict cyber-attack based on previous history. This proposed solution is to avoid future cyber middle man attack in network transmission.

4. Smelling our appetite? The influence of food odors on congruent appetite, food preferences and intake:

https://www.sciencedirect.com/science/article/pii/S0950329320302287

ABSTRACT: We are surrounded by sensory food cues, such as odors, that may trigger (un)conscious decisions and even lead to (over)eating, it is therefore crucial to better understand the effect of food odors on behavioral responses. Food odor exposure has been shown to enhance appetite for food products with similar properties: sensory-specific appetite. This suggests that based on previous encounters with foods, we have learned to detect the nutritional content of foods, through our sense of smell. We investigated the influence of aware exposure of macronutrient-related odors on various measures of eating behavior, in a cross-over intervention study. Thirty two normal-weight healthy and unrestrained Dutch females took part in five test sessions. On each test session, they were exposed to one of five conditions (active smelling of clearly noticeable odors representing food high in carbohydrates, protein, and fat, low in calories, and a no-odor condition for 3-min) and assessed on specific appetite, food preferences and intake. Odor exposure increased congruent appetite after protein-related odor exposure. Similarly, protein-related odor exposure influenced the liking for protein foods and the preference ranking for savory products. However, food intake was not affected by smelling congruent food odors. Together this indicates that exposure to (aware) food odors may mostly influence appetite, but does not impact subsequent food intake. Moreover, appetite seems to be triggered by taste qualities rather than macronutrient information of the food, as signaled by olfactory cues. Future studies should investigate the role of awareness in more detail, to fully understand how odors might be used to steer people towards healthier food choices.

2.5 Deep learning in multi-object detection and tracking: state of the art:

https://www.isical.ac.in/~sankar/paper/Invited-2021.pdf

ABSTRACT: Object detection and tracking is one of the most important and challenging branches in computer vision, and have been widely applied in various fields, such as health-care monitoring, autonomous driving, anomaly detection, and so on. With the rapid development of deep learning (DL) networks and GPU's computing power, the performance of object detectors and trackers has been greatly improved. To understand the main development status of object detection and tracking pipeline thoroughly, in this survey, we have critically analyzed the existing DL network-based methods of object detection and tracking and described various benchmark datasets. This includes the recent development in granulated DL models. Primarily, we have provided a comprehensive overview of a variety of both generic object detection and specific object detection models. We have enlisted various comparative results for obtaining the best detector, tracker, and their combination. Moreover, we have listed the traditional and new applications of object detection and tracking showing its developmental trends.

CHAPTER-3

EXISTING SYSTEM

3.1 OVERVIEW:

With just two pictures (one is from the side and one is from the highest point) of the food and a solitary coin, individuals will actually want to know the calorie of the food that they are eating. In this investigation we discover and characterize the food and make an expectation about the volume of the food. At last, we figure the calorie of the food dependent on the volume that models have anticipated. Nonetheless, we found that assessing the calories straightforwardly was giving us much precise outcomes. But, in most cases, unfortunately people face difficulties in estimating and measuring the amount of food intake due to the mainly lack of nutritional information, which includes manual process of writing down this information, and other reasons. As such, a system to record and measure the number of calories consumed in a meal is of a great help. Hence accurate prediction of food calorie is equally important in such cases.

1. **Dietary Intake Assessment of Children and Adolescents:** The 24-hour dietary recall is a widely-used tool where individuals report to a dietitian or instructor what they consumed in the past 24 hours. This method is generally conducted through face-to-face interviews or self-reported online forms

2. **Calorie Card-Based System:** This system requires the user to place a special calorie card next to the food before taking a picture. The card, marked with standard measurements, helps in calibrating the size of the food and determining its portion size.

3. **Recognition of Multiple:** Food Images by Detecting Candidate Regions In the field of dietary assessment, recognizing multiple food items in a single image is a critical challenge. To address this, advanced systems have been developed to detect candidate regions within food images, which allow for the identification of multiple food items in a single picture.

3.2 DISADVANTAGES OF EXISTING SYSTEM:

1. **Reliance on Manual User Input and Lack of Real-Time Processing**
    - Many existing calorie tracking apps require users to manually enter their food details, which is time-consuming and prone to human error.
    - Lack of real-time processing makes it inconvenient for users who want instant feedback on their meals.
    - This manual dependency discourages regular usage, leading to inconsistent dietary tracking.
2. **Struggles with Lighting, Angles, and Food Presentation**
    - Variations in lighting conditions (dim vs. bright, natural vs. artificial) affect the accuracy of food recognition algorithms.
    - Different camera angles and perspectives can distort the perceived size and shape of the food, leading to incorrect portion estimation.
    - Food presentation differences (e.g., plated meals vs. takeout containers) introduce additional complexity in food identification.
3. **Difficulty in Recognizing Complex Dishes and Varying Portion Sizes**
    - Multi-ingredient dishes (e.g., salads, mixed rice bowls, or stews) are difficult to analyze due to overlapping food items.
    - Estimating portion sizes accurately is challenging, especially when the food lacks a clear boundary (e.g., mashed potatoes vs. a solid fruit).
    - Standard datasets often focus on individual food items rather than composite meals, limiting recognition accuracy.
4. **Limited Scalability and Inconsistent Accuracy for New Food Items**
    - Most food recognition models are trained on a fixed dataset and may not generalize well to new or uncommon food items.
    - Dishes from different cultures, regions, or homemade variations may not be correctly classified, leading to errors.
    - Expanding the database to include new foods requires additional training, making scalability an ongoing challenge.

5. **Dataset-Specific Performance and Generalization Issues**
    - Many machine learning models perform well on the dataset they are trained on but struggle with unseen food types, regional cuisines, or unique presentation styles.
    - Food variability in texture, color, and preparation methods makes it difficult for models to recognize dishes outside their training data.
    - Without continuous learning and adaptive datasets, these models fail to maintain high accuracy across diverse food environments.

CHAPTER-4
PROPOSED SYSTEM:

1. Overview:

In the last three years, object classification and detection capabilities have dramatically improved due to advances in deep learning and convolutional networks. Harnessing this technology to accurately classify and detect food objects is significantly essential for a healthy and fit life. But to always refer to the nutritional content in each food item is an extremely tedious task. In this project, we use a deep learning-based fruit image recognition algorithm to improve the accuracy of dietary assessment and analyze each of the network architecture.

The objective of estimating calories in foods is to help individuals make informed decisions about their diet and nutrition. By knowing the calorie content of different foods, people can better manage their caloric intake, maintain a healthy weight, and meet their dietary goals.

This information is particularly useful for individuals looking to lose weight, gain muscle, or simply maintain a balanced diet. Estimating calories in foods allows for better meal planning, portion control, and overall awareness of the nutritional value of what we eat, contributing to improved health and well-being.

Develop a deep learning model capable of Accurate recognizing and categorizing various types of food in images using Convolutional Neural Networks (CNNs) trained on a diverse dataset.

1. Advantages of proposed system:

1. CNN-Based Automated Approach for Real-Time Calorie Estimation

- Eliminates the need for manual input by automating food recognition and calorie estimation.
- Uses Convolutional Neural Networks (CNNs) to detect and classify food items instantly from images.
- Provides real-time analysis, making dietary tracking more efficient and user-friendly.

2. **Enhanced Feature Extraction for Improved Classification Accuracy**
    - The model captures essential food attributes such as texture, shape, and color, ensuring better recognition.
    - Overcomes challenges related to lighting variations, food presentation, and complex dish recognition.
    - Uses multi-layer feature extraction to distinguish between visually similar food items.
3. **Data Augmentation and Fine-Tuned CNN Architectures**
    - Data augmentation (rotation, scaling, flipping, contrast adjustment) increases the model's robustness.
    - Fine-tuning deep CNN architectures such as Res Net, Inception, and Efficient Net enhances classification accuracy.
    - Helps recognize food items from different cultural cuisines, ensuring wider applicability.
4. **Transfer Learning for Generalization Across Diverse Food Categories**
    - Uses pre-trained models on large-scale datasets to improve recognition of new and regional food items.
    - Enables the system to adapt quickly without requiring an extensive labeled dataset for training.
    - Enhances generalization, reducing the likelihood of misclassification when encountering unfamiliar food.
5. **Multi-View Analysis for Accurate Portion Estimation**
    - Uses dual-view images (top and side views) to estimate food volume more accurately.
    - Incorporates depth estimation techniques to overcome inaccuracies caused by camera angle variations.
    - Improves portion size assessment by using a reference object (e.g., a coin or standardized marker).

4.2 SYSTEM ARCHITECTURE

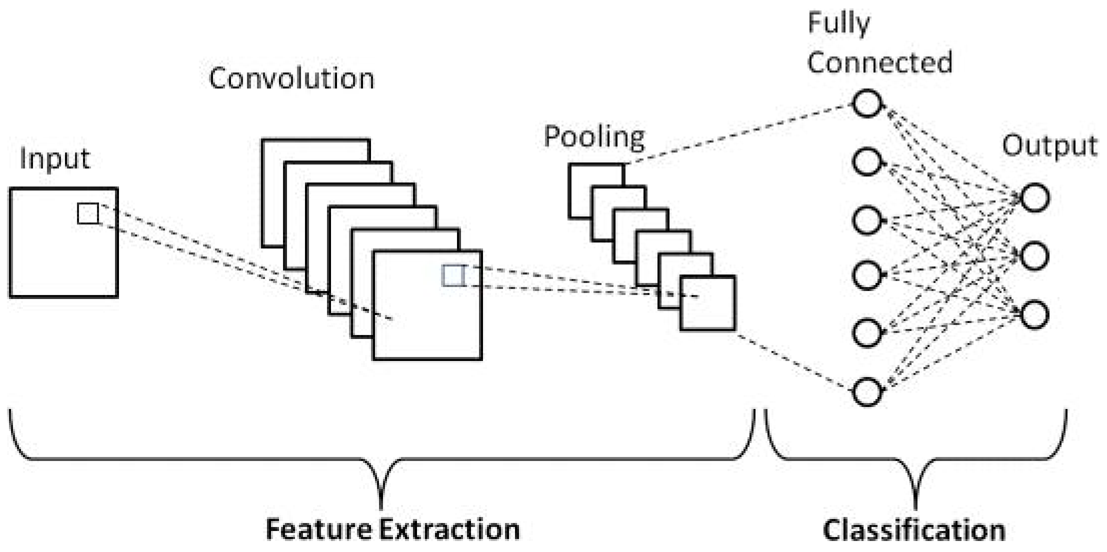


Fig.4.1.1 System architecture

1. Input Layer

- The CNN takes an image as input, represented as a matrix of pixel values.
- If the image is grayscale, it has a single channel (height × width). If it's RGB, it has three channels (height × width × 3).
- The input image is pre processed by normalizing pixel values to a certain range (e.g., [0,1] or [-1,1]) for better training efficiency.

2. Feature Extraction Phase

This phase focuses on learning important patterns from the image. It involves Convolutional Layers and Pooling Layers.

a) Convolution Layer

- A set of learnable filters (kernels) slides over the input image to detect low-level patterns such as edges, textures, and shapes.
- Each kernel is a small matrix (e.g., 3×3 or 5×5) that applies a dot product operation to the corresponding region of the image.
- The result of this operation forms a feature map that represents the detected pattern.

- The deeper convolutional layers detect more complex features, such as object parts and entire objects.
- Activation Function (ReLU): After convolution, a Rectified Linear Unit (ReLU) function is applied to introduce non-linearity, which helps the network learn complex patterns.

b) Pooling Layer

- Pooling reduces the spatial dimensions of the feature maps while retaining important features.
- The two most common pooling techniques are:
  - Max Pooling: Selects the maximum value in a region (e.g., a 2×2 window).
  - Average Pooling: Computes the average value in a region.
- Benefits of Pooling:
  - Reduces computational complexity.
  - Helps the network become more invariant to small translations.
  - Prevents overfitting by down sampling unnecessary details.

4.2.3. Classification Phase

Once the feature maps are extracted, the network needs to classify the image.

a) Fully Connected (FC) Layer

- The flattened feature maps (converted into a 1D vector) are passed into fully connected layers.
- Each neuron in the fully connected layer is connected to all neurons in the previous layer, enabling the model to learn high-level representations.
- The weights and biases are learned through training using backpropagation and gradient descent.

b) Output Layer

- The final fully connected layer produces scores (logits) for different classes.
- A Soft max activation function is applied to convert these scores into probabilities, ensuring that the sum of all probabilities equals 1.
- The class with the highest probability is selected as the final prediction.

4. Training Process of CNN

1. Forward Propagation:
    - The input image passes through convolution, pooling, and fully connected layers.
    - The network predicts an output class.
2. Loss Calculation:
    - The loss function (e.g., Cross-Entropy Loss) calculates the difference between the predicted and actual label.
3. Backward Propagation:
    - The network updates its weights using Gradient Descent and Backpropagation to reduce the loss.
4. Iteration Over Epochs:

    The process repeats for multiple epochs until the model converges and achieves high accuracy.

CHAPTER-5

UML DIAGRAMS

5. UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS OF UML :

The Primary goals in the design of the UML are as follows:

1. Clearly define system structure through visual representation of components, classes, and interactions for better understanding.
2. Provide a standardized language to facilitate communication between all stakeholders, ensuring alignment on system functionalities.
3. Identify and resolve potential issues early in the design process by visualizing interactions and data flow.
4. Model system behaviour to simulate scenarios and predict outcomes, enabling informed decision-making in system development.
5. Assist in system documentation, providing a reference for future modifications, updates, and maintenance tasks.
6. Improve system architecture by capturing relationships between components and ensuring appropriate scalability and flexibility.

5.1 DATA FLOW DIAGRAM:

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

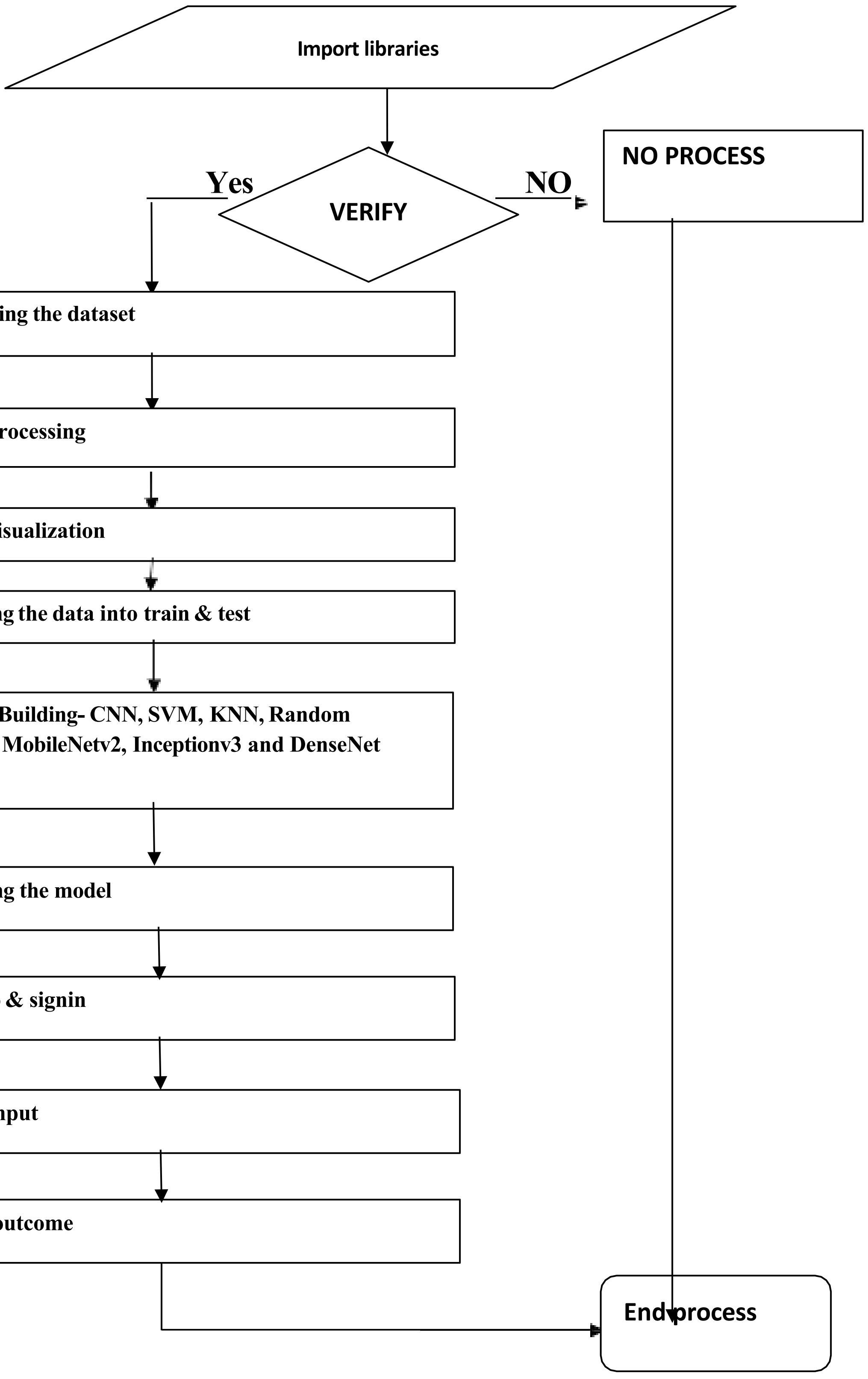


Fig.5.1.1 Dataflow diagram

5.2 Use case diagram:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

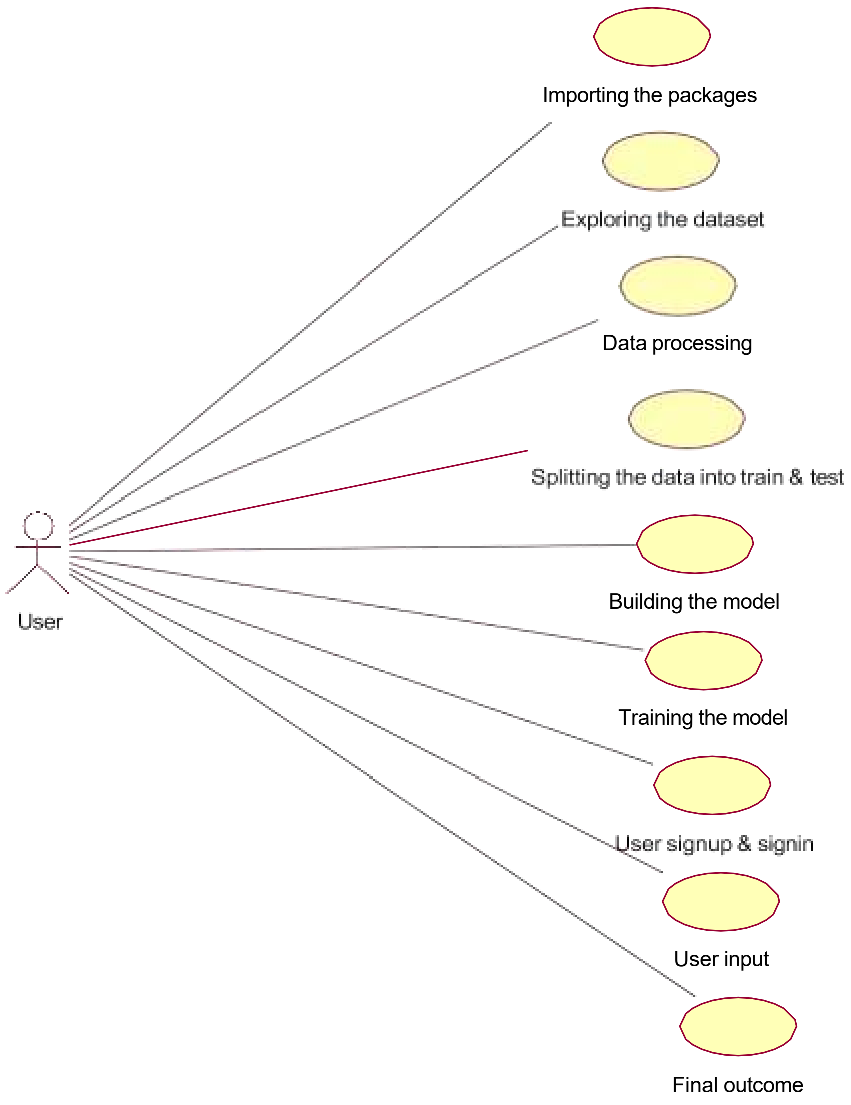


Fig.5.2.1 Use case diagram

5.3 Class diagram:

The class diagram is used to refine the use case diagram and define a detailed design of the system. The class diagram classifies the actors defined in the use case diagram into a set of interrelated classes. The relationship or association between the classes can be either an "is-a" or "has-a" relationship. Each class in the class diagram may be capable of providing certain functionalities. These functionalities provided by the class are termed "methods" of the class. Apart from this, each class may have certain "attributes" that uniquely identify the class.

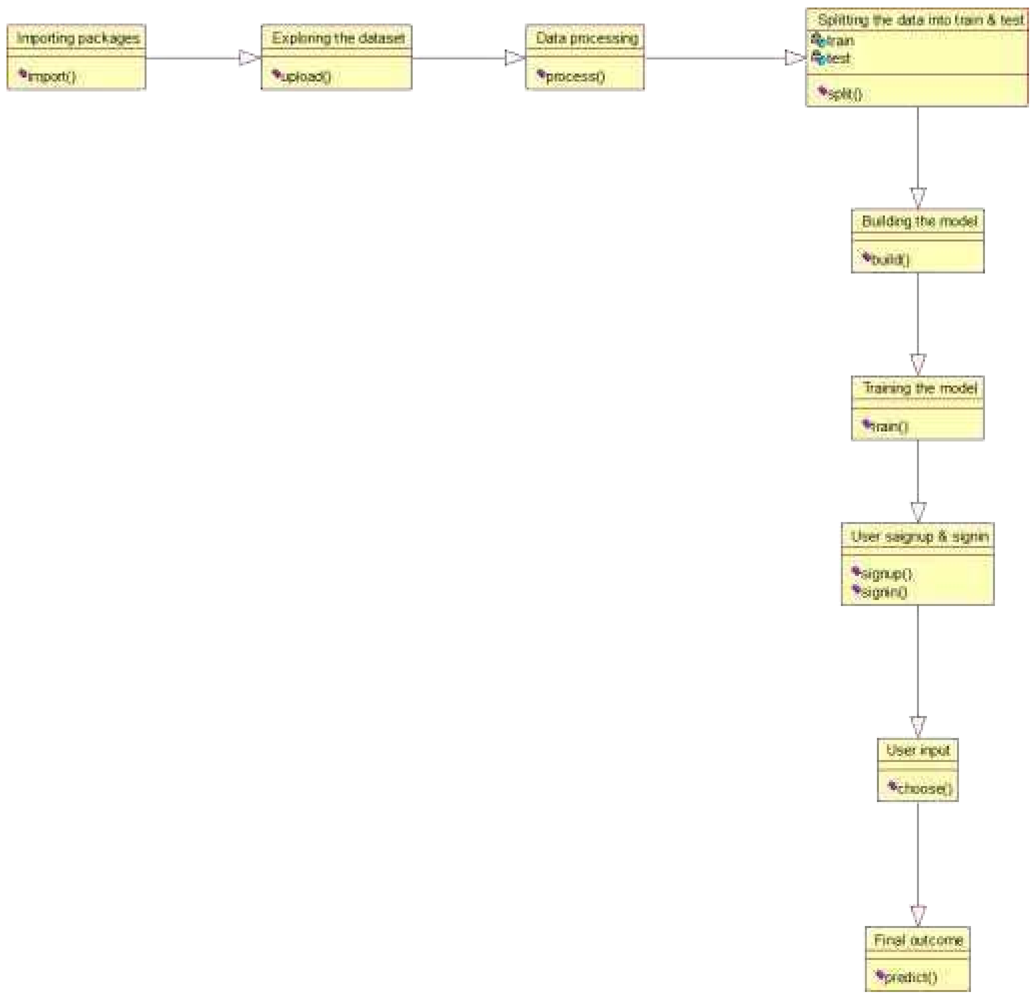


Fig.5.3.1 Class diagram

5.4 Activity diagram:

The process flows in the system are captured in the activity diagram. Similar to a state diagram, an activity diagram also consists of activities, actions, transitions, initial and final states, and guard conditions.

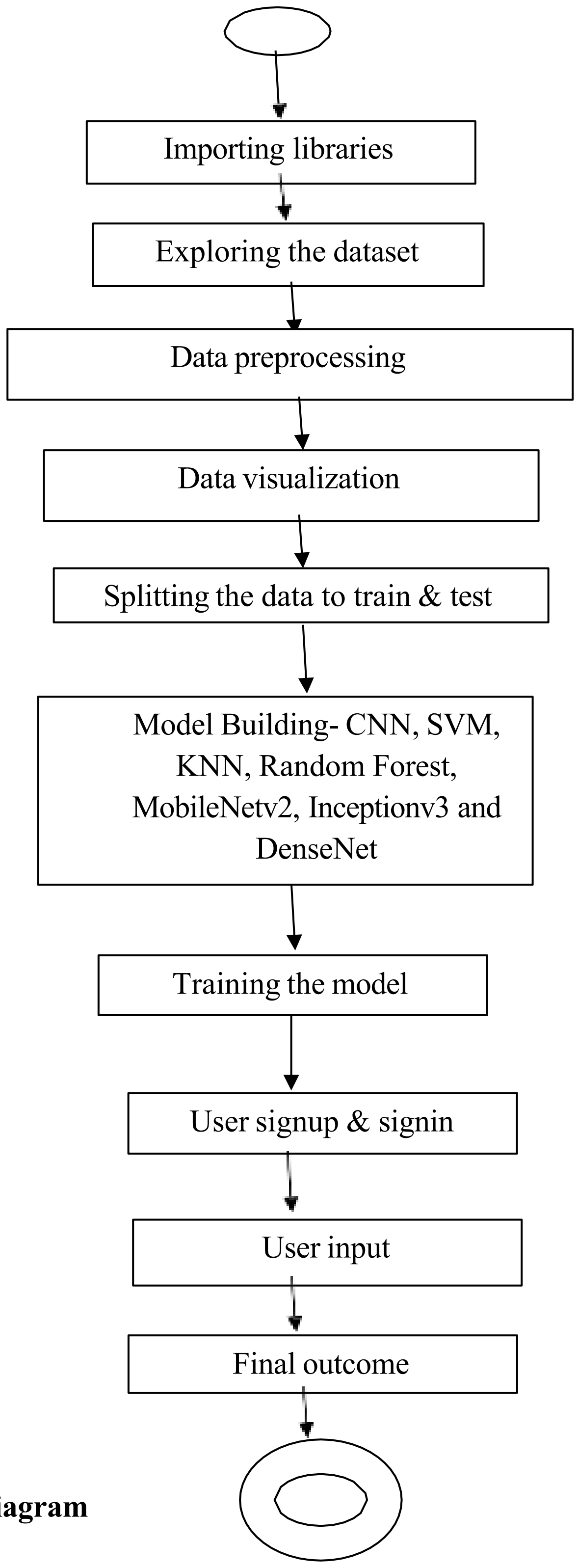


Fig.5.4.1 Activity diagram

5.5 Sequence diagram:

A sequence diagram represents the interaction between different objects in the system. The important aspect of a sequence diagram is that it is time-ordered. This means that the exact sequence of the interactions between the objects is represented step by step. Different objects in the sequence diagram interact with each other by passing "messages".

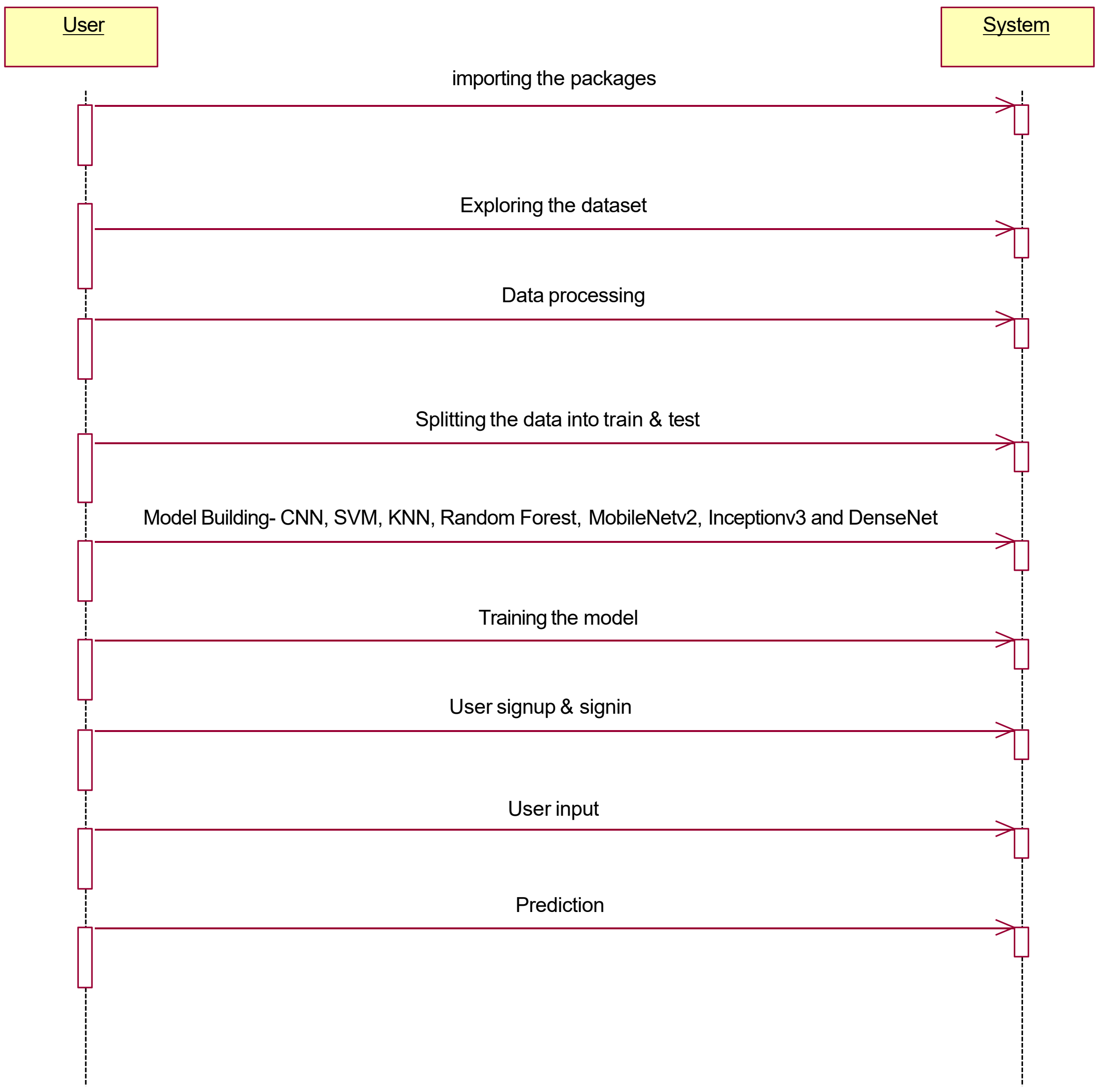


Fig.5.5.1 Sequence diagram

5.6 Collaboration diagram:

A collaboration diagram groups together the interactions between different objects. The interactions are listed as numbered interactions that help to trace the sequence of the interactions. The collaboration diagram helps to identify all the possible interactions that each object has with other objects.

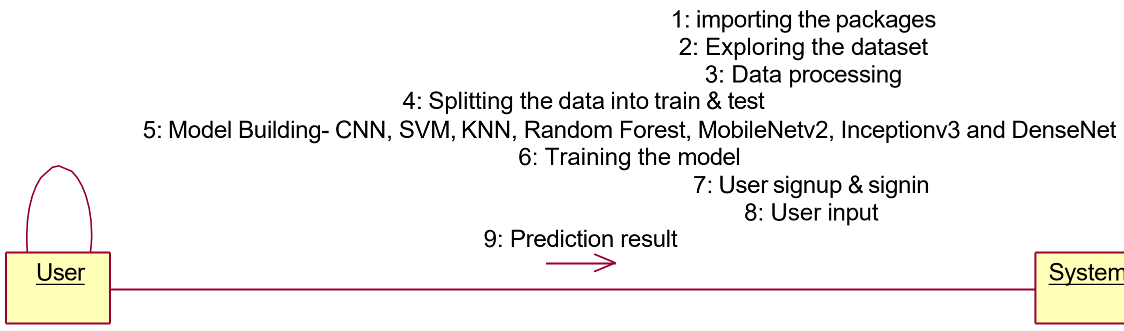

1: importing the packages
2: Exploring the dataset
3: Data processing
4: Splitting the data into train & test
5: Model Building- CNN, SVM, KNN, Random Forest, MobileNetv2, Inceptionv3 and DenseNet
6: Training the model
7: User signup & signin
8: User input
9: Prediction result
User
System

Fig.5.6.1 Collaboration diagram

5.7 Component diagram:

The component diagram represents the high-level parts that make up the system. This diagram depicts, at a high level, what components form part of the system and how they are interrelated. A component diagram depicts the components culled after the system has undergone the development or construction phase.

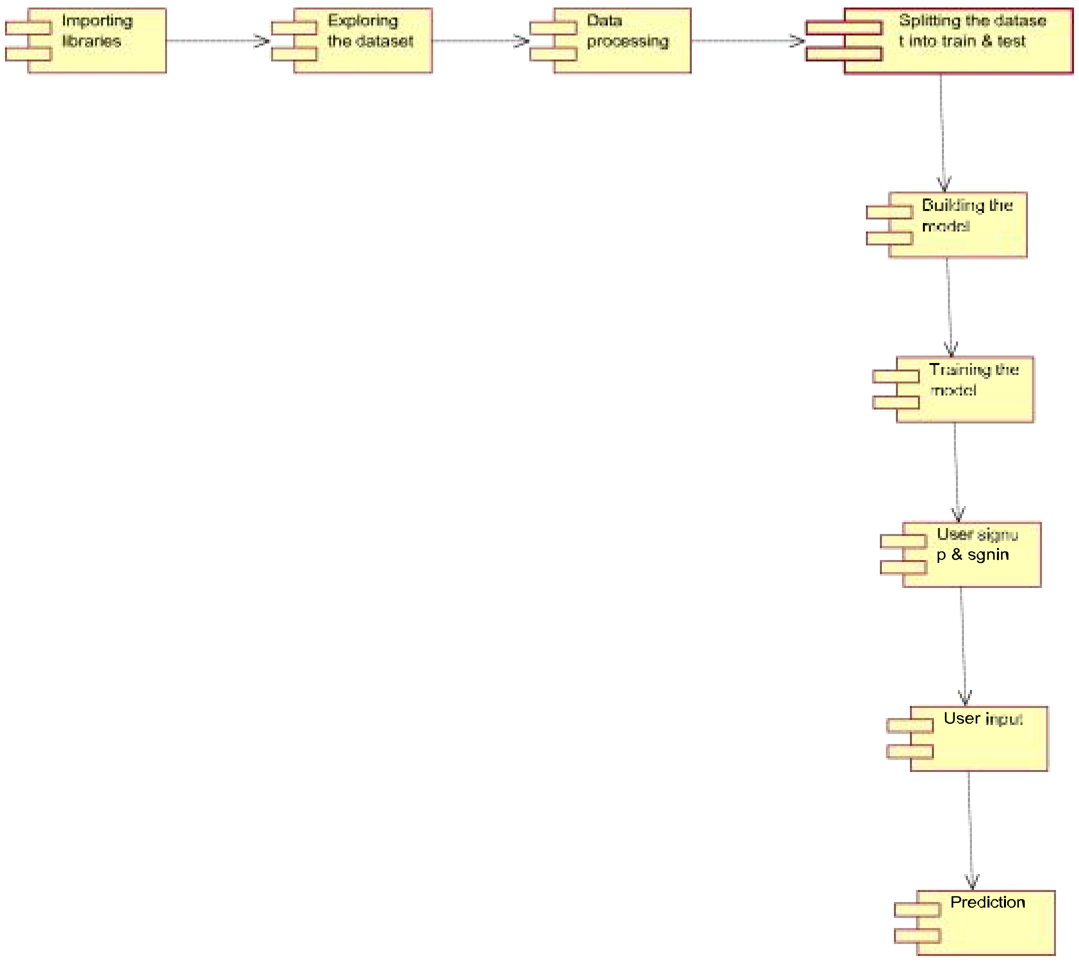


Fig.5.7.1 Component diagram

5.8 Deployment diagram:

The deployment diagram captures the configuration of the runtime elements of the application. This diagram is by far most useful when a system is built and ready to be deployed.

User
System

Fig.5.8.1 Deployment diagram

CHAPTER-6

SOFTWARE ENVIRONMENT

1. PYTHON LANGUAGE:

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed. Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

Python is a dynamic, high-level, free open source, and interpreted programming language. It supports object-oriented programming as well as procedural-oriented programming. In Python, we don’t need to declare the type of variable because it is a dynamically typed language. For example, x = 10 Here, x can be anything such as String, int, etc.

1. Features in Python:

There are many features in Python, some of which are discussed below as follows:

1. Free and Open Source

Python language is freely available at the official website and you can download it from the given download link below click on the Download Python keyword. Download Pyhton Since it is open-source, this means that source code is also available to the public. So you can download it, use it as well as share it.

2. Easy to code

Python is a high-level programming language Python is very easy to learn the language as compared to other languages like C, C#, Javascript, Java, etc. It is very easy to code in the Python language and anybody can learn Python basics in a few hours or days. It is also a developer-friendly language.

3. Easy to Read

As you will see, learning Python is quite simple. As was already established, Python's syntax is really straightforward. The code block is defined by the indentations rather than by semicolons or brackets.

4. Object-Oriented Language

One of the key features of Python is object-Oriented Programming. Python supports object-oriented language and concepts of classes, object encapsulation, etc.

5. GUI Programming Support

Graphical User interfaces can be made using a module such as PyQt5,PyQt4, wxPython, or Tk in python. PyQt5 is the most popular option for creating graphical apps with Python.

6. High-Level Language

Python is a high-level language. When we write programs in Python, we do not need to remember the system architecture, nor do we need to manage the memory.

7. Extensible feature

Python is an Extensible language. We can write some Python code into C or C++ language and also we can compile that code in C/C++ language.

8. Easy to Debug

Excellent information for mistake tracing. You will be able to quickly identify and correct the majority of your program's issues once you understand how to interpretPython's error traces. Simply by glancing at the code, you can determine what it is designed to perform.

9. Python is a Portable language

Python language is also a portable language. For example, if we have Python code for windows and if we want to run this code on other platforms such as Linux Unix, and Mac then we do not need to change it, we can run this code on any platform.

10. Python is an Integrated language

Python is also an Integrated language because we can easily integrate Python with other languages like C, C++ etc.

11. Interpreted Language:

Python is an Interpreted Language because Python code is executed line by line at a time. like other languages C, C++, Java etc. there is no need to compile Python code this makes it easier to debug our code. The source code of Python is converted into an immediate form called bytecode.

12. Large Standard Library

Python has a large standard libraryt hat provides a rich set of modules and functions so you do not have to write your own code for every single thing. There are many libraries present in Python such as regular expressions, unit-testing, web browsers, etc.

13. Dynamically Typed Language

Python is a dynamically-typed language. That means the type (for example- int, double, long, etc.) for a variable is decided at run time not in advance because of this feature we don't need to specify the type of variable.

14. Frontend and backend development

With a new project py script, you can run and write Python codes in HTML with the help of some simple tags <py-script>, <py-env>, etc. This will help you do frontend development work in Python like javascript. Backend is the strong forte of Python it's extensively used for this work cause of its frameworks like Django and Flask.

15. Allocating Memory Dynamically

In Python, the variable data type does not need to be specified. The memory is automatically allocated to a variable at runtime when it is given a value. Developers do not need to write int y = 18 if the integer value 15 is set to y. You may just type y=18.

6.1.2 LIBRARIES/PACKAGES :-

Tensorflow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

Numpy

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the t humbnail gallery. For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

CHAPTER 7
SYSTEM REQUIREMENTS SPECIFICATIONS

7.1 SOFTWARE REQUIREMENTS

Software requirements deal with defining software resource requirements and prerequisites that need to be installed on a computer to provide optimal functioning of an application. These requirements or prerequisites are generally not included in the software installation package and need to be installed separately before the software is installed.

Platform – In computing, a platform describes some sort of framework, either in hardware or software, which allows software to run. Typical platforms include a computer's architecture, operating system, or programming languages and their runtime libraries.

Operating system is one of the first requirements mentioned when defining system requirements (software). Software may not be compatible with different versions of same line of operating systems, although some measure of backward compatibility is often maintained. For example, most software designed for Microsoft Windows XP does not run on Microsoft Windows 98, although the converse is not always true. Similarly, software designed using newer features of Linux Kernel v2.6 generally does not run or compile properly (or at all) on Linux distributions using Kernel v2.2 or v2.4.

APIs and drivers – Software making extensive use of special hardware devices, like high-end display adapters, needs special API or newer device drivers. A good example is DirectX, which is a collection of APIs for handling tasks related to multimedia, especially game programming, on Microsoft platforms.

Web browser – Most web applications and software depending heavily on Internet technologies make use of the default browser installed on system. Microsoft Internet Explorer is a frequent choice of software running on Microsoft Windows, which makes use of ActiveX controls, despite their vulnerabilities.

1)Visual Studio Community Version

2)Nodejs (Version 12.3.1)

3)Python IDEL (Python 3.7)

7.2 HARDWARE REQUIREMENTS

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware, A hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatible, and sometimes incompatible hardware devices for a particular operating system or application. The following sub-sections discuss the various aspects of hardware requirements.

Architecture – All computer operating systems are designed for a particular computer architecture. Most software applications are limited to particular operating systems running on particular architectures. Although architecture-independent operating systems and applications exist, most need to be recompiled to run on a new architecture. See also a list of common operating systems and their supporting architectures.

Processing power – The power of the central processing unit (CPU) is a fundamental system requirement for any software. Most software running on x86 architecture define processing power as the model and the clock speed of the CPU. Many other features of a CPU that influence its speed and power, like bus speed, cache, and MIPS are often ignored. This definition of power is often erroneous, as AMD Athlon and Intel Pentium CPUs at similar clock speed often have different throughput speeds. Intel Pentium CPUs have enjoyed a considerable degree of popularity, and are often mentioned in this category.

Memory – All software, when run, resides in the random access memory (RAM) of a computer. Memory requirements are defined after considering demands of the application, operating system, supporting software and files, and other running processes. Optimal performance of other unrelated software running on a multi-tasking computer system is also considered when defining this requirement.

Secondary storage – Hard-disk requirements vary, depending on the size of software installation, temporary files created and maintained while installing or running the software, and possible use of swap space (if RAM is insufficient).

Display adapter – Software requiring a better than average computer graphics display, like graphics editors and high-end games, often define high-end display adapters in the system requirements.

Peripherals – Some software applications need to make extensive and/or special use of some peripherals, demanding the higher performance or functionality of such peripherals. Such peripherals include CD-ROM drives, keyboards, pointing devices, network devices, etc.

1)Operating System : Windows Only

2)Processor : i5 and above

3)Ram : 4gb and above

4)Hard Disk : 50 GB

7.3 FUNCTIONAL REQUIREMENTS

Functional requirements define the essential tasks that the system must perform. These requirements describe the inputs, processing steps, and expected outputs of the system.

1. Data Collection

- The system must gather images of food items from multiple sources, including:
  - User-uploaded photos
  - Open-source datasets (e.g., Food-101, UEC-Food100)
  - Mobile camera inputs
- Ensure metadata collection, such as image timestamps, geolocation (if enabled), and food category.
- The system should support batch image uploads and real-time food scanning.

2. Data Preprocessing

- Image data should undergo cleaning and normalization for consistent input.
- Steps involved in preprocessing:
    - Resizing: Standardizing image dimensions for uniform model input.
    - Noise Reduction: Applying filters to remove distortions.
    - Contrast Adjustment: Enhancing food textures.
    - Data Augmentation: Rotation, flipping, and cropping for robust training.
- Convert images into numerical tensor representations suitable for CNN input.

3. Training and Testing

- Implement Convolutional Neural Networks (CNNs) to extract features from food images.
- Use Supervised Learning to train the model on labeled food datasets.
- Split data into training (80%) and testing (20%) subsets.
- Evaluate model accuracy using metrics such as:
    - Precision, Recall, F1-score
    - Confusion Matrix
    - Loss Function Optimization (Cross-Entropy, Mean Squared Error)
- Train the model iteratively, adjusting hyperparameters like learning rate, dropout, and batch size.

4. Modelling

- Fine-tune deep learning architectures such as Res Net, Mobile Net, or Efficient Net.
- Implement Transfer Learning to leverage pre-trained models on food classification.
- Use multi-class classification techniques to identify multiple food items in a single image.

- Store trained models in optimized formats (ONNX, TensorFlow SavedModel) for real-time deployment.

5. Predicting

- The model must accurately predict food items in a given image.
- Perform calorie estimation by integrating food database values.
- Display confidence scores for each prediction.
- Enable real-time feedback and allow user corrections to improve accuracy.
- Implement API-based predictions for mobile and web applications.

7.4 NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements (NFRs) define the system's quality attributes, performance, and constraints. They ensure system reliability, efficiency, and security.

1. Usability Requirement

- The system should have a user-friendly interface for seamless interaction.
- Provide multilingual support for diverse users.
- Ensure accessibility compliance (WCAG 2.1 standards for visually impaired users).

2. Serviceability Requirement

- The system should support automated model updates to improve performance.
- Implement error handling mechanisms to provide meaningful feedback.

3. Manageability Requirement

- The system should allow remote monitoring and debugging.
- Provide an admin dashboard for tracking model performance and usage statistics.

4. Recoverability Requirement

- Implement automatic backups for model training and prediction logs.
- Ensure disaster recovery mechanisms for data loss prevention.

5. Security Requirement

- Use end-to-end encryption for protecting sensitive user data.
- Implement role-based access control (RBAC) for secure data access.
- Ensure GDPR and HIPAA compliance for data privacy.

6. Data Integrity Requirement

- Prevent unauthorized modifications of training datasets.
- Use checksum validation for dataset consistency.

7. Capacity Requirement

- The system should handle large-scale datasets (millions of images).
- Optimize storage using cloud-based solutions (AWS S3, Google Cloud Storage).

8. Availability Requirement

- Ensure 99.9% uptime with failover mechanisms.
- Implement load balancing to distribute traffic efficiently.

9. Scalability Requirement

- Support horizontal scaling by adding new servers as user demand increases.
- Use serverless computing for efficient resource utilization.

10. Interoperability Requirement

- Ensure compatibility with multiple platforms (web, mobile, IoT devices).
- Provide REST API and Graph QL support for integration with third-party applications.

11. Reliability Requirement

- The model should provide consistent accuracy (above 85%) across different food categories.
- Implement fault tolerance mechanisms to prevent system crashes.

12. Maintainability Requirement

- Support version control for model updates.
- Provide automated logging and debugging tools for quick issue resolution.

13. Regulatory Requirement

Ensure compliance with ISO/IEC 27001 for data security.

- Follow FDA guidelines for food calorie estimation tools.

14. Environmental Requirement

- Optimize computational resources to reduce energy consumption.
- Use green AI practices by implementing energy-efficient model training

7.5 Feasibility Study

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

1. ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

2. TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

7.5.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

CHAPTER-8

SYSTEM TESTING

8. SYSTEM TESTING

System testing, also referred to as system-level tests or system-integration testing, is the process in which a quality assurance (QA) team evaluates how the various components of an application interact together in the full, integrated system or application. System testing verifies that an application performs tasks as designed. This step, a kind of black box testing, focuses on the functionality of an application. System testing, for example, might check that every kind of user input produces the intended output across the application.

Phases of system testing:

A video tutorial about this test level. System testing examines every component of an application to make sure that they work as a complete and unified whole. A QA team typically conducts system testing after it checks individual modules with functional or user-story testing and then each component through integration testing.

If a software build achieves the desired results in system testing, it gets a final check via acceptance testing before it goes to production, where users consume the software. An app-dev team logs all defects, and establishes what kinds and amount of defects are tolerable.

1. **Software Testing Strategies:**

Optimization of the approach to testing in software engineering is the best way to make it effective. A software testing strategy defines what, when, and how to do whatever is necessary to make an end-product of high quality. Usually, the following software testing strategies and their combinations are used to achieve this major objective:

Static Testing:

The early-stage testing strategy is static testing: it is performed without actually running the developing product. Basically, such desk-checking is required to detect bugs and issues that are present in the code itself. Such a check-up is important at the pre-deployment stage as it helps avoid problems caused by errors in the code and software structure deficits.

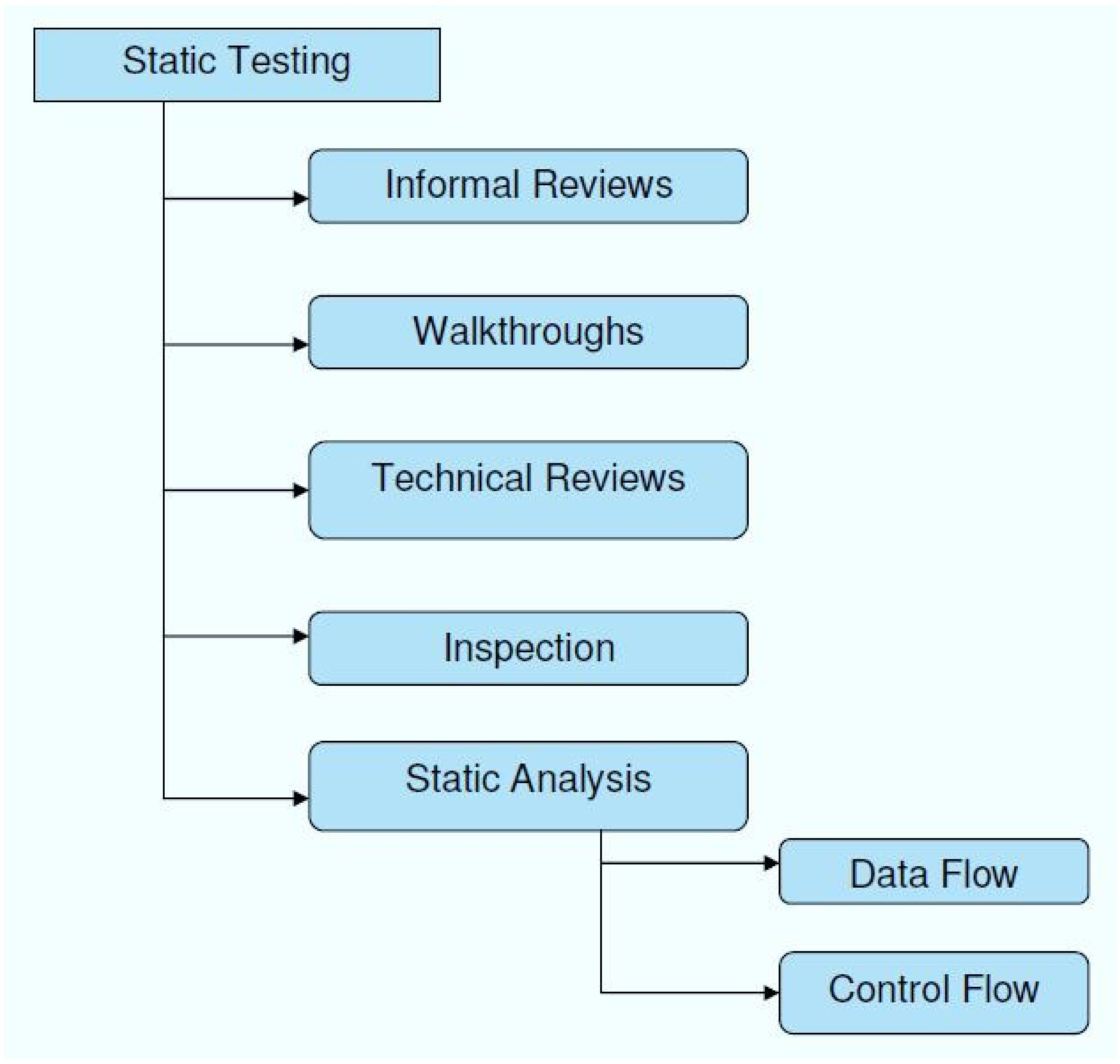


Fig 8.1 Static Testing

Structural Testing:

It is not possible to effectively test software without running it. Structural testing, also known as white-box testing, is required to detect and fix bugs and errors emerging during the pre-production stage of the software development process. At this stage, unit testing based on the software structure is performed using regression testing. In most cases, it is an automated process working within the test automation framework to speed up the development process at this stage. Developers and QA engineers have full access to the software's structure and data flows (data flows testing), so they could track any changes (mutation testing) in the system's behavior by comparing the tests' outcomes with the results of previous iterations (control flow testing).

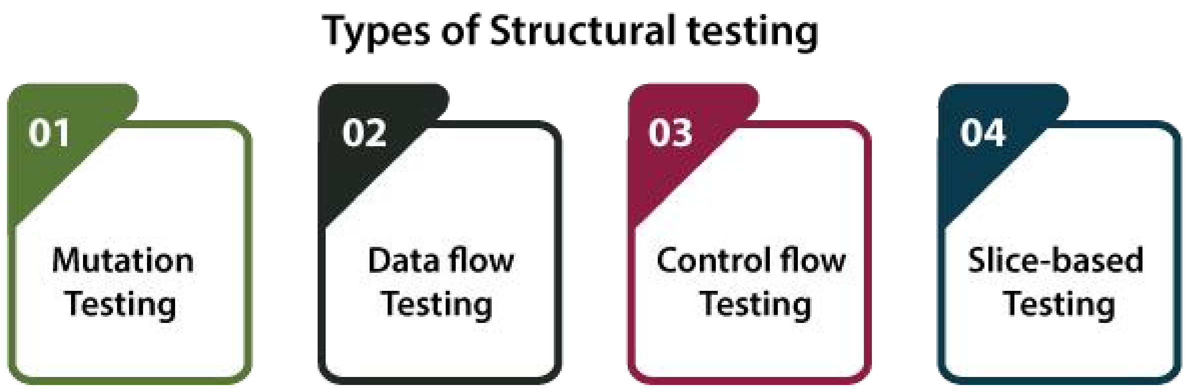


Fig 8.1 Structural Testing

Behavioral Testing:

The final stage of testing focuses on the software's reactions to various activities rather than on the mechanisms behind these reactions. In other words, behavioral testing, also known as black-box testing, presupposes running numerous tests, mostly manual, to see the product from the user's point of view. QA engineers usually have some specific information about a business or other purposes of the software ('the black box') to run usability tests, for example, and react to bugs as regular users of the product will do. Behavioral testing also may include automation (regression tests) to eliminate human error if repetitive activities are required. For example, you may need to fill 100 registration forms on the website to see how the product copes with such an activity, so the automation of this test is preferable.

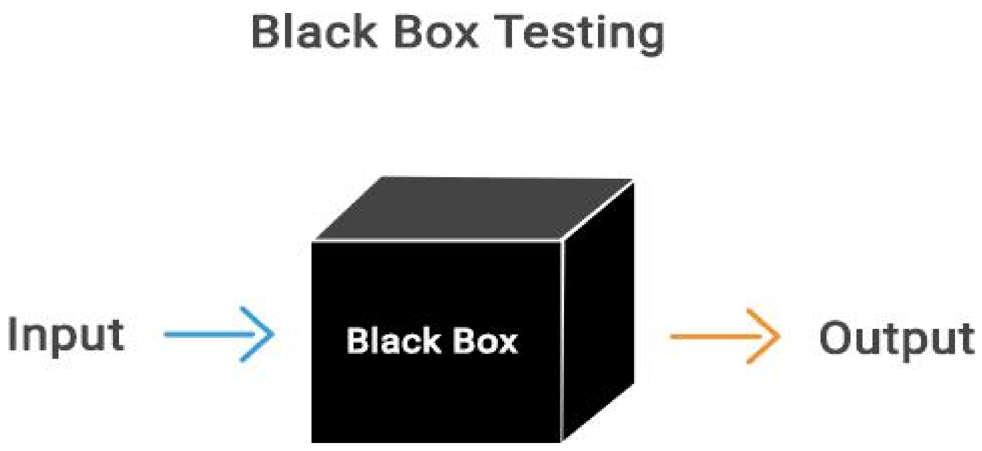


Fig 8.1 Black Box Testing

8.2 TEST CASES:

S.NO	INPUT	If available	If not available
1	User signup	User get registered into the application	There is no process
2	User signin	User get login into the application	There is no process
3	Enter input for prediction	Prediction result displayed	There is no process

CHAPTER-9

METHODOLOGY

1. MODULES:

- **Data exploration**: using this module we will load data into system
- **Processing**: Using the module we will read data for processing
- **Splitting data into train & test**: using this module data will be divided into train & test
- **Model generation**: Model Building- CNN, SVM, KNN, Random Forest, MobileNetv2, Inceptionv3 and Dense Net. Algorithms accuracy calculated
- **User signup & login**: Using this module will get registration and login
- **User input**: Using this module will give input for prediction
- **Prediction**: final predicted displayed

2. ALGORITHMS:

CNN: A CNN is a kind of network architecture for deep learning algorithms and is specifically used for image recognition and tasks that involve the processing of pixel data. There are other types of neural networks in deep learning, but for identifying and recognizing objects, CNNs are the network architecture of choice.

SVM: Support Vector Machine(SVM) is a supervised machine learning algorithm used for both classification and regression. Though we say regression problems as well its best suited for classification. The objective of SVM algorithm is to find a hyperplane in an N-dimensional space that distinctly classifies the data points

KNN: The k-nearest neighbors algorithm, also known as KNN or k-NN, is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point.

Random Forest: A Random Forest Algorithm is a supervised machine learning algorithm that is extremely popular and is used for Classification and Regression problems in Machine Learning. We know that a forest comprises numerous trees, and the more trees more it will be robust.

MobileNetv2: MobileNetV2 is a classification model developed by Google. It provides real-time classification capabilities under computing constraints in devices like smartphones. This implementation leverages transfer learning from ImageNet to your dataset.

Inceptionv3: The Inception V3 is a deep learning model based on Convolutional Neural Networks, which is used for image classification. The inception V3 is a superior version of the basic model Inception V1 which was introduced as GoogLe Net in 2014. As the name suggests it was developed by a team at Google.

Dense Net: Dense Net was developed specifically to improve the declined accuracy caused by the vanishing gradient in high-level neural networks. In simpler terms, due to the longer path between the input layer and the output layer, the information vanishes before reaching its destination.

CHAPTER 10

SOURCE CODE

```
import cv2
    import numpy as np
    import os

    def getAreaOfFood(img1):
        data=os.path.join(os.getcwd(),"images")
        if os.path.exists(data):
            print('folder exist for images at ',data)
        else:
            os.mkdir(data)
            print('folder created for images at ',data)

        cv2.imwrite('{}\\1 original image.jpg'.format(data),img1)
        img = cv2.cvtColor(img1, cv2.COLOR_BGR2GRAY)
        cv2.imwrite('{}\\2 original image BGR2GRAY.jpg'.format(data),img)
        img_filt = cv2.medianBlur( img, 5)
        cv2.imwrite('{}\\3 img_filt.jpg'.format(data),img_filt)
        img_th =
cv2.adaptiveThreshold(img_filt,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRESH_B
INARY,21,2)
        cv2.imwrite('{}\\4 img_th.jpg'.format(data),img_th)  contours,
        hierarchy = cv2.findContours(img_th, cv2.RETR_LIST,
cv2.CHAIN_APPROX_SIMPLE) #make change here


        # find contours. sort. and find the biggest contour. the biggest
contour corresponds to the plate and fruit.
        mask = np.zeros(img.shape, np.uint8)
        largest_areas = sorted(contours, key=cv2.contourArea)
        cv2.drawContours(mask, [largest_areas[-1]], 0, (255,255,255,255), -1)
        cv2.imwrite('{}\\5 mask.jpg'.format(data),mask)
        img_bigcontour = cv2.bitwise_and(img1,img1,mask = mask)
        cv2.imwrite('{}\\6 img_bigcontour.jpg'.format(data),img_bigcontour)

        # convert to hsv. otsu threshold in s to remove plate
        hsv_img = cv2.cvtColor(img_bigcontour, cv2.COLOR_BGR2HSV)
        cv2.imwrite('{}\\7 hsv_img.jpg'.format(data),hsv_img)
        h,s,v = cv2.split(hsv_img)
        mask_plate = cv2.inRange(hsv_img, np.array([0,0,50]),
np.array([200,90,250]))
        cv2.imwrite('{}\\8 mask_plate.jpg'.format(data),mask_plate)
        mask_not_plate = cv2.bitwise_not(mask_plate)
        cv2.imwrite('{}\\9 mask_not_plate.jpg'.format(data),mask_not_plate)
        fruit_skin = cv2.bitwise_and(img_bigcontour,img_bigcontour,mask =
mask_not_plate)
        cv2.imwrite('{}\\10 fruit_skin.jpg'.format(data),fruit_skin)

        #convert to hsv to detect and remove skin pixels
        hsv_img = cv2.cvtColor(fruit_skin, cv2.COLOR_BGR2HSV)
        cv2.imwrite('{}\\11 hsv_img.jpg'.format(data),hsv_img)
        skin = cv2.inRange(hsv_img, np.array([0,10,60]),
np.array([10,160,255])) #Scalar(0, 10, 60), Scalar(20, 150, 255)
        cv2.imwrite('{}\\12 skin.jpg'.format(data),skin)
        not_skin = cv2.bitwise_not(skin); #invert skin and black
        cv2.imwrite('{}\\13 not_skin.jpg'.format(data),not_skin)
        fruit = cv2.bitwise_and(fruit_skin,fruit_skin,mask = not_skin) #get
only fruit pixels
        cv2.imwrite('{}\\14 fruit.jpg'.format(data),fruit)

        fruit_bw = cv2.cvtColor(fruit, cv2.COLOR_BGR2GRAY)
        cv2.imwrite('{}\\15 fruit_bw.jpg'.format(data),fruit_bw)
        fruit_bin = cv2.inRange(fruit_bw, 10, 255) #binary of fruit
        cv2.imwrite('{}\\16 fruit_bw.jpg'.format(data),fruit_bin)

        #erode before finding contours
        kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(5,5))
        erode_fruit = cv2.erode(fruit_bin,kernel,iterations = 1)
        cv2.imwrite('{}\\17 erode_fruit.jpg'.format(data),erode_fruit)

        #find largest contour since that will be the fruit
        img_th =
cv2.adaptiveThreshold(erode_fruit,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRES
H_BINARY,11,2)
        cv2.imwrite('{}\\18 img_th.jpg'.format(data),img_th) contours,
        hierarchy = cv2.findContours(img_th, cv2.RETR_LIST,
cv2.CHAIN_APPROX_SIMPLE)
        mask_fruit = np.zeros(fruit_bin.shape, np.uint8)
        largest_areas = sorted(contours, key=cv2.contourArea)
        cv2.drawContours(mask_fruit, [largest_areas[-2]], 0, (255,255,255), -
1)
        cv2.imwrite('{}\\19 mask_fruit.jpg'.format(data),mask_fruit)

        #dilate now
        kernel2 = cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(5,5))
        mask_fruit2 = cv2.dilate(mask_fruit,kernel2,iterations = 1)
        cv2.imwrite('{}\\20 mask_fruit2.jpg'.format(data),mask_fruit2)
        fruit_final = cv2.bitwise_and(img1,img1,mask = mask_fruit2)
        cv2.imwrite('{}\\21 fruit_final.jpg'.format(data),fruit_final)

        #find area of fruit
        img_th =
cv2.adaptiveThreshold(mask_fruit2,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRES
H_BINARY,11,2)
        cv2.imwrite('{}\\22 img_th.jpg'.format(data),img_th) contours,
        hierarchy = cv2.findContours(img_th, cv2.RETR_LIST,
cv2.CHAIN_APPROX_SIMPLE)
        largest_areas = sorted(contours, key=cv2.contourArea)
        fruit_contour = largest_areas[-2]
        fruit_area = cv2.contourArea(fruit_contour)


        #finding the area of skin. find area of biggest contour
        skin2 = skin - mask_fruit2
        cv2.imwrite('{}\\23 skin2.jpg'.format(data),skin2)
        #erode before finding contours
        kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(5,5))
        skin_e = cv2.erode(skin2,kernel,iterations = 1)
        cv2.imwrite('{}\\24 skin_e .jpg'.format(data),skin_e )
        img_th =
cv2.adaptiveThreshold(skin_e,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRESH_BIN
ARY,11,2)
        cv2.imwrite('{}\\25 img_th.jpg'.format(data),img_th) contours,
        hierarchy = cv2.findContours(img_th, cv2.RETR_LIST,
cv2.CHAIN_APPROX_SIMPLE)
        mask_skin = np.zeros(skin.shape, np.uint8)
        largest_areas = sorted(contours, key=cv2.contourArea)
        cv2.drawContours(mask_skin, [largest_areas[-2]], 0, (255,255,255), -1)
        cv2.imwrite('{}\\26 mask_skin.jpg'.format(data),mask_skin)


        skin_rect = cv2.minAreaRect(largest_areas[-2])
        box = cv2.boxPoints(skin_rect)
        box = np.int0(box)
        mask_skin2 = np.zeros(skin.shape, np.uint8)
        cv2.drawContours(mask_skin2,[box],0,(255,255,255),  -1)
        cv2.imwrite('{}\\27 mask_skin2.jpg'.format(data),mask_skin2)

        pix_height = max(skin_rect[1])
        pix_to_cm_multiplier = 5.0/pix_height
        skin_area = cv2.contourArea(box)


        return fruit_area,fruit_bin ,fruit_final,skin_area, fruit_contour,
pix_to_cm_multiplier


    if __name__ == '__main__':
        img1 = cv2.imread(r"C:\Users\piya\Desktop\model2\Orange\2.jpg")
        img = cv2.resize(img1,(1000,1000))
        area, bin_fruit, img_fruit, skin_area, fruit_contour,
pix_to_cm_multiplier = getAreaOfFood(img)
        cv2.imshow('img',img_fruit)
        cv2.waitKey(0)
        cv2.destroyAllWindows()
```

CHAPTER-11

RESULTS AND DISCUSSION

11.1 SCREENSHOTS

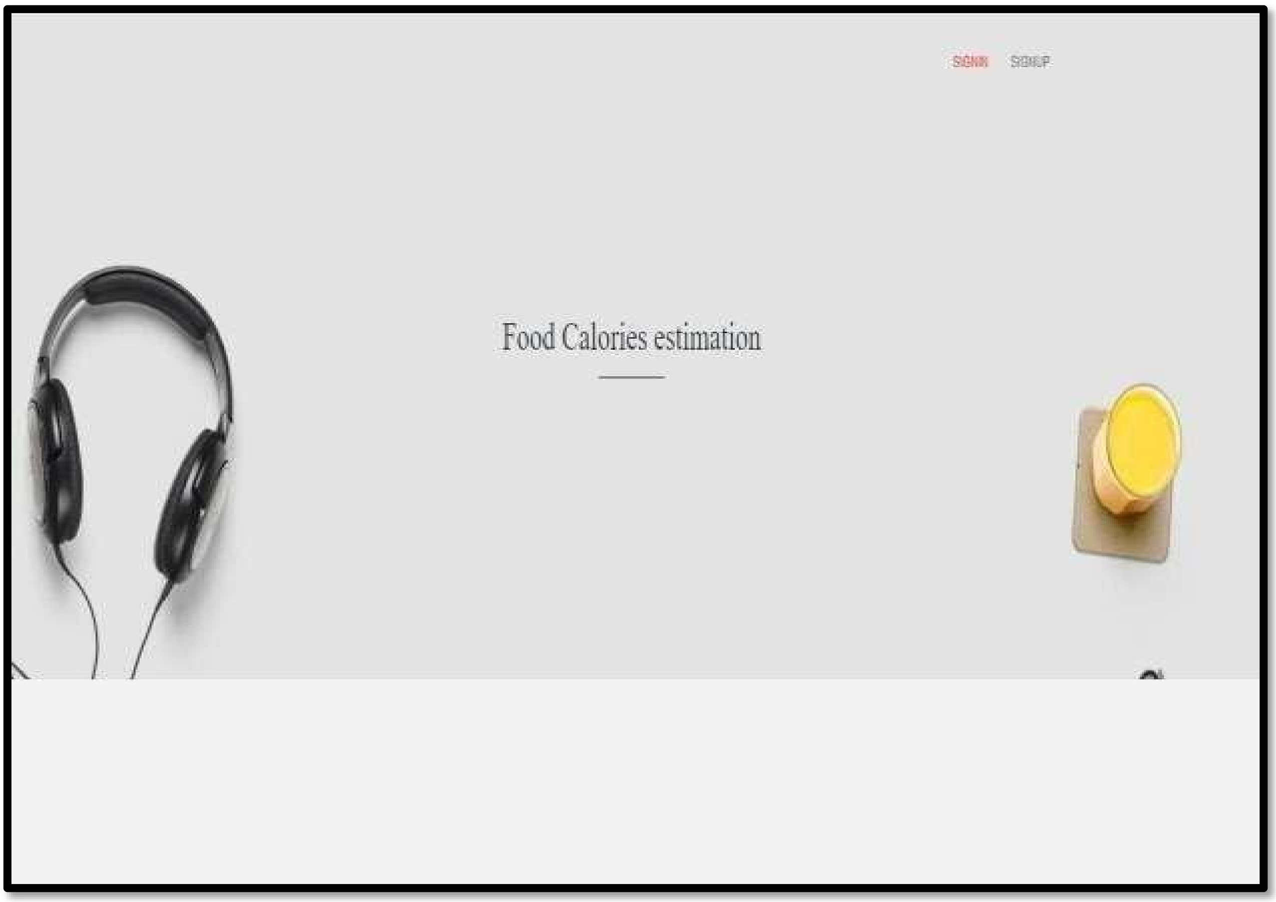


Fig 11.1.1 Cover Page

The image displays a web interface for a **Food Calories Estimation** system. The design has a clean and minimalistic aesthetic, with a light gray background and centered text that reads *"Food Calories Estimation"*. In the top right corner, there are **"SIGNIN"** and **"SIGNUP"** options, allowing users to log in or register. The **"SIGNIN"** text is highlighted in red,indicating the active or selected state.

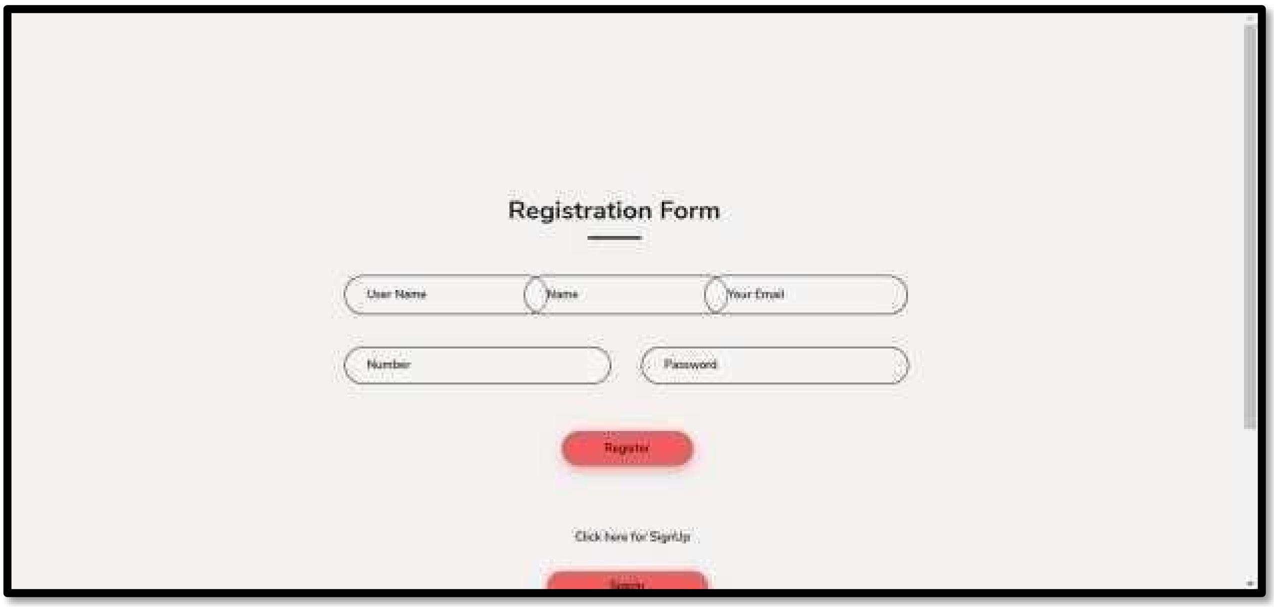


Fig 11.1.2 Registration Form

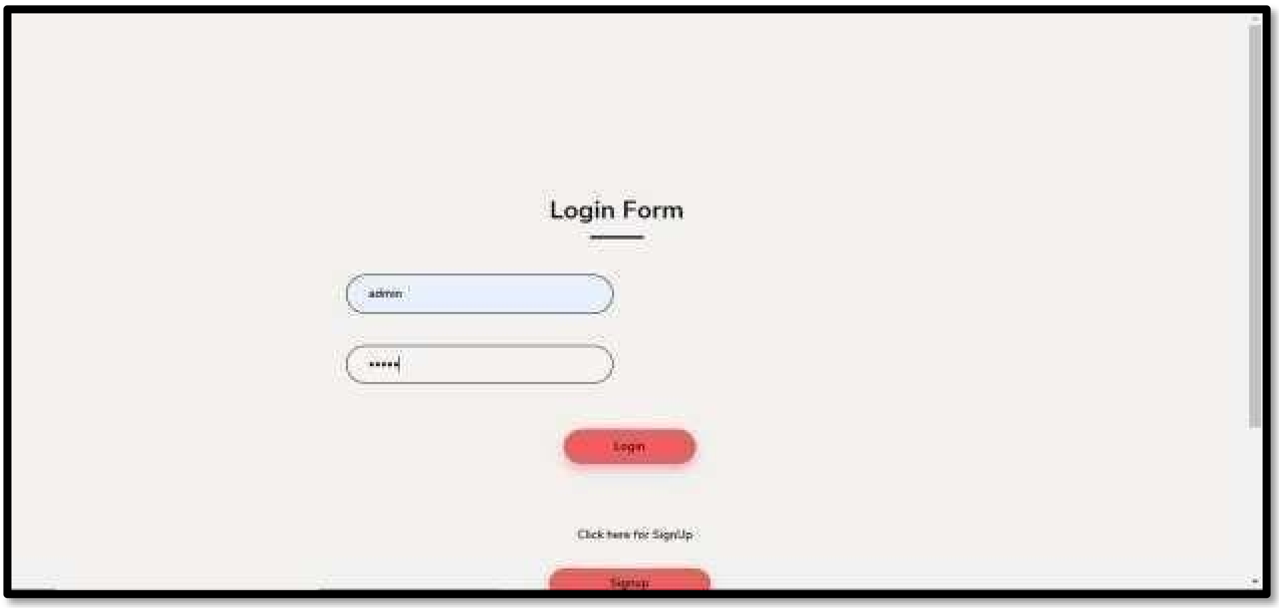
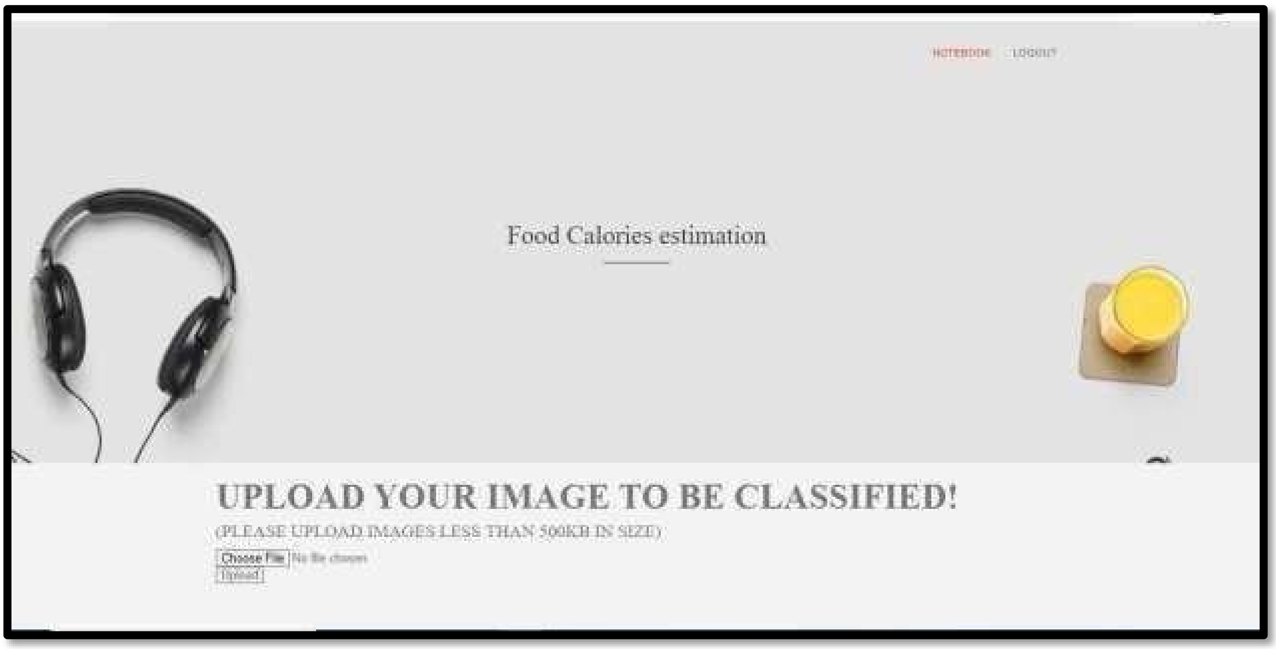


Fig 11.1.3 Login Form

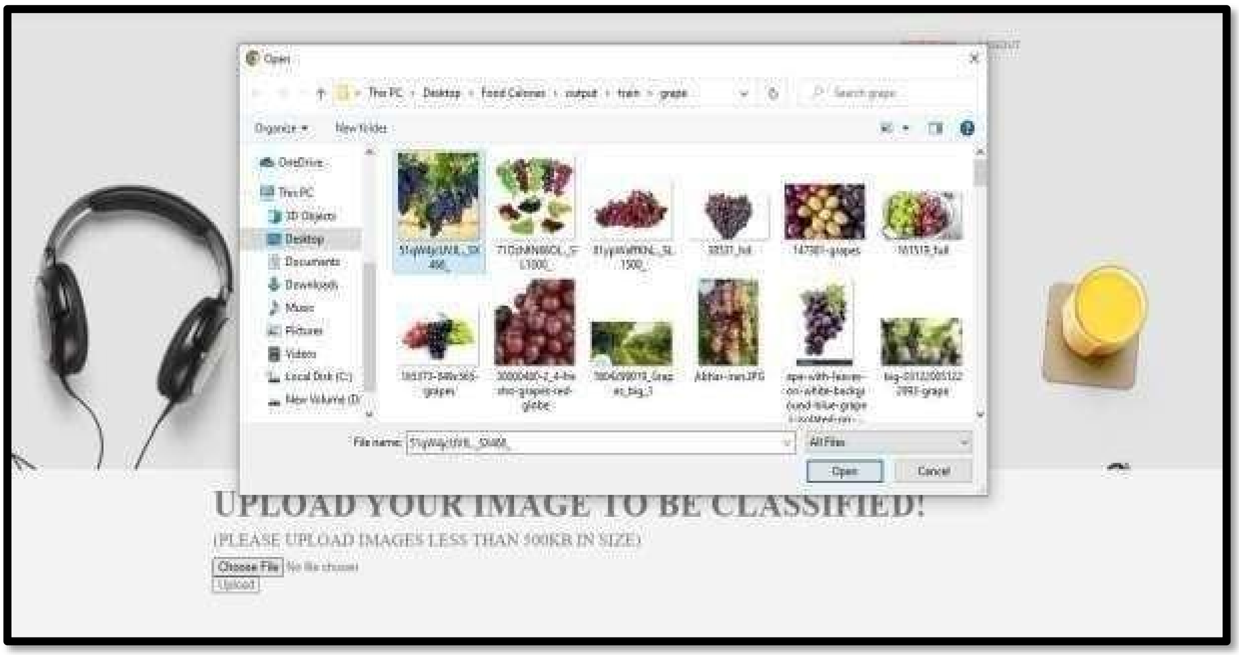


Food Calories estimation

UPLOAD YOUR IMAGE TO BE CLASSIFIED!

(PLEASE UPLOAD IMAGES LESS THAN 500KB IN SIZE)

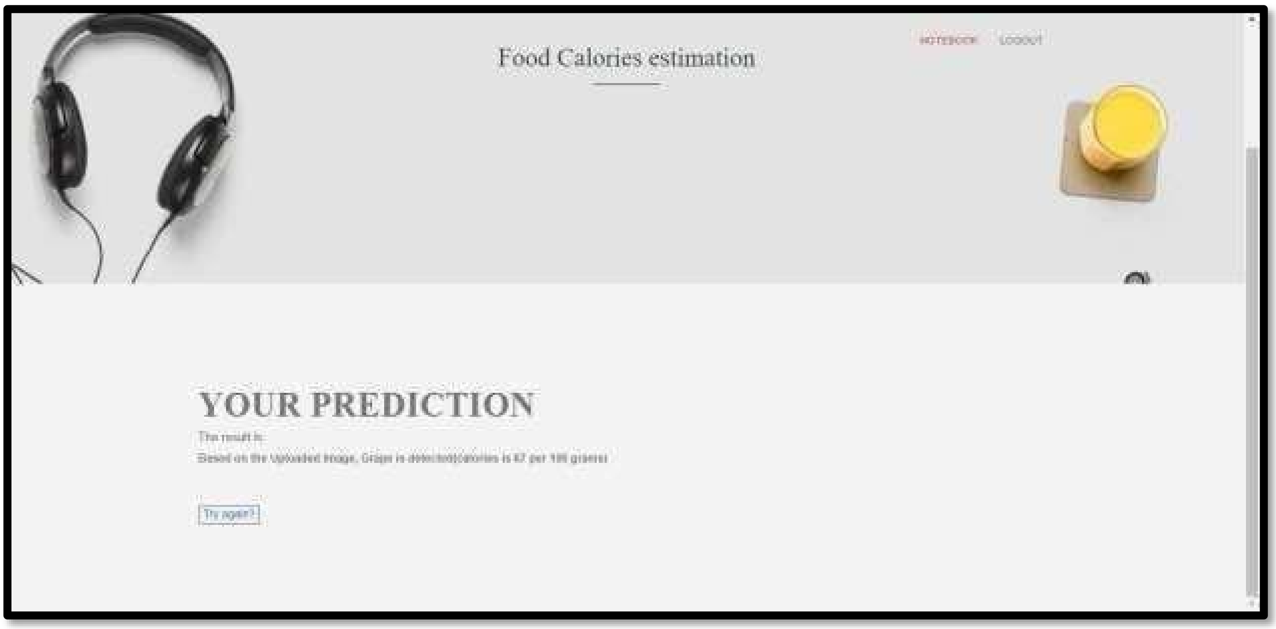
Fig 11.1.4 Upload Image For Classification



UPLOAD YOUR IMAGE TO BE CLASSIFIED!

(PLEASE UPLOAD IMAGES LESS THAN 500KB IN SIZE)

Fig 11.1.5 Upload Image

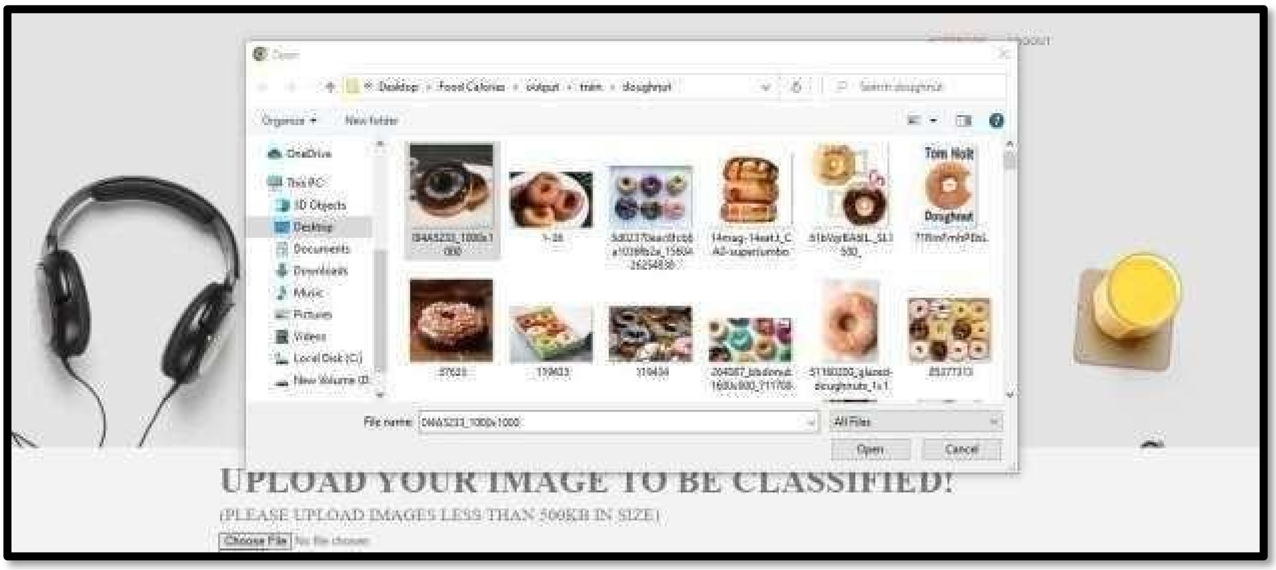


YOUR PREDICTION

The result is:

Based on the Uploaded Image, Grape is detected(calories is 67 per 100 grams)

Fig 11.1.6 Prediction and Outcome



UPLOAD YOUR IMAGE TO BE CLASSIFIED!

(PLEASE UPLOAD IMAGES LESS THAN 500KB IN SIZE)

CHAPTER 12

CONCLUSION AND FUTURE SCOPE

12.1 CONCLUSION

This proposed method is to create food recognition and detection while using several algorithms. Those algorithms are CNN, Random forest, SVM to get better accuracy and we obtained it. We have used a food image dataset which is publicly available. CNN was used for the image recognition. Also we trained the models using information from dataset. Also accuracy has further improved through optimization, hyper parameter tuning. We have written a function which determines calories based on the fruit detected by taking in consideration the average calorie value of that fruit. We will finish up our work contrasting our outcomes and the benchmark work regarding volume assessment since the outcomes for calorie assessments are not partaken in the paper. In the event that we utilize 30% for testing Random Forest model is the most ideal model for our concern with the mean mistake of 13.12 though KNN has a mean blunder of 21.06. We can see the correlation between benchmark results and both of our models. As we can see both of our models beats the benchmark work. Since volumes determined with math equations in our standard work, some natural product that are near amazing shape like lemon has higher precision than our models, anyway this is simply restricted to food sources with ellipsoid shapes. At the point when we look at our volume assessment techniques, we can find that irregular woods model is marginally outflanking the KNN model. The motivation behind why the K Nearest Neighbors technique is however great as the Random Forest strategy seems to be that the informational index and the quantity of highlights that we use is little and in this manner the model doesn't experience the ill effects of revile of dimensionality. Since our informational collection is moderately little we can grow our informational index which will diminish the mistake considerably more as a future work. Additionally eliminating the imbalanced conveyance in our informational index will diminish the blunder too.

12.2 FUTURE SCOPE

1. **Dataset Expansion**

- Incorporate more diverse food categories, regional cuisines, and different presentation styles.
- Increase dataset size to improve model generalization and accuracy.

2. **Data Augmentation**

- Apply techniques like rotation, flipping, contrast adjustments, and noise addition.
- Improve recognition under various lighting and angles.

3. **Advanced Deep Learning Models**

- Implement Transformer-based vision models and architectures like EfficientNet and ResNet.
- Improve feature extraction and classification performance.

4. **Multi-Modal Learning**

- Combine image recognition with textual descriptions and nutritional databases.
- Enhance calorie estimation accuracy with cross-modal information.

5. **Computational Optimization**

- Improve model efficiency for real-time recognition.
- Utilize edge computing to reduce dependency on cloud-based processing.

6. **Commercial Applications**

- Deploy in restaurant menu scanning and AI-driven meal planning.
- Assist in dietary monitoring for healthcare and fitness sectors.

7. **Error Reduction & Model Refinement**

- Address dataset imbalance to minimize classification errors.
- Increase dataset diversity and tuning to enhance model robustness.

CHAPTER- 13

REFERENCES

[1] Winter-Jensen, M., Afzal, S., Jess, T., Nordestgaard, B. G., & Allin, K. H. Body mass index and risk of infections: a Mendelian randomization study of 101,447 individuals. European journal of epidemiology, 35(4), 347-354. (2020)

[2] Bai, C., Liu, T., Xu, J., Ma, X., Huang, L., Liu, S., & Gu, X. Effect of high calorie diet on intestinal flora in LPS-induced pneumonia rats. Scientific reports, 10(1), 1-12. (2020)

[3] Sree, S. R., Vyshnavi, S. B., & Jayapandian, N. Real-world application of machine learning and deep learning. In 2019 International Conference on Smart Systems and Inventive Technology (ICSSIT) (pp. 1069-1073). IEEE. (2019)

[4] Natarajan, J. Cyber Secure Man-in-the-Middle Attack Intrusion Detection Using Machine Learning Algorithms. In AI and Big Data's Potential for Disruptive Innovation (pp. 291-316). IGI Global. (2020)

[5] Morquecho-Campos, P., de Graaf, K., & Boesveldt, S. Smelling our appetite? The influence of food odors on congruent appetite, food preferences and intake. Food Quality and Preference, 85, 103959.(2020)

[6] Pal, S. K., Pramanik, A., Maiti, J., & Mitra, P. (2021). Deep learning in multi-object detection and tracking: state of the art. Applied Intelligence, 1-30. (2021)

[7] Fahira, P. K., Rahmadhani, Z. P., Mursanto, P., Wibisono, A., & Wisesa, H. A. Classical Machine Learning Classification for Javanese Traditional Food Image. In 2020 4th International Conference on Informatics and Computational Sciences (ICICoS) (pp. 1-5). IEEE. (2020)

[8] Grattarola, D., & Alippi, C. Graph Neural Networks in TensorFlow and Keras with Spektral [Application Notes]. IEEE Computational Intelligence Magazine, 16(1), 99-106. (2021)

[9] Talukdar, J., Gupta, S., Rajpura, P. S., & Hegde, R. S. Transfer learning for object detection using state-of-the-art deep neural networks. In 2018 5th International Conference on Signal Processing and Integrated Networks (SPIN) (pp. 78-83). IEEE. (2018)

[10] Bakke, A. J., Carney, E. M., Higgins, M. J., Moding, K., Johnson, S. L., & Hayes, J. E. Blending dark green vegetables with fruits in commercially available infant foods makes them taste like fruit. Appetite, 150, 104652.(2020)

[11] Zheng, L., Lawlor, B., Katko, B. J., McGuire, C., Zanteson, J., & Eliasson, V. Image processing and edge detection techniques to quantify shock wave dynamics experiments. Experimental Techniques, 1-13. (2020)

[12] Asante-Okyere, S., Shen, C., Ziggah, Y. Y., Rulegeya, M. M., & Zhu, X. Principal component analysis (PCA) based hybrid models for the accurate estimation of reservoir water saturation. Computers & Geosciences, 145, 104555. (2020)

[13] Huynh-The, T., Hua, C. H., & Kim, D. S. Encoding pose features to images with data augmentation for 3-D action recognition. IEEE Transactions on Industrial Informatics, 16(5), 3100-3111. (2019)

[14] Vo, H. V., Pérez, P., & Ponce, J. Toward unsupervised, multi-object discovery in large-scale image collections. In European Conference on Computer Vision (pp. 779-795). Springer, Cham. (2019)

[15] Grinvald, M., Furrer, F., Novkovic, T., Chung, J. J., Cadena, C., Siegwart, R., & Nieto, J. Volumetric instance-aware semantic mapping and 3D object discovery. IEEE Robotics and Automation Letters, 4(3), 3037-3044. (2019)