# 42028: Deep Learning and Convolutional Neural Network

## Assignent-3 Project Report

## Project Title: Real-Time Food Recognition and Classification Using Neural Networks

**Team Name and Project Number:** 32-The Hunger Games

**Team Members:**

| | |
|---|---|
| Nazmul Kaonine | 13300912 |
| Fernanda Santosa | 13406537 |
| Minrui Li | 12439586 |

**Date:** 18/06/2020

# Table of Contents

**Abstract** 3

**1. Introduction and Background** 4

**2 Overview of the architecture/system** 12

**3 Results and Evaluation** 22

**4 Discussion and Conclusions** 27

**5 References** 27

**Appendices** 28

# Abstract

With increasing number of health concerns and dietary problems, more individuals have used food tracking applications now than ever before. Food tracking application usually consists of food name options including its categories and detailed nutrition. By having these informations, users can track their dietary intake and maintain their health well-being. There are lots of mobile food tracking applications in the market yet it requires manual steps to scan the food which is not very efficient. In this project, we aim to develop a real-time food detection system which it can detect multiple foods on images in real-time via cameras or from the uploaded images. Food image classification and detection has been successfully implemented by using Mask R-CNN method which we obtain a result of very small amount of loss 1.375. This indicates Mask R-CNN successfully carried out with a little bit of time and patience.

# 1. Introduction and Background

With increasing rates of sub-health and disease, individuals tend to pay more attention towards their health lifestyle especially in their eating habit. They use food tracking mobile applications because of personal interests or for medical purposes. In the existing market, food tracking mobile applications are mostly done by manual input. This process is often called as logging. This allows user to track their food intake by inputting the food names including its amount to the application. This action can also be done by scanning the food barcode which sometimes requires them to manually input the barcode number.

The usage of object detection with deep learning may provide the process faster for users without require them to log their food intake manually. The food detection system that we are trying to implement allow exact shape of multiple object detection on images or even videos in real-time. Model training and testing is required in order to acquire best prediction model based on its accuracy rate. This method can be potentially used on mobile devices to build the real-time food detection process faster and users can track their dietary intake by solely scanning the food, instead of performing it manually. This situation eventually allow the users to track their dietary intake in more effective and efficient way which leads them to have a healthier lifestyle.

## 1.1 The Problems

As mentioned, user usually need to manually input their food name and its amount on food tracking application. However, users find these manual steps troublesome and exhausting because it requires long and detailed process. This situation eventually causes the users to no longer using food tracking application while ironically, these applications were built in purpose to help the user to make the whole steps easier. By providing image classification and object detection with deep learning, specifically Mask R-CNN, these features will help the efficiency of manual input entry in food tracking application.

## 1.2 Motivation

Increasing incidence of diseases such cancer, diabetes, and obesity influence the rising of health awareness more than ever before. It causes individuals to put more efforts and attentions towards their health-wellbeing. This situation leads the rising of food tracking application demand which allows them to record their daily dietary intake. There are benefits proven from using these food tracking applications such as; gain more valuable insights regarding daily diets and eating habits, improve health awareness within the community and the most important of all, to have a healthier lifestyle.

These applications would also be very useful for medical purposes. It would help patients to monitor their diets better which simultaneously provide helpful supports for health professionals. For future possibilities, these applications may support individuals to save their money on personal health care as well as reduce the financial cost and burden on world's healthcare system.

However, the existing food tracking mobile applications are not able to provide real-time food detection via built-in mobile cameras or the uploaded images. Users are required to manually input the food information or scan the food barcode with limited database. This process sometimes can be time-consuming for users who have busy lifestyles. Thus, as a group, we aim to build a system for real-time food detection to fill the gap of users' demands and need with the end purpose to provide more efficient and effective way of managing dietary intake that leads to healthier lifestyle.

## 1.3 Application

Real-time food recognition application has huge potential implementation on mobile devices. This feature carries more efficient method in tracking users dietary intake by providing food category class and its nutrition facts based on the users' phone camera or uploaded food images. Using Mask R-CNN allows exact shape food detection which allow the system to locate multiple foods on one food image. Object detection and segmentation techniques were implemented to build, train and test the model. Then, the model is implemented on real-time food recognition that provides food image classification of one or multiple food item(s) at a time and developed on Graphical User Interface (GUI) Python.

## 1.4 Dataset

The dataset is obtained from food recognition challenge on AIcrowd website and contains 61 food categories. The training dataset contains 7949 food images with the corresponding annotations (AIcrowd n.d.) and the validation dataset has 418 images along with the annotations. All the images are in RGB format and annotations are in MS-COCO JSON format. The annotation file contains food classification and segmentation. The dataset link is in the reference and the following figures are the sample images from the used dataset.

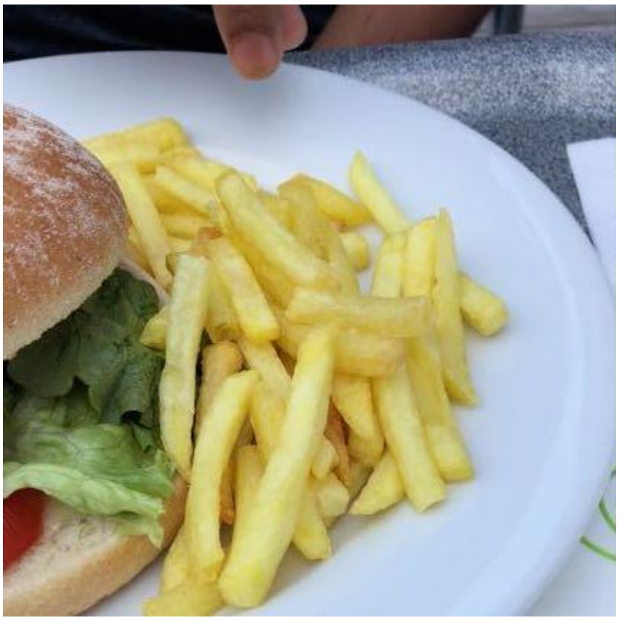


*Figure 1 Example image*

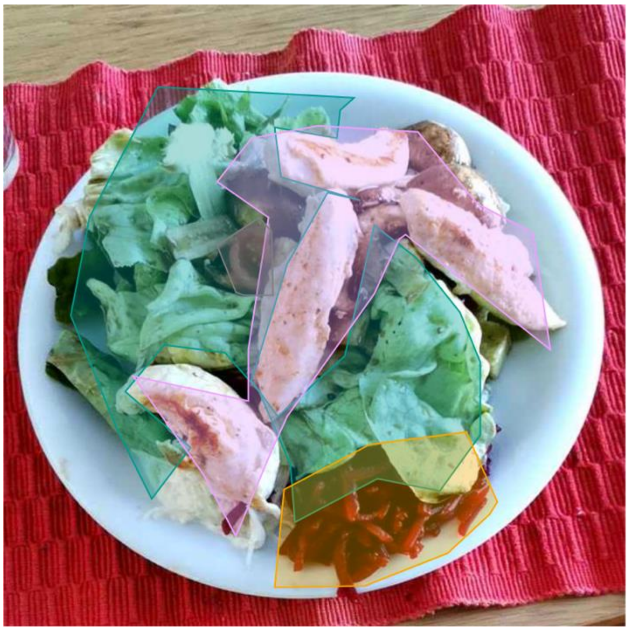


*Figure 2 Image with mask (AIcrowd n.d.)*

The annotations contain information for bounding box and the figure below is sample images with bounding box.

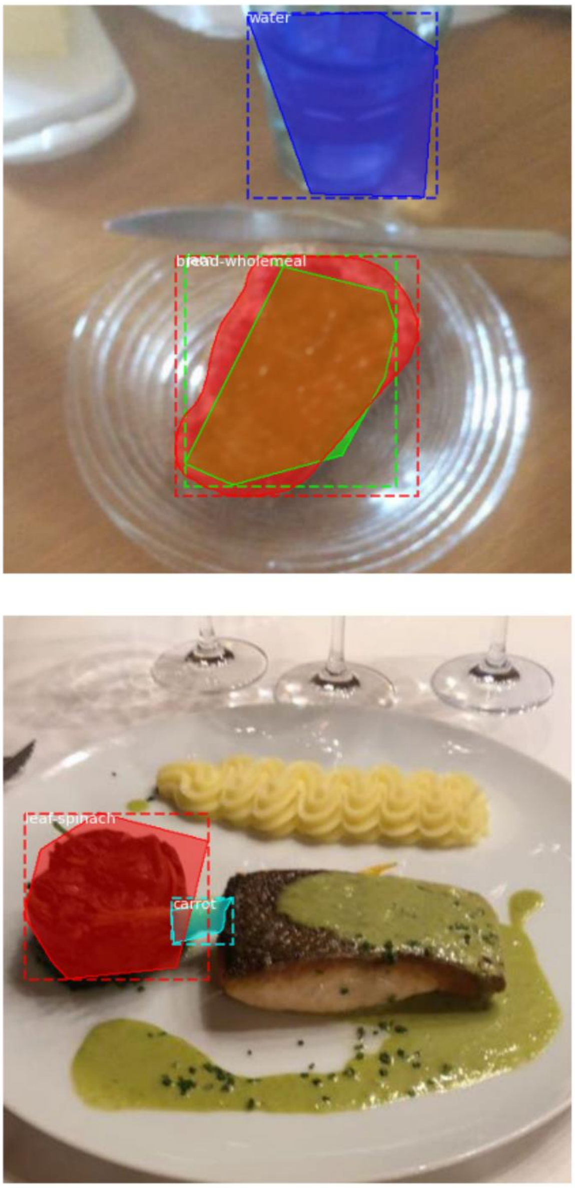
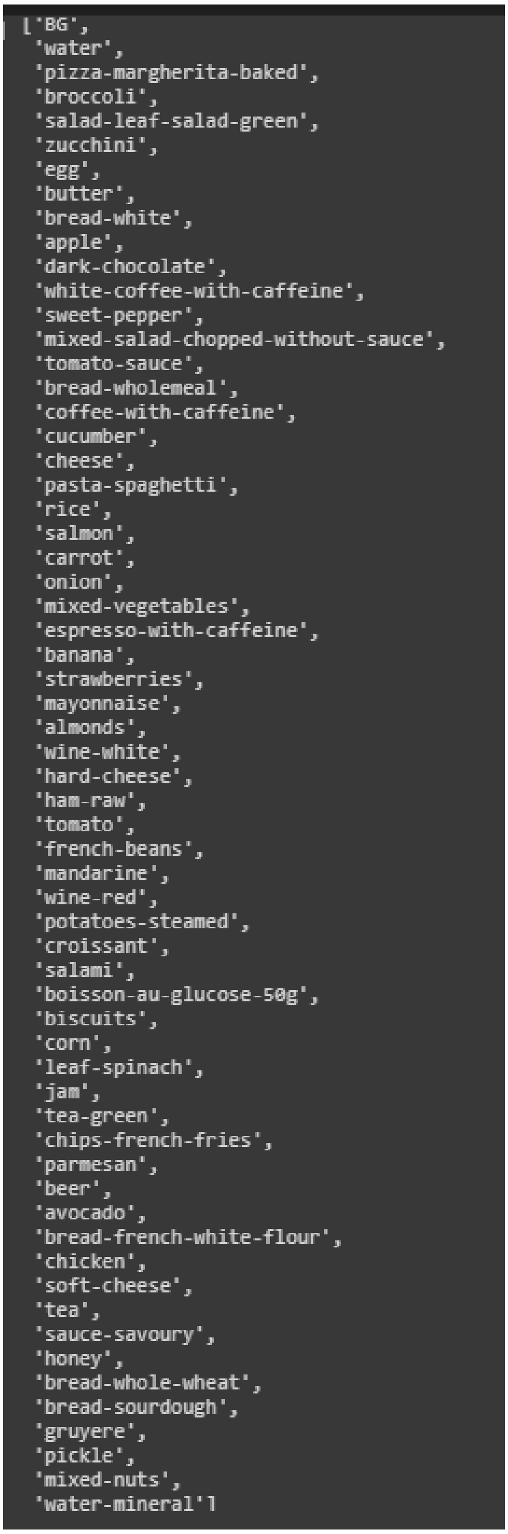


*Figure 3 Sample images with bounding boxes*

Here is the image of all the classes in the dataset, there are 61 food classes  in total.



```
['BG',
 'water',
 'pizza-margherita-baked',
 'broccoli',
 'salad-leaf-salad-green',
 'zucchini',
 'egg',
 'butter',
 'bread-white',
 'apple',
 'dark-chocolate',
 'white-coffee-with-caffeine',
 'sweet-pepper',
 'mixed-salad-chopped-without-sauce',
 'tomato-sauce',
 'bread-wholemeal',
 'coffee-with-caffeine',
 'cucumber',
 'cheese',
 'pasta-spaghetti',
 'rice',
 'salmon',
 'carrot',
 'onion',
 'mixed-vegetables',
 'espresso-with-caffeine',
 'banana',
 'strawberries',
 'mayonnaise',
 'almonds',
 'wine-white',
 'hard-cheese',
 'ham-raw',
 'tomato',
 'french-beans',
 'mandarine',
 'wine-red',
 'potatoes-steamed',
 'croissant',
 'salami',
 'boisson-au-glucose-50g',
 'biscuits',
 'corn',
 'leaf-spinach',
 'jam',
 'tea-green',
 'chips-french-fries',
 'parmesan',
 'beer',
 'avocado',
 'bread-french-white-flour',
 'chicken',
 'soft-cheese',
 'tea',
 'sauce-savoury',
 'honey',
 'bread-whole-wheat',
 'bread-sourdough',
 'gruyere',
 'pickle',
 'mixed-nuts',
 'water-mineral']
```

*Figure 4 61 classes*

The dataset also has nnotation information about each image masks and the following figure is sample images with masks. And the 2nd image below is the annotation of one of the images.

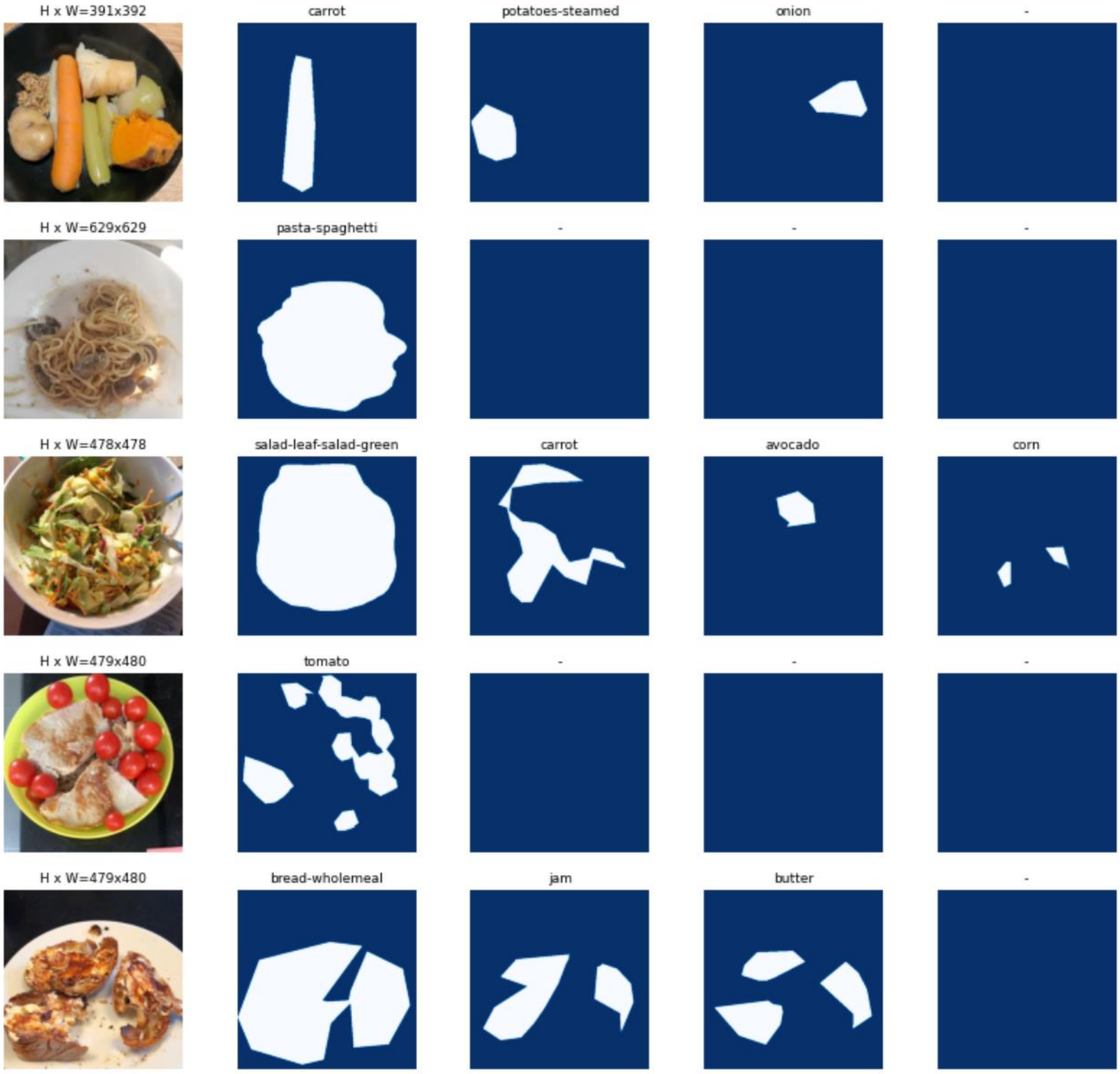
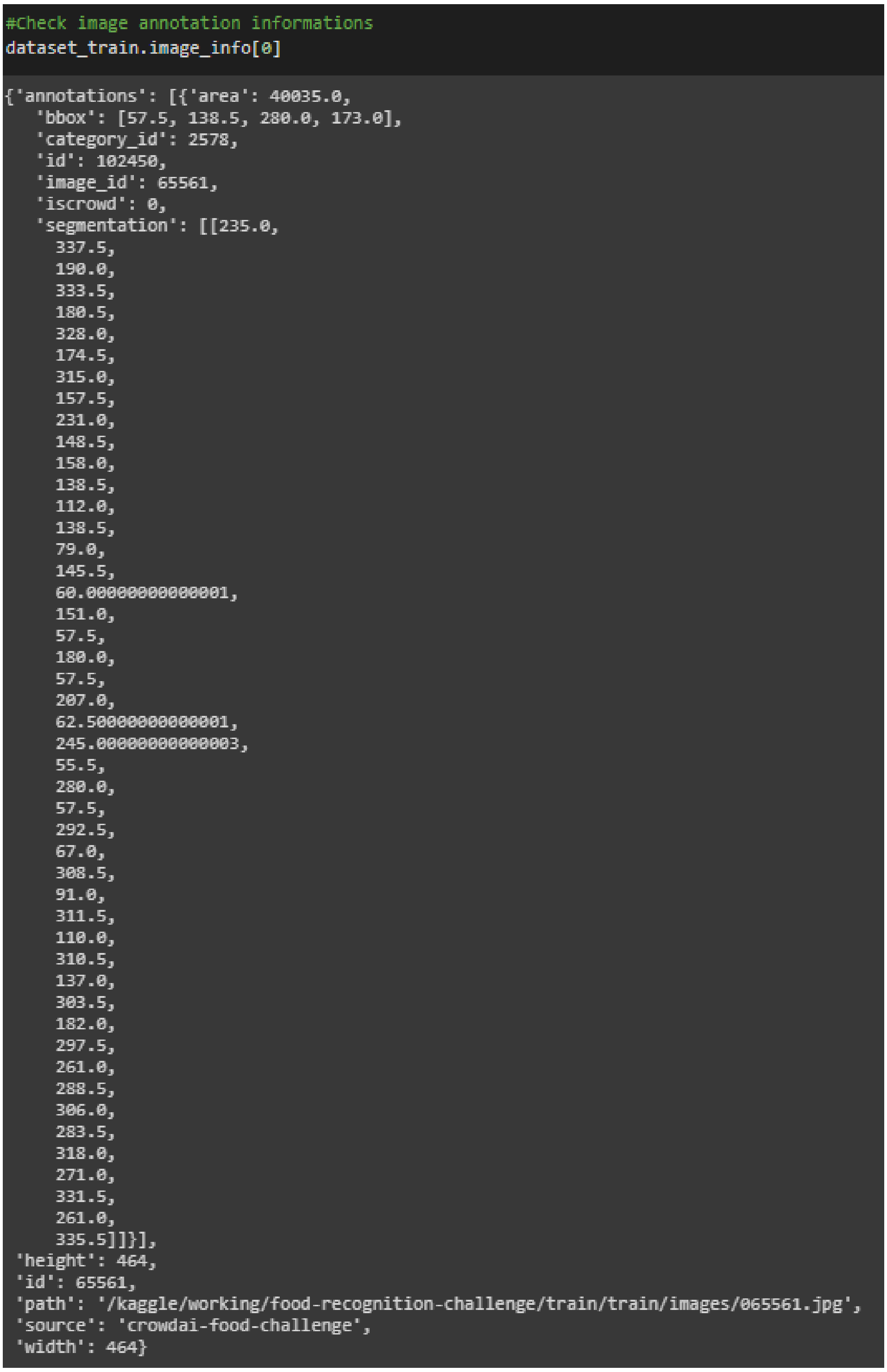


*Figure 5 Images with masks*

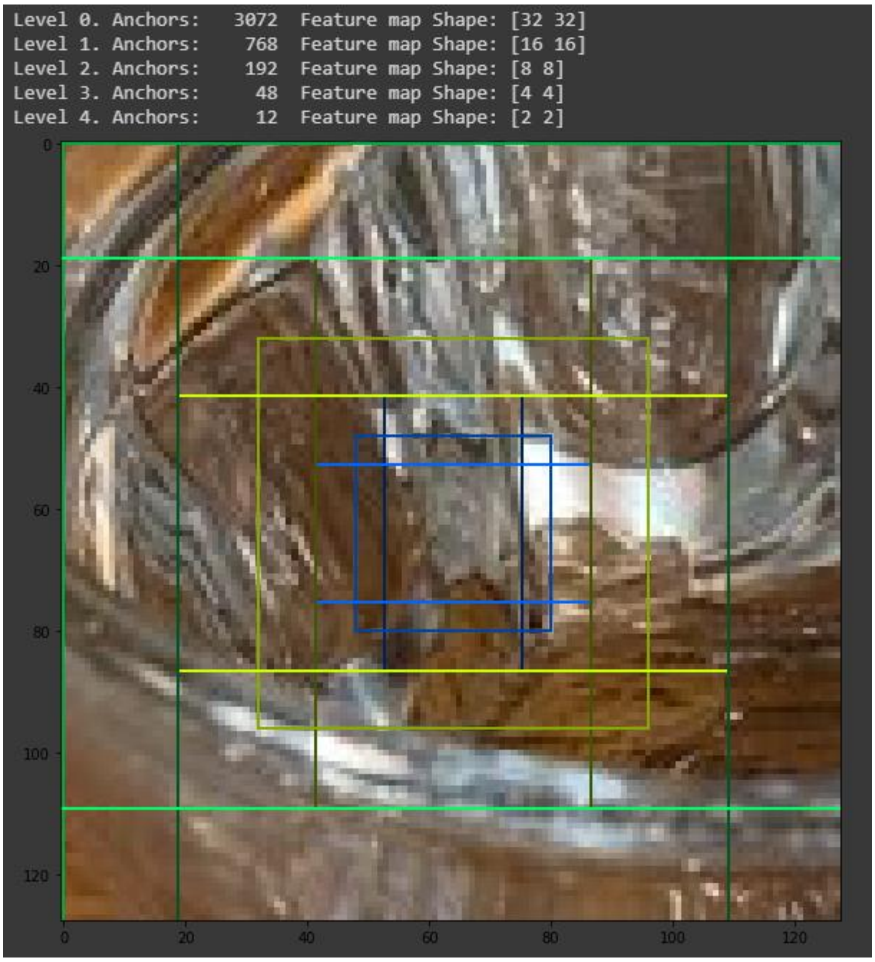


```
#Check image annotation informations
dataset_train.image_info[0]
```

```
{'annotations': [{'area': 40035.0,
   'bbox': [57.5, 138.5, 280.0, 173.0],
   'category_id': 2578,
   'id': 102450,
   'image_id': 65561,
   'iscrowd': 0,
   'segmentation': [[235.0,
     337.5,
     190.0,
     333.5,
     180.5,
     328.0,
     174.5,
     315.0,
     157.5,
     231.0,
     148.5,
     158.0,
     138.5,
     112.0,
     138.5,
     79.0,
     145.5,
     60.00000000000001,
     151.0,
     57.5,
     180.0,
     57.5,
     207.0,
     62.50000000000001,
     245.00000000000003,
     55.5,
     280.0,
     57.5,
     292.5,
     67.0,
     308.5,
     91.0,
     311.5,
     110.0,
     310.5,
     137.0,
     303.5,
     182.0,
     297.5,
     261.0,
     288.5,
     306.0,
     283.5,
     318.0,
     271.0,
     331.5,
     261.0,
     335.5]]}],
 'height': 464,
 'id': 65561,
 'path': '/kaggle/working/food-recognition-challenge/train/train/images/065561.jpg',
 'source': 'crowdai-food-challenge',
 'width': 464}
```

*Figure 6 Image annotation*

Except annotation and bounding box information, our approach of object detection using Mask R-CNN also allows formation of anchor and it is shown in the following figure.



```
Level 0. Anchors:   3072  Feature map Shape: [32 32]
Level 1. Anchors:    768  Feature map Shape: [16 16]
Level 2. Anchors:    192  Feature map Shape: [8 8]
Level 3. Anchors:     48  Feature map Shape: [4 4]
Level 4. Anchors:     12  Feature map Shape: [2 2]
```

*Figure 7 Example anchor*

# 2. Overview of the architecture/system

## 2.1 Flow Diagram

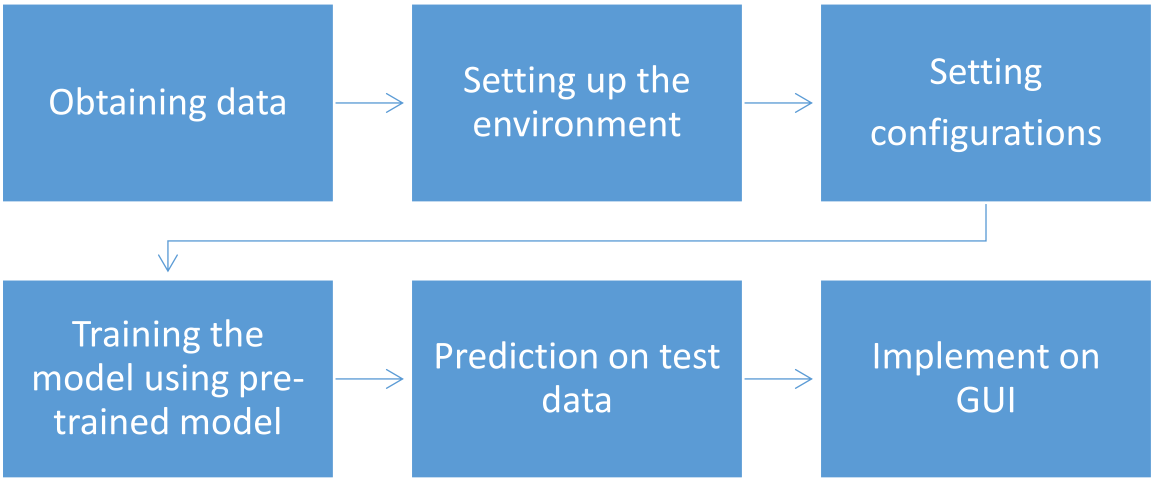


*Figure 8 Workflow of the project*

As mentioned in the last section, the dataset includes images with complete annotations that has bounding box and segmentation annotations is from AIcrowd website with 61 categories of food. The figure above illustrates the flow of the project.

## 2.2 CNN Architecture Design

The task is to implement object detection and segmentation on food images. The algorithm chosen is Mask R-CNN, which is based on Faster R-CNN, a region-based convolutional neural network with outputs of bounding boxes, class names and confidence score. The algorithm first performs feature extraction on the entire image with backbone CNN architecture. After feature extraction, regional proposal network (RPN) scans feature maps to output regions of interest (ROIs). RPN's role is to identify areas that potentially contain object by applying sliding windows and obtaining anchors and allow these areas to be further processed in the ROIAlign. The prediction of bounding box (classification and box offset) and mask prediction are in parallel and generated by ROI align (He et al. 2017).

The backbone CNN architecture used in this project is ResNet50. ResNet 50 is one of the pre-trained networks trained on ImageNet dataset, which contains general images including food and uses deep residual network consisting of 16 residual blocks. The following figure is the structure of residual network at the top, ResNet 50 has the same structure except each residual block consists of 3 layers.

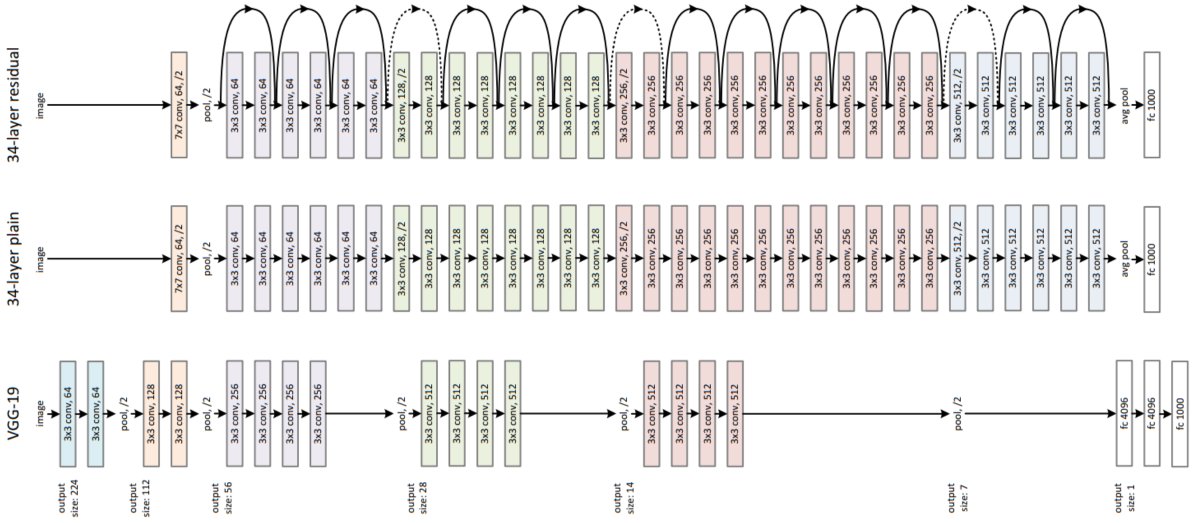


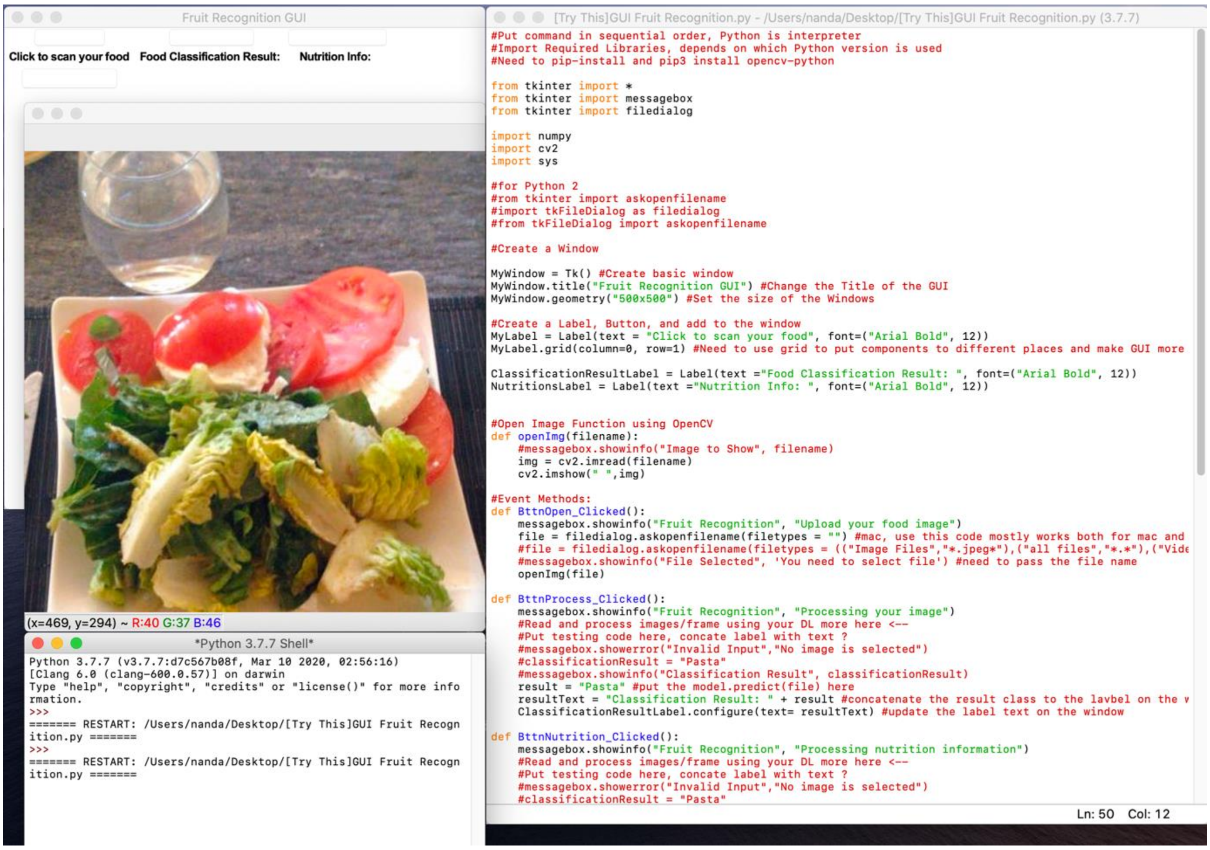
*Figure 9 Network architecture (He et al. 2016)*

## 2.3 GUI Design

Here is our first attempt on creating our Food Tracking GUI Design. We are able to open images, and aim to create the system to perform process images and provide nutrition information based on prediction model. We managed to create fully-functioned GUI that able to process images.

### First GUI Design Attempt on Mac

Attached step-by-step images of using GUI to implement classification.


Fruit Recognition GUI
Click to scan your food
Food Classification Result:
Nutrition Info:
[Try This]GUI Fruit Recognition.py - /Users/nanda/Desktop/[Try This]GUI Fruit Recognition.py (3.7.7)
*Python 3.7.7 Shell*
Ln: 50
Col: 12

```
#Put command in sequential order, Python is interpreter
#Import Required Libraries, depends on which Python version is used
#Need to pip-install and pip3 install opencv-python

from tkinter import *
from tkinter import messagebox
from tkinter import filedialog

import numpy
import cv2
import sys

#for Python 2
#rom tkinter import askopenfilename
#import tkFileDialog as filedialog
#from tkFileDialog import askopenfilename

#Create a Window

MyWindow = Tk() #Create basic window
MyWindow.title("Fruit Recognition GUI") #Change the Title of the GUI
MyWindow.geometry("500x500") #Set the size of the Windows

#Create a Label, Button, and add to the window
MyLabel = Label(text = "Click to scan your food", font=("Arial Bold", 12))
MyLabel.grid(column=0, row=1) #Need to use grid to put components to different places and make GUI more

ClassificationResultLabel = Label(text ="Food Classification Result: ", font=("Arial Bold", 12))
NutritionsLabel = Label(text ="Nutrition Info: ", font=("Arial Bold", 12))


#Open Image Function using OpenCV
def openImg(filename):
    #messagebox.showinfo("Image to Show", filename)
    img = cv2.imread(filename)
    cv2.imshow(" ",img)

#Event Methods:
def BttnOpen_Clicked():
    messagebox.showinfo("Fruit Recognition", "Upload your food image")
    file = filedialog.askopenfilename(filetypes = "") #mac, use this code mostly works both for mac and
    #file = filedialog.askopenfilename(filetypes = (("Image Files","*.jpeg*"),("all files","*.*"),("Vide
    #messagebox.showinfo("File Selected", 'You need to select file') #need to pass the file name
    openImg(file)

def BttnProcess_Clicked():
    messagebox.showinfo("Fruit Recognition", "Processing your image")
    #Read and process images/frame using your DL more here <--
    #Put testing code here, concate label with text ?
    #messagebox.showerror("Invalid Input","No image is selected")
    #classificationResult = "Pasta"
    #messagebox.showinfo("Classification Result", classificationResult)
    result = "Pasta" #put the model.predict(file) here
    resultText = "Classification Result: " + result #concatenate the result class to the lavbel on the v
    ClassificationResultLabel.configure(text= resultText) #update the label text on the window

def BttnNutrition_Clicked():
    messagebox.showinfo("Fruit Recognition", "Processing nutrition information")
    #Read and process images/frame using your DL more here <--
    #Put testing code here, concate label with text ?
    #messagebox.showerror("Invalid Input","No image is selected")
    #classificationResult = "Pasta"
```

```
(x=469, y=294) ~ R:40 G:37 B:46
```

```
Python 3.7.7 (v3.7.7:d7c567b08f, Mar 10 2020, 02:56:16)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license()" for more info
rmation.
>>>
======= RESTART: /Users/nanda/Desktop/[Try This]GUI Fruit Recogn
ition.py =======
>>>
======= RESTART: /Users/nanda/Desktop/[Try This]GUI Fruit Recogn
ition.py =======
```

*Figure 10 GUI Process*

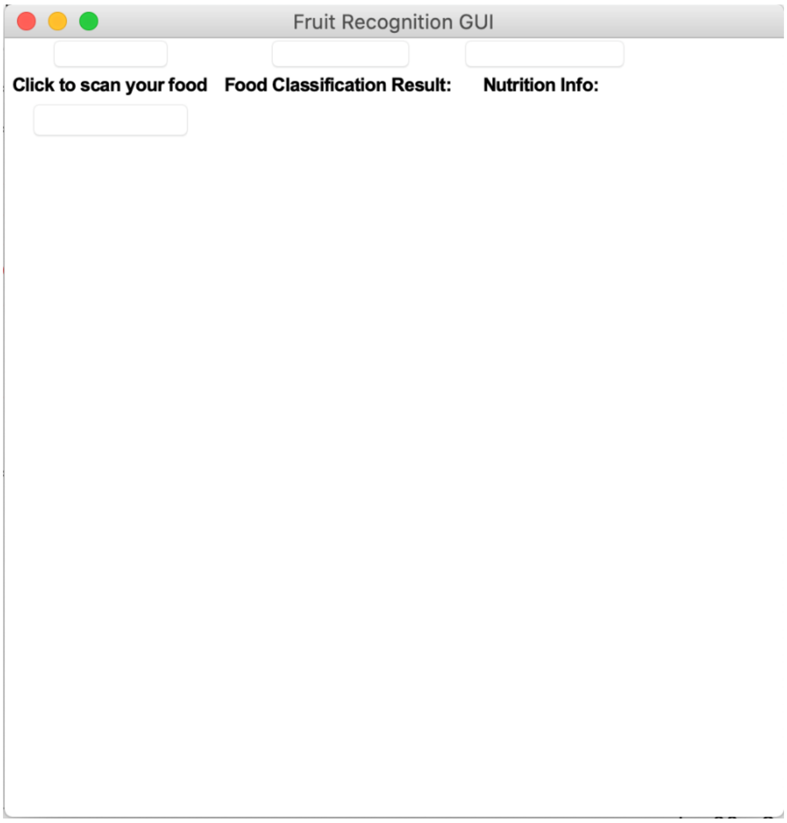
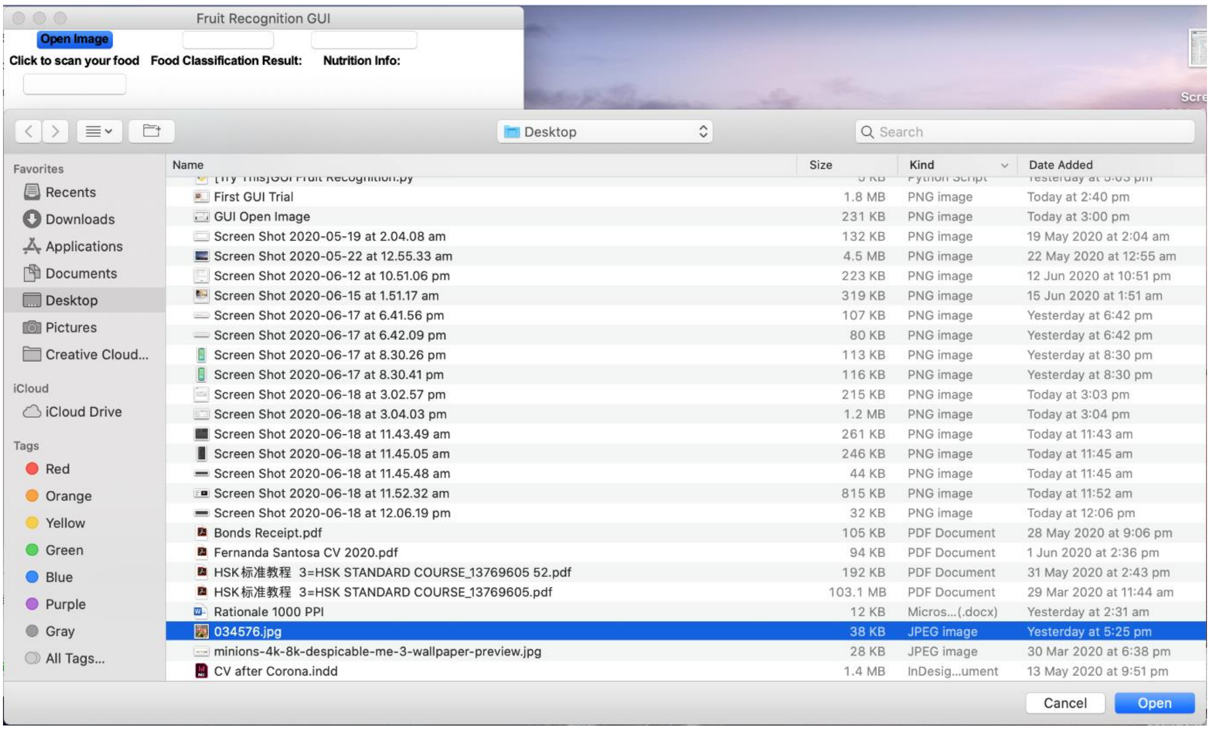
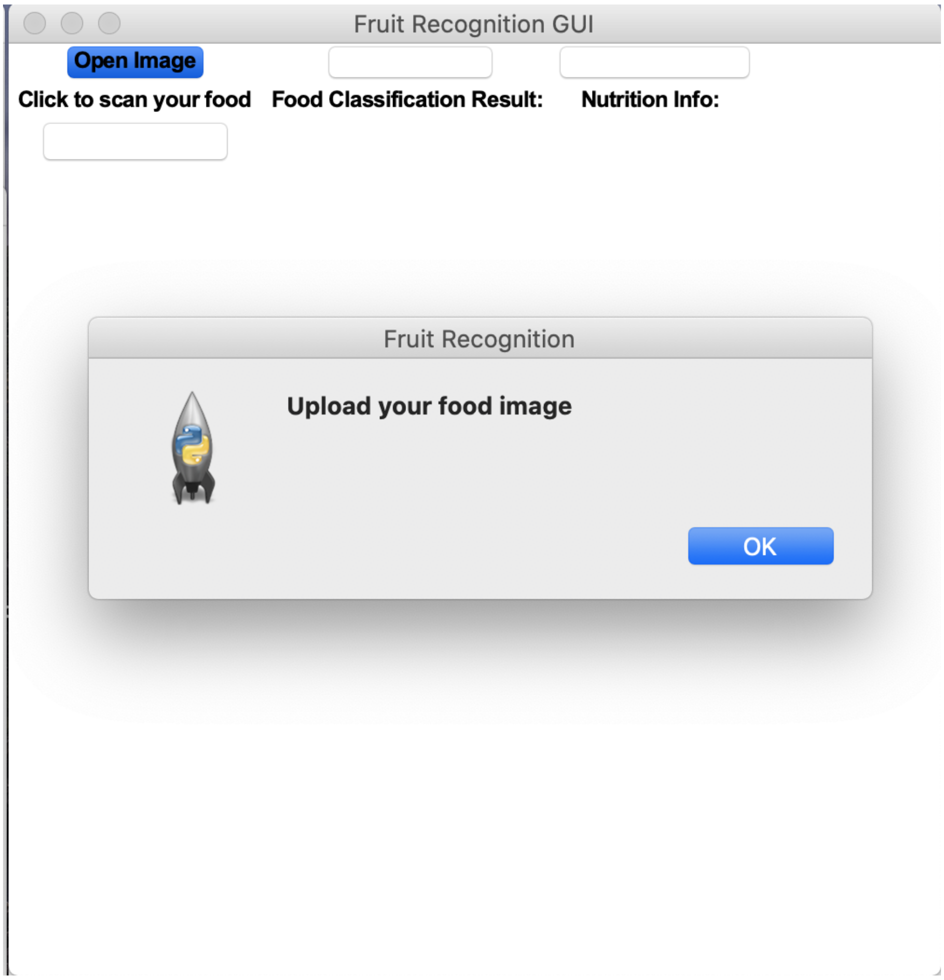


*Figure 11 Step 1 GUI Home screen*



| Name | Size | Kind | Date Added |
|---|---|---|---|
| [Try This]GUI Fruit Recognition.py | 5 KB | Python Script | Yesterday at 5:03 pm |
| First GUI Trial | 1.8 MB | PNG image | Today at 2:40 pm |
| GUI Open Image | 231 KB | PNG image | Today at 3:00 pm |
| Screen Shot 2020-05-19 at 2.04.08 am | 132 KB | PNG image | 19 May 2020 at 2:04 am |
| Screen Shot 2020-05-22 at 12.55.33 am | 4.5 MB | PNG image | 22 May 2020 at 12:55 am |
| Screen Shot 2020-06-12 at 10.51.06 pm | 223 KB | PNG image | 12 Jun 2020 at 10:51 pm |
| Screen Shot 2020-06-15 at 1.51.17 am | 319 KB | PNG image | 15 Jun 2020 at 1:51 am |
| Screen Shot 2020-06-17 at 6.41.56 pm | 107 KB | PNG image | Yesterday at 6:42 pm |
| Screen Shot 2020-06-17 at 6.42.09 pm | 80 KB | PNG image | Yesterday at 6:42 pm |
| Screen Shot 2020-06-17 at 8.30.26 pm | 113 KB | PNG image | Yesterday at 8:30 pm |
| Screen Shot 2020-06-17 at 8.30.41 pm | 116 KB | PNG image | Yesterday at 8:30 pm |
| Screen Shot 2020-06-18 at 3.02.57 pm | 215 KB | PNG image | Today at 3:03 pm |
| Screen Shot 2020-06-18 at 3.04.03 pm | 1.2 MB | PNG image | Today at 3:04 pm |
| Screen Shot 2020-06-18 at 11.43.49 am | 261 KB | PNG image | Today at 11:43 am |
| Screen Shot 2020-06-18 at 11.45.05 am | 246 KB | PNG image | Today at 11:45 am |
| Screen Shot 2020-06-18 at 11.45.48 am | 44 KB | PNG image | Today at 11:45 am |
| Screen Shot 2020-06-18 at 11.52.32 am | 815 KB | PNG image | Today at 11:52 am |
| Screen Shot 2020-06-18 at 12.06.19 pm | 32 KB | PNG image | Today at 12:06 pm |
| Bonds Receipt.pdf | 105 KB | PDF Document | 28 May 2020 at 9:06 pm |
| Fernanda Santosa CV 2020.pdf | 94 KB | PDF Document | 1 Jun 2020 at 2:36 pm |
| HSK标准教程 3=HSK STANDARD COURSE_13769605 52.pdf | 192 KB | PDF Document | 31 May 2020 at 2:43 pm |
| HSK标准教程 3=HSK STANDARD COURSE_13769605.pdf | 103.1 MB | PDF Document | 29 Mar 2020 at 11:44 am |
| Rationale 1000 PPI | 12 KB | Micros...(.docx) | Yesterday at 2:31 am |
| 034576.jpg | 38 KB | JPEG image | Yesterday at 5:25 pm |
| minions-4k-8k-despicable-me-3-wallpaper-preview.jpg | 28 KB | JPEG image | 30 Mar 2020 at 6:38 pm |
| CV after Corona.indd | 1.4 MB | InDesig...ument | 13 May 2020 at 9:51 pm |

*Figure 12 Step 2 choose image*



Upload your food image

*Figure 13 Step 3 Open image*

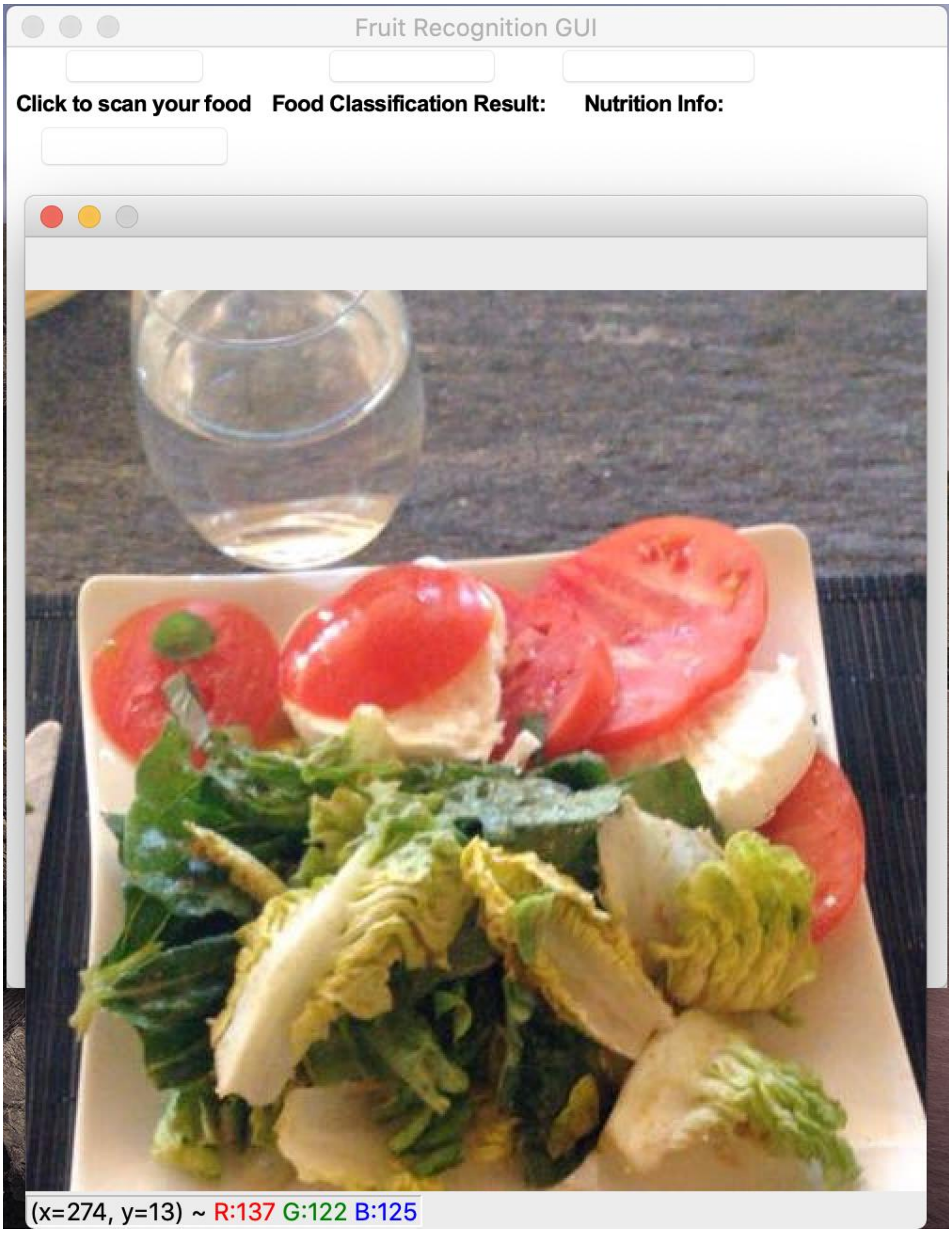
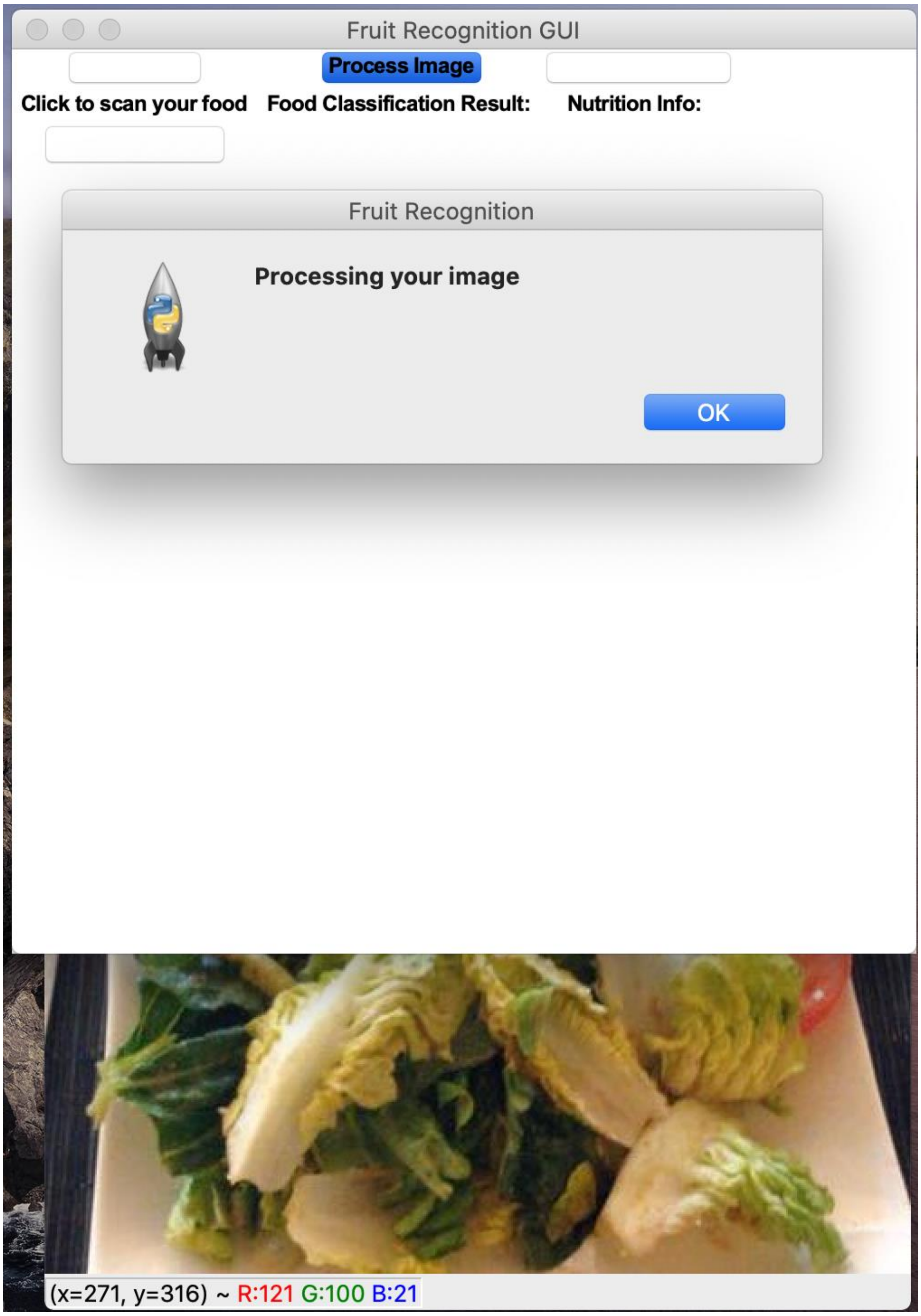

Fruit Recognition GUI
Click to scan your food
Food Classification Result:
Nutrition Info:
(x=274, y=13) ~ R:137 G:122 B:125

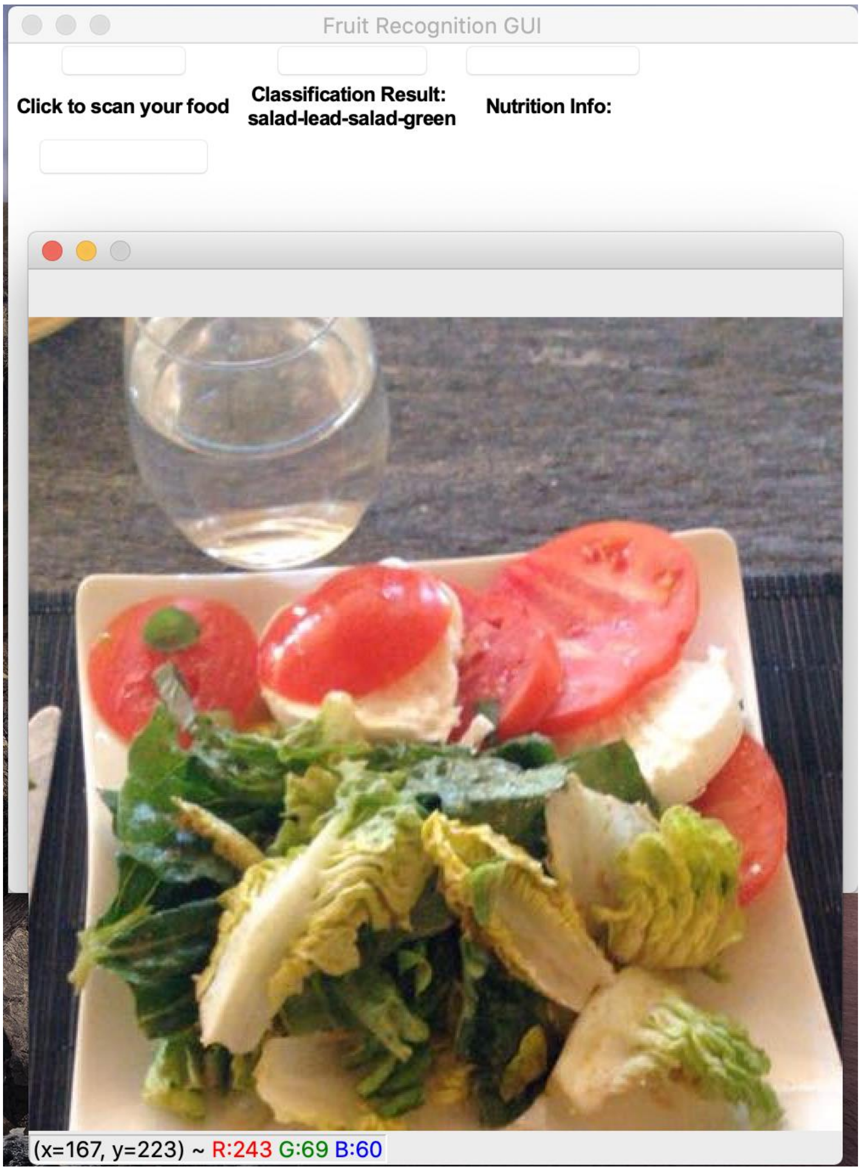
*Figure 14 Step 4 image opened*



Processing your image

```
(x=271, y=316) ~ R:121 G:100 B:21
```

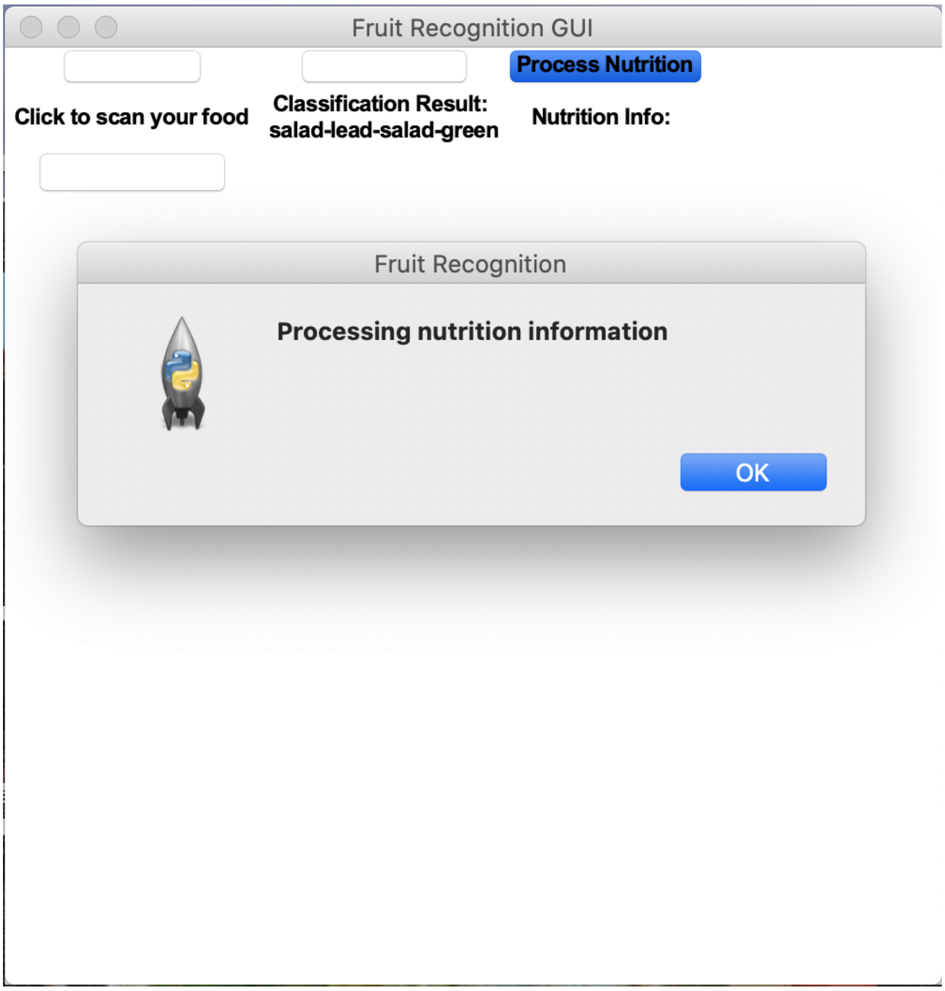
*Figure 15 Step 5 process image*



Classification Result:
salad-lead-salad-green

(x=167, y=223) ~ R:243 G:69 B:60

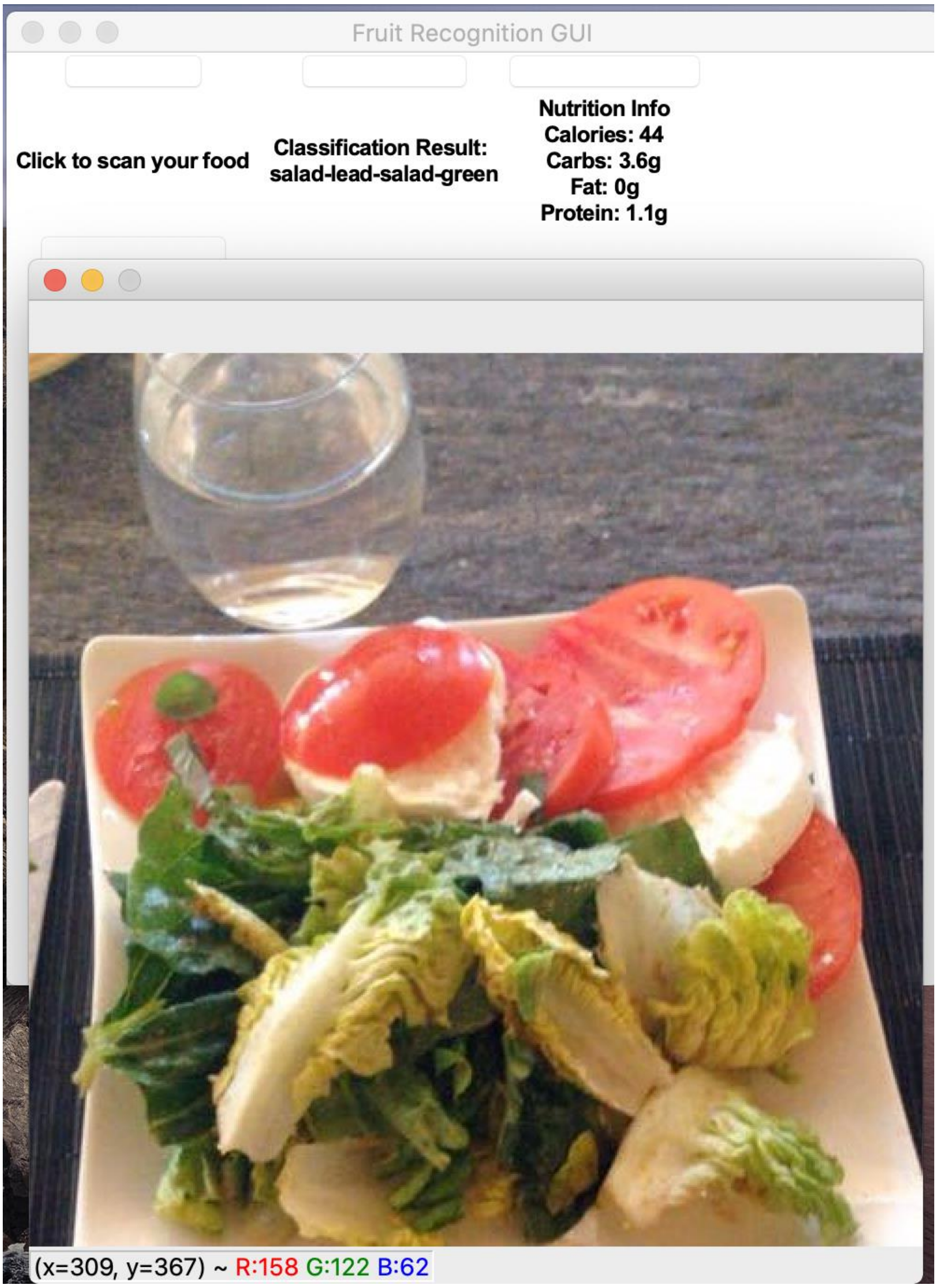
*Figure 16 Step 6 classification result*



Classification Result:
salad-lead-salad-green

Processing nutrition information

*Figure 17 Step 7 Process Nutrition*



Fruit Recognition GUI

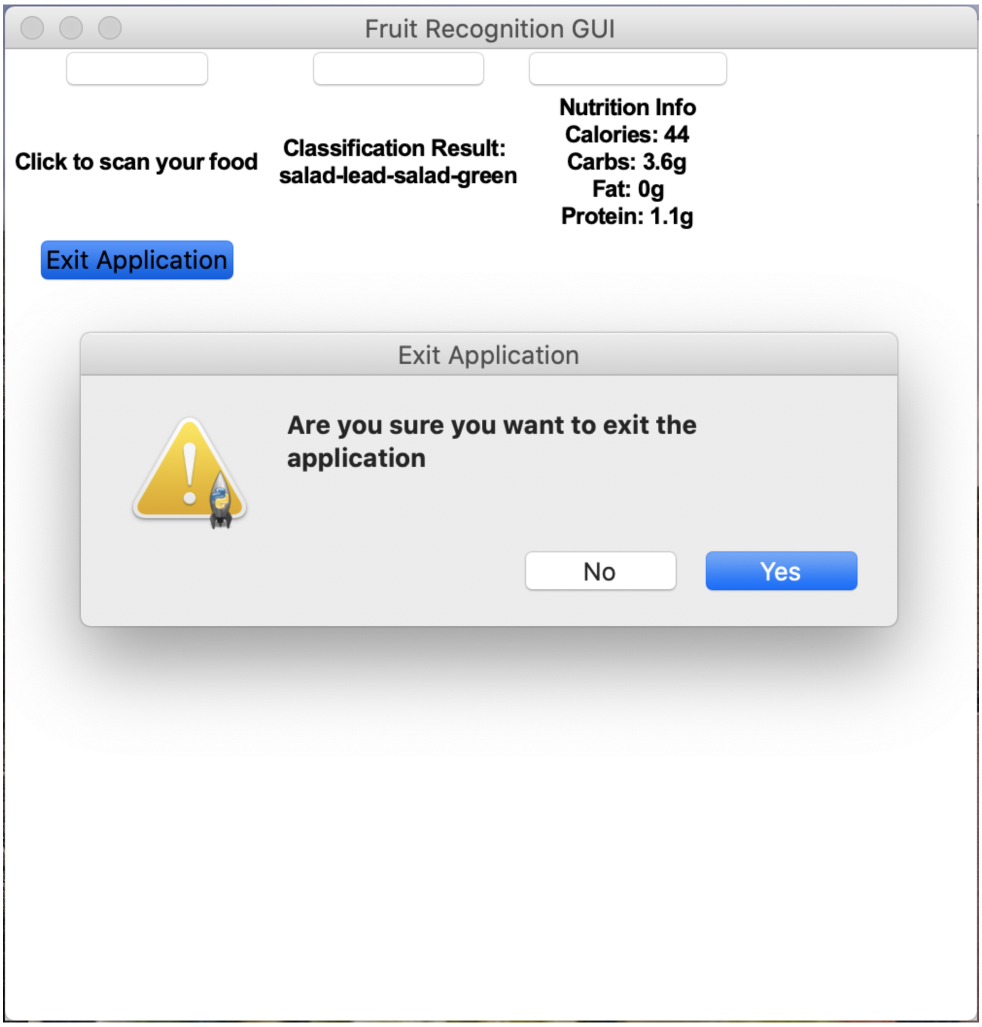
Click to scan your food

Classification Result:
salad-lead-salad-green

Nutrition Info
Calories: 44
Carbs: 3.6g
Fat: 0g
Protein: 1.1g

(x=309, y=367) ~ R:158 G:122 B:62

*Figure 18 Step 8 Final Step*



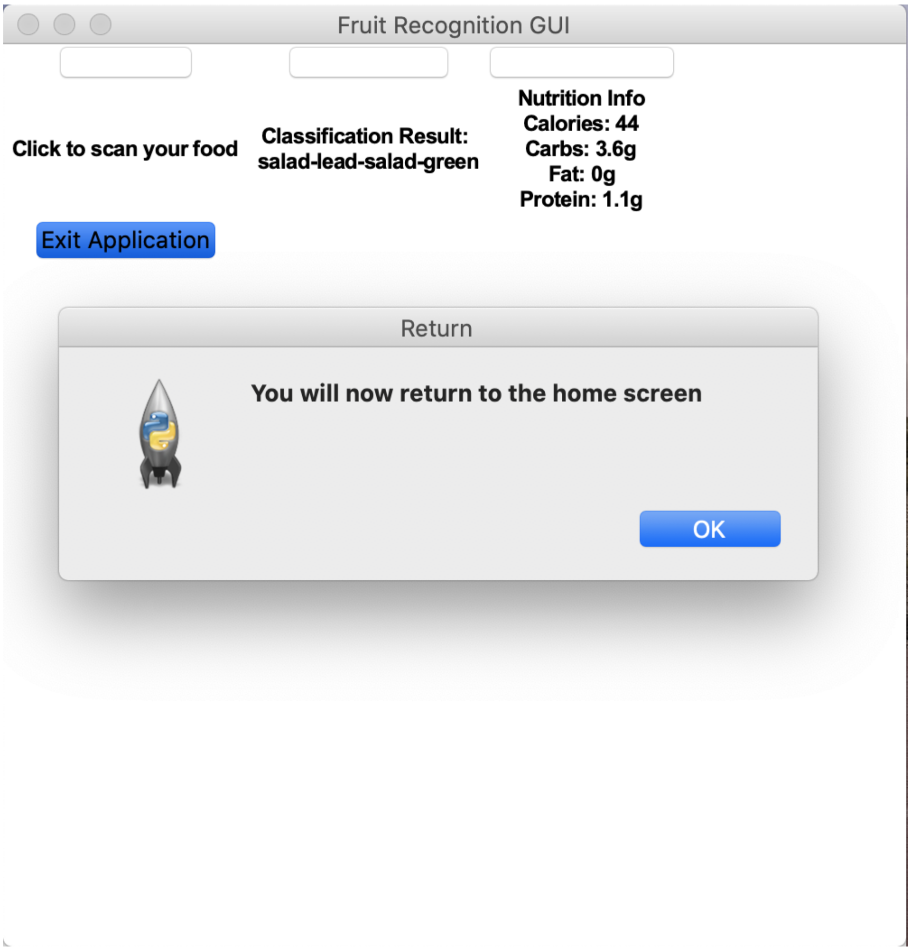
Click to scan your food

Classification Result:
salad-lead-salad-green

Nutrition Info
Calories: 44
Carbs: 3.6g
Fat: 0g
Protein: 1.1g

Are you sure you want to exit the application

*Figure 19 Step 9 Exit Application Yes*



Classification Result:
salad-lead-salad-green

Nutrition Info
Calories: 44
Carbs: 3.6g
Fat: 0g
Protein: 1.1g

You will now return to the home screen

*Figure 20 Step 10 Exit Application No*

### Final GUI Design on Windows

The picture is the final screen of our GUI.

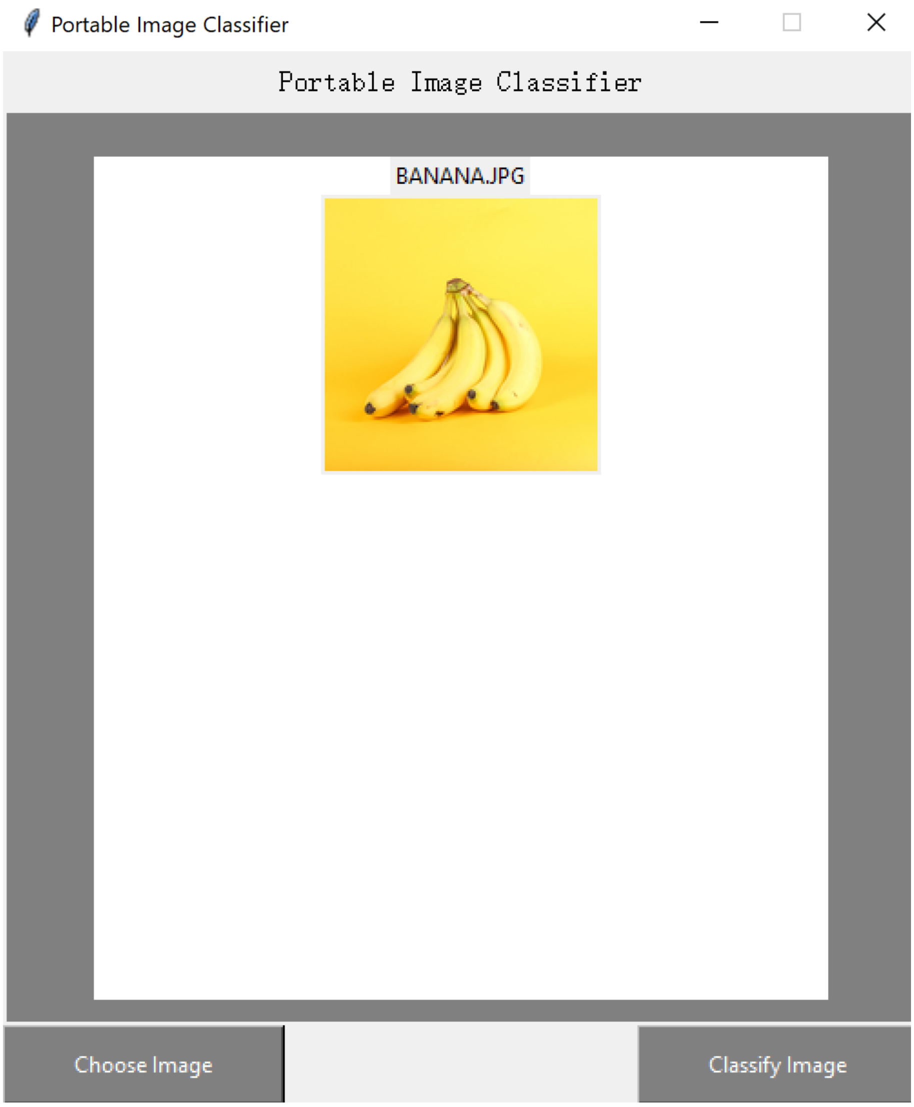


*Figure 21 Final GUI*

# 3. Results and Evaluation

## 3.1 Experimental Settings

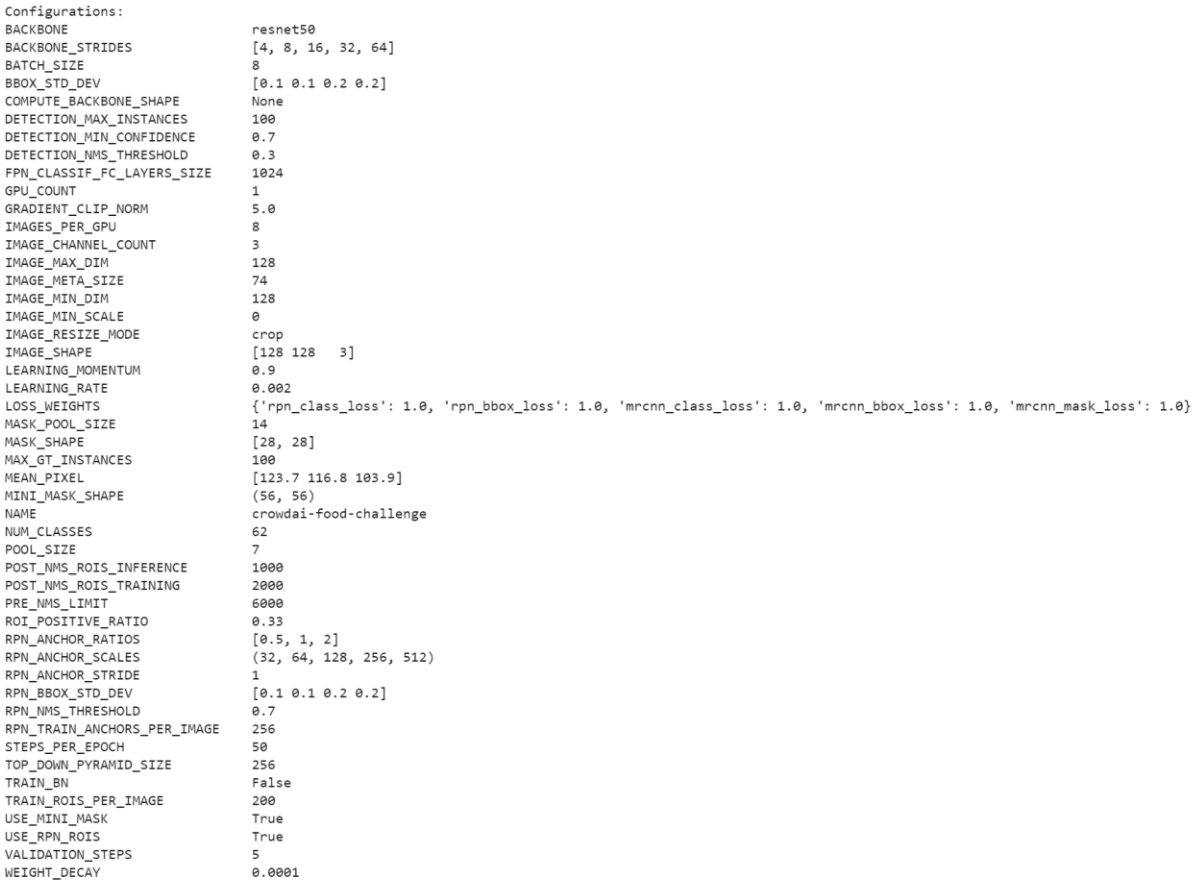
[Briefly describe the Experimental results with Sample images]

The following table is the hyper parameter settings of the experiment. The use of relatively small learning rate of 0.002 prevents our model from overshooting. The learning momentum allows the acceleration of training. Using ResNet 101 as CNN backbone might give a better outcome but because Mask R-CNN is designed for obtaining good accuracy, it is not a light-weight algorithm and there is limitation with memory and hardware, thus, ResNet 50 is chosen.

*Table 1 Hyperparameter settings*

| | Hyper-parameters |
|---|---|
| Epoch | 15 |
| Backbone | ResNet 50 |
| Steps_per_epoch | 50 |
| Validation_steps | 5 |
| Image_resize_mode | “crop” |
| Learning_rate | 0.002 |
| Image_max_dim | 128 |
| Image_min_dim | 128 |
| Num_classes | 62 (1 Background + 61 classes) |
| Images_per_GPU | 8 |
| Learning_momentum | 0.9 (default) |
| Other settings | Default |

The screenshot below is the complete configurations of the training.



```
Configurations:
BACKBONE                       resnet50
BACKBONE_STRIDES               [4, 8, 16, 32, 64]
BATCH_SIZE                     8
BBOX_STD_DEV                   [0.1 0.1 0.2 0.2]
COMPUTE_BACKBONE_SHAPE         None
DETECTION_MAX_INSTANCES        100
DETECTION_MIN_CONFIDENCE       0.7
DETECTION_NMS_THRESHOLD        0.3
FPN_CLASSIF_FC_LAYERS_SIZE     1024
GPU_COUNT                      1
GRADIENT_CLIP_NORM             5.0
IMAGES_PER_GPU                 8
IMAGE_CHANNEL_COUNT            3
IMAGE_MAX_DIM                  128
IMAGE_META_SIZE                74
IMAGE_MIN_DIM                  128
IMAGE_MIN_SCALE                0
IMAGE_RESIZE_MODE              crop
IMAGE_SHAPE                    [128 128   3]
LEARNING_MOMENTUM              0.9
LEARNING_RATE                  0.002
LOSS_WEIGHTS                   {'rpn_class_loss': 1.0, 'rpn_bbox_loss': 1.0, 'mrcnn_class_loss': 1.0, 'mrcnn_bbox_loss': 1.0, 'mrcnn_mask_loss': 1.0}
MASK_POOL_SIZE                 14
MASK_SHAPE                     [28, 28]
MAX_GT_INSTANCES               100
MEAN_PIXEL                     [123.7 116.8 103.9]
MINI_MASK_SHAPE                (56, 56)
NAME                           crowdai-food-challenge
NUM_CLASSES                    62
POOL_SIZE                      7
POST_NMS_ROIS_INFERENCE        1000
POST_NMS_ROIS_TRAINING         2000
PRE_NMS_LIMIT                  6000
ROI_POSITIVE_RATIO             0.33
RPN_ANCHOR_RATIOS              [0.5, 1, 2]
RPN_ANCHOR_SCALES              (32, 64, 128, 256, 512)
RPN_ANCHOR_STRIDE              1
RPN_BBOX_STD_DEV               [0.1 0.1 0.2 0.2]
RPN_NMS_THRESHOLD              0.7
RPN_TRAIN_ANCHORS_PER_IMAGE    256
STEPS_PER_EPOCH                50
TOP_DOWN_PYRAMID_SIZE          256
TRAIN_BN                       False
TRAIN_ROIS_PER_IMAGE           200
USE_MINI_MASK                  True
USE_RPN_ROIS                   True
VALIDATION_STEPS               5
WEIGHT_DECAY                   0.0001
```

*Figure 22 Configurations*

# 3.2 Pre-Processing

Our pre-processing parts include image data exploration and performed research with the related pre-trained model.

## 3.3 Experimental Results

The following table is the result obtained of our experiment. The evaluation metrics are based on loss from intermediate step of RPN and the result of mask R-CNN of classification, bounding box, and mask.

| Metrics | Values |
|---|---|
| Total loss | 1.3758 |
| rpn_class_loss | 0.0236 |
| rpn_bbox_loss | 0.3910 |
| mrcnn_class_loss | 0.3104 |
| mrcnn_bbox_loss | 0.3183 |
| mrcnn_mask_loss | 0.3325 |
| val total loss | 1.3855 |
| val_rpn_class_loss | 0.0203 |
| val_rpn_bbox_loss | 0.3609 |
| val_mrcnn_class_loss | 0.2980 |
| val_mrcnn_bbox_loss | 0.3545 |
| val_mrcnn_mask_loss | 0.3518 |

*Figure 23 Result*

By setting inference configuration, few images are predicted based on our trained model, they are shown below. The left is ground truth and the right is prediction result. The model is able to form bounding boxes and masks, although they are not predicted accurately. This might be due to the model trained with small learning rates and not enough training steps and epochs. The other reason might be the pre-trained model we used for training has different setting as we had while training. In addition, it might be because we used different inference configurations as we set for training model.

```
(512, 512, 3)
[ 8 33  7 44 20 27]
```

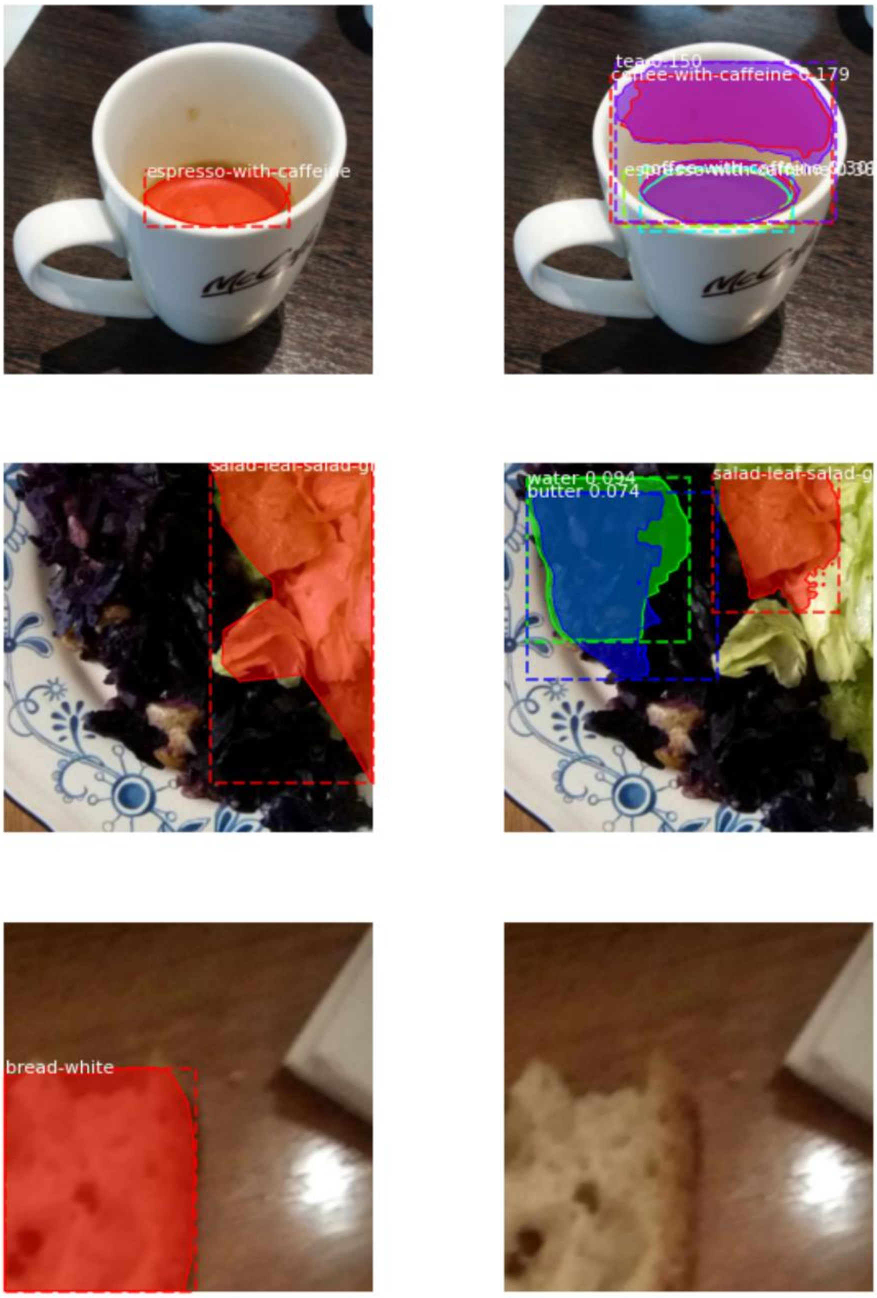


*Figure 24 Prediction result*

## 3.4 Limitations

Due to the nature of R-CNN family, Mask R-CNN is not aimed to train for speed but the accuracy, mask prediction and bounding box prediction are done separately, it requires long time to train. It is generally expected with high accuracy comparing to regression/classification-based object detection model such as Single Shot Detector (SSD) predicting both together. The only two limitations will be time and hardware. The algorithm itself and the CNN backbone used are heavy and the dataset is not small, this requires large amount of time and even though model can be saved during training, using Google Colab will disconnect automatically after 12 hours, this brings inconvenience. Model with not huge number of training may not be able to give good result as shown in the last section. In our case, we used ResNet 50, if better results are desired, heavier CNN backbone such ResNet 101 can be used, however, this may require even more advanced hardware and time. Besides, due to the heavy architecture of Mask R-CNN, it is unable to apply on mobile devices at this stage because of the hardware and limited computational power.

# 4 Discussion and Conclusions

To sum up, the food recognition model trained by us works for detecting food items presented in the images, forming bounding boxes, outputting labels and masks, although the result is not as good as expected. Although Mask RCNN is a heavy model, we were successful in implementing it after a few hours of training.

In the future, to improve the model performance, different configurations and more training can be done, potentially may use more complex CNN backbone such as ResNet 101 with better hardware. To be more beneficial for users, detailed nutrient information will be added such as calories, minerals and macronutrients, and the system can be improved by having more complex algorithm and giving users dietary suggestions and linking to the related websites. This can be used to assist doctors to monitor or advise patients who require to adjust about the diets and help the treatment for medical purpose. For more general purposes, people can be beneficial from the application by logging multiple food items at a time. The detection system can be further developed as a mobile application and compatible with mobile devices for more efficient food tracking while advanced hardware is available for mobile devices in the near future, so that users can log food by using this model at any time.

# 5 References

AIcrowd n.d., *AIcrowd*, Food Recognition Challenge, viewed 17 June 2020, <https://www.aicrowd.com/challenges/food-recognition-challenge>.

He, K., Gkioxari, G., Dollar, P. & Girshick, R. 2017, ‘Mask R-CNN’, *2017 IEEE International Conference on Computer Vision (ICCV)*, IEEE, Venice, Italy, pp. 2980-8.

He, K., Zhang, X., Ren, S. & Sun, J. 2016, ‘Deep Residual Learning for Image Recognition’, *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, pp. 770-8.

# Appendices (Mandatory Section)

## A. Individual Contribution

## B. Individual Contribution Split

| Team Member Name | Percentage |
|---|---|
| Nazmul Kaonine | **100%** |
| Fernanda Santosa | **100%** |
| Minrui Li | **100%** |

All the team members have discussed and agreed on the above individual contribution to the project.

**Nazmul:** Group members were very contributing. Minrui really helped in analysing the results while Fernanda contributed in the model building. The GUI design was also Fernanda's idea and I personally think it is beautiful. Minrui helped organizing the submissions and kept everything in check. Overall we all looked upon gaps wherever we could fill in. Everyone participated in making this project stand up. We had punctual group meetings even during the pandemic. We conducted our team meetings at UTS and worked for full days. The meetings and work were organised and I am really happy to be a part of this group. Also, I believe I learned from my group members. Whenever we faced a problem, we worked together as a team to solve it. Lastly but most importantly, we all love food! It was a challenge in taking a heavy model like Mask RCNN but I am happy that we were successful in classifying and detecting the food objects. I would love to work again with this group in the future if the time comes.

**Minrui:**

I have written part of the report based on the results that we as team able to implement and we successfully create prediction model to classify including object food images. I also contribute and work collaboratively within the team for almost every group project that we planned. I also contributed in researching on image classification and object detection model to help the other team members creating successful prediction model and GUI.

**Fernanda :**

I have researched the project dataset that has full annotations which we able to implement on this dataset to our project. I have fully attend on every group work discussion and work collaboratively within the team throughout the course. I researched about the pretrained model and how does the food recognition works by using my knowledge in Deep Learning. I have also created GUI first attempt design and for the final GUI Design, we as a group were work together in creating the all the Image Classification Object Detection model including with the GUI.

**Nazmul Kaonine :** *Muhit*

**Fernanda Santosa:** Nanda

**Minrui Li:** Minrui