

ASSESSING CALORIE INTAKE IN PATIENT TO AVOID UNWANTED HEALTH ISSUES

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Technology degree in Biomedical Engineering.

By

MANOBALA P (37240050)



**DEPARTMENT OF BIOMEDICAL ENGINEERING
SCHOOL OF BIO AND CHEMICAL ENGINEERING**

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

MARCH -2021



SATHYABAMA

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

**Accredited with “A” grade by NAAC | 12B Status by UGC | Approved by AICTE
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119
www.sathyabama.ac.in**

DEPARTMENT OF BIOMEDICAL ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Manobala P (37240050)**, who carried out the project entitled “**Assessing Calorie Intake In Patient To Avoid Unwanted Health Issues**” under my supervision from **October 2020 to March 2021**

Dr.S.KRISHNAKUMAR, M.Sc., Ph.D.,

Internal Guide

Dr. T. SUDHAKAR, M.Sc., Ph.D.,

Head of the Department

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

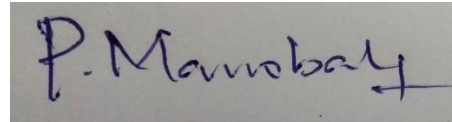
DECLARATION

I, **Manobala P (37240050)** hereby declare that the Project Report entitled **“Assessing Calorie Intake In Patient To Avoid Unwanted Health Issues ”** done by me under the guidance of **Dr. S. Krishna Kumar**, is submitted in partial fulfilment of the requirements for the award of Bachelor of Technology degree in Biomedical Engineering.

DATE: 10.4.2021

PLACE: Chennai

1.

A rectangular box containing a handwritten signature in blue ink. The signature appears to be 'P. Manobala' written in a cursive style.

SIGNATURE OF THE CANDIDATES

ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to the **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 and **Dr. J. Premkumar** M.Sc.,Ph.D., **Dr. T. Sudhakar** M.Sc., Ph.D., **Head of the Department, Dept. of Biomedical Engineering** for providing me 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. S. Krishnakumar**, M.Sc., Ph.D., **Department of Bio Medical Engineering** for his valuable guidance, suggestions and constant encouragement paving the way for the successful completion of our project work.

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

ABSTRACT

Many Hospitalized patients are sub optimal due to nutrient intake have been reported. The nutritional intake of patient"s in general medical wards is very important. To their needs many patients may not consume proper food even though the hospital provides adequate energy and nutrients. Approximately one-third of the patients' estimated energy intake was very low, and vitamin C, calcium, and zinc intakes were also of concern. At least one-third of the patients who arrive at the hospital are underweight. This may indicate that a patient's diet lacks the proper amount or ratio of calories, macronutrients, and micronutrients to facilitate sufficient healing. If this continuous then it may be a risk to the patient"s health and may not be a good sign. In this project, we proposed a deep learning system to monitor the nutrient intake for hospitalized patients. A Camera will scan the amount of food given to the patient, when the patient finishes the meal again the camera will again scan the amount of food the patient has taken. It will also note down the number of calories the patient has intake and the amount of calories the patient has left. This will help hospitalized patient to be healthy and will also help hospital staffs to monitor with ease.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO.
	ABSTRACT	i
	LIST OF FIGURES	iv
	LIST OF SYMBOLS AND ABBREVIATION	v
1	INTRODUCTION	1
	1.1 GENERAL	1
	1.2 TECHNOLOGY USED	3
	1.2.1 Deep learning	3
	1.2.1.1 Deep Learning Process	4
	1.2.1.2 Classification of Neural Networks	5
	1.2.1.3 Convolutional Neural Networks (CNN)	8
2	LITERATURE SURVEY	10
3	AIM AND SCOPE	15
	3.1 AIM	15
	3.2 SCOPE	15
	3.3 EXISTING SYSTEM	16
	3.4 DISADVANTAGE OF EXISTING SYSTEM	16
	3.5 PROPOSED SYSTEM	17
	3.6 ADVANTAGE OF PROPOSED SYSTEM	17
	3.7 APPLICATIONS	17

4	MATERIALS AND METHOD	18
	4.1 SYSTEM ARCHITECTURE	18
	4.2 WORKING	18
	4.3 MODULE DESCRIPTION	19
	4.3.1 Data collection	21
	4.3.2 Data pre-processing	21
	4.3.3 Training with the algorithm	22
	4.3.4 Object detection	23
	4.3.5 Object segmentation	25
	4.3.6 Calorie prediction	25
	4.4. SOFTWARE DESCRIPTION	26
	4.4.1 Microsoft visual studio	26
5	RESULTS AND DISCUSSION	28
	5.1 Final results obtained	28
	5.2 Discussion	33
6	SUMMARY AND CONCLUSION	34
	6.1 SUMMARY	34
	6.2 CONCLUSION	34
	REFERENCES	35
	APPENDIX	37

LIST OF FIGURES

FIGURE NO	NAME OF THE FIGURE	PAGE NO.
1.1	Deep Learning Layers	3
1.2	Deep Learning Process	4
1.3	Types of Deep Learning Networks	6
1.4	CNN	8
1.5	Types of CNN	9
4.1	Block Diagram	21
4.2	Data Pre Processing	25
4.3	Mask R-CNN architecture	26
4.4	Object detection	27
5.1	Dataset Collected	31
5.2	Input food image before segmentation	32
5.3	Input food image after segmentation	32
5.4	Below image shows the calorie content of a food	33
5.5	The calorie prediction of food consumed	33
5.6	Data visualization	34
5.7	Histogram	34
5.8	Model evaluation	35

LIST OF SYMBOLS AND ABBREVIATIONS

CNN	Convolutional Neural Network
RNNS	Recurrent Neural Network
ILSVRC	Image Net Large Scale Visual Recognition Challenge
ReLU	Rectified Linear Unit
SARSA	State-Action-Reward-State-Action
DDPG	Deep Deterministic Policy Gradient

CHAPTER -1

INTRODUCTION

1.1 GENERAL INTRODUCTION:

- People who eat foods that are high in fat and cholesterol are more likely to have heart problems while those who eat excessively are more likely to be overweight or obese. People who make healthy food choices are also bound to be healthier compared to those who do not. One's diet does not only affect the body physically but it also affects the health.
- The foods we eat contain different nutrients such as carbohydrates, proteins, fats, vitamins, and minerals. There are two things to be considered when choosing which foods are best for the health, namely caloric density and nutrient density. Caloric density refers to the amount of calories in food in relation to its weight. Nutrient density, on the other hand, refers to the nutrients found in the food in relation to the amount of calories it contains. High calorie and low nutrient food is considered to be bad for the health while high nutrient and low calorie food is considered to be essential for a healthy diet.
- Foods that are high in calories but low in nutrition should be avoided as much as possible since it adds up to the calories in the body while offering no useful benefits to the body. Also called empty calories, foods that are high in calories but low in nutrition include cookies, candy bars, white flour, sugar, soda, and ice cream. Most of these foods are not only high in calories but contain artificially-produced simple carbohydrates that can cause surges in blood sugar levels. Aside from foods with added sugar, salad dressings, mayonnaise, gravies, and sauces are also high in calories but low in nutritional value.

- Excessive intake of calories leads to weight gain especially if the individual is also physically inactive. Excess calories are processed as fat in the body, and consuming high-calorie, low-nutrition foods on a regular basis can contribute to obesity.
- Obesity is defined as having too much fat in the body. It is different from being overweight, which does not only refer to excess weight in the body from fat but also from bone, muscle, or water in the body. An individual with a body mass index (BMI) of 30 and above is considered to be obese while individuals with a body mass index of 25 to 29 already fall under the category of overweight.
- Being obese and overweight are known risk factors for the development of numerous chronic diseases such as cancer, heart disease, osteoarthritis, and diabetes. Obesity and being overweight cause not only premature mortality and extensive disability, but they may also cause financial hardship due to healthcare costs.
- Making healthy food choices is important to avoid excessive weight gain. Using a calorie calculator can help in monitoring the amount of calories consumed per day while a calorie chart can help you keep track of calories, fat, protein, and carbohydrate content of foods. Reading the nutrition labels of foods can also help you identify foods that are high in calories but low in nutrients.
- Eating the right kinds of food will not only provide the body with all the nutrients it needs but will also help prevent health problems. Foods that are high in calories and have low nutrients can lead to obesity, which in turn can be a risk factor for many health conditions. Aside from a healthy diet, it is also necessary to get regular exercise for effective weight management.

1.2 TECHNOLOGIES USED:

1.2.1 Deep learning

Deep learning is a form of computer software that simulates the brain's network of neurons. It is a branch of machine learning and is referred to as deep learning because it employs deep neural networks.

Deep learning algorithms are constructed with connected layers.

- The Input Layer is the first layer
- The Output Layer is the second.
- All of the layers in between are known as Hidden Layers. "Deep" refers to a network of more than two layers of neurons..

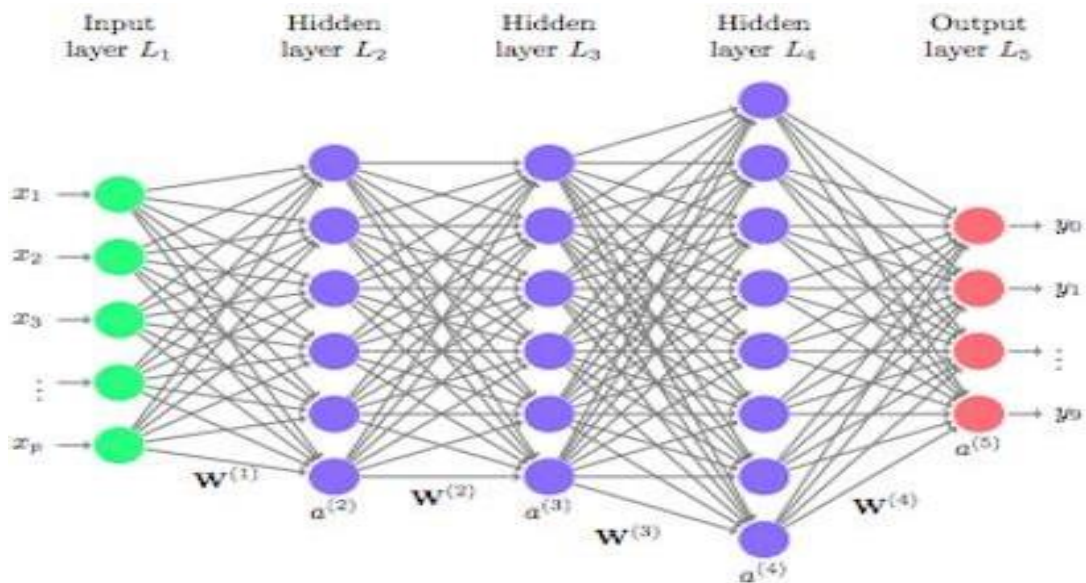


Figure 1.1 Deep Learning Layer

Each Hidden layer is composed of neurons. The neurons are connected to each other. The neuron will process and then propagate the input signal it receives the

layer above it. The strength of the signal given the neuron in the next layer depends on the weight, bias and activation function.

The network collects large volumes of input data and processes it across multiple layers, with each layer learning increasingly complex data features.

Deep learning is a strong method for turning prediction into a useful outcome. Pattern discovery (unsupervised learning) and knowledge-based prediction are two areas where deep learning excels. Big data is the fuel for deep learning .When these two factors are combined, a company can achieve unparalleled levels of growth, revenue, management, and creativity.

Traditional methods can be outperformed by deep learning. Deep learning algorithms, for example, are 41% more accurate in image classification than machine learning algorithms, 27% more accurate in facial recognition, and 25% more accurate in voice recognition

1.2.1.1 Deep learning process

A deep neural network provides state-of-the-art accuracy in many tasks, from object detection to speech recognition. They can learn automatically, without predefined knowledge explicitly coded by the programmers.

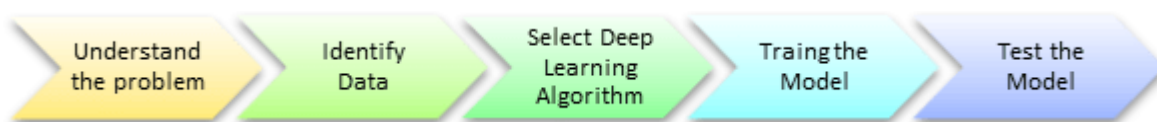


Figure 1.2 Deep Learning Process

Consider a family with a baby and parents to understand the concept of deep learning. With his little finger, the toddler points to things and repeats the word "cat" over and over. Since his parents are worried about his education, they constantly tell him, "Yes, that is a cat" or "No, that is not a cat." The baby continues to point to

things, but with cats, he becomes more precise. Deep down, the little guy doesn't understand why he should tell whether it's a cat or not. He's only recently learned how to rank complicated features on a cat by looking at the whole animal and then focusing on aspects like the tails or the nose before making a decision.

A neural network works quite the same. Each layer represents a deeper level of knowledge, i.e., the hierarchy of knowledge. A four-layer neural network can learn more complex features than a two-layer neural network.

The learning occurs in two phases.

- The first phases consist of applying a nonlinear transformation of the input and create a statistical model as output.
- The second phase aims at improving the model with a mathematical method known as derivative.

The neural network repeats these two phases hundreds to thousands of time until it has reached a tolerable level of accuracy. The repeat of this two-phase is called an iteration.

1.2.1.2 Classification of neural networks

SHALLOW NEURAL NETWORK: The Shallow neural network has only one hidden layer between the input and output.

DEEP NEURAL NETWORK: Deep neural networks have more than one layer. For instance, Google LeNet model for image recognition counts 22 layers.

Deep learning is now used in a variety of applications, including driverless cars, cell phones, Google Search Engine, fraud detection, television, and so on.

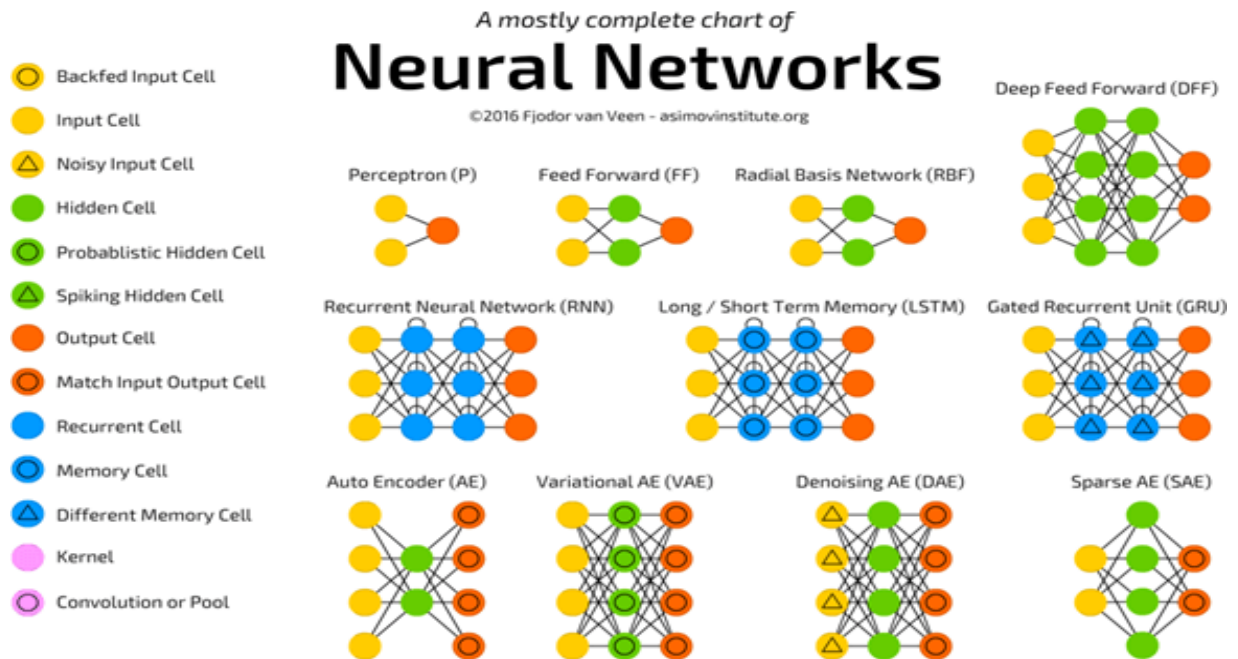


Figure 1.3 Types of Deep Learning Networks

FEED-FORWARD NEURAL NETWORKS

Artificial neural networks in their most basic form. Knowledge flows in just one direction, forward, with this architecture. It means that data flows from the input layer to the "hidden" layers and finally to the output layer. There is no loop in the network. The output layers are where the information comes to a halt.

RECURRENT NEURAL NETWORKS (RNNs)

RNN is a multi-layered neural network that can store information in context nodes, allowing it to learn data sequences and output a number or another sequence. In simpler terms, it's a form of artificial neural network that uses loops to link neurons. RNNs are well-suited to processing input sequences.

The process of RNNs are explained below:

- The RNN neurons can receive a signal pointing to the sentence's beginning..
- The network receives the word "Do" as an input and produces a vector of the number. This vector is fed back to the neuron, giving the network a memory. This stage assists the network in remembering that it obtained "Do" in the first place.
- The network will similarly proceed to the next words. It's a combination of the words "you" and "want." - Time a word is received, the state of the neurons is changed..
- The final stage occurs after receiving the word "a." The neural network will provide a probability for each English word that can be used to complete the sentence. A well-trained RNN probably assigns a high probability

COMMON USES OF RNN

- Help securities traders to generate analytic reports
- Detect abnormalities in the contract of financial statement
- Detect fraudulent credit-card transaction
- Provide a caption for images
- Power chatbots
- The standard uses of RNN occur when the practitioners are working with time-series data or sequences.

1.2.1.3 Convolutional Neural Networks (CNN)

- CNN is a multi-layered neural network with a unique architecture designed to extract increasingly complex features of the data at each layer to determine the output. CNN's are well suited for perceptual tasks

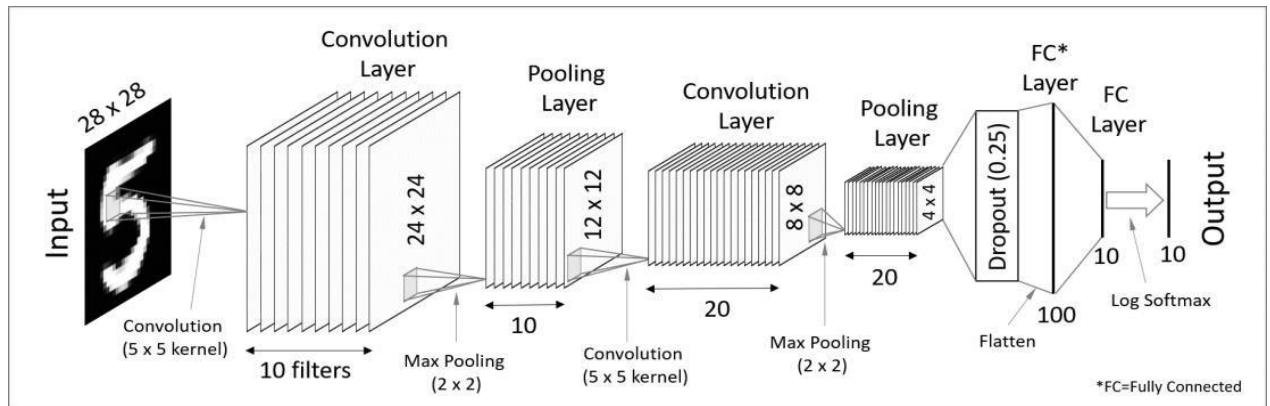


Figure 1.4 CNN

CNN is most often used when professionals need to retrieve information from an unstructured data set (e.g., images).

If the goal is to predict an image caption, for example:

- The CNN receives an image of, say, a cat; this image is a set of pixels in computer terms. In general, a greyscale image has one layer and a colour image has three layers.
- During the feature learning (i.e., hidden layers), the network will identify unique features, for instance, the tail of the cat, the ear, etc.
- When the network has mastered the art of image recognition, it may assign a probability to each image it recognizes. The network's estimate would be the mark with the highest likelihood.

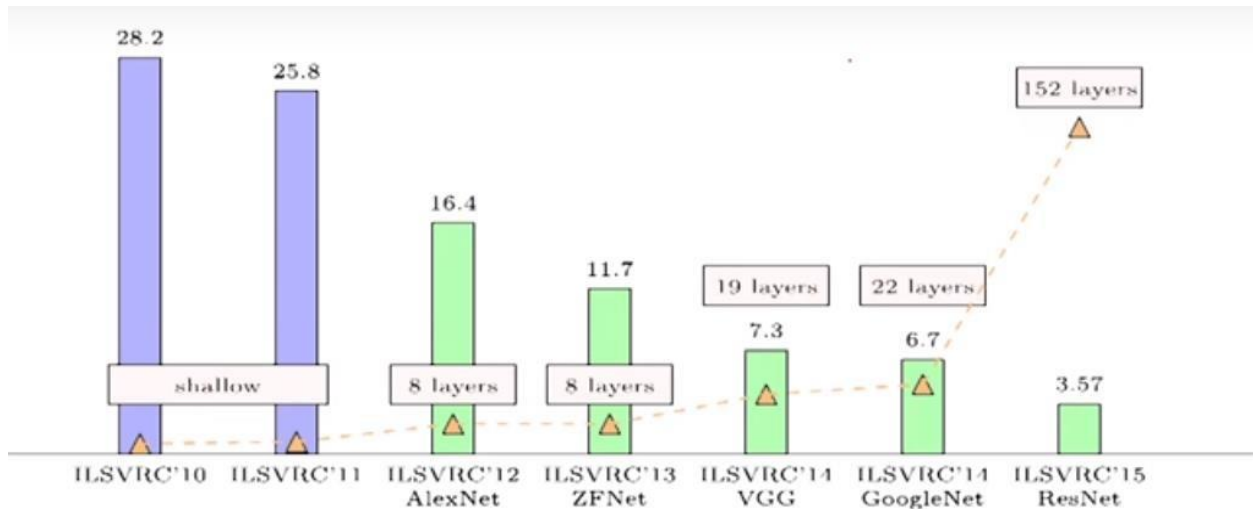


Figure 1.5 Types of CNN

A Convolutional Neural Network (CNN, or ConvNet) is a form of multi-layer neural network that is designed to recognise visual patterns directly from pixel images with little to no pre-processing. The Image Net project is a massive visual database created to aid in the development of visual object recognition software. The ImageNet project hosts the Image Net Large Scale Visual Recognition Challenge (ILSVRC), an annual software competition in which software programmes compete to correctly recognise and detect objects and scenes.

CHAPTER - 2

LITERATURE SURVEY.

2.1 LITERATURE SURVEY:

Mukul Sarkar Nitin Gupta., et al., (2018) reported that The food industry is looking for a quick and inexpensive way to track food quality. To determine the best harvesting period, track changes during storage, and assess the internal quality of the fruits, non-destructive and non-invasive testing is needed. For non-invasive simultaneous measurement of the scattering and absorption properties of turbid media, time-resolved reflectance spectroscopy (TRS) is used. To control the internal consistency of the fruits, a time-resolved phase reflectance dependent non-destructive and non-invasive method is proposed in this paper. The number of internal scatters dependent on the incident light's mean free path length determines the time-resolved reflectance. The change in the mean free path due to change in the pigmentation of the internal pulp, changes the order of the scatters and therefore, affects the scattering profile of the reflected light. Stokes degree of linear polarisation is used to test the scattering profile. As an example, the Red Star and Granny Smith apples are considered and the degradation in the apples due to storage is correlated with the changes in the DoP. The change in the DoP after 7 days is 0.26% and 11.4% for a time resolution of 300 ps while for 500 ps it is 2% and 2.3% for Red Star and Granny Smith respectively. The differences in DoP aid in monitoring the internal pulp content of the two apple varieties under consideration. This proposed method can be used to track internal quality in all agro-based products.

Shota Horiguchi., et al., (2018) published a paper on Personalized classifier for food image recognition. In that Food image recognition tasks are currently assessed using fixed datasets. However, in real-world situations, the number of samples in each class can continue to rise, and samples from new classes may appear. In particular, dynamic datasets in which each user generates samples and continues the updating process often have content that differs significantly from one user to the next. The number of samples available to each user is extremely small. Such complex data

cannot be handled by a single classifier used by all users. The goal of bridging the gap between the laboratory and the real world has yet to be realized on a wide scale. A promising way to do this is to incrementally personalize a classifier for each user. A promising way to do this is to incrementally personalize a classifier for each user. In this paper, we look at the personalization issue, which entails gradually adapting to the user's domain using a small number of samples. A combination of the nearest class mean classifier and the 1-nearest neighbor classifier based on deep features is proposed as a simple but efficient personalization framework.

Bin liu ., et al., (2019) discusses a method for visualizing and modeling internal fixation plates for computer-assisted, individualized, and minimally invasive curved bone fracture surger. Methodologies: We are concentrating on individualized reverse reconstruction of the bone fracture plate using three-dimensional (3D) mesh templates obtained from a 3D optical scanner. The steps of the method are as follows. To begin, a Bezier curve (ridge line) of broken bones is reconstructed using principal component analysis (PCA) and the K-means process. Second, a capsule projection model of the broken bones is proposed to obtain feature details of the broken bone parts based on the geometric structure of the curved broken bones. For preregistration, the Ordering Points to Identify the Clustering Structure (OPTICS) approach is used (rough registration). Fourth, the cross-section points are extracted using a regional self-growth approach. Fifth, for correct registration of the fracture surface models, the iterative closest point (ICP) approach is used. Finally, based on multiple user inputs, a custom internal fixation plate model is reconstructed. The internal fixation plate model can be reconstructed based on the bone parameters of the patient. Many researchers have focused on developing minimally invasive, accurate and personalized surgical schemes to treat bone fractures. To repair curved bone fractures, which frequently occur, we designed an automatic personalized internal fixation plate modeling framework for minimally invasive curved bone fracture surgery based on preregistration with a capsule projection model.

Gert Mertes, et al., (2018) Discover a new plate system. This new plate system is presented that can detect the weight and position of individual bites during meals. A

base station with sensors and a detachable off-the-shelf polymer plate with three compartments make up the device. Individual bites' weights can be precisely calculated and localised on the plate using data from several weight sensors, allowing the compartment from which they were taken to be determined. The device will estimate the nutritional value of the ingested food based on previous knowledge of the weight of the food in each compartment at the start of the meal. The machine measured the weight of ingested food in each compartment with a maximum relative error of 1.4 percent in a test performed in a controlled home setting. The system's aim is to replace conventional monitoring tools and automatically track consumption levels. This is sufficient to estimate the total amount of calories, given that the food present on the plate is known beforehand. In situations where food monitoring may be required, such as treatment of eating disorders, hospitalization or elderly care, food is typically served from a central kitchen with a known menu..

Lixing Wang (2018) reported that The management of food safety is a critical problem in human health and social stability. Because of the large scale of food supply networks and the limited monitoring resources, monitoring all network nodes is prohibitively costly. We propose a guided graph-based social network model for hazard management network architecture optimization. Following the principles of nodes in social networks, the concepts and calculating methods of k-step observability/controllability and accessible node sets are proposed to measure node accessibility. A 0-1 linear programming model is designed for the selection of the optimal monitoring points based on the above concepts. With limited monitoring services, the chosen combination of monitoring points will access the entire food supply network. A popular commercial software package can be used to solve the model. To assess the model's validity, we apply it to a specific local supply network of infant foods. A hazard substance monitoring network is an efficient means to guarantee food safety. Given the constrained monitoring resources, the nodes in the network should be optimally selected as the monitoring points to cover whole food supply networks.

Vasileios Papapanagiotou., et al., (2018).suggested that Obesity and eating disorders have been linked to the composition of the accumulated food intake (CFI) curve. This

curve is captured using scales that record the weight loss of a plate on which a subject consumes food; however, their measurements are polluted by additive noise and skewed by some types of objects. This paper describes an algorithm for extracting the clean CFI curve and in-meal eating measures such as total food intake and food intake rate from continuous in-meal weight measurements. The algorithm uses a string of symbols to represent the weight time series, which correspond to events like bites or food additions. The next step is to use a context-free grammar to model a meal as a series of such events. Finally, the expected eating sequence is determined by selecting the most likely parse tree. The algorithm is tested on a dataset of 113 meals obtained using the Manometer, a scale that measures plate weight continuously while feeding. We assess the efficacy of seven measures as well as the identification of bite-instances. We equate our method to three state-of-the-art algorithms and find that ours has the lowest error rates for the majority of indicators (24 g for total meal weight). The proposed algorithm extracts the parameters of the CFI curve automatically, eliminating the need for manual data processing, and thus facilitating large-scale studies of eating behaviour.

Eduardo Aguilar, et al.,(2018) reported that People's growing knowledge of their nutritional habits has drew a lot of attention to the field of automated food analysis. Focusing on self-service restaurants environment, automatic food analysis is not only useful for extracting nutritional information from foods selected by customers, In periods of high demand, it's also essential to improve service by removing the bottleneck that occurs at the cash registers. In this paper, we address the problem of automatic food tray analysis in canteens and restaurants environment, which consists in predicting multiple foods placed on a tray image. We propose Semantic Food Detection, a new method for food analysis based on convolutional neural networks that combines food localization, recognition, and segmentation in a single system. On the public dataset UNIMIB2016, we show that our method enhances state-of-the-art food detection by a large margin, achieving around 90% in terms of F-measure, and thus provides a significant technical advance towards automated billing in restaurant environments. We present a novel framework that performs Semantic Food Detection in the context of self-service restaurant food tray research. More specifically, we

combine both methods, food/non-food semantic segmentation and food detection, by employing two procedures: a probabilistic procedure for removing context detections and a custom non-maximum procedure for detecting food.

Shuqiang Jiang,et al.,(2019) has done a paper on"Multi-Scale Multi-View Deep Feature Aggregation for Food Recognition",It states That Food recognition has recently gotten a lot of attention in image processing and computer vision because of its huge potential in human health. For food recognition, the majority of current methods use Convolutional Neural Networks (CNNs) to extract deep visual features. Such methods neglect food image characteristics, making it difficult to achieve optimum recognition efficiency Food images, in contrast to general object recognition, do not usually have a distinct spatial structure or common semantic patterns. In this paper, we propose a food recognition scheme called MSMVFA (Multi-Scale Multi-View Feature Aggregation). MSMVFA is capable of combining high-level semantic features, mid-level attribute features, and deep visual features into a single representation. As a result, the aggregated features have the best chance of capturing the semantics of food images. We use additional ingredient knowledge to obtain mid-level attribute representation through ingredient-supervised CNNs for that solution. Deep visual features and high-level semantic features are derived from class-supervised CNNs. MSMVFA fuses multi-scale CNN activations for each type of feature to make aggregated features more discriminative and invariant to geometrical deformation, since food images do not always have a distinct spatial structure. Finally, using two levels of fusion, namely multi-scale fusion for each type of feature and multi-view aggregation for various types of features, the aggregated features are more stable, detailed, and discriminative. Furthermore, MSMVFA is a general scheme that can be used for a variety of deep networks.

CHAPTER – 3

AIM AND SCOPE

3.1 AIM

To figure out how much calories and nutrients is eaten. To accurately measure the amount of time it takes to eat the food. Determine a low-cost method of monitoring a person's food intake. Thus, the scope of the project is to determine the nutritional intake of person by calculating their calories intake by implementing the deep learning algorithm and to maintain the nutrition supply.

3.2 SCOPE

In the coming future, we review the application of the recognize the food item types and the nutrient technology in the healthcare field and it can promote for the recognize the food item quality technology with more accuracy compare to this project and in health care field there have more chance to develop or convert this project in many ways. Then in this project we can identify the food item type before and after eating thus the nutrient intake of everyone can be found. The predicted calorie intake can be utilized for maintaining the patient's diet and their nutritional supplements can be maintained for their fast recovery, so the time period of recovery can be reduced. The rate of malnutrition can also falls down at rate by making the proper nutrition supplements to everyone from the detection of calorie intake by this method.

3.3 EXISTING SYSTEM:

The food industry is looking for a quick and inexpensive way to track food quality. To determine the best harvesting period, track changes during storage, and assess the internal quality of the fruits, non-destructive and non-invasive testing is needed. For non-invasive simultaneous measurement of the scattering and absorption properties of turbid media, time-resolved reflectance spectroscopy (TRS) is used. In this work, a time-resolved phase reflectance based non-destructive and non-invasive method is proposed to monitor the internal quality of the fruits. The number of internal scatters dependent on the incident light's mean free path length determines the time-resolved reflectance. Changes in the mean free path caused by changes in the pigmentation of the internal pulp influence the scattering profile of reflected light by changing the order of the scatters. Stokes degree of linear polarisation is used to test the scattering profile. The deterioration of the apples due to storage is associated with changes in the DoP in the Red Star and Granny Smith apples, for example. The change in the DoP after 7 days is 0.26% and 11.4% for a time resolution of 300 ps while for 500 ps it is 2% and 2.3% for Red Star and Granny Smith respectively. The differences in DoP aid in monitoring the internal pulp content of the two apple varieties under consideration. This proposed method can be used to track internal quality in all agro-based product.

3.2 DISADVANTAGES OF EXISTING SYSTEM:

- This technique did not present for all variety of food.
- Project didn't contain more accuracy.
- This methodology focuses on only the food recognition and quality.

3.3 PROPOSED SYSTEM

The foods we eat contain different nutrients such as carbohydrates, proteins, fats, vitamins, and minerals. There are two things to be considered when choosing which foods are best for the health, namely caloric density and nutrient density. Caloric density refers to the amount of calories in food in relation to its weight.

Nutrient density, on the other hand, refers to the nutrients found in the food in relation to the amount of calories it contains. High calorie and low nutrient food is considered to be bad for the health while high nutrient and low calorie food is considered to be essential for a healthy diet. In this project, we recognize the food item type along with its nutrient level in that food, based on the quantity of food available at a specific time. After a certain duration, the food items are again analyzed for the quantity remaining, based on this data the food consumed is calculated along with the nutrient intake. Excessive intake of calories leads to weight gain especially if the individual is also physically inactive. Excess calories are processed as fat in the body, and consuming high-calorie, low-nutrition foods on a regular basis can contribute to obesity. By this project we can identify the amount of food and nutrients consumed by any person.

3.4 ADVANTAGES OF PROPOSED SYSTEM

- This project helps to recognize the food type and calories. .
- Reduces the time required for manual classification.
- Provides efficient way for providing best and fast result of food type.
- Eliminates the human error rate when done manually.
- It was more accuracy.

3.5 APPLICATIONS:

- Used in hotels, restaurants etc...,
- Used in homes while before eating.
- Used in anywhere while eating junk food.

CHAPTER 4

MATERIALS AND METHODS

DETAILED DESIGN OF THE PROJECT

- This chapter describes the overall functioning of the project.

4.1 BLOCK DIAGRAM

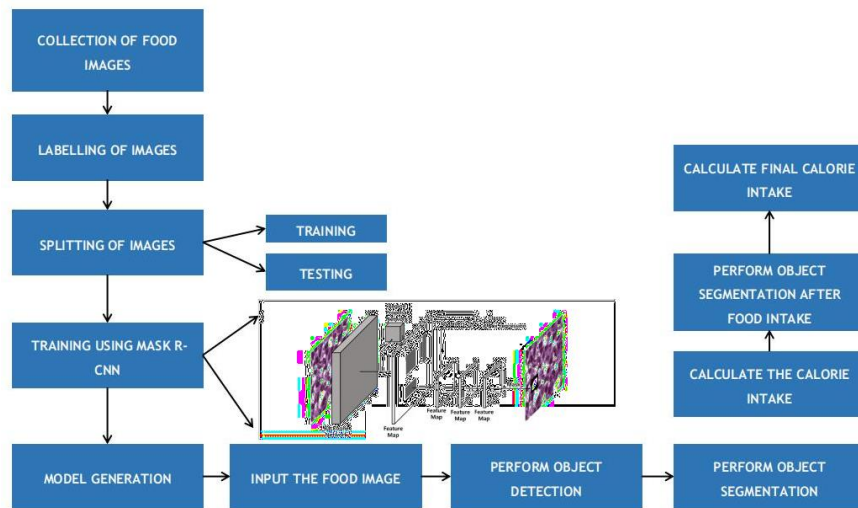


Figure 4.1: Block Diagram

4.2 WORKING

In this project, we are going to determine the nutrient intake and the amount of food consumed by a person by which we can help the elder ones as well as parents of children to monitor the nutrient intake as well as the amount of food consumed by them helping them to regulate their diet. So, the first step in the project will be collecting the dataset from various resources of the types of food and then we will be separating these datasets into training as well as testing dataset where the testing dataset will be kept separate and the training dataset will be used to train the model. Then dataset augmentation will be done which will increase the dataset into multiple numbers. Then these datasets are pre-processed using different techniques to align the datasets into single dimensions. After pre-processing our dataset we will be ready for training with

the architecture. Now, we will be using architecture such as mask R-CNN to train the model. We will be performing loss minimization such as cross entropy and finally validate the mode by giving real time inputs. The type of food is identified using object detection with the quantity along with the nutrient available at a specific recorded time. The difference in the quantity of food after a certain duration of time specifies the amount of food and nutrient intake consumed by the person.

4.3 MODULE DESCRIPTION

- Data collection
- Data pre-processing
- Training with the algorithm
- Object detection
- Object segmentation
- Calorie prediction

4.3.1 Data Collection

A data set is a collection of data Deep Learning has emerged as the preferred tool for tackling a wide range of difficult real-world problems. It is, without a doubt, the most effective approach for computer vision tasks. Deep learning's strength in computer vision is demonstrated in the picture above. A deep network can segment and classify the "key points" of any individual in an image with enough training. These deep learning machines that have been working so well need fuel lots of fuel; that fuel is data. The more labelled data available, the better our model performs. Google has also experimented with the concept of more data contributing to better results on a wide scale, with a dataset of 300 million images! When deploying a Deep Learning model in a real-world application, data must be constantly fed to continue improving its performance. And, in the deep learning era, data is very well arguably the most valuable resource. There are three steps of collecting data.

Scraping From the Web

Because of the amount of human effort involved, manually locating and uploading pictures takes a long time. The task probably has some kind of common

objects are to be detected. And so that becomes the keyword for web-scraping. It also becomes the object's class name. This seems to be very simple for a task like image classification, where the image annotations are relatively coarse. But to do something like instance segmentation. Every single pixel in the image is required. It's best to use some of the many excellent image annotation resources that are already available. The paper demonstrates how to construct a model that can produce pixel labels for segmentation from a rough collection of polygon points around an object. Deep extreme cut is identical to deep extreme cut, except that only the four extreme points around the object are used. This will then give some nice bounding box and segmentation labels. Another alternative is to make use of pre-existing image annotation user interfaces. Label someone very popular where one can draw both bounding boxes and set polygon points for segmentation maps. Amazon Mechanical Turk is also a cheap option.

Third-party:

Since data has become such a valuable asset in the deep learning era, many start-ups have begun to offer their own image annotation services, where they can collect and mark data. Given a description of what kind of data and annotations needed. Mighty, a company that specialises in self-driving car image annotation and has grown to be a major player in the field was also present at CVPR 2018. Payment AI is less specialised than Mighty AI, and it can annotate images from any domain. They also have a few additional features, such as video and landmark annotations.

Pre-trained Networks

Many of us are familiar with the concept of transfer learning, which entails starting with a network that has been pre-trained on a broad dataset and then fine-tuning it on our own. We may apply the same concept to our new dataset collection. These pre-trained networks were trained on massive datasets; just look at the Open Images dataset, which contains over 15 million images labelled with bounding boxes from 600 categories! A network trained on this dataset would be pretty good at detecting objects right away. As a result, we'll be able to use it to draw some bounding

boxes around the objects in our photos. A network trained on this dataset would be pretty good at detecting objects right away. As a result, we can use it to create bounding boxes around the items in our images. All we have to do now is identify the items in the boxes, which cuts our job in half! Moreover, with 600 types, some of the objects you want to detect and classify may have already been detected and classified with high accuracy. The Tensor Flow Object Detection API actually already has a network pre-trained on Open Images if you'd like to try it out.

4.3.2 Data pre-processing

Deep learning has truly come into the mainstream in the past few years. Deep learning makes use of neural networks with a large number of hidden layers (dozens in today's state of the art), as well as a large amount of training data. These models have proven particularly useful in gaining insight and approaching human-level precision in perceptual tasks such as vision, voice, and language processing. Several decades earlier, the theoretical and mathematical foundations were laid. Primarily two phenomena have contributed to the rise of deep learning: a) Availability of huge datasets/training examples in multiple domains and b) Advances in raw compute power and the rise of efficient parallel hardware.

The network architecture as well as the input data format must be carefully considered when creating an efficient neural network model. The number of images, image height, image width, number of channels, and number of levels per pixel are the most common image data input parameters. Typically, there are 3 channels of data corresponding to the colours Red, Green, Blue (RGB). Pixel levels are usually [0,255]. number of images = 100

- image width, image height = 100
- channels, pixel levels ranging from 0 to 255

Uniform aspect ratio: Making sure the pictures are the same size and aspect ratio is one of the first things to do. The majority of neural network models assume a square-

shaped input image, which means that each image must be tested for squareness and cropped accordingly. Cropping can be used to select a square portion of an image, as seen in the example. When cropping, we typically focus on the central portion of the picture.



Figure 4.2: Data pre-processing

4.3.3 Training using mask R-CNN architecture

Faster R-CNN is extended with Mask R-CNN to solve instance segmentation problems. It accomplishes this by adding an object mask prediction branch to the current branch for bounding box recognition. Mask R-CNN is an intuitive extension of Faster R-CNN in theory, but correctly assembling the mask branch is crucial for good results.

Most importantly, The faster R-CNN was not designed to sync network inputs and outputs pixel-by-pixel. RoIPool, the de facto core operation for attending to instances, demonstrates this by performing coarse spatial quantization for function extraction. Mask R-CNN corrects the misalignment by using RoIAlign, a simple, quantization-free layer that faithfully preserves exact spatial positions.

Second, Mask R-CNN separates mask and class prediction: it forecasts a binary mask for each class separately, without competition, and relies on the network's RoI classification branch to predict the group. In contrast, an FCN usually perform per-pixel

multi-class categorization, which couples segmentation and classification.

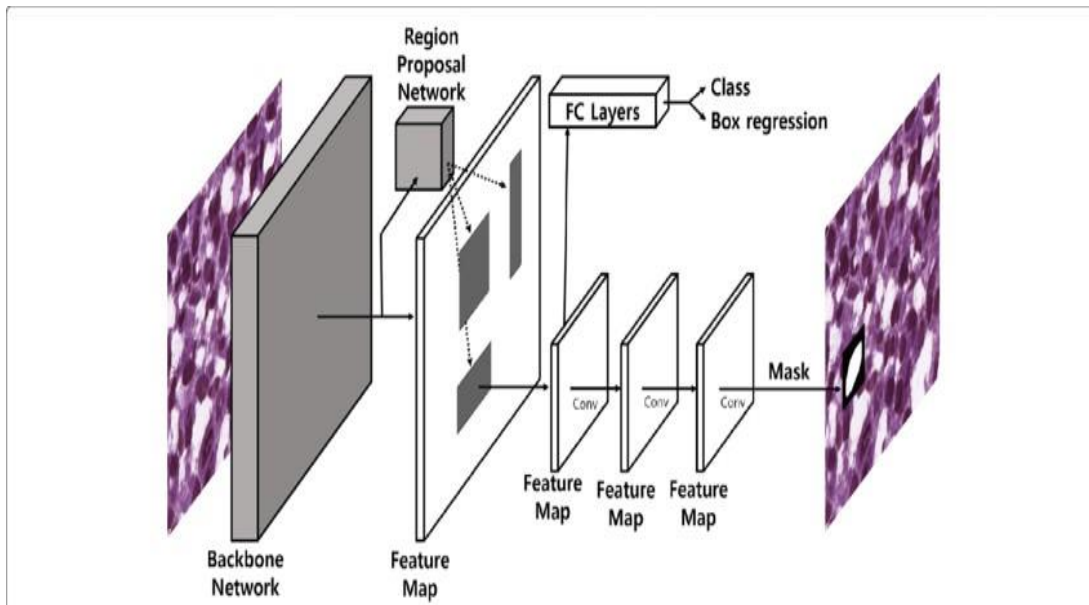


Figure 4.3: Mask R-CNN architecture

4.3.4 Object detection

Object detection is the method of accurately locating all potential instances of real-world objects in photographs or videos, such as human faces, flowers, vehicles, and so on. To identify all instances of an object type, the object detection technique employs derived features and learning algorithms.

Object detection is a technique for identifying, detecting, and localising several visual instances of objects in an image or video. Rather than just simple object classification, it offers a much clearer interpretation of the object as a whole. This approach can be used to count the number of instances of unique objects and mark them as well as count the number of instances of unique objects. The efficiency of this method has greatly improved over time, assisting us with real-time use cases.

Overall, it provides an answer to the query, “What object is located where and how much of it is present?”

Object detection technology evolved in the early 2000s, as did the detectors at the time. They used low-level and mid-level vision, as well as the „recognition-by-components' technique. The features that were taken into consideration were distance transforms, shape contexts, and edgeless, among others. This approach allowed object detection as a measurement of similarities between the object components, shapes, and contours, and the features that were taken into consideration were distance transforms, shape contexts, and edgeless, among others. Things didn't go as planned, so machine detection methods were introduced to help solve the problem

Objects of "different sizes" and "different aspect ratios" had to be taken into account when multi-scale detection was to be achieved. During the early stages of object detection, this was one of the most difficult technological problems. However, the issue was solved after 2014, as a result of increased technological advances. This led to the second step of object detection, where deep learning was used to complete the tasks.

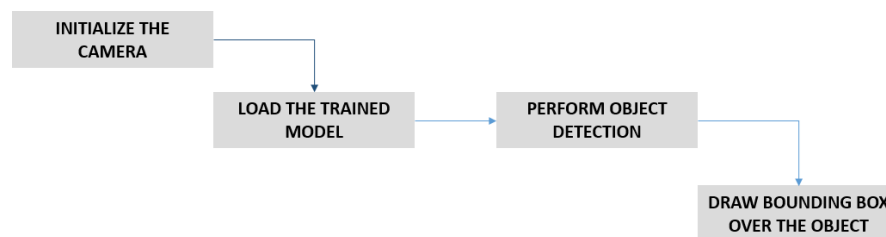


Figure 4.4: Object detection

4.3.5 OBJECT SEGMENTATION

In computer vision, image segmentation is a crucial step. It entails segmenting a visual input to make image analysis easier. Segments are made up of "super-pixels," which are collections of pixels that represent objects or parts of objects. Image segmentation divides pixels into larger components, removing the need to think of individual pixels as observation units. There are three levels of image analysis:

- Classification – includes assigning a category to the entire picture, such as "people," "animals," or "outdoors."
- Object detection - detecting and drawing a rectangle around objects within an image, such as an individual or a sheep.
- Segmentation- identifies image parts and determines which object they belong to. The foundation for object detection and classification is segmentation.

4.3.6 Calorie prediction

- The calorie of a particular food is predefined and the type of food is detected. After a certain time limit the camera module captures the food again based on using the object segmentation, the difference in the quantity of food is achieved. Based on the difference in the quantity of food the amount of calories consumed by a person is detected.

4.4 SOFTWARE DESCRIPTION

- Microsoft Visual studio
- Python

4.4.1 Microsoft Visual Studio

Microsoft Visual Studio is the company's integrated production environment (IDE). It's used to make blogs, web applications, web services, and smartphone apps,

among other things. Windows API, Windows Shapes, Windows Presentation Foundation, Windows Store, and Microsoft Silverlight are some of the Microsoft software development tools used by Visual Studio. It can generate both native and compiled code.

Visual Studio comes with a code editor that supports IntelliSense (code completion) and code refactoring. The integrated debugger can be used as both a source-level and a machine-level debugger. A code profiler, a designer for creating GUI applications, a web designer, a class designer, and a database schema designer are among the other built-in tools. It embraces plug-ins that improve functionality at almost every level, such as adding support for source control systems (such as Subversion and Git) and new toolsets for domain-specific languages, such as editors and visual designers, or toolsets for other aspects of the software development lifecycle (like the Azure DevOps client: Team Explorer).

Visual Studio supports 36 programming languages, and the code editor and debugger can support nearly any programming language (to varying degrees) if a language-specific service is available. C, C++, C++/CLI, Visual Basic.NET, C#, F#, JavaScript, TypeScript, XML, XSLT, HTML, and CSS are among the built-in languages. Plug-ins provide support for other languages such as Python, Ruby, Node.js, and M, among others. In the past, Java (and J#) were supported.

The Community version of Visual Studio is the most basic and is available for free. "Free, fully-featured IDE for students, open-source, and individual developers," says the Visual Studio Community edition's slogan".

The currently supported Visual Studio version is 2019

Features

- Code editor. Like any other IDE, it includes a code editor that supports syntax highlighting and code completion using IntelliSense for variables, functions, methods, loops, and LINQ queries.
- Debugger.

- Designer.
- Other tools.
- Extensibility.
- Previous products.
- Community.
- Professional.

➤ **Python**

Python is a high-level, interpreted programming language that can be used for a variety of tasks. Python was designed by Guido van Rossum and first published in 1991. Its design philosophy emphasizes code readability and makes extensive use of whitespace. Its language constructs and object-oriented style are aimed at assisting programmers in writing simple, logical code for both small and large-scale projects.

Python is garbage-collected and dynamically typed. Procedural, object-oriented, and functional programming are among the programming paradigms supported. Because of its extensive standard library, Python is sometimes referred to as a "batteries included" language.

Python was created in the late 1980s as a replacement for the ABC programming language. List comprehensions and a garbage collection framework capable of collecting reference cycles were implemented in Python 2.0, which was published in 2000. Python 3.0, published in 2008, is a big revision of the language that isn't fully backwards compatible, and most Python 2 code won't run on Python 3.

CHAPTER 5

RESULTS AND DISCUSSION

5.1 RESULTS

- To begin with, testing of the trained model, we can split our project into modules of implementation that is done. Dataset collection involves the process of collecting different types of food images. Various datasets were collected and one example among the collected dataset can be found below

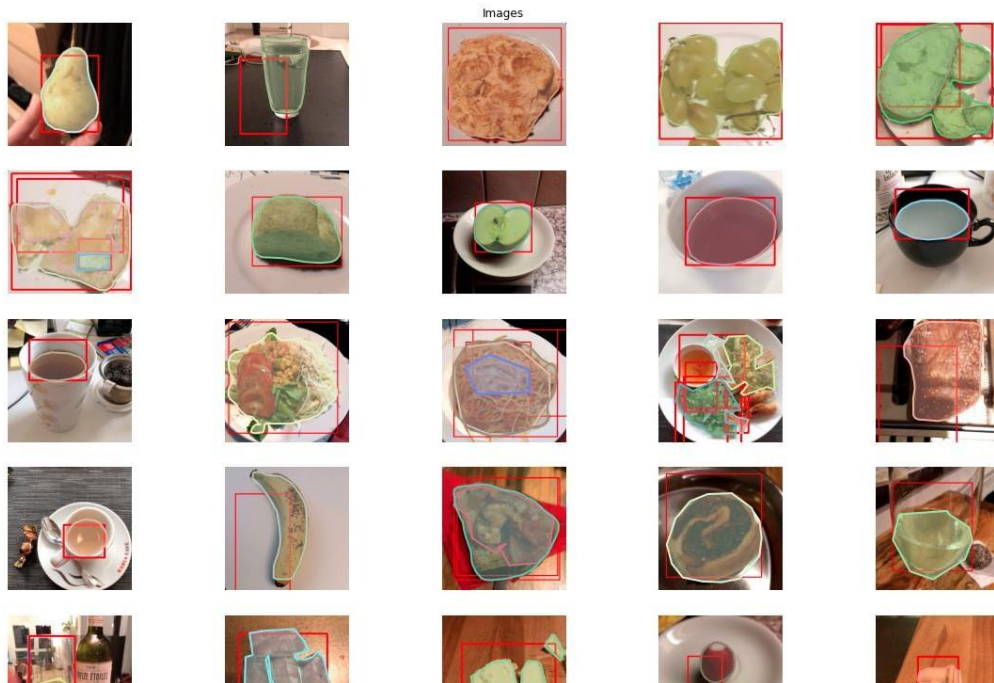


Figure 5.1 Dataset Collected

The goal of image segmentation is to cluster pixels into salient image regions, i.e., regions corresponding to individual surfaces, objects, or natural parts of objects. The below image shows the input food image before segmentation.



Figure 5.2 Input food images before segmentation

The below image shows the input food image after segmentation.

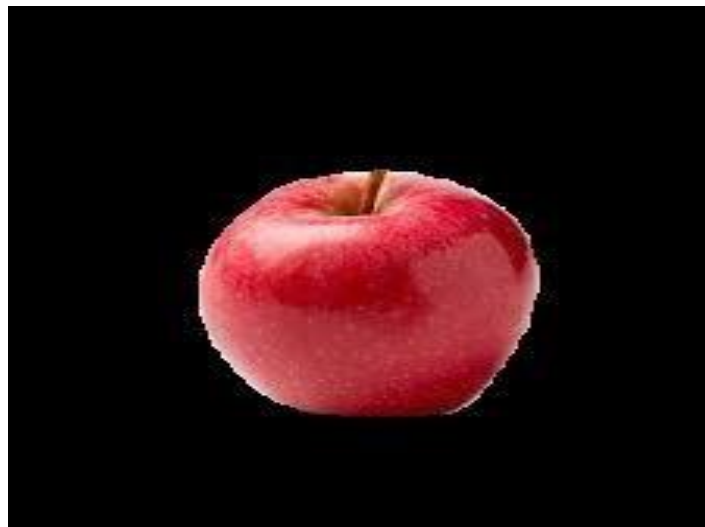


Figure 5.3 Input food images after segmentation

The below image shows the calorie content of the food image on being given as input:

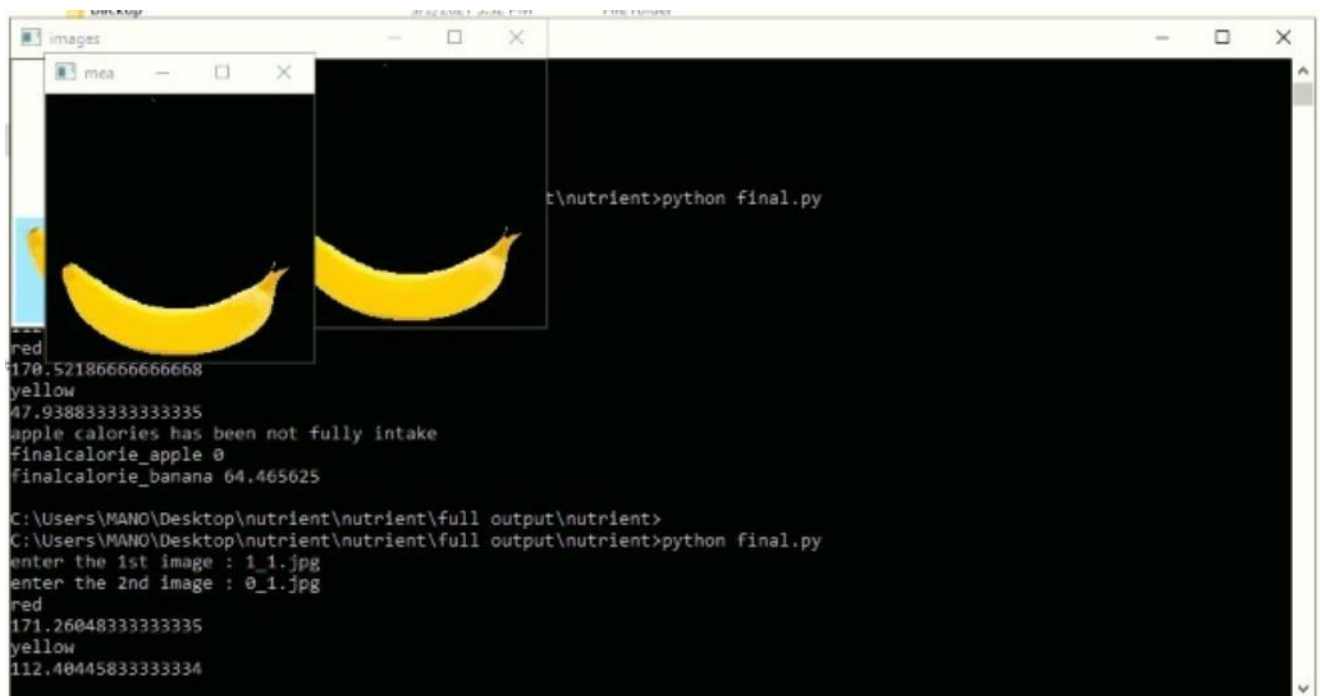


Figure 5.4 image shows the calorie content of a food

The below image shows the calorie prediction of food consumed :

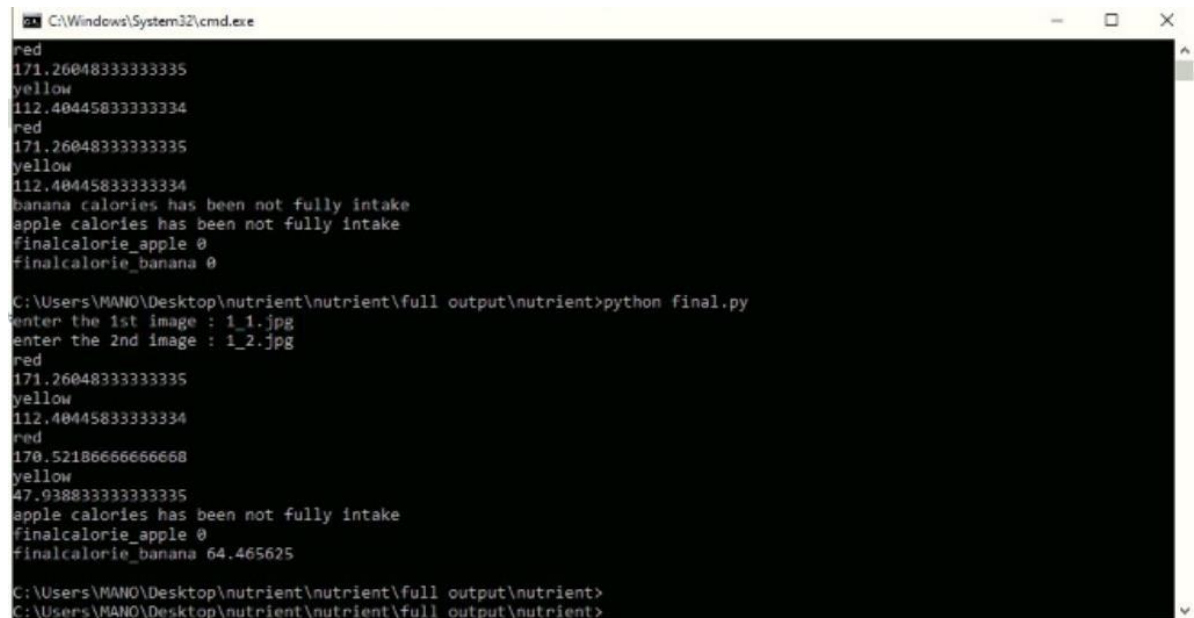


Figure 5.5: the calorie prediction of food consumed

Data visualization is the graphical representation of information and data. The below image shows the data visualisation of the nutrient project:

- ▼ Data Visualisations

```
[ ] fig = go.Figure([go.Bar(x=list(no_images_per_category.keys()), y=list(no_images_per_category.values()))])
fig.update_layout(
    title="No of Image per class",)
fig.show()
```

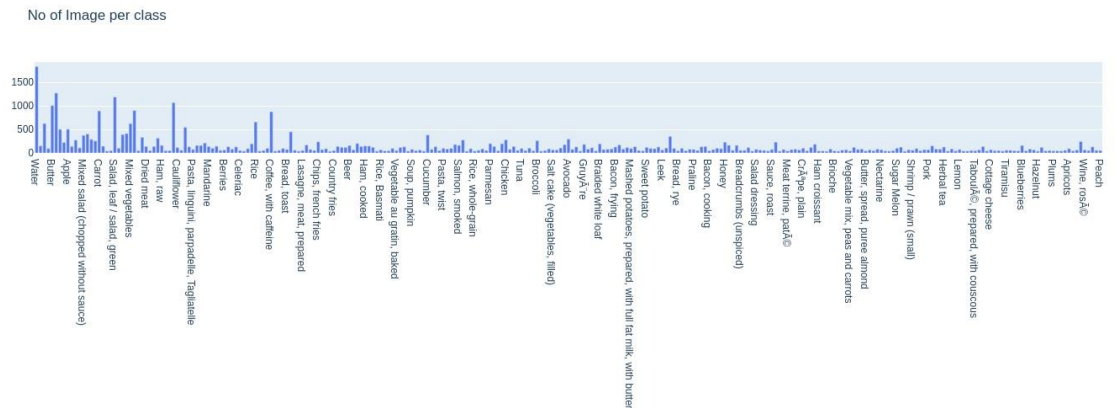


Figure 5.6: Data visualization

A histogram is a graphical display of data using bars of different heights. In a histogram, each bar groups numbers into ranges. The below image shows the histogram of the nutrient project:

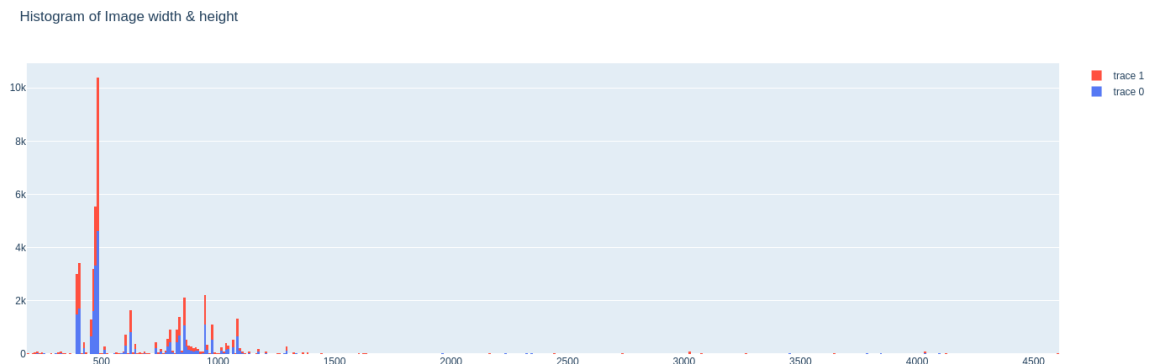


Figure 5.7: Histogram

The below image shows the model evaluation of the nutrient project:

▾ Evaluating the model 🍌

After training is done, we will evaluate our model to see how it's performing in the unseen images!

```
[ ] cfg.MODEL.WEIGHTS = os.path.join(cfg.OUTPUT_DIR, "model_final.pth")
    cfg.MODEL.ROI_HEADS.SCORE_THRESH_TEST = 0.5
    cfg.DATASETS.TEST = ("validation_dataset", )
    predictor = DefaultPredictor(cfg)
```

```
evaluator = COCOEvaluator("validation_dataset", cfg, False, output_dir=cfg.OUTPUT_DIR)
val_loader = build_detection_test_loader(cfg, "validation_dataset")
valResults = inference_on_dataset(trainer.model, val_loader, evaluator)
```

WARNING [02/28 03:54:03 d2.data.datasets.coco]:
Category ids in annotations are not in [1, #categories]! We'll apply a mapping for you.

[02/28 03:54:03 d2.data.datasets.coco]: Loaded 1269 images in COCO format from /content/val/new_ann.json
[02/28 03:54:03 d2.data.build]: Distribution of instances among all 273 categories:

category	#instances	category	#instances	category	#instances
beetroot-st..	2	green-bean-..	3	watermelon-..	9
pizza-with-..	3	pizza-with-..	7	applesauce-..	3
quiche-with-..	1	sweet-potato	6	country-fries	5
potato-gnoc..	4	potatoes-st..	25	chips-frenc..	16
rosti	4	vegetable-m..	1	mixed-veget..	34
ratatouille	8	mixed-salad..	28	leaf-spinach	6
witloof-chi..	7	salad-rocket	5	salad-leaf-..	64
salad-lambs..	4	artichoke	6	eggplant	7
avocado	19	french-beans	7	pickle	9
cucumber	27	pumpkin	3	sweet-pepper	14
tomato	52	zucchini	12	red-radish	10
beetroot-raw	5	carrot	39	celeriac	3
cauliflower	7	broccoli	11	kolhrabi	4
red-cabbage	5	white-cabbage	2	mushroom	1
mushrooms	6	peas	4	corn	4
leek	4	onion	9	fennel	3
green-aspar..	7	white-aspar..	0	alfa-sprouts	1
beans-kidney	1	chickpeas	7	lentils	3
pineapple	1	apple	38	pomegranate	2
apricots	4	banana	28	berries	3
pear	6	dates	5	strawberries	12
fruit-salad	7	blueberries	7	raspberries	0
kiwi	8	mandarine	9	mango	1

Figure 5.8: Model evaluation

Thus from the above image it is clear the project has been successfully implemented and executed to effectively predict the calorie intake by a person on being provided with food. Thus, we have successfully implemented the scope of the project.

5.2 DISCUSSION

The results obtained from , assessing calorie intake in patient to avoid unwanted health issues in which the food images were detected , segmented and calorie intake in predicted using mask RCNN deep learning method with 97.98% accuracy as composed with existing literature. Nondestructive food quality monitoring using phase information in time resolved reflectance spectroscopy (Mukul sarkar Nitin Gupta and Maher Assad 2020) in which the method can be used in all agro based products for internal quality monitoring but this method seems very costly. From the above results, It is clear that the result obtained by mask RCNN deep learning algorithm gives more accuracy values on comparing with the method which is used in th existing literature system. Personalized classifier for food image recognition (Shoto horiguchi., et al.,2018). In which combination of the nearest class mean classifier and the 1- nearest neighbor classifier based on deep features , but this method used a single classifier which cannot handle dynamic data. From the above results, It is clear that the result obtained by mask RCNN deep learning algorithm gives more accuracy values on comparing with the method which is used in th existing literature system. Measuring weight and location of individual bites using a sensor augmented smart plate (Gert mertes ., et al.,2018) in which the total amount of calories are estimated with known beforehand food present on the plate but this method has on error of 1.4% which is not acceptable .On comparing both the results .the Calories obtained by the mask RCNN algorithm gives more accurate values than the existing literature . this shows the mask RCNN provides Better Accuracy. (Lunsherg Ma, et al 2016)

“ A new deep learning based food recognition System for Dietary Assessment On an edge computing Service Infrastructure” in which they developed a practical depp learning based food recognition System for dietary Assessment within the edge computing service infrastructure. This experiment results on two challenging data sets using our system but this method makes use of linear programming making it difficult to process dynamic data. On comparing this, The result obtained from Mask RCNN algorithm gives better accuracy than the existing literature.

CHAPTER – 6

6.1 SUMMARY

In this project we obtained the input images are collected from various sources from the internet. These images are used as dataset and trained. The dataset is split into training and testing dataset in the ratio of 75: 25. An effective deep learning algorithm VGG-16 is used to train these dataset to obtain very high accuracy. For instance segmentation problem in machine learning mask RCNN algorithm is used for food items detection. Since object detection helps in recognition, detection and localization of multiple visual instances of objects in an image or video .Then object segmentation is done to divide pixels into larger components. From this the calorie is predicted by identifying the difference between those two food images using the python implementation. Thus the newly created deep learning method provides better accuracy.

6.2 CONCLUSION

This project is used to recognize the food item type and its showed calories level in that food. By this project people will be saved from those problem and also achieve a healthy environment. And it determined the all variety of foods with full of accuracy. Thus, this project helps in recognize the food item types and calories effectively.

REFERENCES

1. Bin Liu, Song Zhang , Yiqian Yang , Mingzhe Wang , Xiaohui Zhang , Jianxin Zhang , Wen Qi , Liang Yang, "An Automatic Personalized Internal Fixation Plate Modeling Framework for Minimally Invasive Curved Bone Fracture Surgery Based on Preregistration with Capsule Projection Model",Vol.00699,2019.
2. Chang Liu, Yu Cao, Senior Member, IEEE, Yan Luo, Member, IEEE, Guanling Chen, Member, IEEE, Vinod Vokkarane, Senior Member, IEEE, Yunsheng Ma, Songqing Chen, Member, IEEE, Peng Hou,"A New Deep Learning-based Food Recognition System for Dietary Assessment on An Edge Computing Service Infrastructure",IEEE Transactions on Services Computing,Vol.1939-1374,2016.
3. Eduardo Aguilar, Beatriz Remeseiro, Marc Bolanos, and Petia Radeva, Fellow, IAPR,"Grab, Pay and Eat: Semantic Food Detection for Smart Restaurants",2015
4. Eduardo Aguilar, Beatriz Remeseiro, Marc Bolanos, and Petia Radeva, ~ Fellow, IAPR,"Grab, Pay and Eat: Semantic Food Detection for Smart Restaurants",IEEE Transactions on Multimedia,Vol.1520-9210,2018.
5. Gert Mertes, Glenn Christiaensen, Hans Hallez, Sammy Verslype, Wei Chen and Bart Vanrumste,"Measuring weight and location of individual bites using a sensor augmented smart plate",Vol.5386-3646,2018.
6. Kiyoharu Aizawa, Member, IEEE, Yuto Maruyama, He Li, and Chamin Morikawa,"Food Balance Estimation by Using Personal Dietary Tendencies in a Multimedia Food Log",2015
7. Lixing Wang,"Design Optimization of Food Safety Monitoring System With Social Network Analysis",Ieee Transactions On Computational Social Systems,Vol.2329-924X,2018.
8. Luis Herranz, Ruihan Xu, Shuqiang Jiang,"Modeling restaurant context for food recognition",IEEE Transactions on Multimedia,Vol.1520-9210,2016.

9. Mukul Sarkar, Member, IEEE, Nitin Gupta and Maher Assaad, "Non-destructive food quality monitoring using phase information in time-resolved reflectance spectroscopy" TKDE, vol. 28 , 2020.
10. Ruihan Xu, Luis Herranz, Shuqiang Jiang, Senior Member, IEEE, Shuang Wang, Xinhang Song, Ramesh Jain, Fellow, IEEE,"Geolocalized Modeling for Dish Recognition",IEEE Transactions on Multimedia,Vol.1520-9210,2015.
11. Ruihan Xu, Luis Herranz, Shuqiang Jiang, Senior Member, IEEE, Shuang Wang, Xinhang Song, Ramesh Jain, Fellow, IEEE,"Geolocalized Modeling for Dish Recognition",2016
12. Shota Horiguchi, Member, IEEE, Sosuke Amano, Makoto Ogawa, and Kiyoharu Aizawa, Fellow, IEEE,Vol.00699,2019.
13. Shuqiang Jiang, Senior Member, IEEE, Weiqing Min, Member, IEEE, Linhu Liu and Zhengdong Luo,"Multi-Scale Multi-View Deep Feature Aggregation for Food Recognition",IEEE Transactions on Image Processing,Vol.1057-7149,2019.
14. S.Tanaka, Member, IEEE, T. Ohtani, Y. Narita, Y. Hatsukade, and S. Suzuki,"Development of Metallic Contaminant Detection System Using RF High-Tc SQUIDs for Food Inspection",2016
15. Vasileios Papapanagiotou, Student Member, IEEE, Christos Diou, Member, IEEE, Ioannis Ioakimidis, Per Sodersten, and Anastasios Delopoulos, " Member, IEEE,"Automatic analysis of food intake and meal microstructure based on continuous weight measurements",IEEE Journal of Biomedical and Health Informatics,Vol.2168-2194,2018

APPENDIX

```
# USAGE
# python detect_color.py --image 1.jfif
# import the necessary packages
import numpy as np
import argparse
import cv2
import json
from datetime import datetime

# construct the argument parse and parse the arguments
# ap = argparse.ArgumentParser()
# ap.add_argument("-i", "--image", help = "path to the image")
# args = vars(ap.parse_args())

# load the image
# image = cv2.imread(args["image"])
# while True:
#     # img = (input("enter the image : "))
#     # print(img)

image1 = (input("enter the 1st image : "))
image2 = (input("enter the 2nd image : "))

firstflag = 0
runflag = 0
firstimagevalue_apple = None
firstimagevalue_banana = None
secimagevalue_apple = None
```

```
secimagevalue_banana = None
```

```
while True:
```

```
    currenttime=datetime.now()
```

```
    if firstflag == 0 and runflag == 0:
```

```
        currentTime = currenttime.strftime("%H:%M:%S")
```

```
        with open('time.json', 'w') as f:
```

```
            json.dump(currentTime, f)
```

```
            image = cv2.imread(image1)
```

```
            firstflag = 1
```

```
            runflag = 1
```

```
            image = cv2.resize(image,(200,200))
```

```
            # define the list of boundaries
```

```
            boundaries = [
```

```
                # ([17, 15, 100], [50, 56, 200])
```

```
                # ([30, 146, 190], [70, 180, 250])
```

```
                ( [0, 50, 50], [10, 255, 255],"red"),
```

```
                ([20, 100, 100], [30, 255, 255],"yellow")
```

```
            ]
```

```
            lowerred = ([0, 50, 50])
```

```
            upperred = ([10, 255, 255])
```

```
            loweryel = ([20, 100, 100])
```

```
            upperyel = ([30, 255, 255])
```

```
            # loop over the boundaries
```

```

for (lower, upper,color) in boundaries:
# while True:
    print(color)
    img_hsv=cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    # create NumPy arrays from the boundaries
    # lower = np.array(lower, dtype = "uint8")
    # upper = np.array(upper, dtype = "uint8")
    # red
    lower = np.array(lower)
    upper = np.array(upper)

    # lower = np.array([0,50,50]) #example value
    # upper = np.array([10,255,255])

    # # YELLOW
    # lower = np.array([20, 100, 100])
    # upper = np.array([30, 255, 255])

    # find the colors within the specified boundaries and apply
    # the mask
    mask = cv2.inRange(img_hsv, lower, upper)

    output = cv2.bitwise_and(image, image, mask = mask)
    mean = output.mean(axis=0).mean()
    if color == "red":
        mean = mean*7
        # data = []
        print(mean)
        firstimagevalue_apple = mean
        # im = data.extend(mean)

```

```

        # with open('config.json', 'w') as json_file:
        #     json.dump(im, json_file)
elif color == "yellow":
    mean = mean * 5
    print(mean)
    firstimagevalue_banana = mean

# show the images
cv2.imshow("images", np.hstack([image, output]))
cv2.imshow("mea", output)
cv2.waitKey(200)
# runflag = 2

elif firstflag == 1:

    with open('time.json') as f:
        data = json.load(f)
        timediff = (currenttime - datetime.strptime(data, "%H:%M:%S")).seconds
        # print(timediff)
    if timediff >= 5 and timediff <= 9 and runflag == 1:
        runflag = 1
        image = cv2.imread(image2)
        image = cv2.resize(image, (200, 200))
        # define the list of boundaries
        boundaries = [
            # ([17, 15, 100], [50, 56, 200])

            # ([30, 146, 190], [70, 180, 250])
            ([0, 50, 50], [10, 255, 255], "red"),
            ([20, 100, 100], [30, 255, 255], "yellow")

```

```

]
lowerred = ([0, 50, 50])
upperred = ([10, 255, 255])
loweryel = ([20, 100, 100])
upperyel = ([30, 255, 255])

# loop over the boundaries
for (lower, upper,color) in boundaries:
# while True:
    print(color)
    img_hsv=cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    # create NumPy arrays from the boundaries
    # lower = np.array(lower, dtype = "uint8")
    # upper = np.array(upper, dtype = "uint8")
    # red
    lower = np.array(lower)
    upper = np.array(upper)

    # lower = np.array([0,50,50]) #example value
    # upper = np.array([10,255,255])

    # # YELLOW
    # lower = np.array([20, 100, 100])
    # upper = np.array([30, 255, 255])

    # find the colors within the specified boundaries and apply
    # the mask
    mask = cv2.inRange(img_hsv, lower, upper)

```

```

output = cv2.bitwise_and(image, image, mask = mask)
mean = output.mean(axis=0).mean()
if color == "red":
    mean = mean*7
    # data = []
    print(mean)
    secimagevalue_apple = mean
    # im = data.extend(mean)
    # with open('config.json', 'w') as json_file:
    #     json.dump(im, json_file)
elif color == "yellow":
    mean = mean * 5
    print(mean)
    secimagevalue_banana = mean

# show the images
cv2.imshow("images", np.hstack([image, output]))
cv2.imshow("mea",output)
cv2.waitKey(200)
runflag = 2

elif timediff == 9:
    runflag = 2
    finalcalorie_apple = firstimagevalue_apple - secimagevalue_apple
    finalcalorie_banana = firstimagevalue_banana - secimagevalue_banana
    if finalcalorie_banana < 3.0 :
        print("banana calories has been not fully intake ")
    if finalcalorie_apple < 3.0 :
        print("apple calories has been not fully intake ")

```

```

    if(finalcalorie_apple<1.0):
        finalcalorie_apple=0
        print("finalcalorie_apple",finalcalorie_apple)
    else:
        print("finalcalorie_apple",finalcalorie_apple)
if(finalcalorie_banana<1.0):
    finalcalorie_banana=0
    print("finalcalorie_banana",finalcalorie_banana)
else:
    print("finalcalorie_banana",finalcalorie_banana)
break

# if runflag == 1:
# image = cv2.resize(image,(200,200))
# # define the list of boundaries
# boundaries = [
#     # ([17, 15, 100], [50, 56, 200])
#     # ([30, 146, 190], [70, 180, 250])
#     ( [0, 50, 50], [10, 255, 255],"red"),
#     ([20, 100, 100], [30, 255, 255],"yellow")
# ]
# lowerred = ([0, 50, 50])
# upperred = ([10, 255, 255])
# loweryel = ([20, 100, 100])
# upperyel = ([30, 255, 255])

# # loop over the boundaries
# for (lower, upper,color) in boundaries:
# # while True:
#     print(color)
#     img_hsv=cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

```

```

# # create NumPy arrays from the boundaries
# # lower = np.array(lower, dtype = "uint8")
# # upper = np.array(upper, dtype = "uint8")
# # red
# lower = np.array(lower)
# upper = np.array(upper)

# # lower = np.array([0,50,50]) #example value
# # upper = np.array([10,255,255])

# # # YELLOW
# # lower = np.array([20, 100, 100])
# # upper = np.array([30, 255, 255])

# # find the colors within the specified boundaries and apply
# # the mask
# mask = cv2.inRange(img_hsv, lower, upper)

# output = cv2.bitwise_and(image, image, mask = mask)
# mean = output.mean(axis=0).mean()
# if color == "red":
#     mean = mean*7
#     # data = []
#     print(mean)
#     # im = data.extend(mean)
#     # with open('config.json', 'w') as json_file:
#     #     json.dump(im, json_file)
# elif color == "yellow":
#     mean = mean * 5
#     print(mean)

```

```
#    # show the images
#    cv2.imshow("images", np.hstack([image, output]))
#    cv2.imshow("mea", output)
#    cv2.waitKey(200)
#    runflag = 0
#    0xFF == ord("q"):
#    if img == "q":
#    break
```