A Report
on

Spots Nutrition Tracking

by

Piyush Chanduka 2021A4PS1554P

of
Study-Oriented Project

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE
PILANI, PILANI CAMPUS

ACKNOWLEDGMENT

I express profound gratitude to my project guide and supervisor, Professor M.S. Dasgupta, for introducing me to this rewarding endeavor. His unwavering guidance and support motivated me to exceed expectations throughout the project, cultivating a deep interest in the subject. I am incredibly thankful for his invaluable inputs, which were crucial to the project's success.

I would also like to extend my sincere thanks to Prof. Srikanta Routroy, Head of the Department of Mechanical Engineering at BITS Pilani – Pilani Campus, and Prof. Sachin Belgamwar, Associate Dean of Academic Undergraduate Studies Division at BITS Pilani – Pilani Campus. Their provision of project courses for undergraduate students as part of the academic curriculum has exposed students to research early on, enriching their academic experience.

Introduction -

Athletes and health-conscious individuals prioritize maintaining a healthy lifestyle to optimize their performance and overall well-being. A key aspect of this lifestyle involves managing body weight and nutrition. Body Mass Index (BMI) remains a relevant metric is often considered indicative of obesity, which can impact an athlete's performance and increase the risk of various health issues.

To address the challenges associated with maintaining a healthy weight, athletes often focus on a balanced approach to diet and exercise. The primary cause of weight-related issues is still the imbalance between calorie intake and energy expenditure. In the context of sports and fitness, maintaining an appropriate body weight is crucial for optimal performance and injury prevention.

For athletes and health-conscious individuals, effective weight management involves strategic dietary choices and precise calorie monitoring. Computer vision-based measurement methods have gained popularity in recent years, providing innovative solutions to estimate calorie intake. These methods leverage deep learning, an emerging approach in machine learning that explores multiple levels of representation.

In this project, the focus shifts to the application of deep learning for food classification and recognition within the context of sports nutrition. The primary goals include:

1. Proposing the first recognition system for food.

2. Proposing a complete and effective calorie estimation method.

By tailoring the focus to sports nutrition, this project seeks to contribute valuable insights and tools that empower athletes and health-conscious individuals to make informed dietary choices, ultimately supporting their journey towards peak performance and well-being.

Problem –

Given a set of food images with desired specifications, estimate the calories and provide a personalized analysis.

Objectives:

1. To detect food type by using Convolutional Neural Network (CNN)
2. To estimate food weight and calories of food
3. To provide a personalized application for each user
4. To estimate the calories of the leftovers (overlapping food items).

Dataset –

For training of the model, I used the FOODD dataset. The dataset contains images taken with different cameras, illuminations, and angles. Having a wide variety of food and foods gives a better and more reliable dataset in order to increase the accuracy of calorie food measurement systems. In the dataset, the images are divided into 6 categories considering the capturing device, background, and lighting condition:

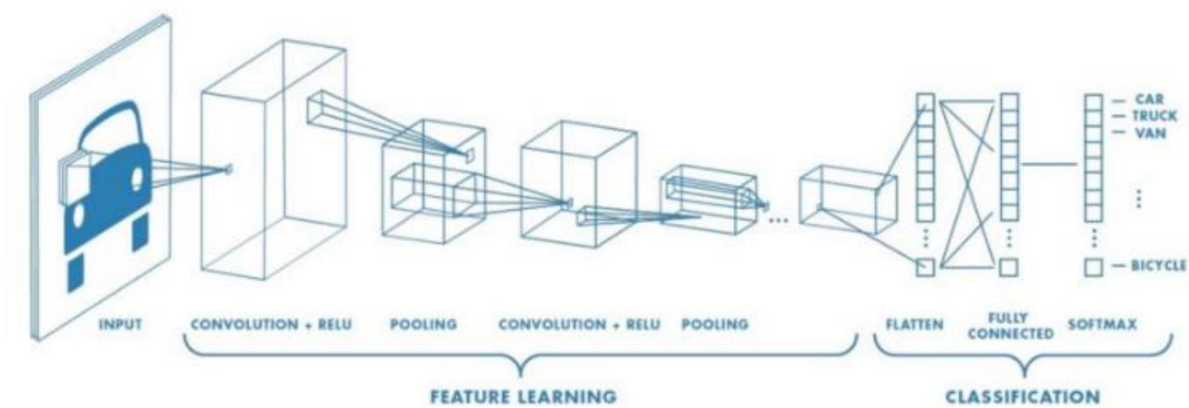
Samsung-S4 Light Environment
Samsung-S4 Dark Environment
IOS-4 Light Environment
IOS-4 Dark Environment
CanonSD1400 Light Environment
CanonSD1400 Dark Environment

In this project I used 7 food items:

Foods	Density (g/cm^3)	Calorie (kcal/g)	Label	Shape
Apple	0.609	0.52	1	Sphere
Banana	0.94	0.89	2	Cylinder
Carrot	0.641	0.41	3	Cylinder
Cucumber	0.641	0.16	4	Cylinder
Onion	0.513	0.40	5	Sphere
Orange	0.482	0.47	6	Sphere
Tomato	0.481	0.18	7	Sphere

Food Recognition Model –

Food Recognition deals with recognition of food item when given an image. For this problem I used Convolutional Neural Network (CNN). The Architecture of CNN given



In this study, I implemented a neural network architecture comprising 5 convolutional layers with ReLU activations, dropout functionality, and SoftMax layers. The process of refining the model on the dataset was completed within approximately 2 hours, with a single Windows 10 Pro CPU equipped with 8GB of memory. The training phase involved a dataset containing 100 images for each food category, each image sized at 300*300 pixels. The optimization process employed the Adam optimizer and a categorical cross-entropy loss function, with a learning rate set at 0.0001. This approach facilitated the calculation and minimization of loss while enhancing the overall accuracy of the model.

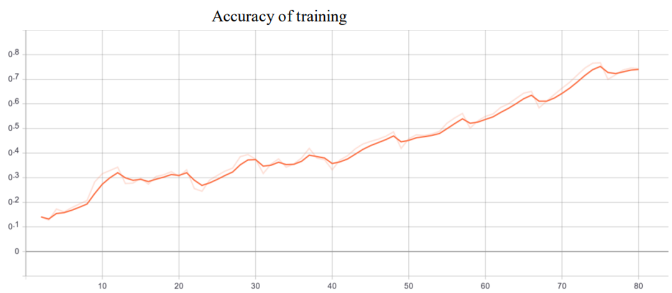
The model summary:

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 300, 300, 32)      2432
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 150, 150, 32)      0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 150, 150, 64)      51264
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 50, 50, 64)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 50, 50, 128)       204928
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 17, 17, 128)       0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 17, 17, 64)        204864
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 5, 5, 64)          0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 5, 5, 32)          51232
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 2, 2, 32)          0
_________________________________________________________________
flatten_1 (Flatten)          (None, 128)               0
_________________________________________________________________
dense_1 (Dense)              (None, 1024)              132096
_________________________________________________________________
dropout_1 (Dropout)          (None, 1024)              0
_________________________________________________________________
dense_2 (Dense)              (None, 7)                 7175
=================================================================
Total params: 653,991
Trainable params: 653,991
Non-trainable params: 0
```

The training of the model -

```
Training Step: 79  | total loss: 1.33835 | time: 60.100s
| Adam | epoch: 010 | loss: 1.33835 - acc: 0.7306 -- iter: 448/488
Training Step: 80  | total loss: 1.23289 | time: 71.804s
| Adam | epoch: 010 | loss: 1.23289 - acc: 0.7470 | val_loss: 0.42227 - val_acc: 0.8365 -- iter: 488/488
--
INFO:tensorflow:C:\Users\user\Desktop\jupyter\model\Fruits_dectector-0.001-5conv-basic.model is not in all_model_checkpoint_paths. Manually adding it.
Model Save At model\Fruits_dectector-0.001-5conv-basic.model
```

Accuracy of this model is **86.06%** with total loss **1.11.**



We know that, our thumb size is approximately 5*2.3cm is a skin multiplier (say). We calculate pixel to cm multiplier by using Maximum pixel height.

$$pix\_to\_cm\_multiplier = \frac{5}{pix\_height}$$

We have 3 factors from image segmentation

1. Foods pixel area
2. Skin pixel area
3. Actual skin area (skin multiplier)

From this factors food estimated area is given below:

$$Estimated\ Food\ Area = \frac{Foods\ Pixel\ Area * Actual\ Skin\ Area}{Skin\_Pixel\_Areat}$$

We have two types of shape of foods

1. Sphere - like apple, orange, tomato, onion
2. Cylinder – like banana, cucumber, carrot

Volume estimation for Sphere:

$$Estimated\_Radius = \sqrt{\frac{Estimated\_Food\_Area}{\pi}}$$

$$Estimated\_Volume = \frac{4}{3}\pi(Estimated\_Radius)^3$$

Volume Estimation for Cylinder:

$$Estimated\ Height = Pixel\ Height * Pix\_To\_Cm\_Multiplier$$

$$Estimated\ Radius = \frac{Estimated\ Food\ Area}{2 * Estimated\ Height}$$

$$Estimated\ Volume = \pi * Estimated\ Height * Estimated\ Radius2$$

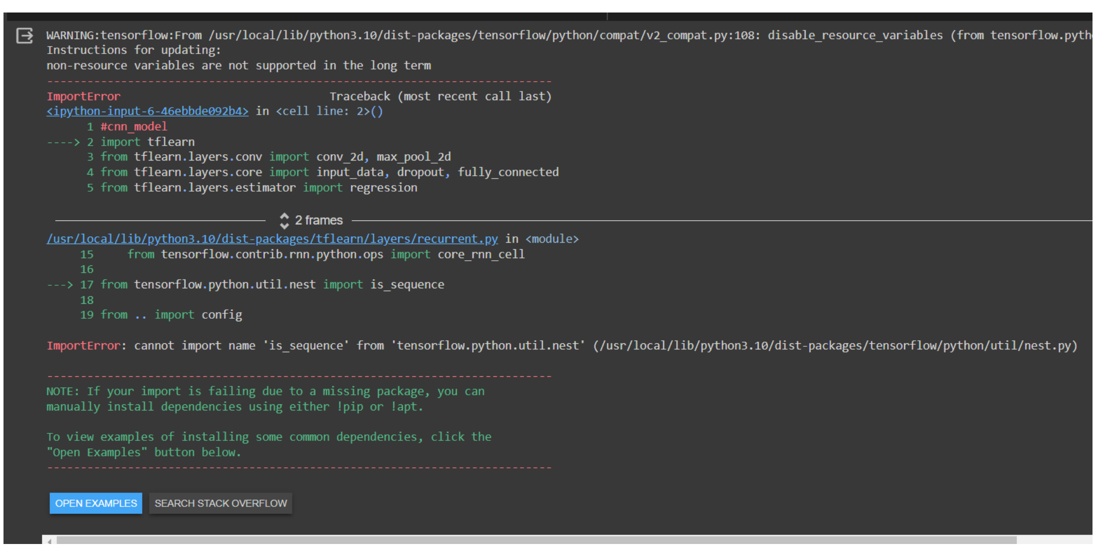
Weight And Calories Estimation of Food:

$$Estimated\ Weight = Actual\ Density * Estimated\ Volume$$

$$Estimated\ Calories = \frac{Estimated\ Weight * Calories\ per\ 100gm}{100}$$

Code –

Previous code has much errors as it was running only for one time, showing warnings and import error.



```
WARNING:tensorflow:From /usr/local/lib/python3.10/dist-packages/tensorflow/python/compat/v2_compat.py:108: disable_resource_variables (from tensorflow.pyth
Instructions for updating:
non-resource variables are not supported in the long term
---------------------------------------------------------------------------
ImportError                               Traceback (most recent call last)
<ipython-input-6-46ebbde092b4> in <cell line: 2>()
      1 #cnn_model
----> 2 import tflearn
      3 from tflearn.layers.conv import conv_2d, max_pool_2d
      4 from tflearn.layers.core import input_data, dropout, fully_connected
      5 from tflearn.layers.estimator import regression

2 frames
/usr/local/lib/python3.10/dist-packages/tflearn/layers/recurrent.py in <module>
     15     from tensorflow.contrib.rnn.python.ops import core_rnn_cell
     16
---> 17 from tensorflow.python.util.nest import is_sequence
     18
     19 from .. import config

ImportError: cannot import name 'is_sequence' from 'tensorflow.python.util.nest' (/usr/local/lib/python3.10/dist-packages/tensorflow/python/util/nest.py)

---------------------------------------------------------------------------
NOTE: If your import is failing due to a missing package, you can
manually install dependencies using either !pip or !apt.

To view examples of installing some common dependencies, click the
"Open Examples" button below.
---------------------------------------------------------------------------

OPEN EXAMPLES   SEARCH STACK OVERFLOW
```

The updated code is free from errors and can be run multiple times. The code link can be found here –

https://colab.research.google.com/drive/1ypUrp91iyDbqOO7hNWUMzvKzBieJ4Fm4

The first and second objectives are completed. For fourth objective there are no much research papers about the overlapping problems. The third objective's code is given in the code link but it needs more dataset with labelled sample names.

Limitations –

The personalized user interface is not fully implemented as it needs labelled dataset.

```
[2]: !pip3 install opencv-python
     !pip install tflearn

Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: opencv-python in c:\users\user\appdata\roaming\python\python310\site-packages (4.8.1.78)
Requirement already satisfied: numpy>=1.17.3 in c:\users\user\appdata\roaming\python\python310\site-packages (from opencv-python) (1.23.5)
WARNING: You are using pip version 21.3.1; however, version 23.3.2 is available.
You should consider upgrading via the 'C:\Program Files\Python310\python.exe -m pip install --upgrade pip' command.
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: tflearn in c:\users\user\appdata\roaming\python\python310\site-packages (0.5.0)
Requirement already satisfied: numpy in c:\users\user\appdata\roaming\python\python310\site-packages (from tflearn) (1.23.5)
Requirement already satisfied: six in c:\users\user\appdata\roaming\python\python310\site-packages (from tflearn) (1.16.0)
Requirement already satisfied: Pillow in c:\users\user\appdata\roaming\python\python310\site-packages (from tflearn) (9.1.0)
WARNING: You are using pip version 21.3.1; however, version 23.3.2 is available.
You should consider upgrading via the 'C:\Program Files\Python310\python.exe -m pip install --upgrade pip' command.
```

```
[3]: #image_segment
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_BINARY,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)
```

```python
# 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.THRESH_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)
```

```python
#find area of fruit
img_th = cv2.adaptiveThreshold(mask_fruit2,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRESH_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_BINARY,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
```

```
#caleries
import cv2
import numpy as np
#density - gram / cm^3
density_dict = { 1:0.609, 2:0.94, 3:0.641,  4:0.641,5:0.513, 6:0.482,7:0.481}
#kcal
calorie_dict = { 1:52, 2:89,  3:41,4:16,5:40,6:47,7:18 }
#skin of photo to real multiplier
skin_multiplier = 5*2.3

def getCalorie(label, volume): #volume in cm^3
    calorie = calorie_dict[int(label)]
    density = density_dict[int(label)]
    mass = volume*density*1.0
    calorie_tot = (calorie/100.0)*mass
    return mass, calorie_tot, calorie #calorie per 100 grams

def getVolume(label, area, skin_area, pix_to_cm_multiplier, fruit_contour):
    area_fruit = (area/skin_area)*skin_multiplier #area in cm^2
    label = int(label)
    volume = 100
    if label == 1 or label == 5 or label == 7 or label == 6 : #sphere-apple,tomato,orange,kiwi,onion
        radius = np.sqrt(area_fruit/np.pi)
        volume = (4/3)*np.pi*radius*radius*radius
        #print (area_fruit, radius, volume, skin_area)

    if label == 2 or label == 4 or (label == 3 and area_fruit > 30): #cylinder like banana, cucumber, carrot
        fruit_rect = cv2.minAreaRect(fruit_contour)
        height = max(fruit_rect[1])*pix_to_cm_multiplier
        radius = area_fruit/(2.0*height)
        volume = np.pi*radius*radius*height

    if (label==4 and area_fruit < 30) : # carrot
        volume = area_fruit*0.5 #assuming width = 0.5 cm

    return volume

def calories(result,img):
    img_path =img # "C:/Users/M Sc-2/Desktop/dataset/FooD/"+str(j)+"_"+str(i)+".jpg"
    fruit_areas,final_f,areaod,skin_areas, fruit_contours, pix_cm = getAreaOfFood(img_path)
    volume = getVolume(result, fruit_areas, skin_areas, pix_cm, fruit_contours)
    mass, cal, cal_100 = getCalorie(result, volume)
    fruit_volumes=volume
    fruit_calories=cal
    fruit_calories_100grams=cal_100
    fruit_mass=mass
    #print("\nfruit_volumes",fruit_volumes,"\nfruit_calories",fruit_calories,"\nruit_calories_100grams",fruit_calories_100grams,"\nfruit_mass",fruit_mas
    return fruit_calories
```

```
!pip install tensorflow==2.12
```

```python
#cnn_model
import tflearn
from tflearn.layers.conv import conv_2d, max_pool_2d
from tflearn.layers.core import input_data, dropout, fully_connected
from tflearn.layers.estimator import regression
import tensorflow as tf

def get_model(IMG_SIZE,no_of_fruits,LR):
    a = tf.constant(1)
    with tf.Session() as sess:
        tf.reset_default_graph()
    # try:
    #   tf.reset_default_graph()
    # except:
    #   print("tensorflow")
    convnet = input_data(shape=[None, IMG_SIZE, IMG_SIZE, 3], name='input')

    convnet = conv_2d(convnet, 32, 5, activation='relu')

    convnet = max_pool_2d(convnet, 5)

    convnet = conv_2d(convnet, 64, 5, activation='relu')

    convnet = max_pool_2d(convnet, 5)

    convnet = conv_2d(convnet, 128, 5, activation='relu')
    convnet = max_pool_2d(convnet, 5)

    convnet = conv_2d(convnet, 64, 5, activation='relu')
    convnet = max_pool_2d(convnet, 5)


    convnet = conv_2d(convnet, 32, 5, activation='relu')
    convnet = max_pool_2d(convnet, 5)

    convnet = fully_connected(convnet, 1024, activation='relu')
    convnet = dropout(convnet, 0.8)

    convnet = fully_connected(convnet, no_of_fruits, activation='softmax')
    convnet = regression(convnet, optimizer='adam', learning_rate=LR, loss='categorical_crossentropy', name='targets')

    model = tflearn.DNN(convnet, tensorboard_dir='log')

    return model
```

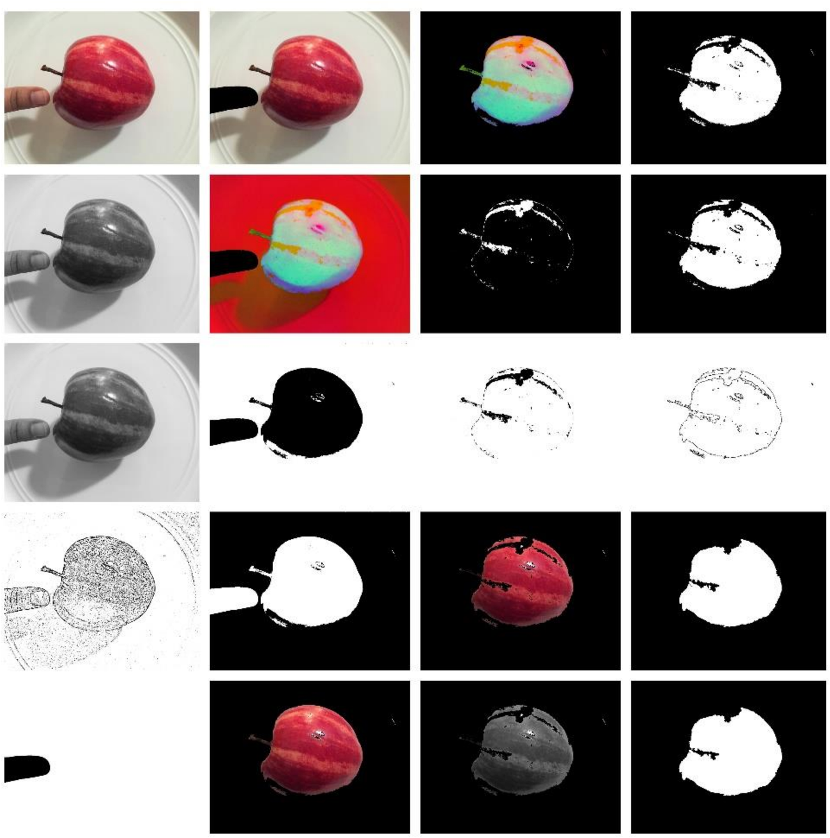
```python
import glob
import cv2

path = r'./dataset'
IMG_SIZE = 400
LR = 1e-3
#Fruits_dectector-{}-{}.model
MODEL_NAME = 'Fruits_dectector-{}-{}.model'.format(LR, '5conv-basic')
no_of_fruits=7
percentage=0.3
no_of_images=100

def create_train_data(path):
    training_data = []
    folders=os.listdir(path)[0:no_of_fruits]
    for i in range(len(folders)):
        label = [0 for i in range(no_of_fruits)]
        label[i] = 1
        print(folders[i])
        k=0
        for j in glob.glob(path+"\\"+folders[i]+"\\*.jpg"):
            if(k==no_of_images):
                break
            k=k+1
            img = cv2.imread(j)
            img = cv2.resize(img, (IMG_SIZE,IMG_SIZE))
            training_data.append([np.array(img),np.array(label)])
    np.save('training_{}_{}_{}.npz'.format(no_of_fruits,no_of_images,IMG_SIZE),training_data)
    shuffle(training_data)
    return training_data,folders

training_data,labels=create_train_data(path)
# training_data=np.load('training_{}_{}_{}.npz'.format(no_of_fruits,no_of_images,IMG_SIZE))
size=int(len(training_data)*percentage)
train = training_data[:-size]
test=training_data[-size:]

X = np.array([i[0] for i in train]).reshape(-1,IMG_SIZE,IMG_SIZE,3)
Y = [i[1] for i in train]

test_x = np.array([i[0] for i in test]).reshape(-1,IMG_SIZE,IMG_SIZE,3)
test_y = [i[1] for i in test]

model=get_model(IMG_SIZE,no_of_fruits,LR)

model.fit({'input': X}, {'targets': Y}, n_epoch=10, validation_set=({'input': test_x}, {'targets': test_y}),
    snapshot_step=500, show_metric=True, run_id=MODEL_NAME)

model_save_at=os.path.join("model",MODEL_NAME)
model.save(model_save_at)
print("Model Save At",model_save_at)
```

```python
import os
import cv2
import numpy as np
# import tflearn
# import tensorflow as tf
# from cnn import get_model
from calorie import calories

IMG_SIZE = 400
LR = 1e-3
no_of_fruits=7

MODEL_NAME = 'Fruits_dectector-{}-{}.model'.format(LR, '5conv-basic')

model_save_at=os.path.join("model",MODEL_NAME)
model=get_model(IMG_SIZE,no_of_fruits,LR)

model.load(model_save_at)
labels=list(np.load('labels.npy'))

test_images = ['test_image1.JPG']

for test_data in test_images:
    img=cv2.imread(test_data)
    img1=cv2.resize(img,(IMG_SIZE,IMG_SIZE))
    model_out=model.predict([img1])
    result=np.argmax(model_out)
    name=labels[result]
    cal=round(calories(result+1,img),2)
    import matplotlib.pyplot as plt
    plt.imshow(img)
    plt.title('{}({}kcal)'.format(name,cal))
    plt.axis('off')
    plt.show()
```

```
INFO:tensorflow:Restoring parameters from C:\Users\user\Desktop\jupyter\model\Fruits_dectector-0.001-5conv-basic.model
folder exist for images at  C:\Users\user\Desktop\jupyter\images
```

Apple(211.35kcal)

The following pictures are analyzed for calculating the calories of the sample apple -



References –

1. https://viso.ai/deep-learning/mask-r-cnn/

2. https://github.com/vinayaksable2399/Food-Calories-Estimation-Using-Image-Processing

3. P.Pouladzadeh, S.Shirmohammadi, and R.Almaghrabi, "Measuring Calorie and Nutrition from Food Image", IEEE Transactions on Instrumentation & Measurement, Vol.63, No.8, p.p. 1947 – 1956, August 2014.

4. Parisa Pouladzadeh, Abdulsalam Yassine, and Shervin Shirmohammadi, "Foodd: An image-based food detection dataset for calorie measurement," in InternationalConferenceonMultimediaAssistedDietaryManagement, 2015

5. Meghana M Reddy, "Calorie-estimation-from-food-images-opencv", Git repo , May 2016