Shivaji University, Kolhapur

Department of Statistics

A

Project Report

On

“Foods Calories Estimation Using Image Processing”

Submitted

By

Mr. Sable Vinayak Machhindra

MSc.II (ASI)

Under the Guidance of

Dr. S.B.Mahadik

2018-2019

ACKNOWLEDGEMENT

We wish to thank all teachers; non-teaching staff of department we would like express our profound gratitude to Dr. S. B. Mahadik sir, for valuable guidance in completion of this project.

We would like to thank Dr. H.V. Kulkarni, Head of Department, for providing us necessary facilities, their support, suggestions and guidance for this project.

We are also thankful to Dr. D. T. Shirke, Dr. D. N. Kashid, Mr. S. D. Powar and Mr. S.V. Rajguru for their timely suggestion and Encouragements.

Finally, we would like to thank HOD, teachers, non-teaching staff and students of Departments and also thanks to Dr. S.B. Mahadik sir for their support and motivations.

Contents

INTRODUCTION

People are generally considered obese when their Body Mass Index (BMI) is over 30 kg/m2. High BMI is associated with the increased risk of diseases, such as heart disease. Unfortunately, more and more people will meet criteria for obesity. The main cause of obesity is the imbalance between the amount of food in-take and energy consumed by the individuals.

Obesity treatment requires the patients to eat healthy food and decrease the amount of daily calorie intake, which needs patients to calculate and record calories from foods every day. While computer vision-based measurement methods were introduced to estimate calories from images directly according to the calibration object and foods information, obese patients have benefited a lot from these methods.

In recent years, there are a lot of methods based on computer vision proposed to estimate calories. Among these methods, the accuracy of estimation result is determined by two main factors: object detection algorithm and volume estimation method.

In this project, I study the application of deep learning for food classification and recognition. Deep learning is an emerging approach from machine learning, and has been proposed in recent years to move machine learning systems towards the discovery of multiple levels of representation.

The main contributions of this project are listed as follows:

1. Proposing the first recognition system for food.
2. Proposing a complete and effective calorie estimation method.

Motivation:

Computer vision has been introduced to estimate calories from food images. But current food image datasets don’t contain volume and mass records of foods, which leads to an incomplete calorie estimation.

Current obesity treatment techniques require the patient to record all food intakes per day. In most of the cases, unfortunately patients have troubles in estimating the amount of food intake because of the self-denial of the problem, lack of nutritional information, the manual process of writing down this information (which is tiresome and can be forgotten), and other reasons.

Problem:

The problem can be simply stated as, given a set of food images with calibration object thumb with the food name and an unlabeled set of food images from the same group of food, identify food andestimate food volume and calories intake.

Objectives:

1. To detect food type by using Convolutional Neural Network (CNN)
2. To estimate food weight and calories of food

Data collection:

For this project I used two datasets:

1. FOODD
2. ECUST Food Dataset (ECUSTFD)

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. Sample data given in figure 1.In the dataset, the images are divided into 6 categories considering the capturing device, background, and lighting condition

Table 1: Different Food Categories

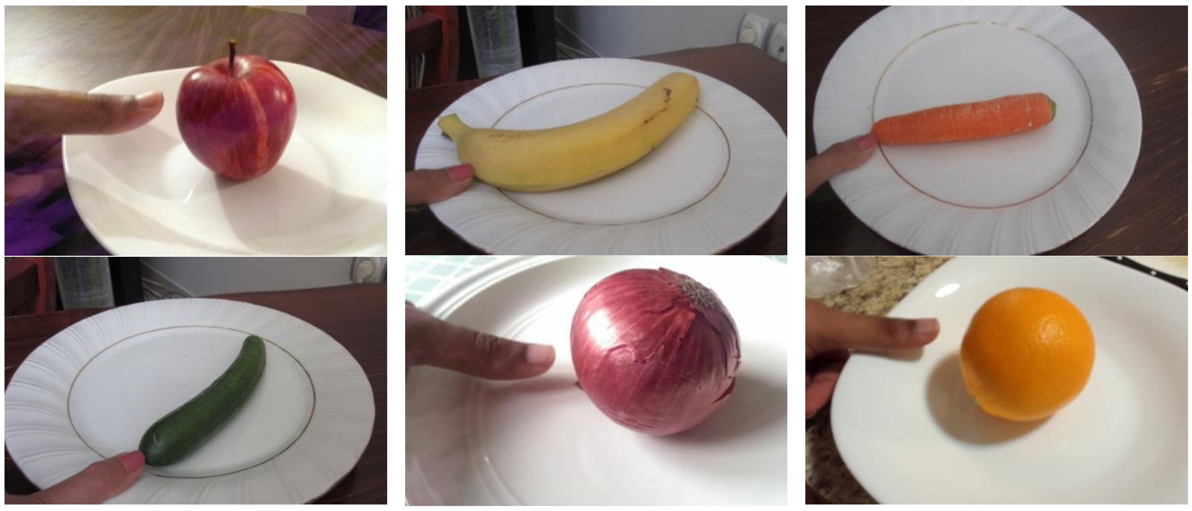
Category	Camera	Lighting
1	Samsung-S4	Light Environment
2	Samsung-S4	Dark Environment
3	IOS-4	Light Environment
4	IOS-4	Dark Environment
5	CanonSD1400	Light Environment
6	CanonSD1400	Dark Environment

In this project I used 7 food items like apple, banana, carrot, cucumber, onion, orange and tomato which details given in table 2

Table 2: Food Nutritional Information

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**

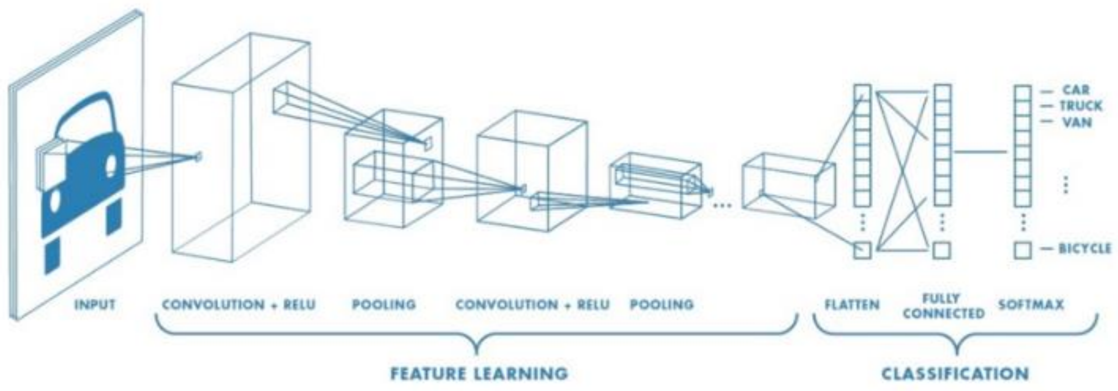
Figure 1 Sample food images in dataset



Recognition method

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 below figure 2

Figure 2: Architecture of CNN



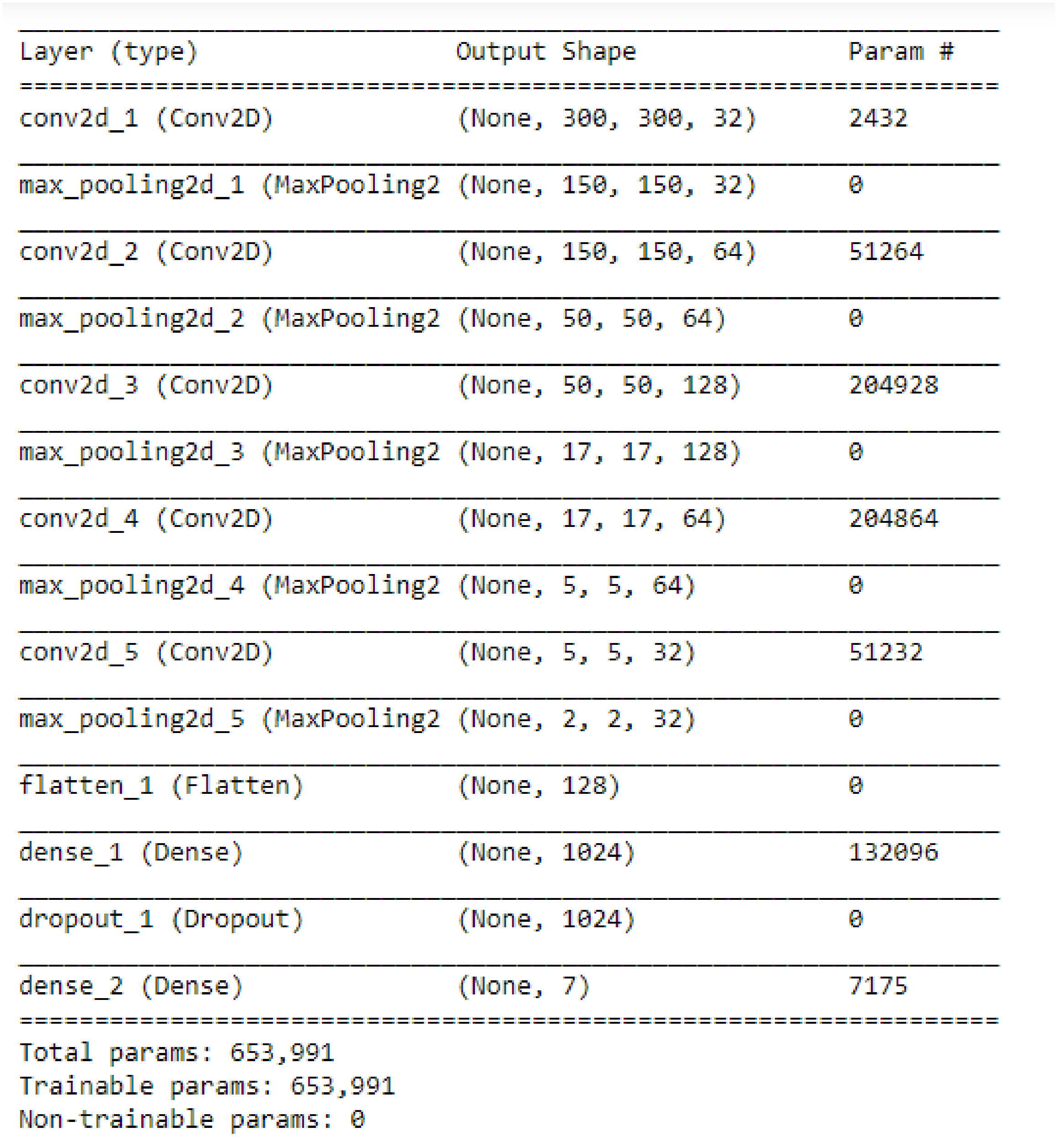
For this project I used 5 convolution layers with ReLU activations,dropout, and softmax layers. Fine tuning the model on our dataset took about 2 hours on a single Windows 10 Pro CPU with 4GB of memory.For this training I used 100 imagesof each food with 300*300 image size. All this work done in Python 3.7.1 with Anaconda Distribution 4.6.11

Also used Adam optimizer and categorical cross entropy loss function with learning rate 0.0001 to calculate and minimize loss as well as optimize model accuracy.

Model Summary of CNN

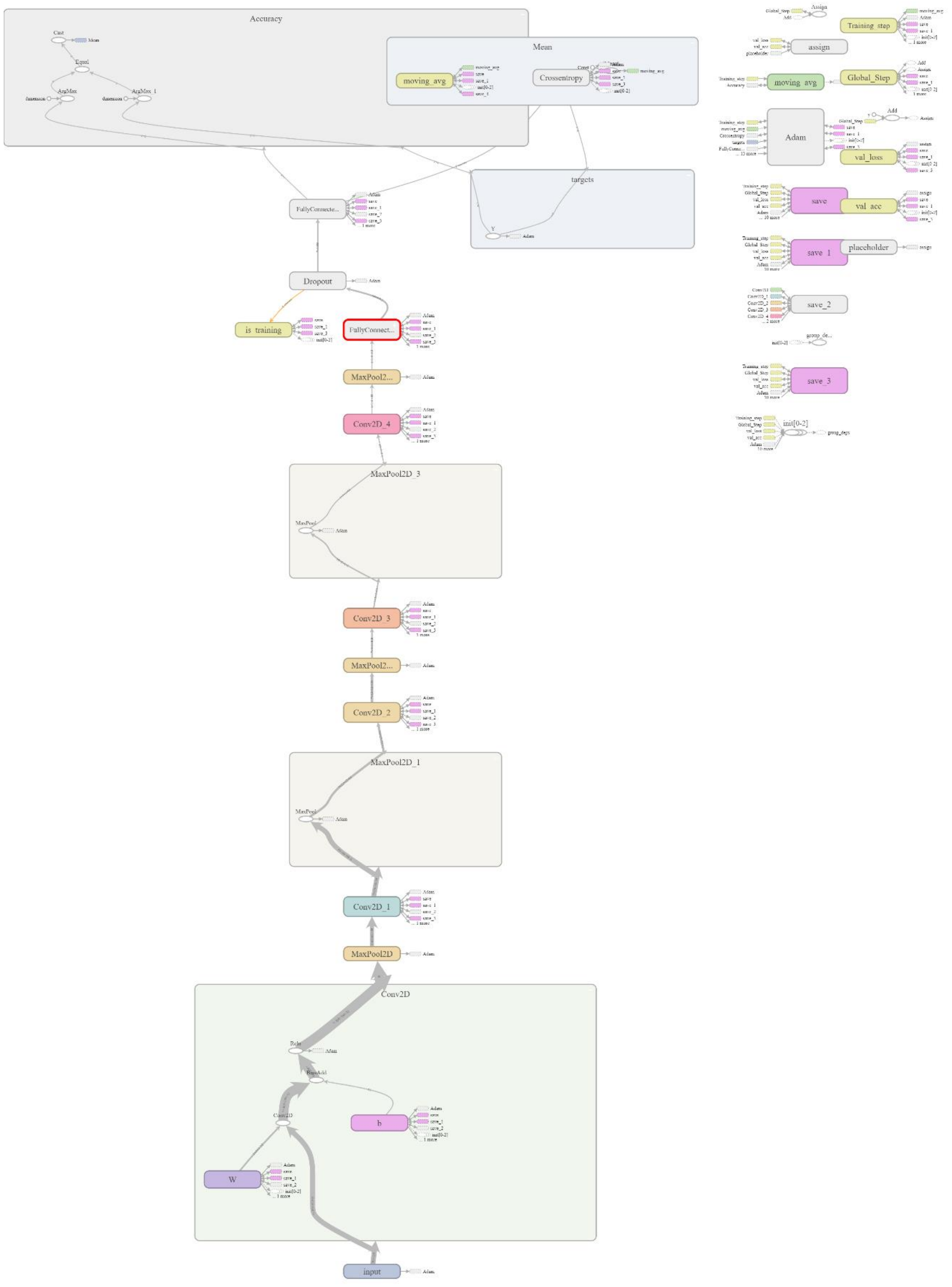
Model summary of CNN show in below figure 3

Figure 3 CNN Model Summaries

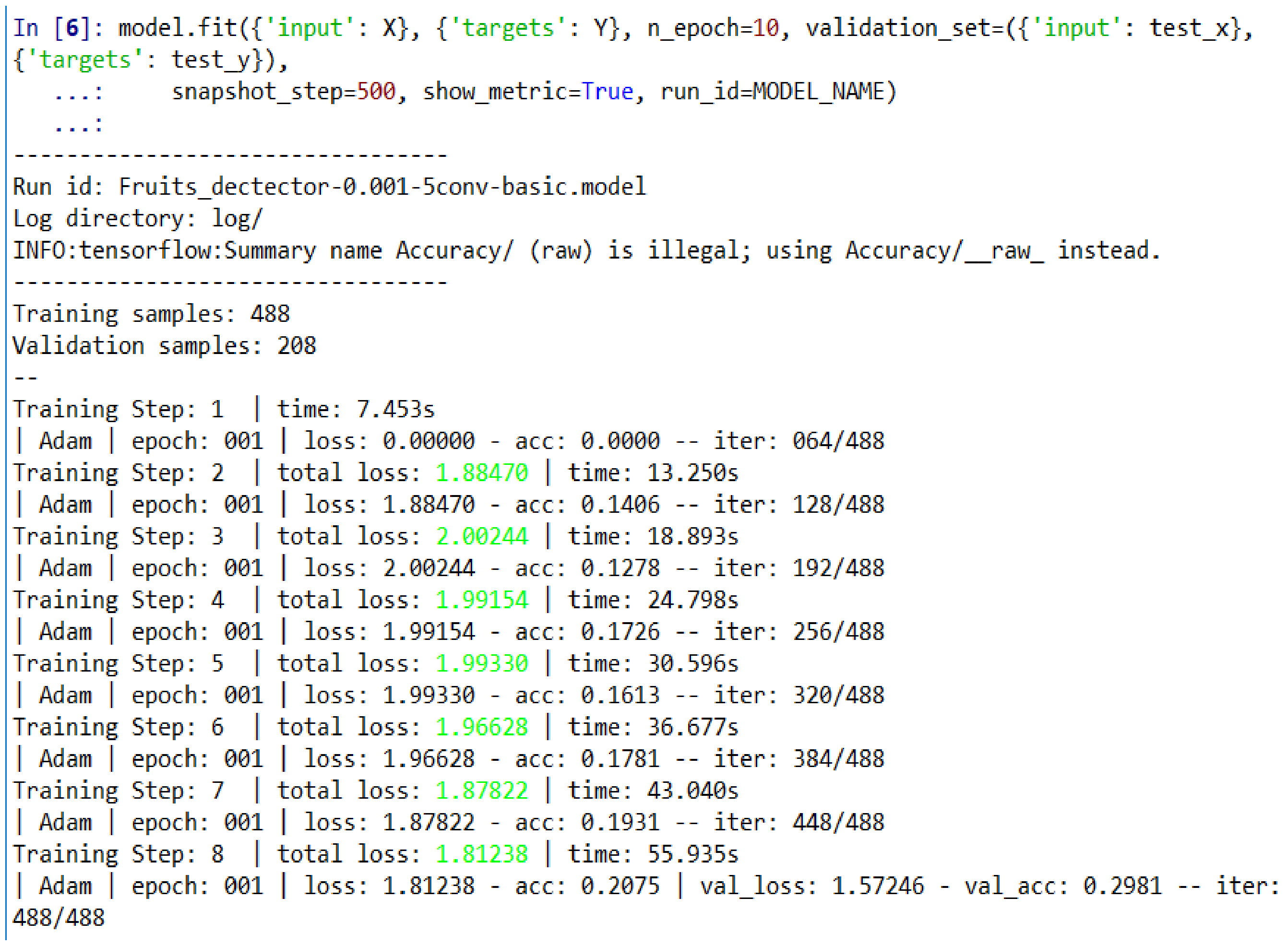


```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 300, 300, 32)      2432
_________________________________________________________________
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
```

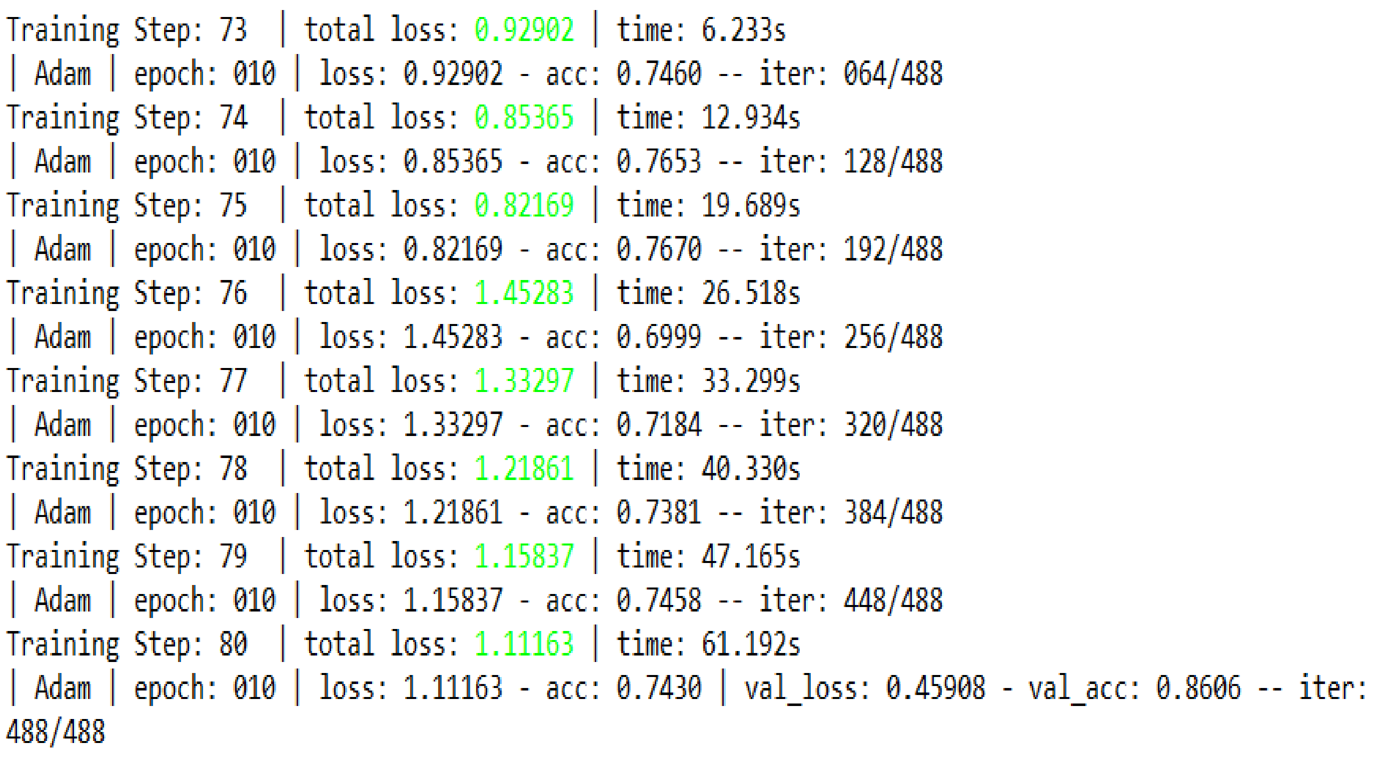
Model Summary by Graph



Training of Model



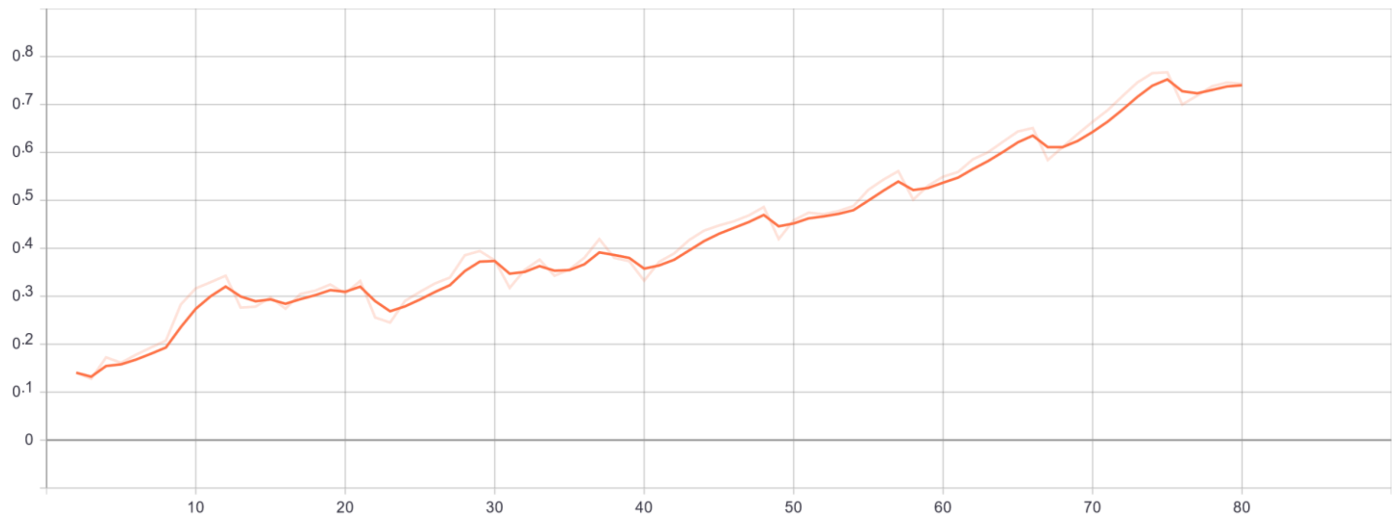
```
In [6]: model.fit({'input': X}, {'targets': Y}, n_epoch=10, validation_set=({'input': test_x},
{'targets': test_y}),
   ...:     snapshot_step=500, show_metric=True, run_id=MODEL_NAME)
   ...:
---------------------------------
Run id: Fruits_dectector-0.001-5conv-basic.model
Log directory: log/
INFO:tensorflow:Summary name Accuracy/ (raw) is illegal; using Accuracy/__raw_ instead.
---------------------------------
Training samples: 488
Validation samples: 208
--
Training Step: 1  | time: 7.453s
| Adam | epoch: 001 | loss: 0.00000 - acc: 0.0000 -- iter: 064/488
Training Step: 2  | total loss: 1.88470 | time: 13.250s
| Adam | epoch: 001 | loss: 1.88470 - acc: 0.1406 -- iter: 128/488
Training Step: 3  | total loss: 2.00244 | time: 18.893s
| Adam | epoch: 001 | loss: 2.00244 - acc: 0.1278 -- iter: 192/488
Training Step: 4  | total loss: 1.99154 | time: 24.798s
| Adam | epoch: 001 | loss: 1.99154 - acc: 0.1726 -- iter: 256/488
Training Step: 5  | total loss: 1.99330 | time: 30.596s
| Adam | epoch: 001 | loss: 1.99330 - acc: 0.1613 -- iter: 320/488
Training Step: 6  | total loss: 1.96628 | time: 36.677s
| Adam | epoch: 001 | loss: 1.96628 - acc: 0.1781 -- iter: 384/488
Training Step: 7  | total loss: 1.87822 | time: 43.040s
| Adam | epoch: 001 | loss: 1.87822 - acc: 0.1931 -- iter: 448/488
Training Step: 8  | total loss: 1.81238 | time: 55.935s
| Adam | epoch: 001 | loss: 1.81238 - acc: 0.2075 | val_loss: 1.57246 - val_acc: 0.2981 -- iter:
488/488
```



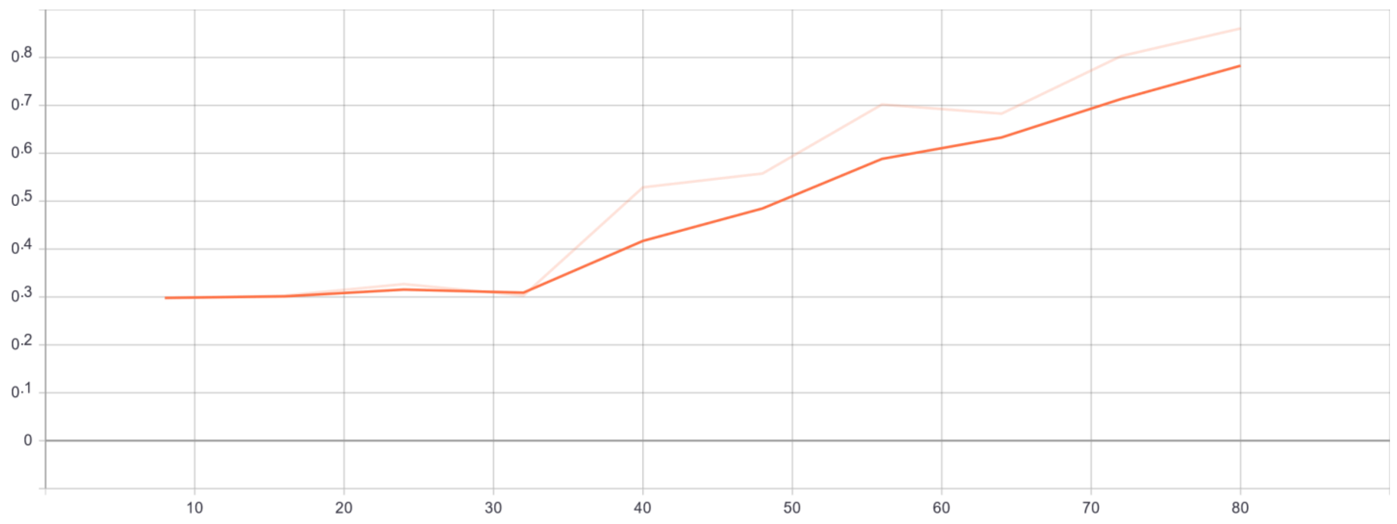
```
Training Step: 73  | total loss: 0.92902 | time: 6.233s
| Adam | epoch: 010 | loss: 0.92902 - acc: 0.7460 -- iter: 064/488
Training Step: 74  | total loss: 0.85365 | time: 12.934s
| Adam | epoch: 010 | loss: 0.85365 - acc: 0.7653 -- iter: 128/488
Training Step: 75  | total loss: 0.82169 | time: 19.689s
| Adam | epoch: 010 | loss: 0.82169 - acc: 0.7670 -- iter: 192/488
Training Step: 76  | total loss: 1.45283 | time: 26.518s
| Adam | epoch: 010 | loss: 1.45283 - acc: 0.6999 -- iter: 256/488
Training Step: 77  | total loss: 1.33297 | time: 33.299s
| Adam | epoch: 010 | loss: 1.33297 - acc: 0.7184 -- iter: 320/488
Training Step: 78  | total loss: 1.21861 | time: 40.330s
| Adam | epoch: 010 | loss: 1.21861 - acc: 0.7381 -- iter: 384/488
Training Step: 79  | total loss: 1.15837 | time: 47.165s
| Adam | epoch: 010 | loss: 1.15837 - acc: 0.7458 -- iter: 448/488
Training Step: 80  | total loss: 1.11163 | time: 61.192s
| Adam | epoch: 010 | loss: 1.11163 - acc: 0.7430 | val_loss: 0.45908 - val_acc: 0.8606 -- iter:
488/488
```

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

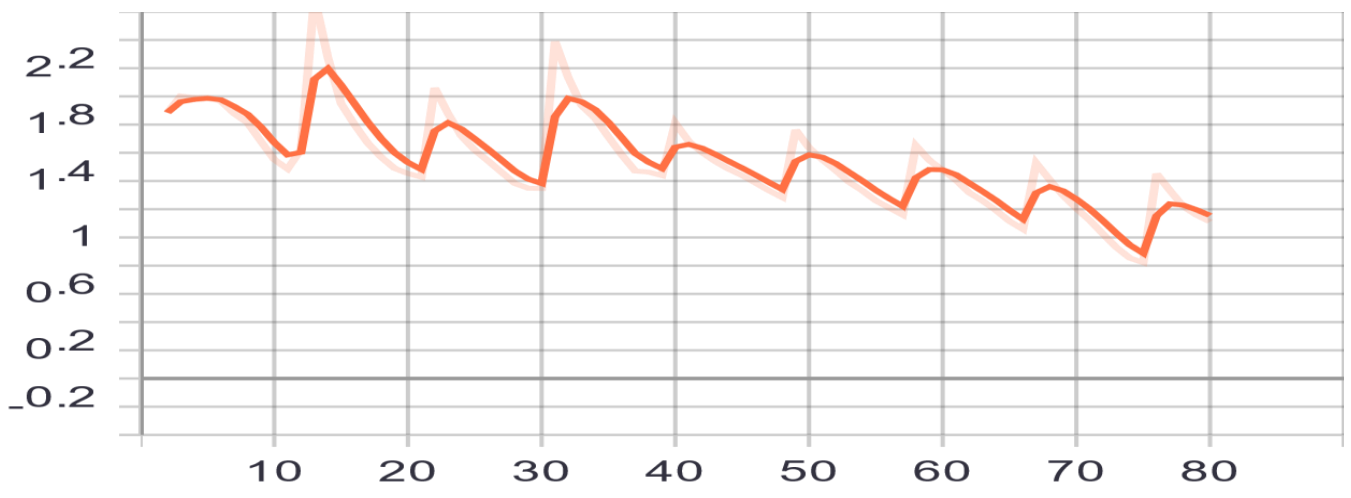
Accuracy of training



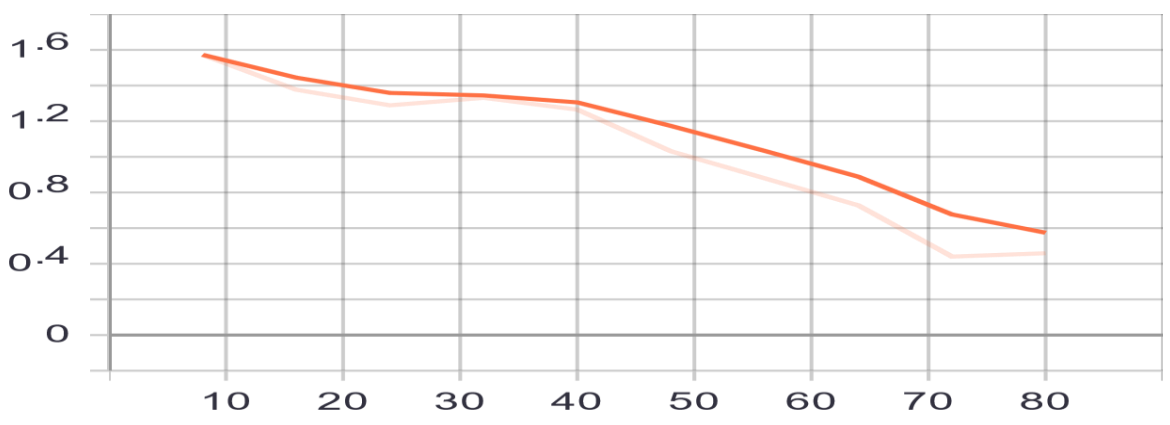
Accuracy of validation



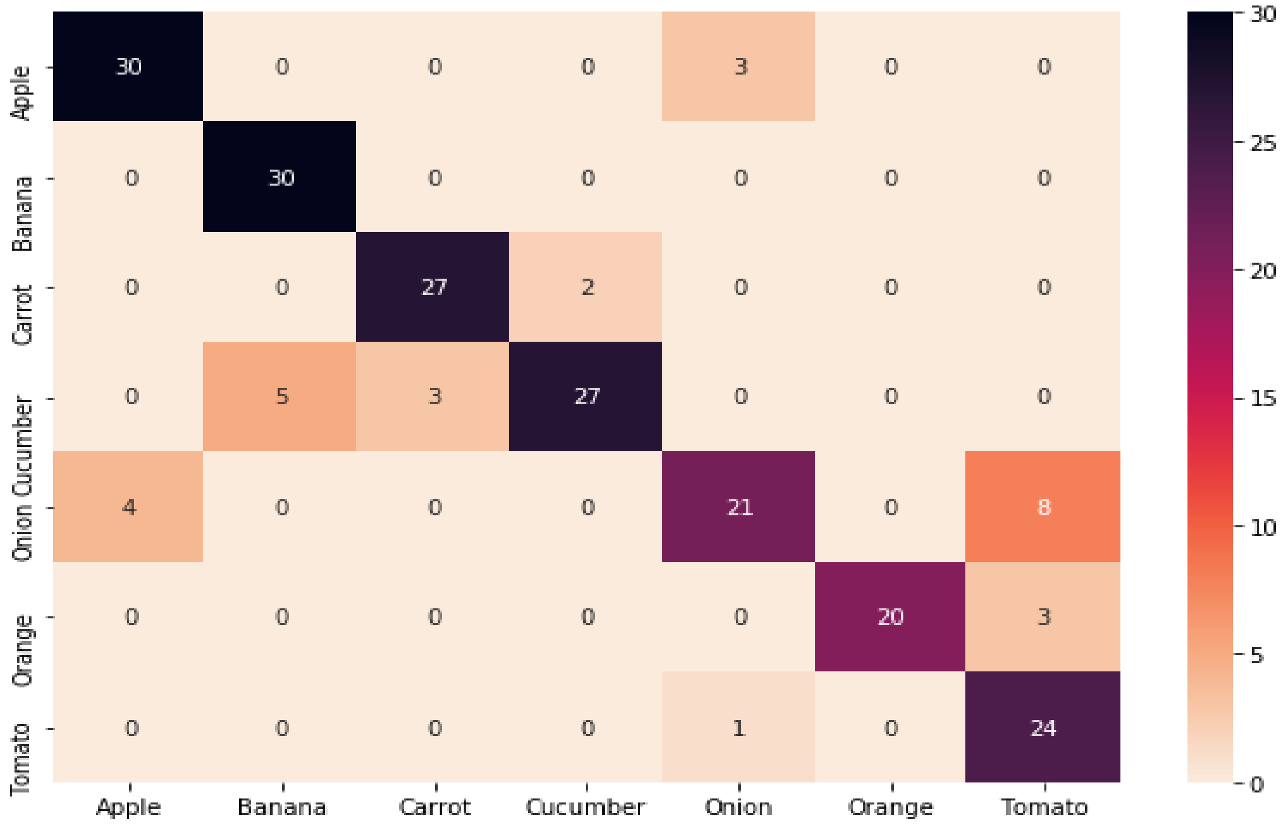
Loss of training



Loss of Validation



Confusion Matrix:



From confusion matrix we see that most of fruits are correctly classified

Estimation Method:

Image Segmentation:

A mixture of methods including canny edge detection, watershed segmentation, morphological operators and Otsu’s method were used to segment the food item to obtain the contour of the food and the contour of the thumb. We use the thumb finger for calibration purposes. The thumb is placed next to the dish while clicking the photo and this thumb gives us the estimate of the real-life size of the food item and helps estimate volume accurately.

Image Segmentation procedure given below:

Figure 4 Original Image

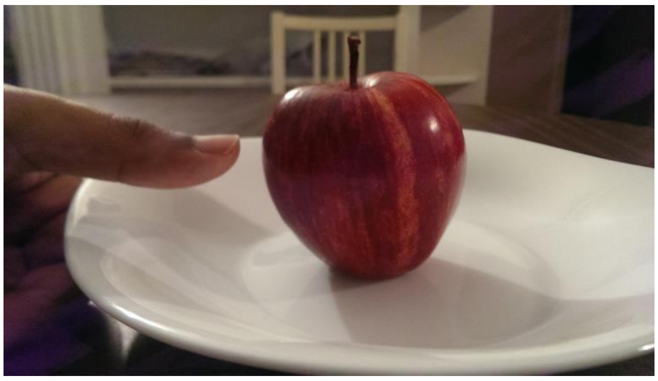


Figure 5 Gray Image

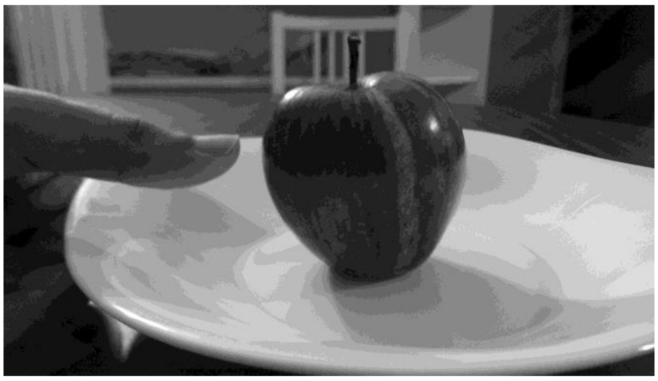


Figure 6 Filter Image

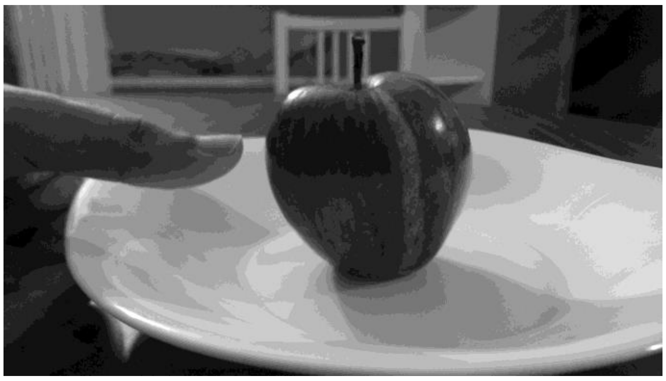
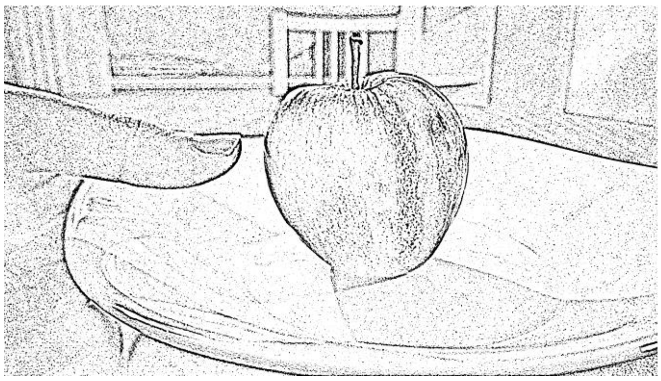


Figure 7 Threshold Image



Find contours, sort and find the biggest contour. The biggest contour corresponds to the plate and food.

Figure 8 Contours of plate

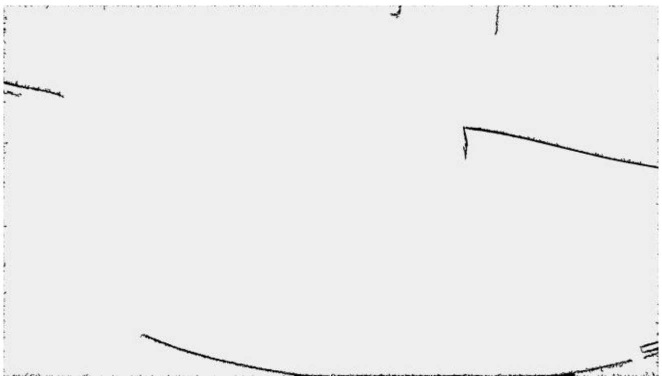
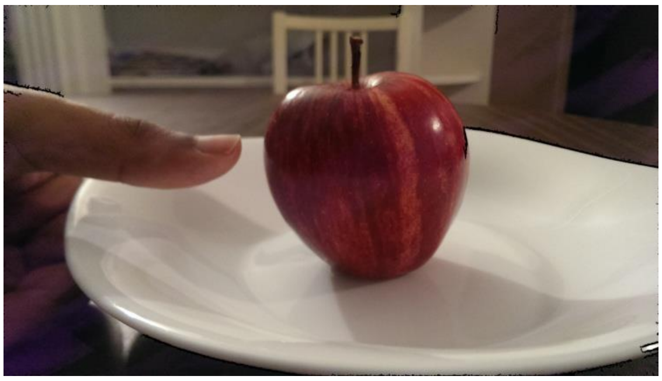


Figure 9Big Contours of Plate



Convert to HSV. Otsu threshold in S to remove plate

Figure 10 HSV Image

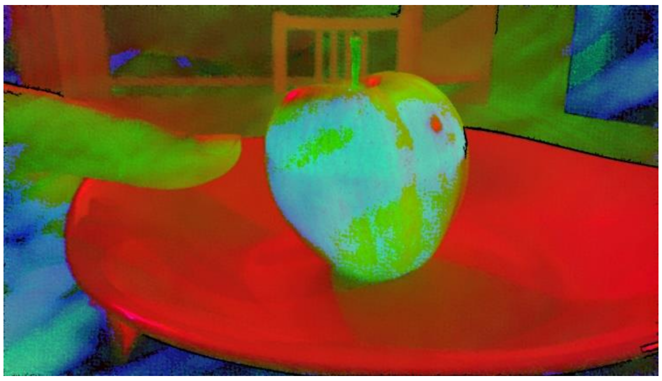


Figure 11 Mask Plate

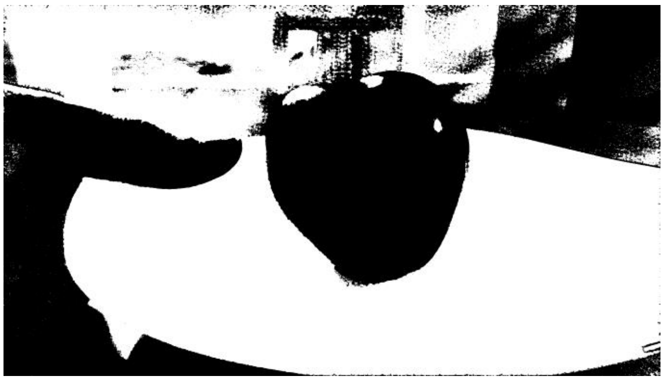


Figure 12 Mask Not Plate

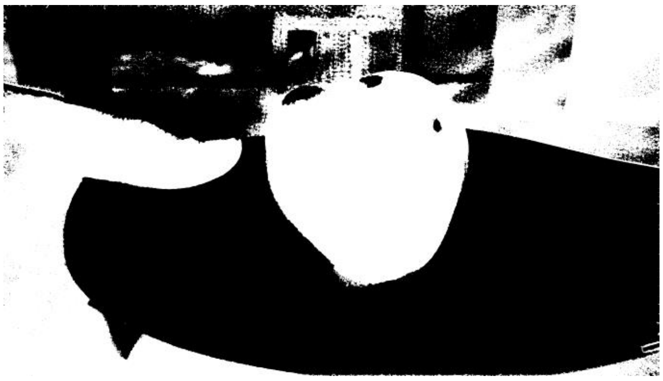
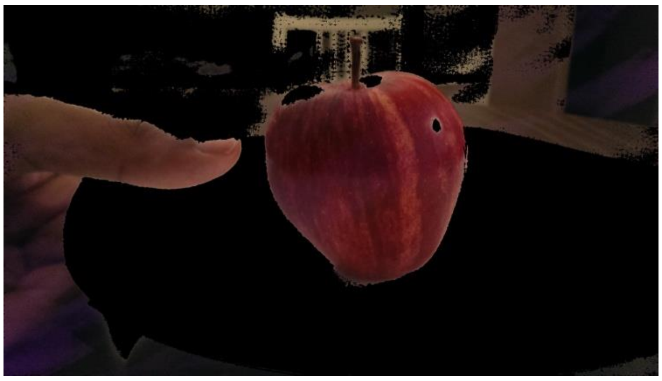


Figure 13Food Skin



Convert to HSV to detect and remove skin pixels

Figure 14HSV Image

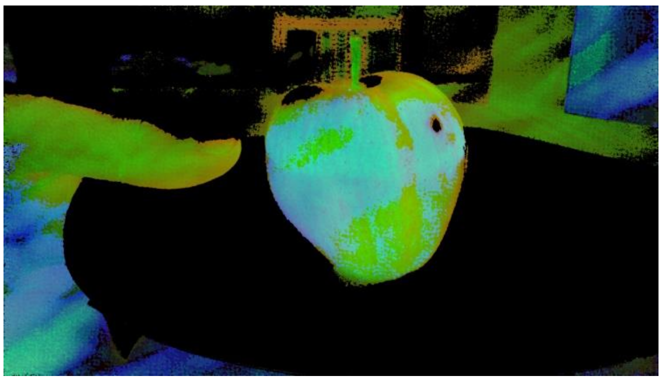


Figure 15 Skin Image

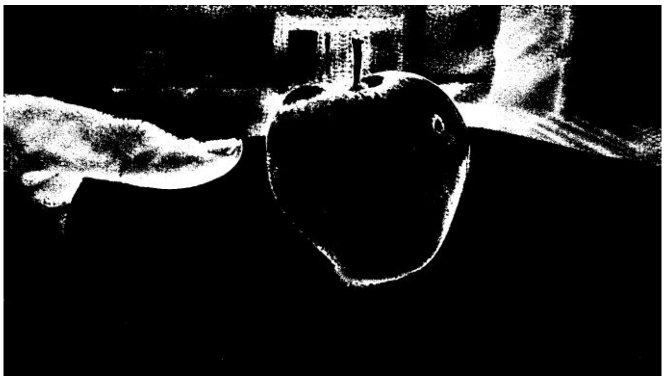


Figure 16 Not Skin Image

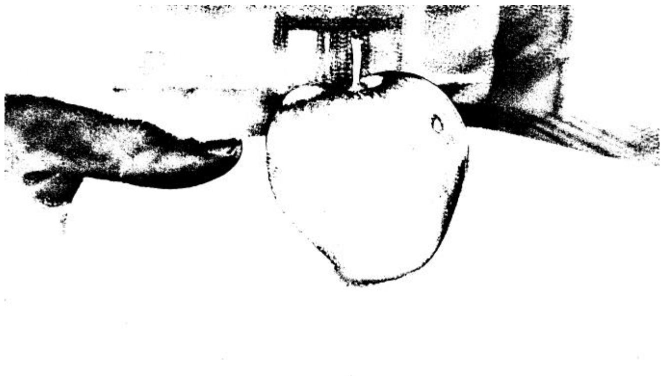


Figure 17food Image

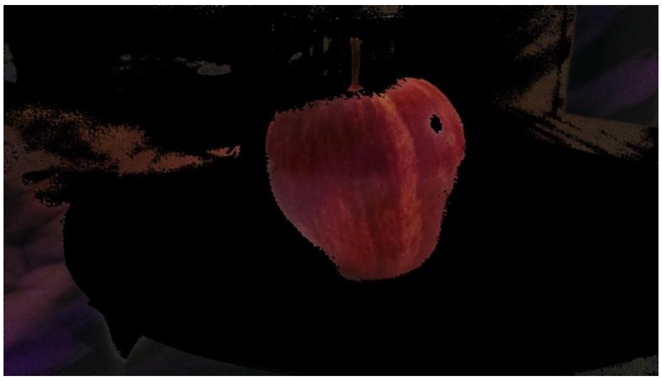


Figure 18Food Structure Image

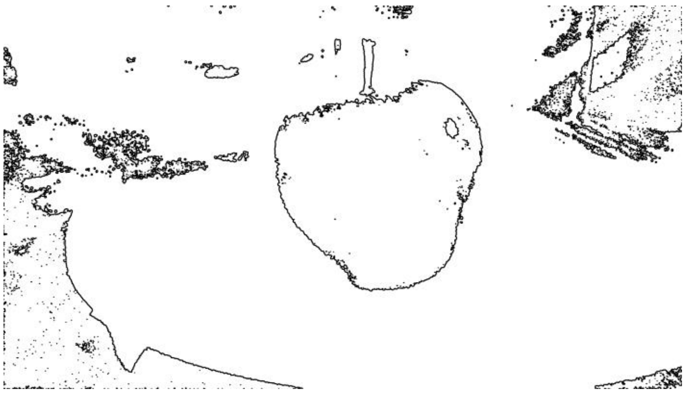


Figure 19 Mask Food Image

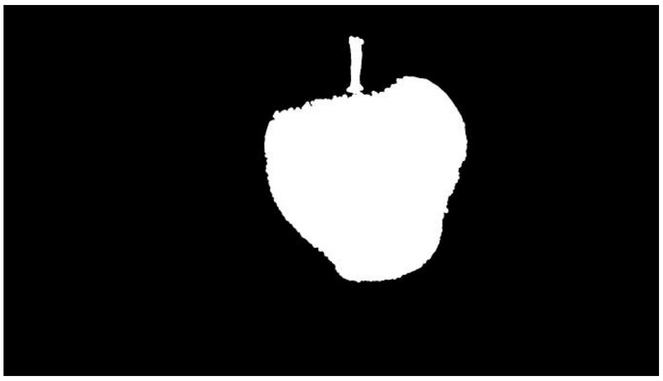


Figure 20 Final Food Image

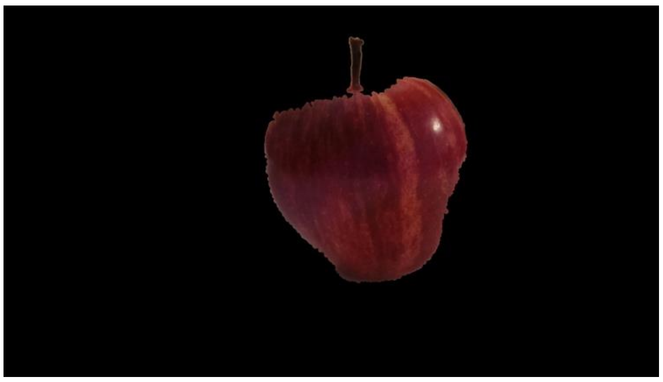


Figure 21 Area of Food Image

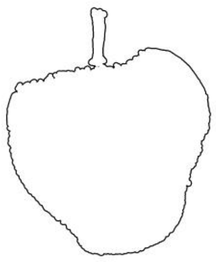


Figure 22Big Contours of Skin

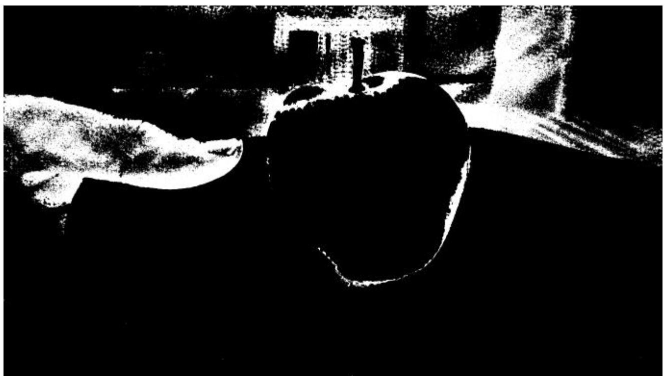


Figure 23 Threshold Image

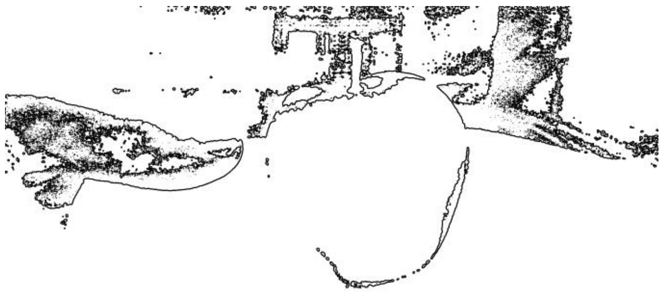


Figure 24 Mask Skin

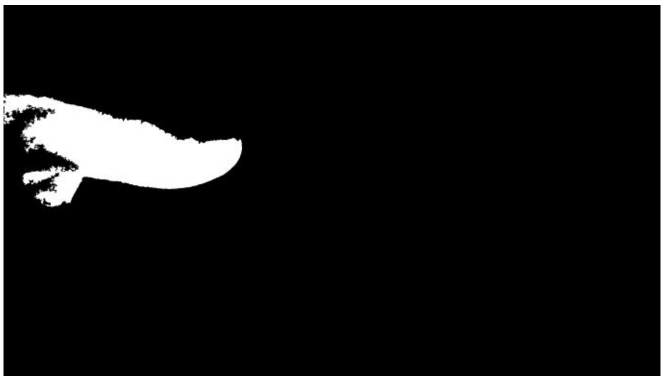
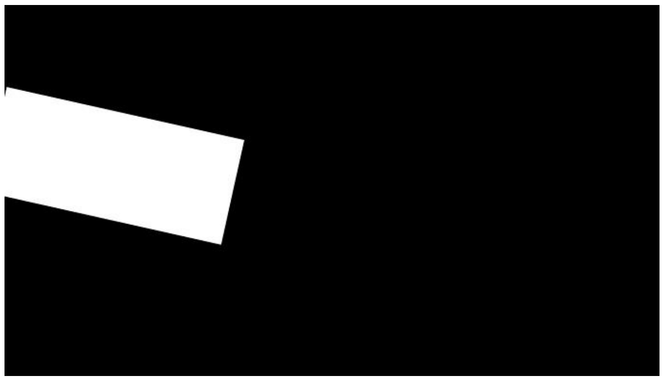


Figure 25 BoxFor Skin



We know that, our thumb size is approximately 5*2.3cm is a skin multiplier (say)
From figure 25 we calculate pixel to cm multiplier by using Maximum pixel height

$$\text{pix\_to\_cm\_multiplier} = \frac{5.0}{\text{pix\_height}}$$

Area of food image:

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:

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

We have two type of shape of foods

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

Volume estimation for Sphere:

$$\text{Estimated Radius} = \sqrt{\frac{\text{Estimated Food Area}}{\Pi}}$$

$$\text{Estimated Volume} = \frac{4}{3} * \Pi * \text{Estimated Radius}^3$$

Volume Estimation ForCylinder:

$$\text{Estimated Height} = \text{Pixel Height} * \text{ Pix\_To\_Cm\_Multiplier}$$

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

$$\text{Estimated Volume} = \Pi * \text{Estimated Height} * \text{Estimated Radius}^2$$

Weight AndCalories Estimation Of Food:

From table 2 we know fooddensity (g/cm^3) and food calories (kcal/g), using this information we can estimate weight and calories intake in given food.

$$\text{Estimated Weight} = \text{ Actual Density} * \text{Estimated Volume}$$

$$\text{Estimated Calories} = \frac{\text{Estimated Weight } * \text{Calories Per 100 Gr}}{100}$$

Result

Food Type	Calories	Estimated Calories
Apple	53.96	40.42
Banana	170.88	188.81
Carrot	31.16	26.28
Cucumber	29.44	37.65
Onion	44.88	37.13
Orange	69.09	71.92
Tomato	17.46	13.82

Limitation and Scope:

Limitations:

1. Actual weight and calories can’t find due to image quality
2. Difficult To Find Appropriate Angle Between Food And Camera
3. Lighting conditioni.e pixel changes with respect to light

Scope:

1. Estimate the calorie from all types of foods.
2. Minimize error of calories estimation

Reference:

[1]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.

[2]Parisa Pouladzadeh, Abdulsalam Yassine, and Shervin Shirmohammadi, “Foodd: An image-based food detection dataset for calorie measurement,” in InternationalConferenceonMultimediaAssistedDietaryManagement, 2015

[3]Meghana M Reddy, “Calorie-estimation-from-food-images-opencv”, github repo, May 2016

Appendix

The code of this project can be downloaded at this website

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