

EzzeRecipe: Ingredient detection and Recipe Recommendation Mobile Application Using CNN

by

Naznin Nahar

&

ID: 171014028

A project report submitted in partial fulfillment of the requirements for the degree of
Bachelor of Science in
Computer Science and Engineering

Supervised By:

Mohammed Ashikur Rahman, PhD



Computer Science and Engineering
University of Liberal Arts Bangladesh
688 Beribadh Road
Mohammadpur
Dhaka - 1207, Bangladesh
Bangladesh.

20, October 2022

DECLARATION

This project report is submitted to the Computer Science and Engineering, University of Liberal Arts Bangladesh in partial fulfillment of the requirements for the degree of Bachelor of Science. So, I hereby, declare that this project report is based on the surveys found by me. Materials of work found by other researchers are mentioned by reference. This project report, neither in whole, nor in part, has been previously submitted for any degree.

Naznin Nahar

ID: 171014028

Email Address: naznin.nahar.cse@ulab.edu.bd

CERTIFICATE OF APPROVAL

The project report entitled “EzzeRecipe” is submitted to the **Computer Science and Engineering**, University of Liberal Arts Bangladesh, Dhaka in partial fulfillment of the requirements for the degree of Bachelor of Science.

Dated: October 2022

Head of Department
Muhammad Golam Kibria, PhD
Associate Professor & Head (Acting), Computer
Science and Engineering
Bachelor of Science, Computer Science and
Engineering (CSE)
University of Liberal Arts Bangladesh

Signature & Date

Mohammed Ashikur Rahman, PhD
Computer Science and Engineering
University of Liberal Arts Bangladesh

Signature & Date

ACKNOWLEDGEMENT

First and foremost, I want to thank my creator for giving me strength, direction, and a sense of purpose throughout the journey to finish this project. I can't thank enough to my project supervisor Dr. Mohammed Ashikur Rahman sir. His patience, guidance, and support during the execution of this project boosted my inner strength to work harder. Through his expert guidance, inspiration, I was able to overcome all the obstacles that I encountered in the last three months. In fact, he always gave me immense hope every time I consulted with him over problems relating to my project. I am also grateful to my honorable faculties, friends, and my esteemed parents for encouraging and supporting me during the completion of this project.

ABSTRACT

The most fascinating fact of this “EzzeRecipe” Application is it implements the Convolutional Neural Network (CNN) method. This application is aimed to make cooking easy and interesting for beginners and build basic cooking skills. This mobile application recommends mouthwatering recipes by detecting ingredient images taken using mobile camera or gallery. CNN used for ingredient detection. The CNN model is based on custom ingredient dataset. The dataset includes 10 types of vegetable and other ingredient images captured with mobile phone. After labelling, preprocessing dataset was Segmented into train (70%) validation (10%) and test (20%) dataset. Then dataset was trained to build CNN model and converted into TensorflowLite model. Finally integrated to flutter Application. With 3300 images dataset, the model produces an excellent accuracy of 95.91%. Also tried EfficientNet on my custom dataset and got 60% accuracy. Flutter application is able to Fetch Recipe from database and display on mobile screen.

LIST OF TABLES

Table 2.1: Literature Review	5
Table 2.2: Difference between EzzeRecipe and Vegidiaries	8
Table 3.1: Camera Specification.....	11
Table 3.2: Data Segmentation Details	12

LIST OF FIGURES

Figure 1-1 System Architecture	1
Figure 2-1: Veggidiaries UI(1)	8
Figure 2-2: Vegediries UI	8
Figure 2-4: Yummly UI to scan ingredients	9
Figure 3-1 : Methodology	10
Figure 3-2: Preprocessing code.....	11
Figure 3-3: Dataset Preprocessing Code.....	12
Figure 3-4: Convolutional Neural Network.....	13
Figure 3-5: Model summary	14
Figure 3-6: Example of Maxpooling.....	15
Figure 3-7:Model Architecture	16
Figure 3-8: Model architecture code.....	17
Figure 3-9: Model Conversion using TensorFlow and keras.....	18
Figure 3-10 :Flow Diagram	19
Figure 3-11: Use Case Diagram.....	20
Figure 4-1 Accuracy Loss graph.....	21
Figure 4-2 Accuracy on Test Dataset.....	22
Figure 4-3 Epoch-Loss in different models	22
Figure 4-4 Test result plot 1	23
Figure 4-5 Test Result Plot 2	24
Figure 4-6: Initial Screen (GUI) of Ezzerecipe.....	25
Figure 4-7: GUI After Invalid Input of Ezzerecipe	25
Figure 4-8: GUI After detection (1) of Ezzerecipe	26
Figure 4-9:GUI After detection (2) of Ezzerecipe	27

Table of Contents

DECLARATION.....	I
CERTIFICATE OF APPROVAL	II
ACKNOWLEDGEMENT.....	III
ABSTRACT.....	IV
LIST OF TABLES	V
LIST OF FIGURES	VI
1. INTRODUCTION.....	1
1.1 Aim	2
1.2 OBJECTIVES	2
1.3 MOTIVATION	2
1.4 PROJECT SPECIFICATION	3
1.4 Listed are the specifications of the “EzzeRecipe” project	3
1.5 Set of requirements that are needed to achieve the specifications.....	3
2. LITERATURE REVIEW	4
2.1 EXISTING SIMILAR SYSTEMS	7
2.2 DIFFERENCE BETWEEN VEGGIDIARIES AND EZZEREcipe.....	8
3. METHODOLOGY	10
3.1 DATA COLLECTION	10
3.2 PRE-PROCESSING	11
3.3 MODEL TRAINING	13
4.3.1 CONVOLUTIONAL NEURAL NETWORK (CNN).....	13
3.3.2 Convolution Layers.....	14
3.3.3 Pooling Layers	15
3.3.4 Fully connected Layers	15
3.4 THE NEURAL NETWORK STRUCTURE USED IN EXPERIMENT	15
3.5 MODEL CONVERSION.....	18
3.6 INTEGRATION TO APPLICATION.....	18
3.7 FLOW DIAGRAM	18

4. RESULT AND ANALYSIS.....	21
4.1 ACCURACY AND LOSS GRAPH	21
4.1.1 Accuracy and loss graph of training and validation dataset	21
4.1.2 Accuracy on training and test dataset	22
4.2 TEST RESULT OF TRAINED TENSORFLOW MODEL	22
4.3 GUI	24
5. CONCLUSION AND FUTURE WORK	28
5.1 CONCLUSION.....	28
5.2 SOCIAL, LEGAL, ETHICAL & ENVIRONMENTAL ISSUES	28
5.3 FUTURE WORK	28
REFERENCES.....	29

1. INTRODUCTION

Expansion in the technology industry is bringing the world closer and easier by the day, and it is difficult to remain isolated without the touch of technology (Javed et al. 2019). Since the invention of the computer, smart people's daily lives have altered. We can now see that smart phones are taking over the world. It has surpassed the great computer innovation. Almost every computer function and many functions may now be accessed through a small tablet or smartphone. Apps on smart phones are one of the primary reasons that people are making their lives easier. As a result of increasing popularity of smartphones, the number of app developers is increasing by the day. Over the last five years, this expansion has also been felt in Bangladesh, where millions of people have begun to use smart phones and Android apps for the first time. These apps are mainly used to replace manual labor. The purpose of machine learning and computer vision is to enable computers to perceive data, comprehend data, and act on prior and current outcomes. Computer vision and machine learning research is still in its beginning phases. For recognizing the ingredients in real-time images, the Smartphone app employs a trained deep CNN model. On a customized ingredient recognition dataset, I trained the deep CNN ingredient recognition model. The tailored food identification dataset includes 15 different vegetable and ingredient variations. After examining the performance of various CNN models, the customized ingredient and vegetable dataset reached an ingredient recognition accuracy of 95%. The most fascinating part of this system is it can detect scanned ingredient and predict the ingredient and fetch recipe from the cloud database.

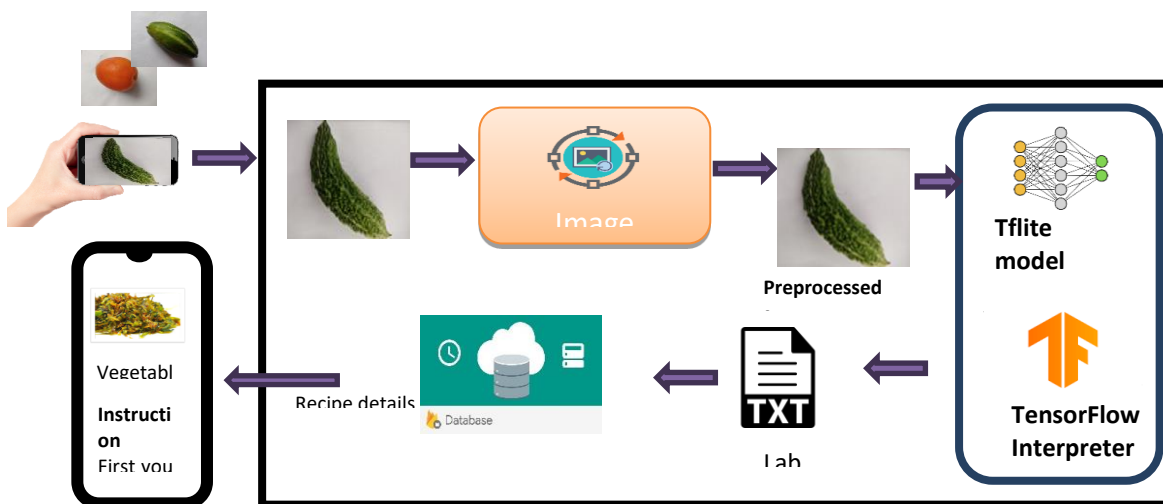


Figure 1-1 System Architecture

1.1 Aim

The system is addressed to save time and make tiring cooking interesting especially people who cooks daily by their own. Let's imagine a situation where someone returns home after a long tiring day and must cook dinner. which becomes very puzzling to decide to cook based on ingredients in fridge. But it would be much easier if by only scanning ingredient he/ she gets mouthwatering recipes displayed on his/her mobile screen. Fast food or restaurant food are not good for health. Also, the number of single household people increasing day by day many of them doesn't have basic cooking skill. This app is great help for beginners who does their cooking by themselves. This app helps to grow interest to cook at home in an effective way. It becomes more interesting to cook a delicious food with recipe details.

The main aim is to build a computer vision base mobile application which makes cooking easier.

1.2 Objectives

My technical objectives in case of developing this system are listed below-

1. Develop a cross platform computer vision-based application to detect ingredient by taking image with camera or from gallery.
2. Matching detected ingredient from cloud database and display recipe details with appealing image.

1.3 Motivation

Compared to quick meals and takeaway, homemade food is healthier, safer. When you cook at home, the produce's nutritious content is preserved. The takeaway you buy will typically contain a lot of oil, butter, and spices, fats which may not be healthy, especially if consumed frequently. But cooking can be boring and tiring. To reduce this boringness "EzzeRecipe" application makes cooking alluring and fun. Along with that what motivated most that is to build a computer vision-based application with comparatively unique idea.

1.4 PROJECT SPECIFICATION

1.4 Listed are the specifications of the “EzzeRecipe” project

- I. User can take photo using mobile camera
- II. User can upload photo from gallery
- III. User can see predicted ingredient name
- IV. By pressing Recipe button user can see recipe details corresponding with ingredient predicted.

1.5 Set of requirements that are needed to achieve the specifications

Programming Language: Dart, Python

Framework: Flutter

Database: Firebase

Software

Android-Studio

Jupiter Notebook

Hardware

Computer with 16 GB RAM

Android Phone 8 GB RAM

Android Phone Camera 64 MP

2. LITERATURE REVIEW

I researched about 25 papers. I shortlisted 15 papers about CNN image classification and object detection. From those 90% paper used vegetable and fruit dataset 10% papers used other data sets. 5 paper was about existing similar project available in market.

Anandajayam et al. have published a paper to find a reliable detection method by making use of tensor flow library, Faster R-CNN algorithm and SoftMax classifier used to detect the quality of fruit in 2018. Authors used CNN and SVM as machine learning model for their detection. In their research, researchers have got 95% accuracy with customized model of CNN and 73% accuracy for SVM. The authors planned to hybrid two ML techniques like CNN and SVM for getting better accuracy.

Adrish et al. have published a paper to detect vegetable image to recommend vegetarian food. In their research they used CNN with MobileNetV2 to detect vegetable in 2022. In their project researcher got about 95.78% accuracy. Authors planned to largen dataset to cover more vegetables. The research produced a website-based application to detect food with given image and recommend vegetarian food.

Fakhrou et al. published a paper where a smartphone-based system for recognizing the food dishes and fruits. The paper was published in 2021. Application targets to help for children with visual impairments. In this research Multiple state of art deep CNN ImageNet method was used to detect images. Authors achieved 95.55% accuracy to detect food and fruit images.

Canchila et al. published a paper to classify Colombian fruit image using CNN with AlexNet in 2020. Research achieved 98.12% accuracy with their model. In this research dataset was custom containing fruits with transparent bags and without bags images to enhance practical uses of the system proposed system.

Gill et al. published a paper to introduce a new, high-quality, fruit image dataset to better accuracy on CNN model. Authors also built a custom CNN model to classify fruit and vegetable images based on their dataset. Custom model produced 99.98 accuracy on test

dataset. Authors continuously largening 360degree dataset made it public. Many amazing projects used this dataset.

Aykul et al. published a paper on CNN image classification to deteect food images fro a diet tracking application. They used UEC Food datasets. Built a CNN model using DenseNet and MobileNetV2 architectures. Authors achieved 82% accuracy. Authors had plan to largen their dataset to get better accuracy.

Table 2.1: Literature Review

Author, Year	Objective	Method	Result	Limitation
(Anandajayam, and Kalaivani 2018)	To find a reliable detection method by making use of tensor flow library, Faster R-CNN algorithm and SoftMax classifier used to detect the quality of fruit.	CNN SVM	95% Accuracy 73% Accuracy	to get better accuracy they planned to try hybrid classification technique using CNN & SVM.
(Ardisa, Prabowo, and Rustad 2022)	Website-based system was developed to identify vegetable photos and suggest vegetarian dishes. The Kaggle vegetable image dataset was downloaded to build the CNN model.	CNN with MobileNetV2	95.78%	Largen dataset to cover more vegetable.
(Fakhrou, Kunhoth, and	Proposed a smartphone-based system for recognizing	Multiple state of art deep	95.55 %	Smartphone device less than 2 GB

Al Maadeed 2021)	the food dishes as well as fruits for children with visual impairments.	CNN ImageNet		memory causes delay in prediction.
(Alvarez-Canchila et al. 2020)	proposed an automated classification system for Colombian fruits. Authors trained a convolutional neural network for classification.	CNN AlexNet	98.12%	-----
(Gill et al. 2022)	In this paper, introduced a new, high-quality, fruit image dataset also CNN model to detect Fruit image. Fruit-360 dataset.	CNN 4 COV. Layer 4 MaxPooling Layer 2 Fully connected Layer 1 SoftMax Layer	99.98%	Constantly largening dataset
(Aykul 2019)	A diet tracking application. Trained and built model using CNN on UEC Food datasets.	CNN DenseNet and MobileNetV2 architectures	82%	To take advantage of RGBD more data need to be collected and augmented to get more accurate result.
(Bhavya et al. 2021)	Built a CNN model using Keras on custom dataset having 24 categories total 3924 images.	CNN 100 epochs Batch size 16	95.50%	-----
(Singh et al. 2022)	In this paper, used CNN models to detect and predict postharvest	CNN based (U-Net, DeepLab, Mask R)	94%	Planned to analyze various combination learning algorithms in diverse contexts to boost performance improve experimental

	deterioration zones in stored apple fruits.			outcomes to include more datasets.
(Bindal, Modi, and Kaushik 2020)	In this paper various image processing techniques Are discussed and used for fruit classification.	CNN Mobile Net 10 epochs Batch size 14	98.74 %	Planned to implement some other CNN based models to compare the accuracy also expand the dataset and include more category.
(Darapaneni et al. 2022)	Built models in Using CNN, Mobile Net to detect banana from image, determine sub-family and quality.	CNN NobileNet	93.4% 100%	Create a mode which can detect plastic bag wrapped banana also, creating a REST API

From above papers I learnt in what proportion dataset division should be. I learnt different CNN model architectures such as SVG, MobileNet, RESNET, U-NET, Alex-NET, VGG-16. There are many recipes recommendation app available in market, but I found few published recipe recommendation app which is based on computer vision. I found website-based application to detect vegetable images to recommend vegetarian recipes(Ardisa, Prabowo, and Rustad 2022).Also Yummly app which has a feature to scan ingredient and recommend recipe.

2.1 Existing similar systems

Veggediaries In this study the used convolutional neural network method with MobileNetV2 transfer learning and achieved excellent results in detecting vegetable images. The accuracy of this system to detect food is 95.78%. This (Ardisa, Prabowo, and Rustad 2022)study also produces a website-based application called "veggiediaries" to detect vegetable images and recommend vegetarian food recipes to users. User interface of veggiediaries looks like Figure 2,3.

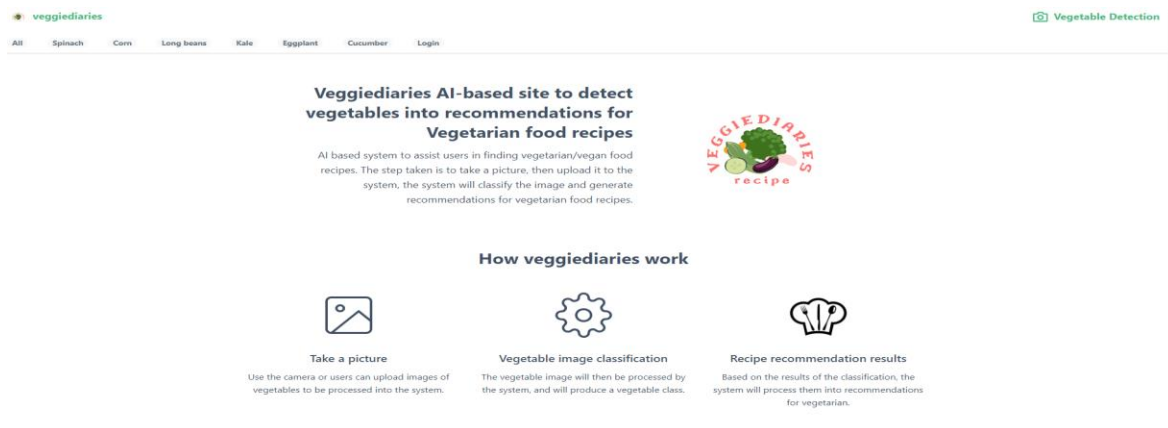


Figure 2-1: Veggidiaries UI(1)

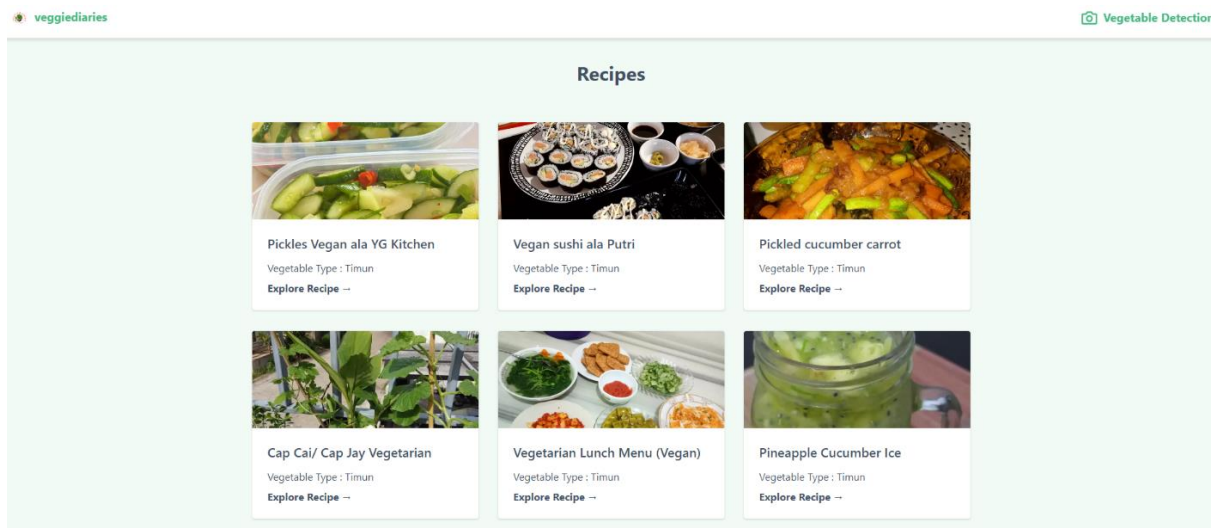


Figure 2-2: Veggidiaries UI

2.2 Difference between Veggidiaries and EzzeRecipe

Table 2.2: Difference between EzzeRecipe and Veggidiaries

	Veggidiaries	EzzeRecipe
Methods, algorithm used	Used CNN with MobileNet Transfer learning.	Used CNN and EfficientNet Custom TensorFlow Keras Model.
Dataset	Dataset downloaded from Kaggle.	Custom dataset with 10 category ingredient images

Type	Website based system	Cross platform mobile application
Result	95.78% accuracy	95.91%



Figure 2-3: Yummly UI to scan ingredients

Yummly

In Figure 4 The software scans different food items with your device's camera and labels each one that it recognizes to come up an appropriate recipe. The software will also provide step-by-step directions to avoid mistakes, as well as demonstrate how to make the right cuts for various sorts of food.

3. METHODOLOGY

As in the literature review chapter we have identified the uniqueness of EzzeRecipe application. I have presented the development methodology using various graphical representation, tables, and diagrams with description throughout this chapter.

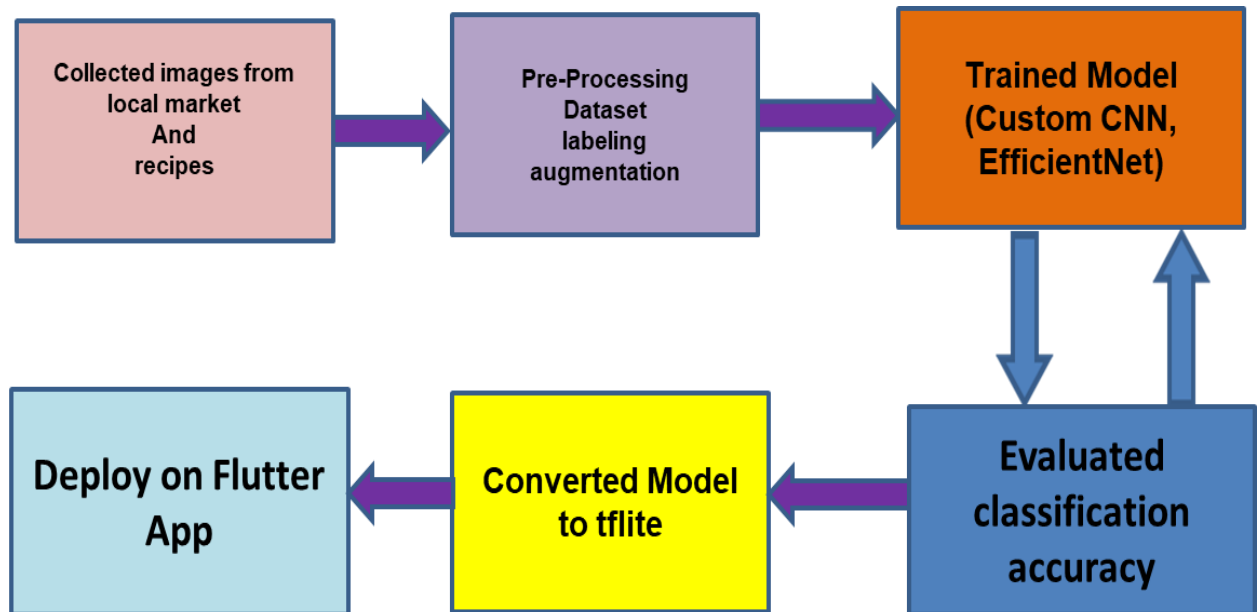


Figure 3-1 : Methodology

3.1 Data Collection

Dataset includes 3300 images of 10 ingredient each ingredient having 330 images in total. Images was taken by mobile camera and saved in 10 different folders named by ingredient name. It was made sure that all ingredient has same number of images. Blurred and partial images was replaced with good quality ingredient images.

```
In [1]: import numpy as np
import tensorflow as tf

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import os

dataset_path = os.listdir('imagesds')
print (dataset_path)

print("Types of classes labels found: ", len(dataset_path))

['Bitter_gourd', 'Capsicum', 'Carrot', 'Cauliflower', 'Egg', 'Lemon', 'Pointed_gourd', 'Potato', 'Spine_gourd', 'Tomato']
Types of classes labels found: 10
```

All the images were clicked with white/black background so that the noise level in learning is reduced, the specifications of the camera used in Table 2:

Table 3.1: Camera Specification

Resolution	4608 x 2592
File Size	5MB to 7 MB
Focal Length	25mm
Aperture	f/1.89
Megapixel	64-MP
ISO	3893

I also collected popular recipe matched with ingredient dataset. Recipe images and details was collected in the initial phage of this project.

3.2 Pre-Processing

At first Dataset segmentation was done. I split dataset by 70% train data ,20% for test data and 10% for validation data. Using get_dataset_partitions_tf method shown below.

```
In [6]: def get_dataset_partitions_tf(ds,train_split=0.7,val_split= 0.1, test_split=0.20, shuffle = True,shuffle_size=1000):  
        ds_size=len(ds)  
        if shuffle:  
            ds = ds.shuffle(shuffle_size, seed = 12)  
        train_size = int(train_split * ds_size)  
        val_size = int(val_split*ds_size)  
  
        train_ds=ds.take(train_size)  
  
        val_ds = ds.skip(train_size).take(val_size)  
        test_ds = ds.skip(train_size).skip(val_size)  
  
        return train_ds, val_ds,test_ds  
  
In [8]: train_ds,val_ds,test_ds = get_dataset_partitions_tf(dataset)
```

Figure 3-2: Preprocessing code

Table 3.2: Data Segmentation Details

Ingredient Name	Number of train images	Number of validation images	Number of Test images
Bitter gourd	231	33	66
Capsicum	231	33	66
Carrot	231	33	66
Cabbage	231	33	66
Egg	231	33	66
Lemon	231	33	66
Pointed gourd	231	33	66
Spine gourd	231	33	66
Tomato	231	33	66

Then I shuffled dataset all three-segment train dataset, validation dataset and test dataset. I resized all images by 256x256 size. Rescaled images. After that data augmentation was done on train dataset with randomly flipping images, random rotation, and random zoom.

```
In [10]: train_ds = train_ds.cache().shuffle(1000).prefetch(buffer_size = tf.data.AUTOTUNE)
        val_ds = val_ds.cache().shuffle(1000).prefetch(buffer_size = tf.data.AUTOTUNE)
        test_ds = test_ds.cache().shuffle(1000).prefetch(buffer_size = tf.data.AUTOTUNE)

In [11]: resize_and_rescale = tf.keras.Sequential([
        layers.experimental.preprocessing.Resizing(IMAGE_SIZE, IMAGE_SIZE),
        layers.experimental.preprocessing.Rescaling(1./255),
    ])

In [12]: data_augmentation = tf.keras.Sequential([
        layers.experimental.preprocessing.RandomFlip("horizontal_and_vertical"),
        layers.experimental.preprocessing.RandomRotation(0.2),
        layers.experimental.preprocessing.RandomZoom(.5, .2),
    ])

In [13]: train_ds = train_ds.map(
        lambda x, y: (data_augmentation(x, training=True), y)
    ).prefetch(buffer_size=tf.data.AUTOTUNE)
```

Figure 3-3: Dataset Preprocessing Code

3.3 Model Training

In this section I described the training process of CNN model. As I used convolutional neural network to build my model, each step described below.

4.3.1 CONVOLUTIONAL NEURAL NETWORK (CNN)

CNN is a multilayered neural network that can recognize, categorize, and identify every feature in an image in real time. CNN is the best image classification algorithm. A CNN can be trained to perform image processing tasks such as object detection, segmentation, classification, and picture processing. CNNs are neural networks, like the human brain. These neurons are made up of weights and biases that build layers and fire in a specific order to produce a final output. Large amounts of data can be fed into the networks to teach them to recognize patterns. This is extremely significant in the field of computer vision because it means that a computer may be taught to recognize various objects.

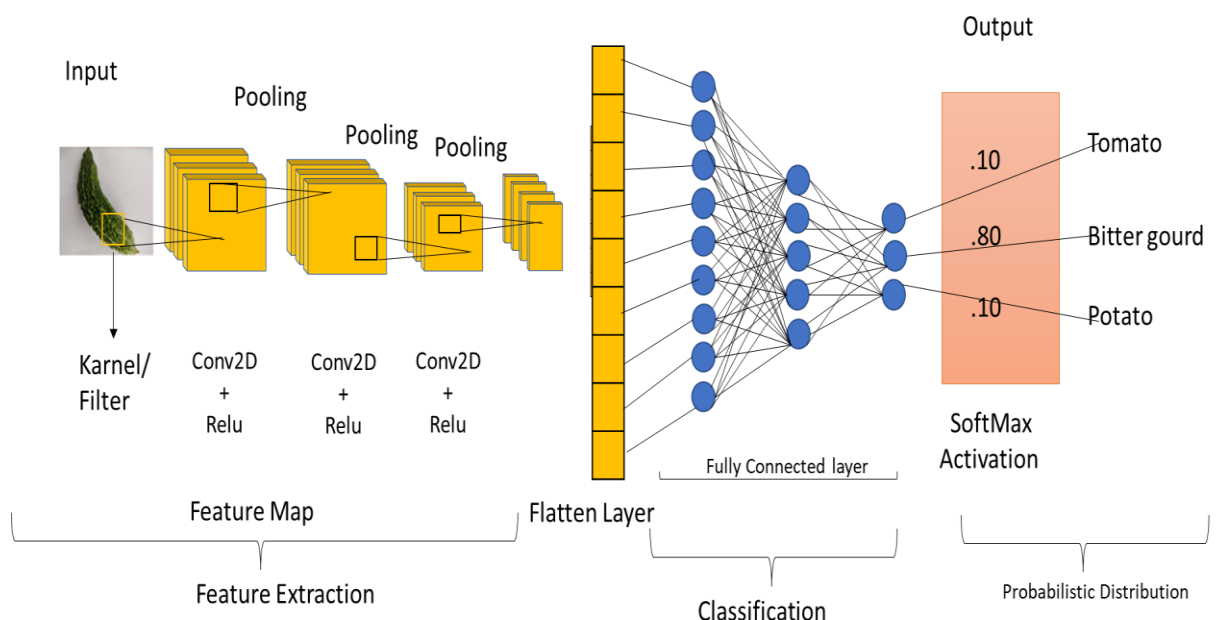


Figure 3-4: Convolutional Neural Network

To train the model I used 6 convolution layers each followed by 6 maxPooling layer, one flattens layer and two dense layers. Total params: 183,682 and total trainable parameters was also 183,682.

```
In [14]: model.summary()
```

```
Model: "sequential_2"
```

Layer (type)	Output Shape	Param #
sequential (Sequential)	(32, 224, 224, 3)	0
conv2d (Conv2D)	(32, 222, 222, 32)	896
max_pooling2d (MaxPooling2D)	(32, 111, 111, 32)	0
conv2d_1 (Conv2D)	(32, 109, 109, 64)	18496
max_pooling2d_1 (MaxPooling2D)	(32, 54, 54, 64)	0
conv2d_2 (Conv2D)	(32, 52, 52, 64)	36928
max_pooling2d_2 (MaxPooling2D)	(32, 26, 26, 64)	0
conv2d_3 (Conv2D)	(32, 24, 24, 64)	36928
max_pooling2d_3 (MaxPooling2D)	(32, 12, 12, 64)	0
conv2d_4 (Conv2D)	(32, 10, 10, 64)	36928
max_pooling2d_4 (MaxPooling2D)	(32, 5, 5, 64)	0
conv2d_5 (Conv2D)	(32, 3, 3, 64)	36928
max_pooling2d_5 (MaxPooling2D)	(32, 1, 1, 64)	0
flatten (Flatten)	(32, 64)	0
dense (Dense)	(32, 64)	4160
dense_1 (Dense)	(32, 3)	195

```

=====
Total params: 171,459
Trainable params: 171,459
Non-trainable params: 0

```

Figure 3-5: Model summary

3.3.2 Convolution Layers

Convolution is an action on two functions in mathematics that results in a third function that is the convolved form of one of the original functions. The foundation of a CNN is

a convolutional layer. It has several filters (or kernels), whose settings must be learned over the course of training. Typically, the filters' size is smaller than the original image. Each filter produces an activation map after it convolves with the image.

3.3.3 Pooling Layers

Pooling layers are used to shrink image, to reduce pixel size to lower the amount of computation done in the network and faster the training process. pooling layers also used to control overfitting. 2x2 with stride 2 is the most used pooling layer.

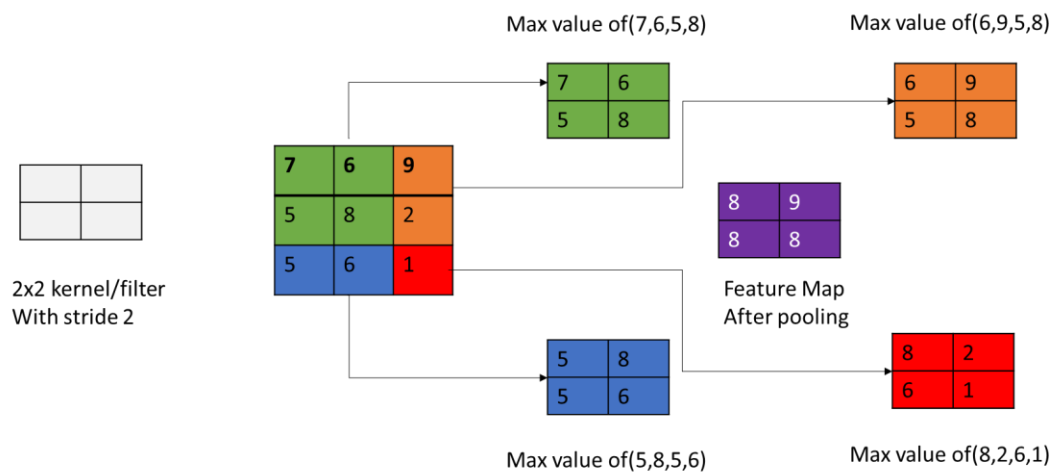


Figure 3-6: Example of Maxpooling

3.3.4 Fully connected Layers

In a neural network, fully connected layers are those in which all the inputs from one layer are connected to every activation unit of the next layer. The final few layers of most popular machine learning models are full connected layers that compile the data extracted by previous layers to form the final output.

3.4 The Neural Network Structure used in experiment

In my project I have used a custom convolutional neural network. As previously described this type of network makes use of convolutional layers, pooling layers, fully connected layers, and dense layers. In a typical CNN architecture, each convolutional layer is followed by a Rectified Linear Unit (ReLU) layer, then a Pooling layer then one

Figure 3-8: Model architecture code

Fig.3 illustrates the visual representation of the neural network used in this project.

- The first layer (Convo2D), which applies 32 x 3 filters, is a convolutional layer. On this layer, I used max pooling with a 2 x 2 stride 2 filter that specifies the pooled zones do not overlap (MaxPooling2D). Additionally, the width and height are both reduced to 50 pixels.
- The 64 x 3 x 3 filters with activation function "Relu" are applied by the second convolutional (Convo2D 1) layer. I use the same MaxPooling2D 1 method, shape 2 x 2, and stride 2 on this layer as I used on the first.
- The third convolutional layer (Convo2D 2) has 64 x 3 x 3 filters. Another max pool layer (MaxPooling2D 2) with shape 2 x 2 and stride 2 follows.
- The fourth convolutional (Convo2D_3) layer applies 64 x 3 x 3 filters after which we apply a final max pool layer (MaxPooling2D_3).
- The fifth convolutional layer (Convo2D 4) contains 64 x 3 x 3 filters, followed by a final max pool layer (MaxPooling2D 4).
- The sixth convolutional (Convo2D_5) layer applies 64 x 3 x 3 filters after which we apply a final max pool layer (MaxPooling2D_5).
- The seventh layer is Flatten layer is used to make the multidimensional input one-dimensional to pass it onto a Dense layer. Flatten layer has 32 x 256 size.
- The eighth layer is the Dense layer, which is a fully connected layer with 64 kernels and the activation function 'Relu.'
- Dense_1 therefore the ninth and the last layer of this convolutional neural network, which is a fully connected layer (Dense_1), takes number of classes here,10 with 'SoftMax' activation function inputs.

EfficientNet

Deep learning architectures seek to uncover more efficient techniques using fewer models. Unlike other cutting-edge models, EfficientNet provides more efficient outcomes by consistently increasing depth, width, and resolution while scaling down the model. The first stage in the compound scaling approach is to look for a grid that will help to figure out the relationship between the several scaling dimensions of the baseline network while working with a fixed resource limitation. A suitable scaling factor for the depth, width, and resolution dimensions is therefore calculated. After that, the coefficients are used to scale the baseline network to the desired target network (Louis 2013). The key building component for EfficientNet is the inverted bottleneck

MBConv, which was first introduced in MobileNetV2, however it is employed significantly more than MobileNetV2 because to the larger FLOPS (floating point operations per second) budget. In MBConv, blocks are made up of a layer that expands and then compresses the channels, resulting in direct connections between bottlenecks that connect many fewer channels than expansion layers.

3.5 Model conversion

After training and testing I converted the TensorFlow model to tflite model with the help of TensorFlow's converter library. Converted model is lighter than original TensorFlow model easy to integrate to mobile application.

```
In [44]: import tensorflow as tf

model = tf.keras.models.load_model('../image_class_blank1.h5')
converter = tf.lite.TFLiteConverter.from_keras_model(model)
tflite_model = converter.convert()
open("converted_model.tflite", "wb").write(tflite_model)
```

Figure 3-9: Model Conversion using TensorFlow and keras

3.6 Integration to Application

I saved the tflite model and label.txt file in app storage in flutter application. Then I used tflite plugin to build connection between tflite model and mobile application. Implementation details and GUI discussed in next chapter.

3.7 Flow Diagram

This is the flow diagram of Application

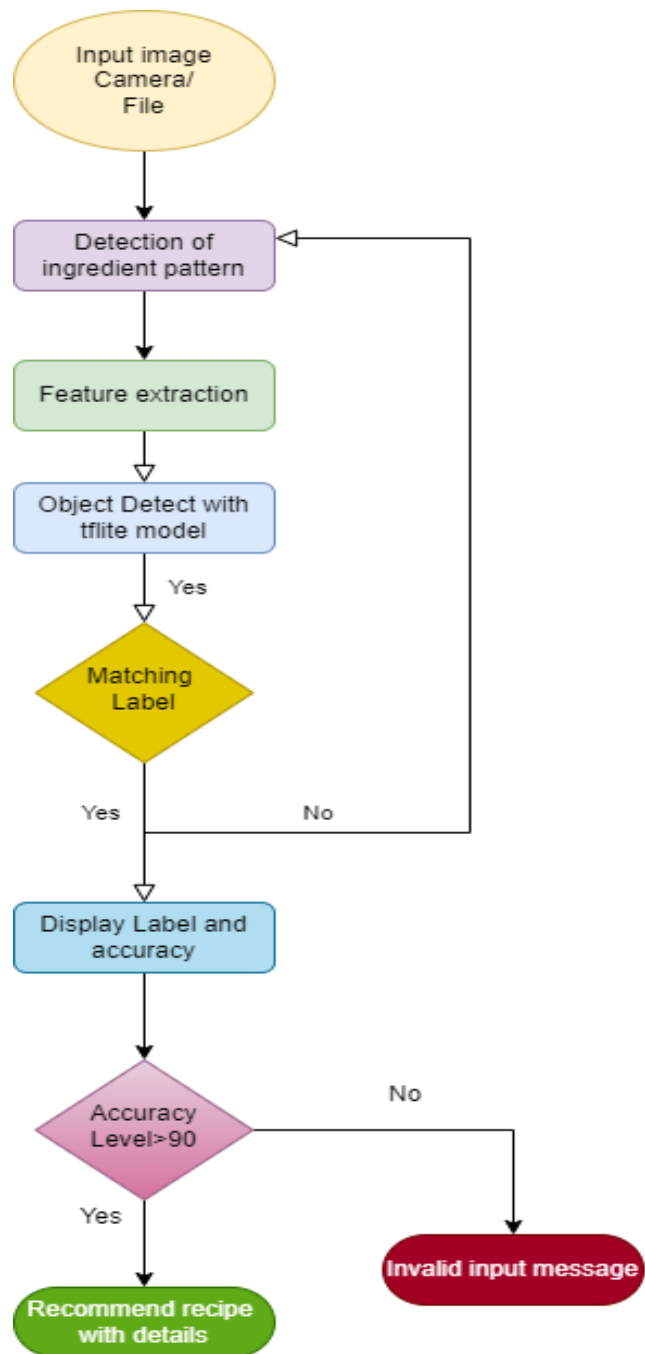


Figure 3-10 :Flow Diagram

The application takes an image as an input detects ingredient pattern, detects features of ingredient with the help of trained model. Displays detected label with accuracy. If accuracy is higher than 90%. Displays recipe details on the other hand accuracy less than 90% gives an invalid input message.

3.8 Use-Case Diagram

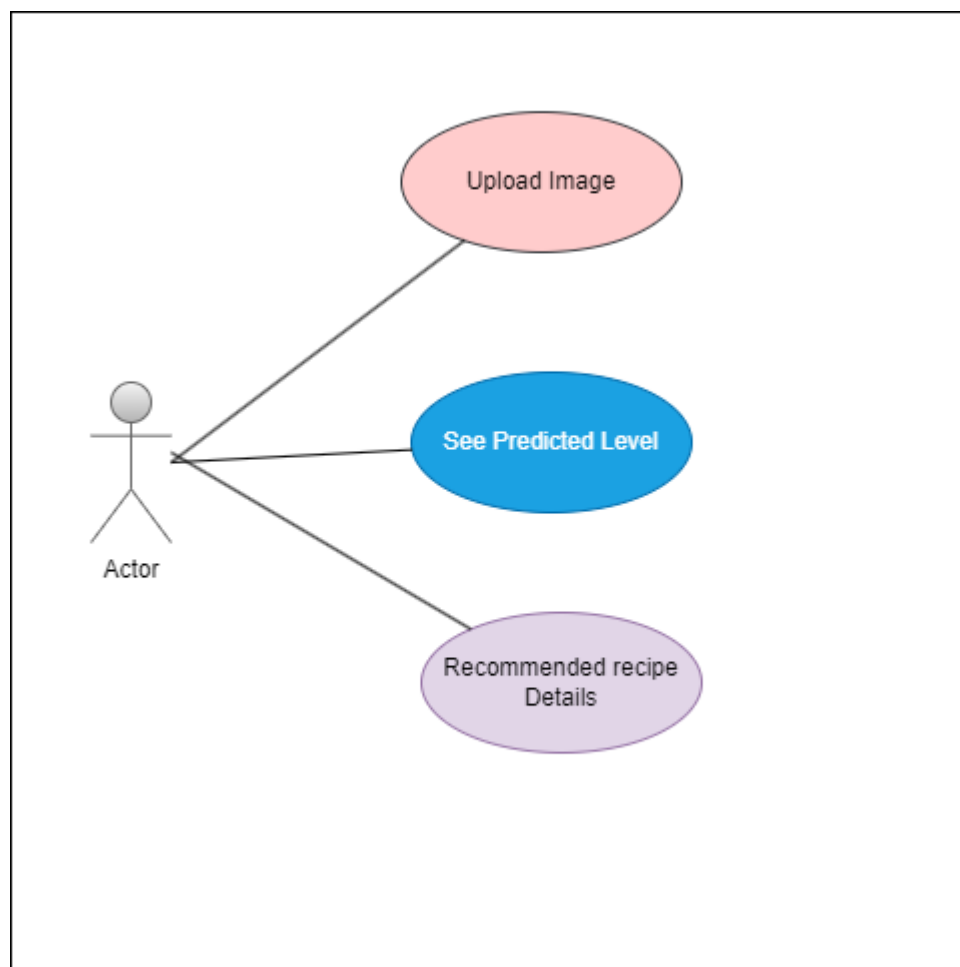


Figure 3-11: Use Case Diagram

In the above use case diagram, we can see there is one actor and three activities. If user open EzzeRecipe app, can upload image from gallery or take a photo using mobile camera. After selecting image user can see predicted label or ingredient name. By clicking Next button user can see recipe details in phone screen.

4. RESULT AND ANALYSIS

The methodology showed the development process of our system. We can see the overall development process can be divided into two major portions, Model training and flutter application. The development consists of two major parts.

- Model training, testing and
- Flutter application.

4.1 Accuracy and loss graph

4.1.1 Accuracy and loss graph of training and validation dataset

Figure 7 illustrates how accuracy and loss increased during training. On a total of 50 epochs, it can be seen that training rapidly advances over the first 10 epochs (accuracy increases to better than 90%). Following that, the advancements slow down over the remaining epochs.

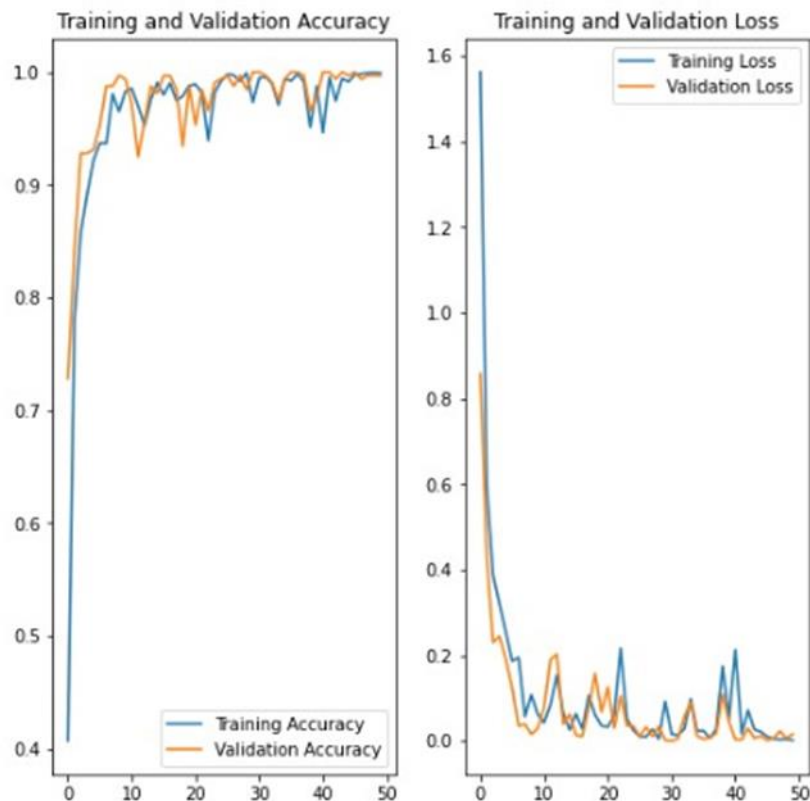
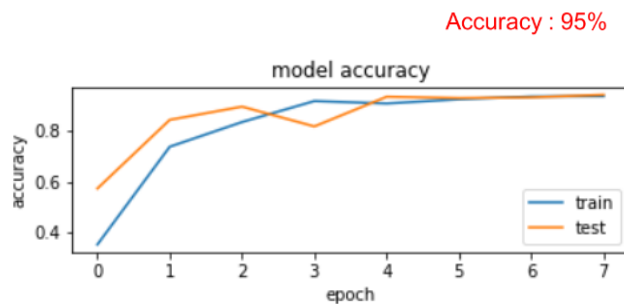


Figure 4-1 Accuracy Loss graph

4.1.2 Accuracy on training and test dataset

In figure 8 accuracy improvement becomes steady after 4 total of 7 epochs.



```
] : scores = model.evaluate(test_ds)
```

```
13/13 [=====] - 50s 147ms/step - loss: 0.1313 - accuracy: 0.9591
```

```
] : scores
```

```
] : [0.13125912845134735, 0.9591346383094788]
```

Figure 4-2 Accuracy on Test Dataset

Epoch Accuracy on the different trained models

I trained more than 11 model to get height accuracy. In figure 16 we can see TensorBoard graphical representation of epoch accuracy on train and validation data while training.

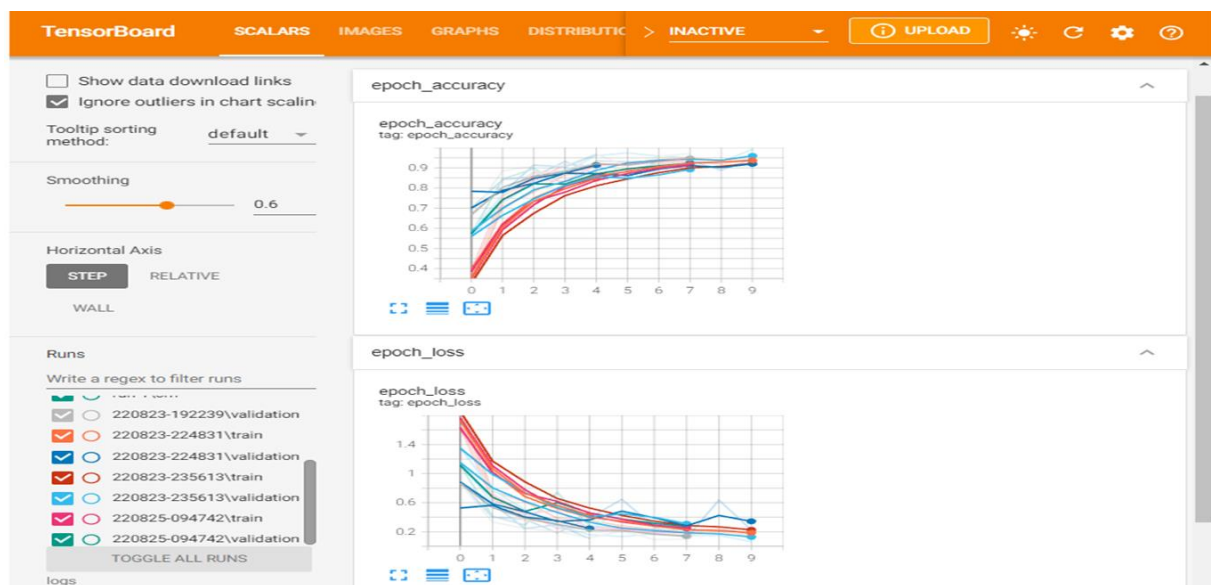


Figure 4-3 Epoch-Loss in different models

4.2 Test Result of trained TensorFlow Model

In below image Figure 19 I plotted 3x3 images with confidence actual and predicted label after training model. I plotted the images with the help of matplotlib.pyplot library.

As we can see in Figure 11 that plotted images are Bitter gourd, Lemon, Carrot, Pointed Gourd, and tomato. The trained model predicted 8 images among 9 successfully with 100% confidence. Only one image Lemon has 99.98% confidence which is very closer to 100%.

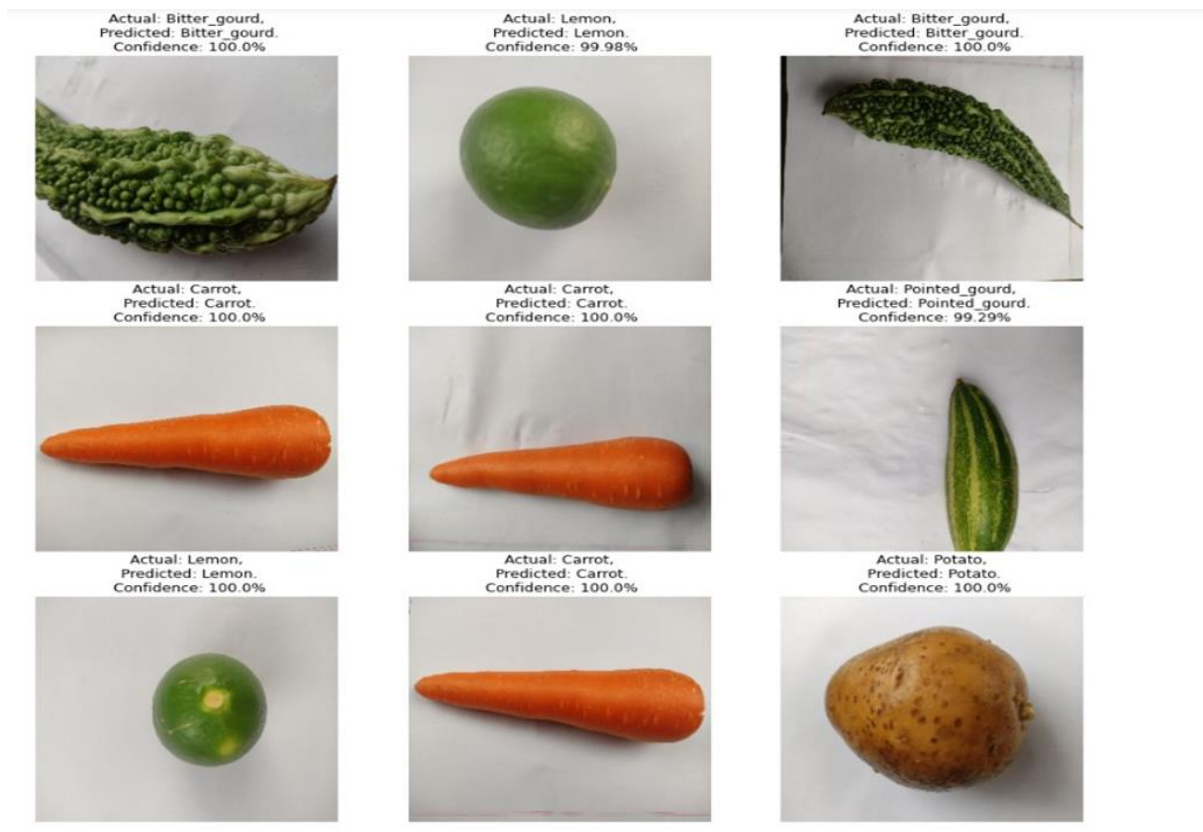


Figure 4-4 Test result plot 1

In Figure 20 image I plotted 2x2 images with confidence and predicted label and Actual label from test dataset which is completely new images for the trained model.

We can see first image's actual label is Potato and model successfully predicted the label of potato with 100% accuracy.

In the first row second image's actual label is lemon and model predicted the label with 96.97% confidence.

In second row first image for Tomato image model did predict the correct label but confidence percentage is quite low as compared to other images.

The last image is a Cauliflower image, and model successfully predicted its label with 100% accuracy.

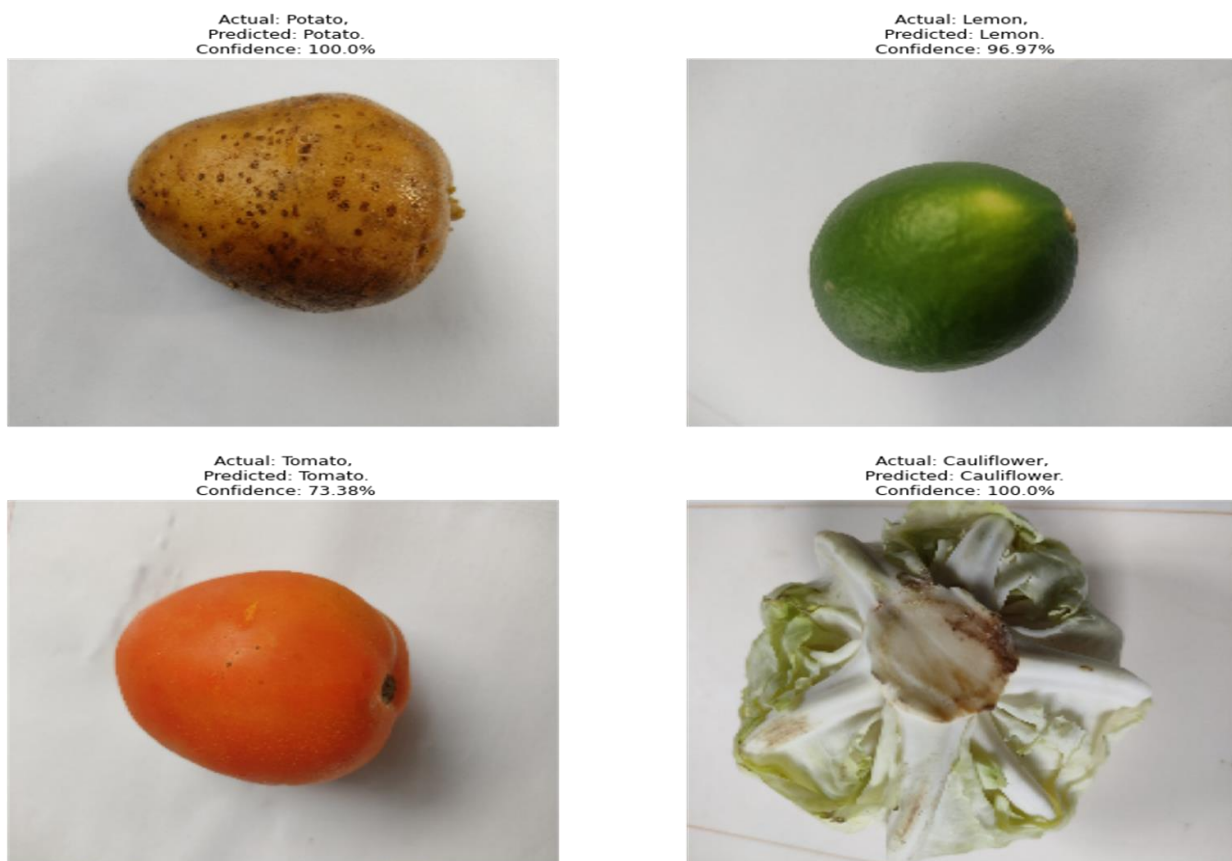


Figure 4-5 Test Result Plot 2

4.3 GUI

I developed my Flutter application in Android studio. The app has mainly 2 screen to avoid complexity and better user experience.

The first screen let user to choose image and second screen displays the recipe details.

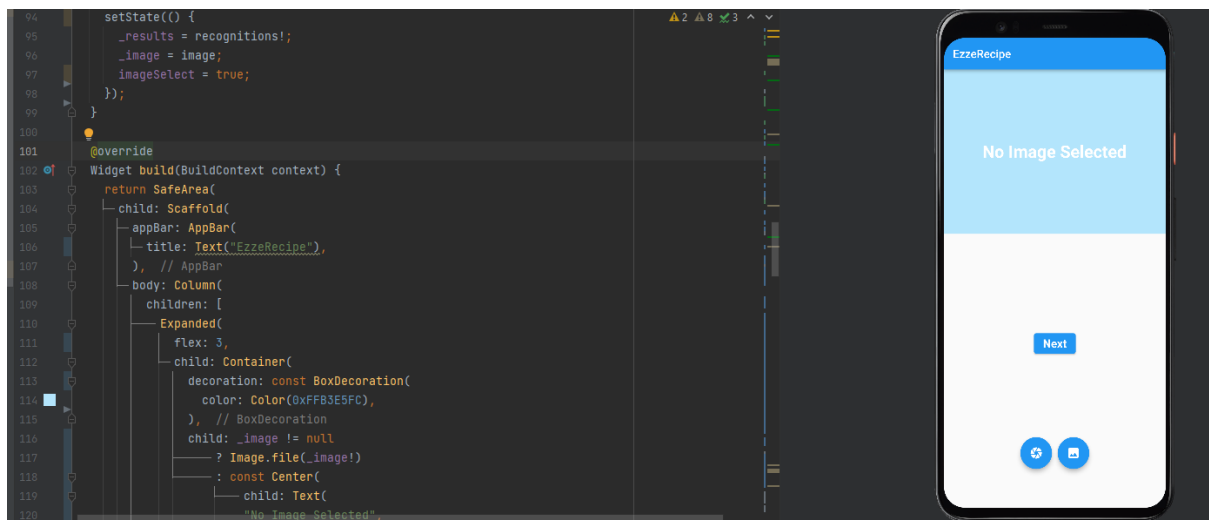


Figure 4-6: Initial Screen (GUI) of Ezzerecipe

Initial Screen (GUI)

User will see screen as Figure 4-6 emulator display screen. Clicking Camera icon will open camera on display to take photo of ingredient, clicking file icon open gallery to select image from gallery.

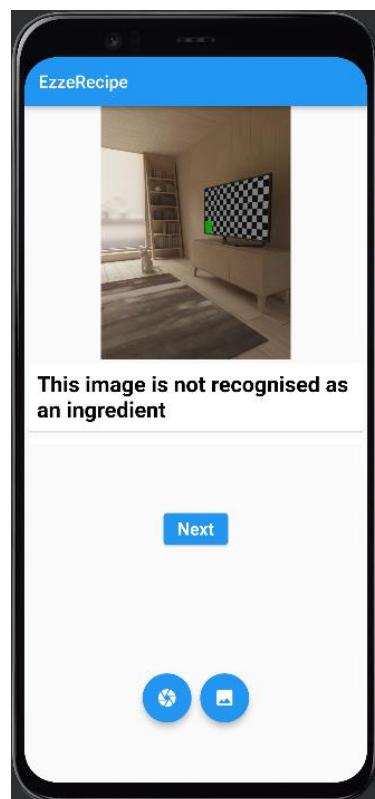


Figure 4-7: GUI After Invalid Input of Ezzerecipe

GUI after taking non ingredient Image

If taken image is not ingredient display will look like as shown below. Camera and File icon will appear to reselect image Figure 14. It displays a message “The image is not recognized as an ingredient image”. Camera and file icon stays for reselecting image.

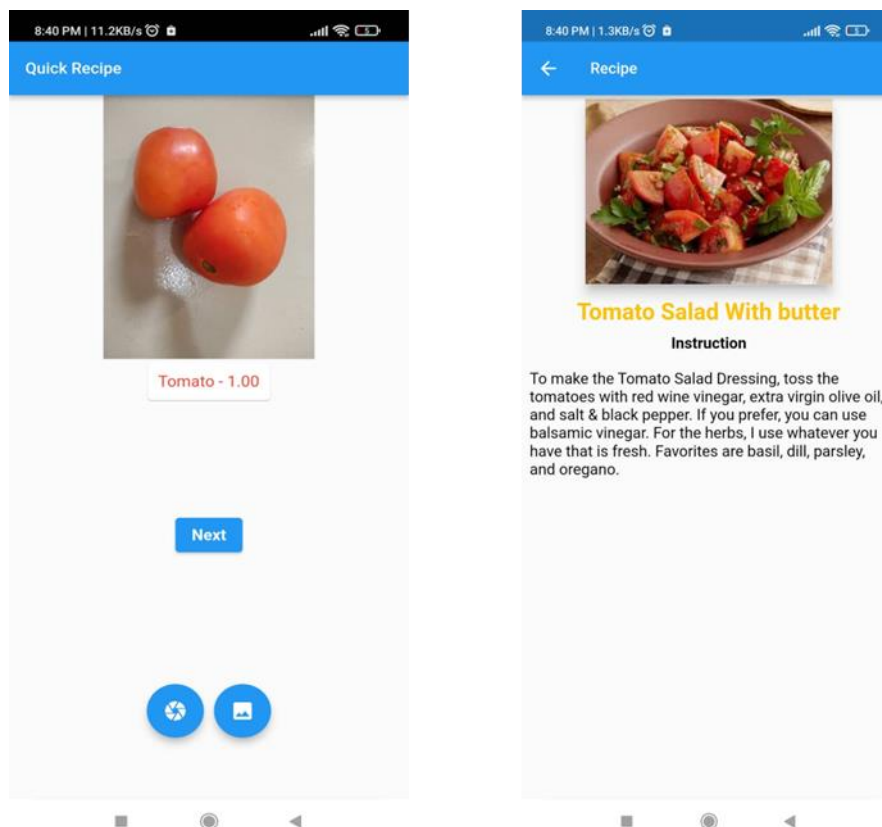


Figure 4-8: GUI After detection (1) of Ezzerecipe

GUI After detecting Image

Predicted label or ingredient name and confidence appears below taken image as in Figure 15. By clicking Next button Recipe details page appears with recipe image. Here we can see application detected Tomato image successfully with 100% confidence. Also, in Next screen Recipe details with image appears. For Tomato ingredient showing Tomato butter salad with instruction.

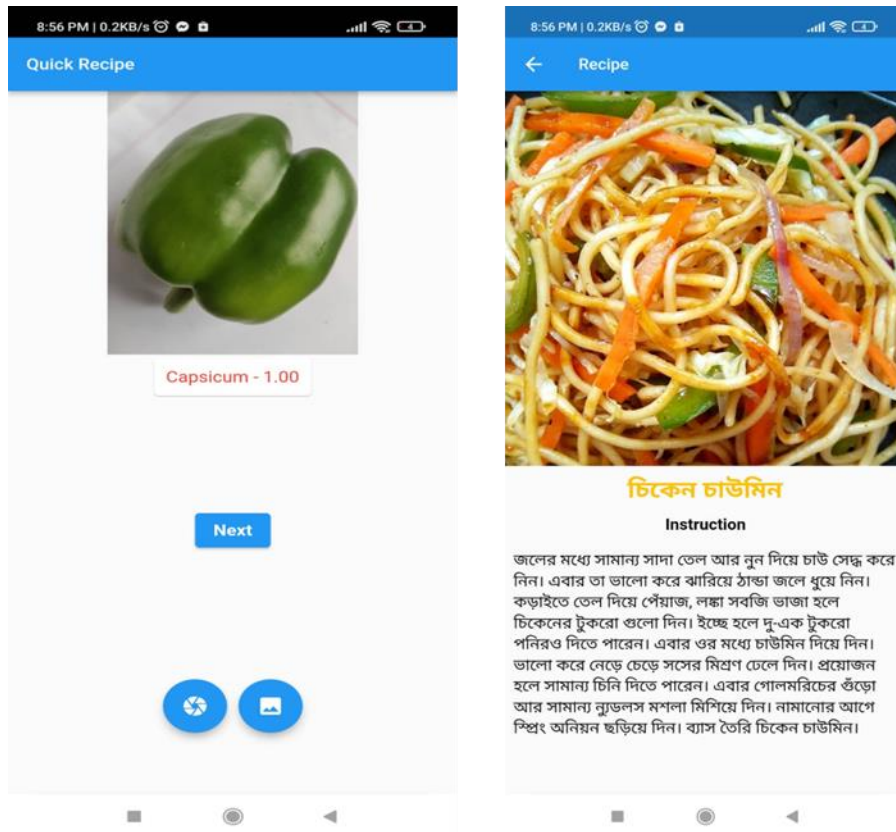


Figure 4-9: GUI After detection (2) of Ezzerecipe

After selecting or taking an image predicted label or class name of ingredient and confidence appears below taken image as in Figure 16. By clicking Next button Recipe details page appears with recipe image. Here we can see application successfully detected Capsicum image successfully with 100% confidence. Also, in Next screen Recipe (Chicken chow mien) details with image appears.

5. CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

This Project concludes that the Convolutional Neural Network (CNN) technique has excellent results in detecting all ingredient images. The level of accuracy reaches a value of 95.91%. This project also produces a mobile-based application called "EzzeRecipe" to detect vegetable images, ingredient imaged and recommend mouthwatering recipes to users. As the number of nuclear-family households grows, it becomes essential for any individual to have a basic knowledge of cooking. Numerous beginners who are just beginning to learn to cook may not know the names of many items or recipes, making it difficult for them to cook. Homemade foods are comparatively much healthier than restaurant food. EzzeRecipe makes cooking easy and interesting.

5.2 SOCIAL, LEGAL, ETHICAL & ENVIRONMENTAL ISSUES

According to our planning of the development of this application, we can ensure that the solution to the design problem is safe to the public and the environment. Primarily, this system will not need any special permission of the user's device like permission for phone calls, messages, and internet connection. It is planning to utilize only the permission to, camera and photos in gallery. It can also have a great social impact, because in today's digital world where everything is getting digitalized people prefer more involvement of technology in every phase of their lives. This project throws a light in this concept, EzzeRecipe will grow more interest to cook and eat homemade food. According to several research, people who cook more frequently rather than get takeout consume a diet that is generally healthier.

5.3 FUTURE WORK

I have plan to largen my dataset which cover many more ingredients to get more accurate result. I will also use Transfer learning to get effective result on larger dataset with Efficient NET, Mobile-Net. I have idea to add more feature in the application such as add to favorite list, user details, cooking alert. Currently app is fetching recipes from in app storage, my future plan is to add firebase cloud store to the flutter application. Where I can store a large recipe database.

REFERENCES

1. Alvarez-Canchila, O. I., D. E. Arroyo-Pérez, A. Patiño-Saucedo, H. Rostro González, and A. Patiño-Vanegas. 2020. "Colombian Fruit and Vegetables Recognition Using Convolutional Neural Networks and Transfer Learning." *Journal of Physics: Conference Series* 1547 (1). <https://doi.org/10.1088/1742-6596/1547/1/012020>.
2. Ardisa, Kevin Aliffanova, Wahyu Aji Eko Prabowo, and Supriadi Rustad. 2022. "Implementation of Convolutional Neural Network Method for Detecting Vegetables as Recommendation for Vegetarian Food Recipes." *Lecture Notes in Engineering and Computer Science* 2244: 38–41.
3. Associate, S Pariselvam, P Anandajayam, and S Kalaivani. 2018. "Detection of Fruit Quality by CNN Algorithm" 20 (6): 80–85. <https://doi.org/10.9790/0661-2006018085>.
4. Aykul, Mete. 2019. "Improving Food Recognition for Mobile Applications via Convolutional Neural Networks and User Habits."
5. Bhavya, J K, B S AC, K Gayithri, B L Keerthi, and M Y MG. 2021. "The Literature Survey on Intra Class Fruits and Vegetable Recognition System Using Deep Learning." *Irjmets.Com*, no. 07: 350–55. https://www.irjmets.com/uploadedfiles/paper/volume3/issue_7_july_2021/14017/1628083545.pdf.
6. Bindal, Namita, Rajat Modi, and Brajesh K. Kaushik. 2020. "Fiscal Classification Using Convolutional Neural Network" 5 (8): 55. <https://doi.org/10.1117/12.2568314>.
7. Darapaneni, Narayana, Arjun Tanndalam, Mohit Gupta, Neeta Taneja, Prabu Purushothaman, Swati Eswar, Anwesh Reddy Paduri, and Thangaselvi Arichandrapandian. 2022. "Banana Sub-Family Classification and Quality Prediction Using Computer Vision." <https://arxiv.org/abs/2204.02581v1>.
8. Fakhrou, Abdalnaser, Jayakanth Kunhoth, and Somaya Al Maadeed. 2021. "Smartphone-Based Food Recognition System Using Multiple Deep CNN Models." *Multimedia Tools and Applications* 80 (21–23): 33011–32. <https://doi.org/10.1007/s11042-021-11329-6>.
9. Gill, Harmandeep Singh, Ganpathy Murugesan, Baljit Singh Khehra, Guna Sekhar Sajja, Gaurav Gupta, and Abhishek Bhatt. 2022. "Fruit Recognition from Images Using Deep Learning Applications." *Multimedia Tools and Applications* 81 (23): 33269–90. <https://doi.org/10.1007/s11042-022-12868-2>.
10. Javed, Aaquib, Monika Zaman, M. Monir Uddin, and Tasnova Nusrat. 2019. *An Analysis on Python Programming Language Demand and Its Recent Trend in Bangladesh. ACM International Conference Proceeding Series*. Springer International Publishing. <https://doi.org/10.1145/3373509.3373540>.
11. Singh, Abha, Gayatri Vaidya, Vishal Jagota, Daniel Amoako Darko, Ravindra Kumar Agarwal, Sandip Debnath, and Erich Potrich. 2022. "Recent Advancement in Postharvest Loss Mitigation and Quality Management of Fruits and Vegetables Using Machine Learning Frameworks." *Journal of Food Quality* 2022. <https://doi.org/10.1155/2022/6447282>.