Mini Project Report

Pesticide Assistant Application

Submitted by

Hritik Rao (111907020)

Mahesh Mali (111907026)

Prathamesh Ganorkar (111907039)

Under Supervision of:

Dr. S. P. Mahajan

Prof. Neelima Kolhare

DEPARTMENT OF ELECTRONICS & TELECOMMUNICATION ENGINEERING

COLLEGE OF ENGINEERING, PUNE

April 2022

CERTIFICATE

This is to certify that the Mini Project entitled

Pesticide Assistant Application

Submitted By

Hritik Rao

Mahesh Mali

Prathamesh Ganorkar

is a bonafide work carried out by them under the supervision of **Prof. Neelima Kolhare** and it is approved for the partial fulfillment of the requirements of T.Y. E&TC Engineering submitted to College Of Engineering, Pune.

The Mini Project work has not been earlier submitted to any other institute or university for the award of degree or diploma.

Prof. Neelima Kolhare	**Dr. S. P. Mahajan**	**Dr. B. B. Ahuja**
Guide	Head	Director
Department of E&TC	Department of E&TC	COEP

Place: Pune

Date: 16 / 04 / 22

Acknowledgements

We are feeling very humble in expressing our gratitude. It will be unfair to bind the precious help and support which we got from many people in a few words. But words are the only media of expressing one's feelings and our feeling of gratitude is absolutely beyond these words. It would be our pride to take this opportunity to say thanks.

We are pleased to acknowledge **Dr. S. P. Mahajan, Head,** Department of E&TC for his co-operation, useful suggestions and his invaluable guidance during this course of project work.

We extend our sincere thanks to **Prof. Neelima Kolhare** who continuously helped us throughout the project and without her guidance, patience and support, this project would have been an uphill task. We would also like to thank **Dr. B. B. Ahuja, Director, College of Engineering Pune**. He always remains a source of inspiration for us to work hard and dedicatedly.

Last but not the least, it is the love and blessings of our families and friends which drove us to complete this project work.

April 2022

Hritik Rao (111907020)

Mahesh Mali (111907026)

Prathamesh Ganorkar (111907039)

Abstract

The Idea of Designing a Pesticide Assistant Application is born with the observation that as the population is increasing exponentially, there is a pressing need for the introduction of advanced technology and equipment to extract maximum of pesticides to better accommodate this increasing demand. We are aiming to have a dedicated Application to recognize pesticides by employing real time image processing.

This application is aimed to process real time data collected from the camera. Once the image is captured it is sent to the image classification algorithm in the backend of the application. After the correct detection of pesticide it is shown to the user for confirmation. After verification as of now we are having only three pesticides listed on our database and two crops options for each of them.

Farmers will get to choose for the specific crop and after choosing he/ she will be shown the pesticide and water connected proportions using the graphical interface.

Here, Android Studio and Pycharm is used to implement the idea and results are obtained. This App will provide the farmer with appropriate information about the amount of pesticides they should add in the given amount of water so that to increase the effectiveness of crops on which the pesticides are going to be applied. It will thus reduce the overuse and the pressing needs on pesticides.

Contents

Chapter 1

Introduction

History

The main objective of this project is to provide the farmers with the android application which provides a good consultant for using pesticides to farmers. As in current days, farmers purchase pesticides, chemicals for spraying on crops but they don't know the actual concentration to be made for proper use. Due to this they get up in the end with no effect on crops or may cause harm to crops due to excess concentration. This App is going to help them to understand how much water and how many chemicals to be added for good results.

Many Apps are made for different purposes. Such as FarmManager, Agro mobile, Krishi Ville and various other Apps. These Apps provide the proper information regarding the pesticides and other related components. Depending on these Apps we took the idea of creating an application which will provide the farmer with appropriate information about the amount of pesticides they should add in the given amount of water so that to increase the effectiveness of crops on which the pesticides are going to be applied. It will thus reduce the overuse and the pressing needs on pesticides.

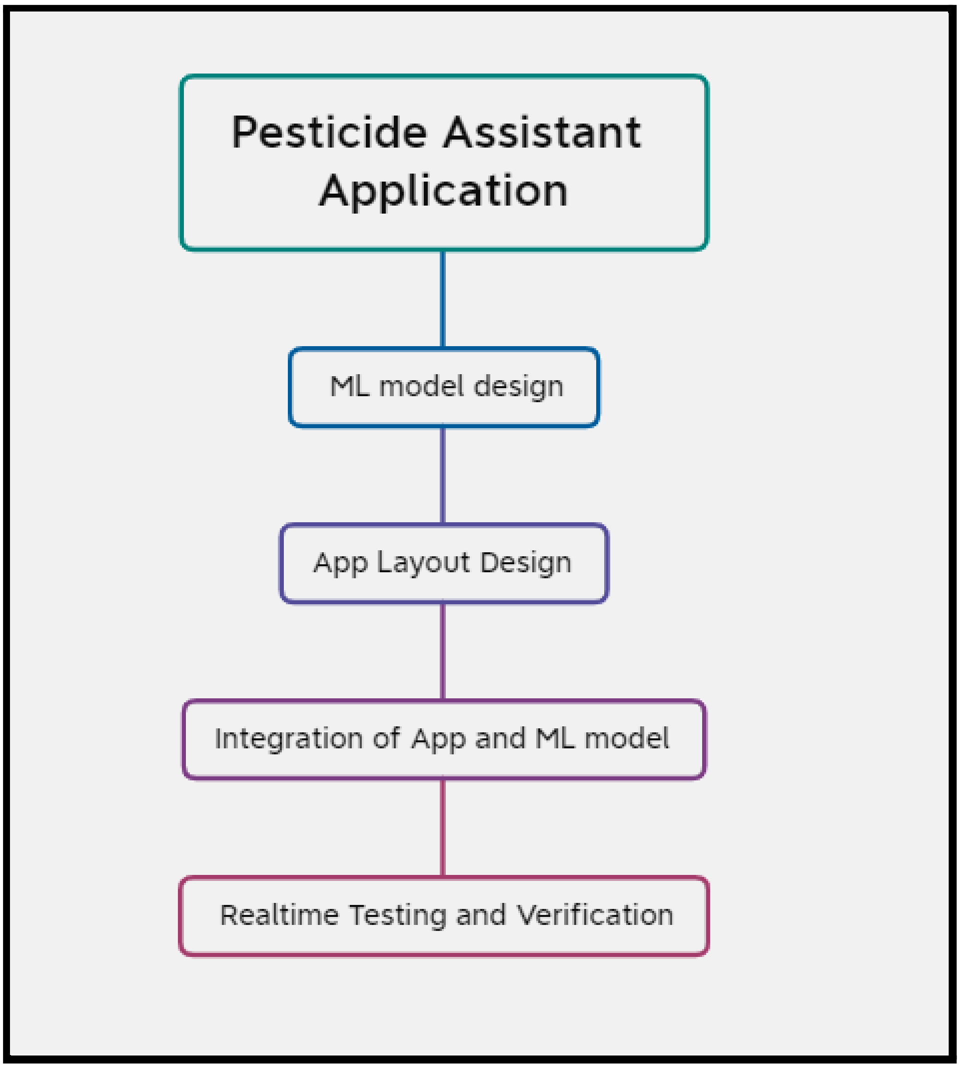
Need Analysis

There are plenty of apps that assist farmers for various purposes such as crop management, farm management and serve for a variety of purposes. They essentially provide the privilege of detecting any crop related problems by scanning the crop. The software is used for achieving fast turnaround times while the hardware is used to speedup critical portions of the system.

Until now, there was no App which provided the information regarding how much pesticide we should feed to a particular crop in a specific amount of given water quantity. With this app it will be easier for farmers to get full fledged effectiveness of the pesticide they were applying than before. It will thus cut down the price of pesticides which they were spending more earlier.

This project deals with the development of such a tool which will assist in the implementation of the above methodology.

Workflow :


Pesticide Assistant Application
ML model design
App Layout Design
Integration of App and ML model
Realtime Testing and Verification

Problem Definition

Just like anything, crops do have vulnerabilities from the excessive use of pesticides. And one way to understand proper and effective use of pesticides that they are going to apply on crops is use of the proper assistant. This App is used to provide proper aid to farmers regarding appropriate usage of pesticides.

Objectives

The final goal of the project was twofold.

1. A dedicated App that will serve the purpose of providing the accurate and handy information regarding spraying appropriate amounts of the pesticides on a specific crop.
2. Along with this, we are also aiming to reduce the pressing needs on the use of fertilizers that is increasing exponentially over the decades.

Technical Specifications

- Application Specification :-
    1. Captures the live image.
    2. Processes on the image fed.
    3. Detects the image with the help of the ML model fed.
    4. Provides information about the appropriate amount of pesticides with the help of block diagrams.

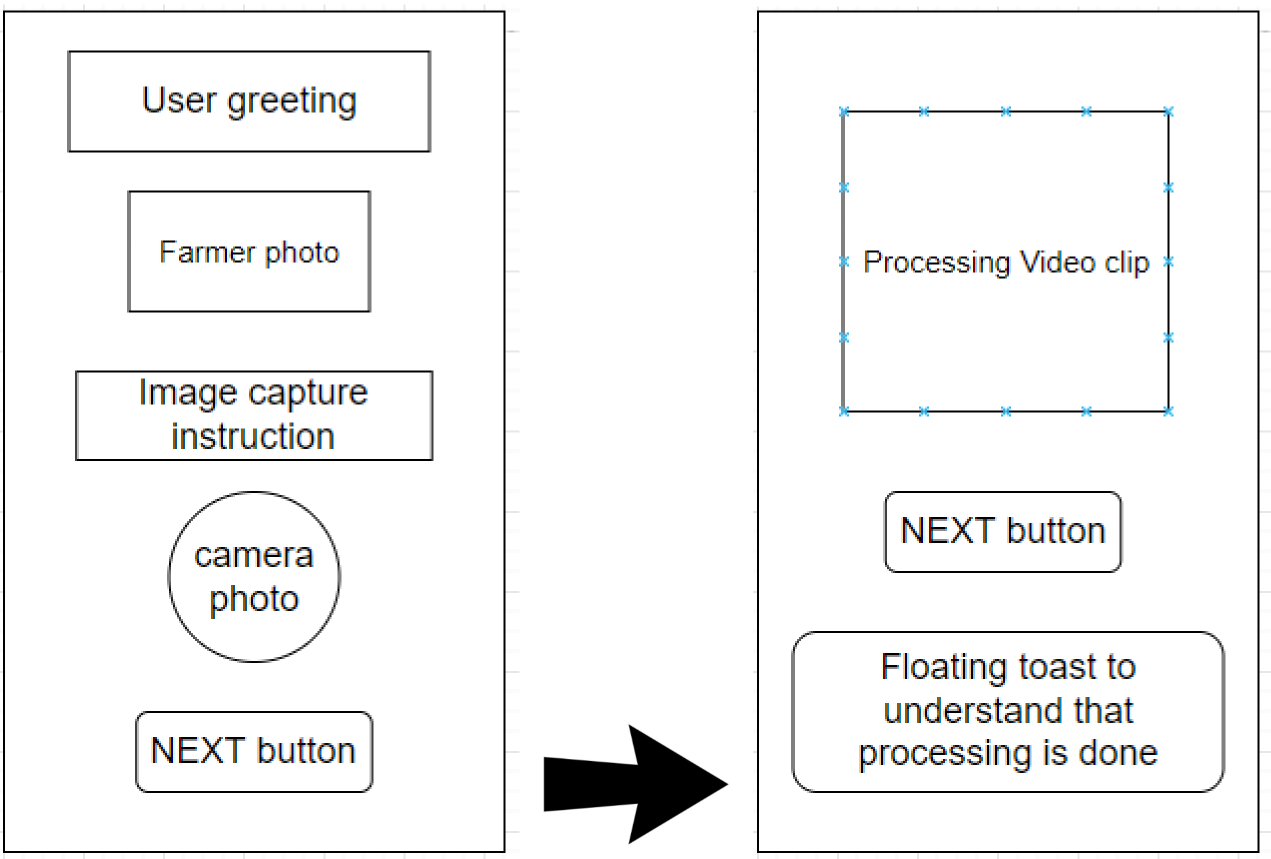
- Hardware Specification :-
    1. The app is compatible with all types of Android devices.

Chapter
2

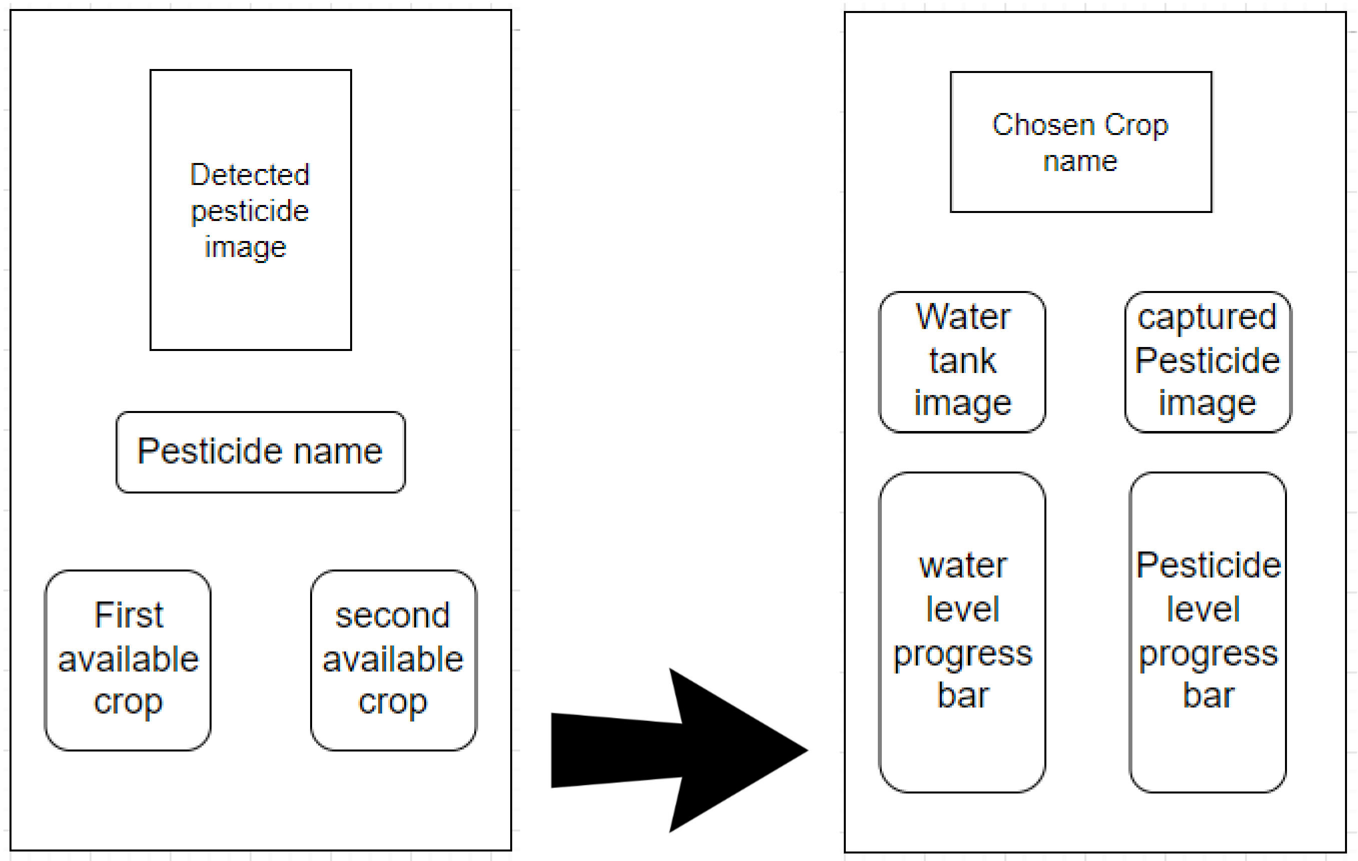
Software Design

A) App Creation :

1. App layout :

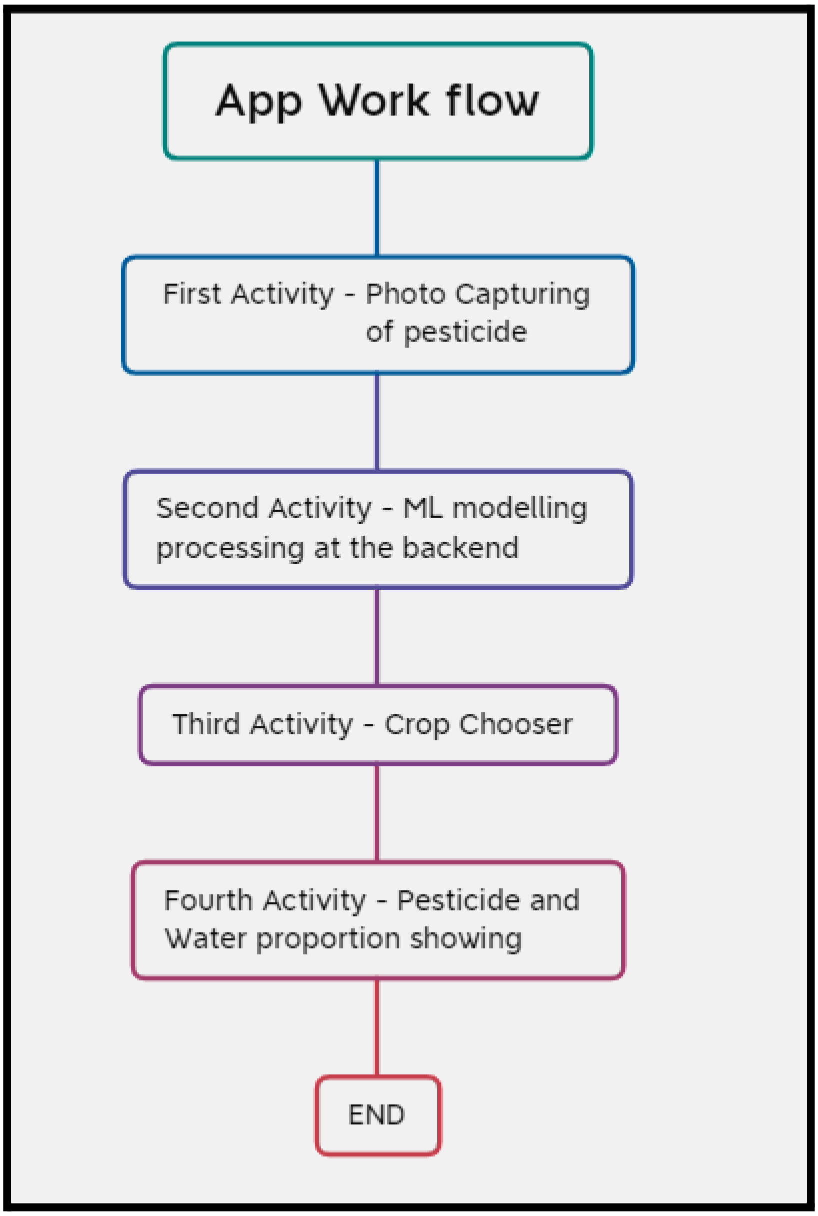


First Activity **Second Activity**



Third Activity **Fourth Activity**

2. App Workflow :



3. Backend Working:

1) First Activity Backend :

In the first activity we are supposed to capture the image using a camera as per the layout. So in the backend there are two textviews for information displaying.

Also two imageview, one of them showing farmer pictures and other for image capture.

There is one button to go for the next activity.

On the camera imageview one onclick listener is placed to invoke the camera intent using the java code. After clicking on the camera image the intent is invoked and the path for the image is saved in the string variable.

Also the new intent is made to send the path to the next activity and it is linked with the "next" button.

As soon as you press the next button the saved path of the image is sent to the next activity.

2) Second Activity Backend :

In the second activity our main brain of the project lies. In the frontend we are having a surfaceholder holding the video to show the processing for the user and the "next" button to go to the next activity.

First of all we collected the path coming from the previous activity and then it is fed to the ML model. The ML model is integrated with the app in the form of TFLite file format using the Android studio's inbuilt feature.

After getting the image path the model predicts which is the class of the captured pesticide and stores it in the string variable.

As you press the "next" button the predicted class is sent to the next activity.

3) **Third Activity Backend:**

In the third activity we are having three imageviews showing the predicted pesticide and the two crops which are available.

At the backend of the two imageview, after clicking on one of the images the encoding code is stored in the string variable. After clicking on the image also the intent is created and the encoded name for the pesticide is sent to the next activity.

4) **Fourth Activity Backend :**

In the fourth activity we are having the two progress bars showing the proportion of the water and pesticide according to the standards. Also we showed the percentage of the quantity in the progress bar.

5) After this our app working ends and if the user wants to go backward and start the process again he may do it or close the app.

B) ML model Design :

1) Dataset specification :

As our project is quite new to the ML feild so we didn't get any ready-made dataset. So, we made it through real time capturing of images and their editing.

a) Consisting of 1800 images of three bottles of pesticides.
b) Their names are "Dudex", "Effect" and "Weeds Super".
c) Which represents the three classes of the model as class0,class1 and class2 respectively.
d) The images in the dataset are taken with different viewpoints, different coloring situations and orientations.

2) Algorithms used :

a) Our main problem statement for the ML model is to classify the given image and tell to which class it belongs from the available classes.
b) At the first site we did a lot of preprocessing work on the images to make them good fit for our model.
c) We used many builtin functions and custom added functions for image preprocessing.
d) Then written functions to split the data into training and testing sets.
e) Trained the model with various epochs to get the better accuracy.
f) The performance observed and analysis curves are attached below.
g) The algorithms used are :
    1) Homography ,feature mapping stage 1
    2) Opencv and keras callbacks intermediate.
    3) 5 layered CNN stage 3 detection.

Code for ML model :

```python
import cv2
import numpy as np
import cv2
import numpy as np
import os
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
from keras.preprocessing.image import ImageDataGenerator
from keras.utils.np_utils import to_categorical
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout,Flatten
from keras.layers.convolutional import Conv2D,MaxPooling2D
from tensorflow import lite

images = []
classes = []

############################
path = "myData1"
path2 = "roi_images"
testratio = 0.1
valratio = 0.1
imageDimensions = (32,32,3)
batchSizeVal = 64
epochsVal = 30
stepsPerEpoch = 25
############################
ind = -1
myList = os.listdir(path)
```

```python
noOfClasses = len(myList)
for x in range(len(myList)):
    myPicList = os.listdir(path+'/'+str(x))
    for y in myPicList:
        curImg = cv2.imread(path+'/'+str(x)+'/'+y)
        curImg = cv2.resize(curImg,(imageDimensions[0],imageDimensions[1]))
        images.append(curImg)
        classes.append(x)
    print(x,end=" ")
images = np.array(images)
classes = np.array(classes)

roi_classes = []
class_names = []

myList2 = os.listdir(path2)
noOfClasses = len(myList2)
for x in myList2:
    roi_det = cv2.imread(path2+'/'+x,cv2.IMREAD_GRAYSCALE)
    roi_classes.append(roi_det)
    class_names.append(os.path.splitext(x)[0])

#extract the key features of the roi
sift = cv2.xfeatures2d.SIFT_create()


#feature matching
index_params = dict(algorithm=0,tree=5)
search_params = dict()
flann = cv2.FlannBasedMatcher(index_params,search_params)
```

```python

roi_kp = []
def calc_desc(images):
    global roi_kp
    desc_list = []
    for i in images:
        kp_roi, desc_roi = sift.detectAndCompute(i, None)
        desc_list.append(desc_roi)
        roi_kp.append(kp_roi)
    return desc_list
def match_desc(desc_list,frame):
    final_val = -1
    good_matches = []
    kp_frame,desc_frame = sift.detectAndCompute(frame,None)
    try:
        for desc_curr in desc_list:
            match = flann.knnMatch(desc_curr, desc_frame, k=2)
            good = []
            for m, n in match:  # m is roi and n is the frame
                if m.distance < 0.75 * n.distance:  # distances between discritors must be small
                    good.append(m)
            good_matches.append(len(good))
    except:
        pass
    if len(good_matches) != 0:
        if max(good_matches) > 15:
            final_val = good_matches.index(max(good_matches))
    return final_val

```

```python


print("\n",images.shape)
print(classes.shape)
x_train,x_test,y_train,y_test = train_test_split(images,classes,test_size=testratio,train_size=1-testratio)
print(x_train.shape)
print(x_test.shape)
x_train,x_validation,y_train,y_validation = train_test_split(x_train,y_train,test_size=valratio,train_size=1-valratio)
print(x_train.shape)
print(x_validation.shape)
samples = []
for t in range(len(myList)):
    samples.append(len(np.where(y_train == t)[0]))
print(samples)
for t in range(len(samples)):
    print("number of samples of %d is %d "%(t, samples[t]))
plt.figure(figsize =(10,5))
plt.bar(range(0,len(samples)),samples)
plt.title("distribution of samples of various classes")
plt.xlabel("class ID")
plt.ylabel("No of samples")
plt.show()


def preprocess(img):
    img = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    img = cv2.equalizeHist(img)
    img = img/255
    return img

```

```python
x_train = np.array(list(map(preprocess,x_train)))
x_test = np.array(list(map(preprocess,x_test)))
x_validation = np.array(list(map(preprocess,x_validation)))

x_train = x_train.reshape(x_train.shape[0],x_train.shape[1],x_train.shape[2],1)  #add a depth of one to train test and validation
x_test = x_test.reshape(x_test.shape[0],x_test.shape[1],x_test.shape[2],1)   #this is done so taht the CNN runs well
x_validation = x_validation.reshape(x_validation.shape[0],x_validation.shape[1],x_validation.shape[2],1)
datagen = ImageDataGenerator(width_shift_range=0.1,
                             height_shift_range=0.1,
                             zoom_range=0.2,
                             shear_range=0.1,
                             rotation_range=10)
datagen.fit(x_train) #To augment the images with shifted,zoomed,rotated changes
print(x_train.shape)
y_train = to_categorical(y_train,len(myList))
y_test = to_categorical(y_test,len(myList))             #1000000000 = 0,0000010000 = 5
y_validation = to_categorical(y_validation,len(myList)) #one hot encoding on the numeric classes

def myModel():
    noOfFilters = 60
    sizeOfFilter1 = (5,5)
    sizeOfFilter2 = (3,3)
    sizeOfPool = (2,2)
    noOfNode = 500

    model = Sequential()
    model.add((Conv2D(noOfFilters,sizeOfFilter1,input_shape = (imageDimensions[0],imageDimensions[1],1),activation = 'relu')))
    model.add((Conv2D(noOfFilters, sizeOfFilter1,activation='relu')))
    model.add(MaxPooling2D(pool_size = sizeOfPool))
    model.add((Conv2D(noOfFilters//2, sizeOfFilter2, activation='relu')))
```

```
    model.add((Conv2D(noOfFilters//2, sizeOfFilter2, activation='relu')))
    model.add(MaxPooling2D(pool_size=sizeOfPool))
    model.add(Dropout(0.5))
    model.add(Flatten())
    model.add(Dense(noOfNode,activation='relu'))
    model.add(Dropout(0.5))
    model.add(Dense(noOfClasses, activation='softmax'))
    model.compile(optimizer='adam',loss='categorical_crossentropy', metrics=['accuracy'])
    return model

model = myModel()
print(model.summary())

history = model.fit_generator(datagen.flow(x_train,y_train,batch_size=batchSizeVal),
                    steps_per_epoch=stepsPerEpoch,
                    epochs=epochsVal,validation_data=(x_validation,y_validation),
                    shuffle=1)
converter = lite.TFLiteConverter.from_keras_model(model)
tfmodel = converter.convert()
open('farmer_assistant.tflite','wb').write(tfmodel)
plt.figure(1)
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.legend('training','validation')
plt.title('Loss')
plt.xlabel('epoch')
plt.figure(2)
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.legend('training','validation')
```

```python
plt.title('Accuracy')
plt.xlabel('epoch')
plt.show()
score = model.evaluate(x_test,y_test,verbose=0)
print("test score is = ",score[0])
print("test accuracy is = ",score[1])

##################

##################

path = "roi_images"
#roi = cv2.imread(path+"/2.jpeg",cv2.IMREAD_GRAYSCALE)
pts2 = np.float32([[0,0],[400,0],[0,400],[400,400]])
biggest = np.array([])
biggest_reordered = np.zeros((4,1,2),dtype = np.int32)
count = 0
curr = 0
arr = np.zeros((4,2),int)
arr2 = np.zeros((4,2),int)
draw = False
create = False
pause = False
submit = False
flag = False



def adjust(event, x, y, flags, params):
```

```python
global draw,create,pause,submit,curr,count,biggest_reordered,ls,flag,ind
if pause:
    frame2 = frame.copy()
if event == cv2.EVENT_RBUTTONDBLCLK:
    pts1 = np.float32([ls[0][0], ls[3][0], ls[1][0], ls[2][0]])
    matrix = cv2.getPerspectiveTransform(pts1, pts2)
    final = cv2.warpPerspective(frame, matrix, (400, 400))
    img = cv2.resize(final, (32, 32))
    img = preprocess(img)
    img = img.reshape(1, 32, 32, 1)
    # predict
    predictions = model.predict(img)
    probval = np.amax(predictions)
    classIndex = np.argmax(predictions, axis=1)
    if (probval > 0.8):
        cv2.putText(img=final, text=str(class_names[ind])+' '+str(probval), org=(100, 100), fontFace=cv2.FONT_HERSHEY_TRIPLEX, fontScale
                    color=(0, 0, 0), thickness=3)
    else:
        cv2.putText(img=final, text='Try again!!', org=(50, 150), fontFace=cv2.FONT_HERSHEY_TRIPLEX, fontScale=1,
                    color=(0, 0, 0), thickness=3)
    cv2.imshow("Final_aligned", final)
if event == cv2.EVENT_LBUTTONDOWN and submit:
    image = cv2.polylines(frame2,[ls],True,(0,0,255),3)
    image = cv2.circle(frame2,ls[0][0],5,(0,100,255),2)
    image = cv2.circle(frame2, ls[1][0],5,(0,100,255),2)
    image = cv2.circle(frame2, ls[2][0],5,(0,100,255),2)
    image = cv2.circle(frame2, ls[3][0],5,(0,100,255),2)
    cv2.imshow("object_tracker",image)
    draw = True
    tt = 0
```

```
    temp = 2000
    for i in range(4):
        if abs(x-ls[i][0][0])+abs(y-ls[i][0][1]) < temp:
            temp = abs(x-ls[i][0][0])+abs(y-ls[i][0][1])
            curr = tt
        tt+=1
    ls[curr][0][0] = x
    ls[curr][0][1] = y
elif event == cv2.EVENT_MOUSEMOVE and submit:
    if draw :
        create = True
        ls[curr][0][0] = x
        ls[curr][0][1] = y
        image = cv2.polylines(frame2,[ls],True,(0,0,255),3)
        image = cv2.circle(frame2, ls[0][0], 5, (0, 100, 255), 2)
        image = cv2.circle(frame2, ls[1][0], 5, (0, 100, 255), 2)
        image = cv2.circle(frame2, ls[2][0], 5, (0, 100, 255), 2)
        image = cv2.circle(frame2, ls[3][0], 5, (0, 100, 255), 2)
        cv2.imshow("object_tracker",image)
elif event == cv2.EVENT_LBUTTONUP and submit:
    if pause and draw and create:
        image = cv2.polylines(frame2,[ls],True,(0,0,255),3)
        image = cv2.circle(frame2, ls[0][0], 5, (0, 100, 255), 2)
        image = cv2.circle(frame2, ls[1][0], 5, (0, 100, 255), 2)
        image = cv2.circle(frame2, ls[2][0], 5, (0, 100, 255), 2)
        image = cv2.circle(frame2, ls[3][0], 5, (0, 100, 255), 2)
        cv2.imshow("object_tracker",image)
    draw = False
    create = False
elif not submit:
```

```
    if event == cv2.EVENT_LBUTTONDOWN:
        img_detected = frame.copy()
        image_edge = cv2.polylines(img_detected,[ls],True,(0,0,255),3)
        cv2.imshow("Contour_image2",image_edge)
        submit = True



namedWindow('object_tracker')
setMouseCallback('object_tracker',adjust)
= cv2.VideoCapture("1.mp4")
_images = calc_desc(roi_classes)

e True:
success,frame = cap.read()
if not success:
    break
grayframe = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
ind = match_desc(desc_images, grayframe)
#homography i.e the box or label is actually present in the video
#give a threshold of atleast 11 good points
if ind != -1:
    h, w = roi_classes[ind].shape
    kp_frame, desc_frame = sift.detectAndCompute(grayframe, None)
    match = flann.knnMatch(desc_images[ind],desc_frame,k=2)
    good_points = []
    for m, n in match:  # m is roi and n is the frame
        if m.distance < 0.75 * n.distance:  # distances between discritors must be small
            good_points.append(m)
    roi_pts = np.float32([roi_kp[ind][m.queryIdx].pt for m in good_points]).reshape(-1, 1, 2)
```

```
    test_pts = np.float32([kp_frame[m.trainIdx].pt for m in good_points]).reshape(-1, 1, 2)
    matrix,mask = cv2.findHomography(roi_pts,test_pts,cv2.RANSAC,5)
    pts = np.float32([[0,0],[0,h-1],[w-1,h-1],[w-1,0]]).reshape(-1,1,2)
    dst = cv2.perspectiveTransform(pts,matrix)
    ls = np.int32(dst)
    homography = cv2.polylines(grayframe,[ls],True,(255,0,0),3)
    cv2.imshow("object_tracker",homography)
:v2.imshow("frame", frame)
esc = cv2.waitKey(10)
if esc == 117:
    pause = True
    cv2.waitKey(0)
if esc == 116:
    pause = False
if esc == 115:
    break
release()
vaitKey(0)
destroyAllWindows()
```

3) App and ML model integration :

Now we are having both app layout and the ML model working fine. We have to integrate them. So, first of all we need the TensorFlowLite file of the ML model to attach it with the java code in the android studio.

So, firstly we converted our ML model to a " tflite" file and saved it using some lines of the python code.
Then created the ML directory inside the android studio project and pasted the model.tflite file there.

For interaction of the model and java code we need to write some lines of java code at the Activity no. 2 in the app layout. The code for integration is attached below.

Basically the code consists of invoking the ML model from the file and then creating the tensorflowImage to feed the model.Then the image is passed to the model and output is taken from the "OutputFeature0" variable by applying the array extraction method on it. The output is then stored in the variable for further usage..

ML model integration code with java :

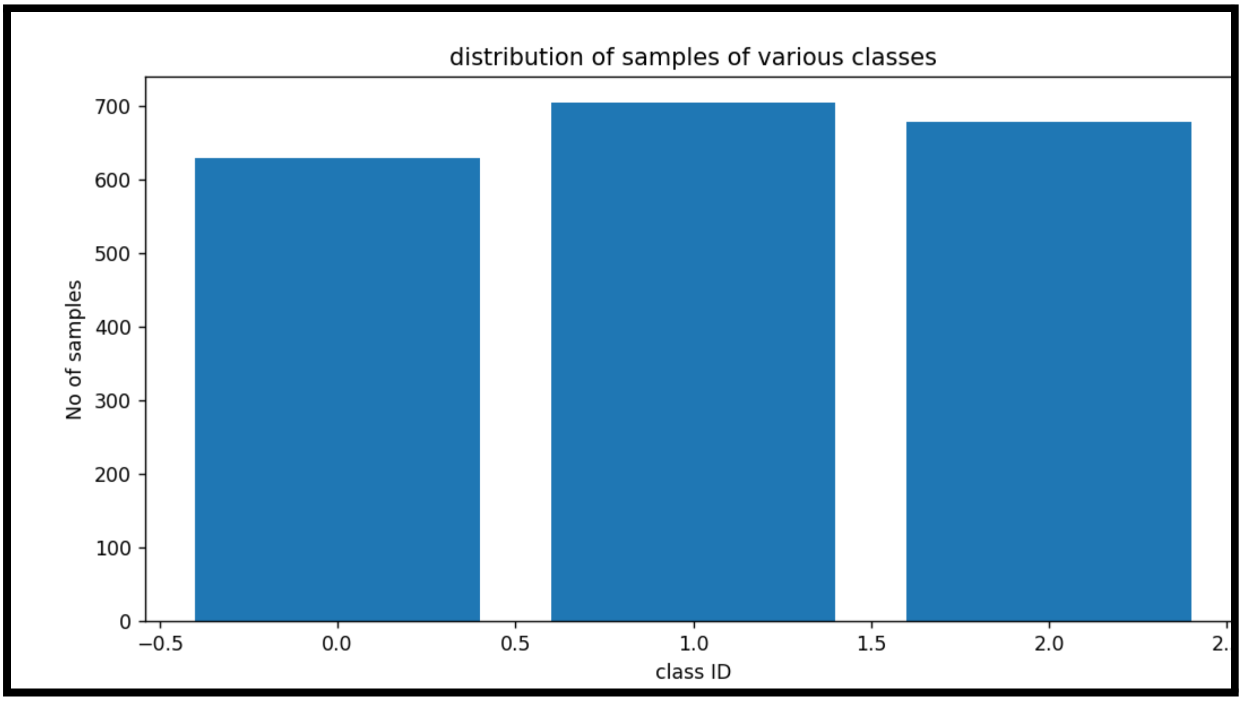
```java
try {
    Model model = Model.newInstance(context);

    // Creates inputs for reference.
    TensorBuffer inputFeature0 = TensorBuffer.createFixedSize(new int[]{1, 224, 224, 3}, DataType.UINT8);
    inputFeature0.loadBuffer(byteBuffer);

    // Runs model inference and gets result.
    Model.Outputs outputs = model.process(inputFeature0);
    TensorBuffer outputFeature0 = outputs.getOutputFeature0AsTensorBuffer();

    // Releases model resources if no longer used.
    model.close();
} catch (IOException e) {
    // TODO Handle the exception
}
```

Chapter
3

Performance Evaluation And Result Analysis

While doing the model design various processes are performed and their analysis is done on each step. Below attached images shows the result/ performance obtained.

a) Distribution of samples of various classes :

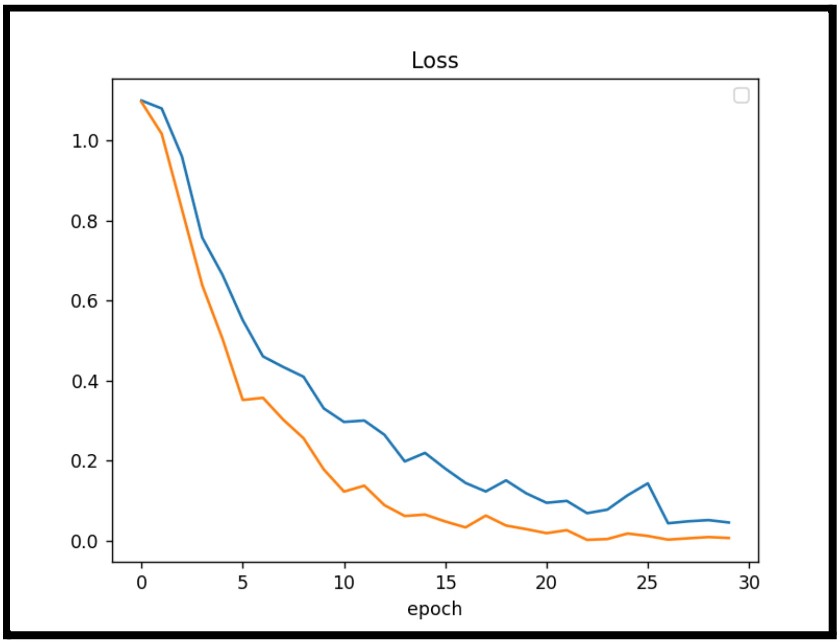


b) Each step epoch performance :

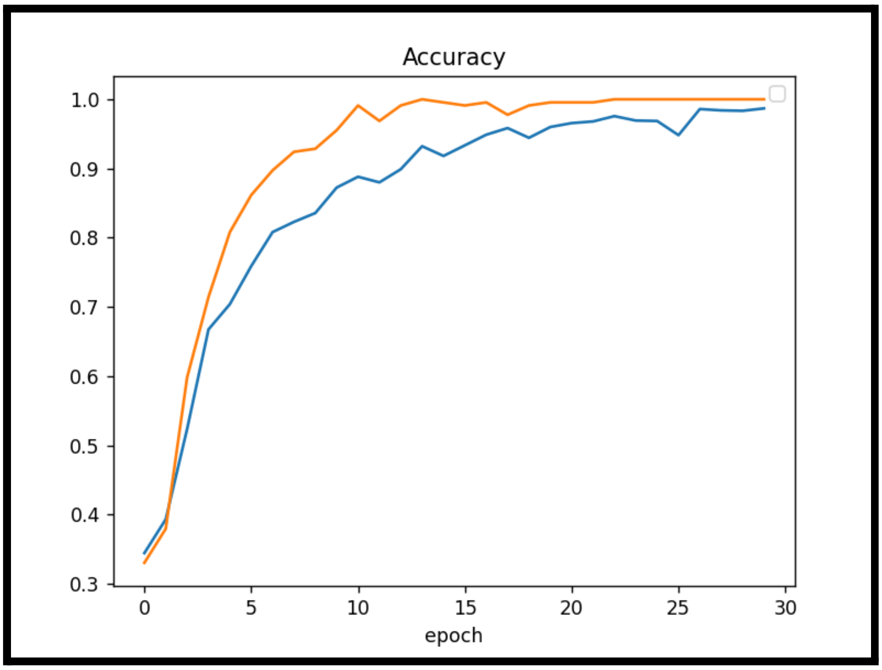
```
Epoch 1/30
25/25 [==============================] - 11s 372ms/step - loss: 1.0994 - accuracy: 0.3444 - val_loss: 1.0957 - val_accuracy: 0.3304
Epoch 2/30
25/25 [==============================] - 8s 324ms/step - loss: 1.0796 - accuracy: 0.3932 - val_loss: 1.0163 - val_accuracy: 0.3795
Epoch 3/30
25/25 [==============================] - 8s 338ms/step - loss: 0.9595 - accuracy: 0.5237 - val_loss: 0.8273 - val_accuracy: 0.5982
Epoch 4/30
25/25 [==============================] - 9s 366ms/step - loss: 0.7571 - accuracy: 0.6675 - val_loss: 0.6383 - val_accuracy: 0.7143
Epoch 5/30
25/25 [==============================] - 9s 356ms/step - loss: 0.6642 - accuracy: 0.7040 - val_loss: 0.5045 - val_accuracy: 0.8080
Epoch 6/30
25/25 [==============================] - 9s 357ms/step - loss: 0.5511 - accuracy: 0.7590 - val_loss: 0.3518 - val_accuracy: 0.8616
Epoch 7/30
25/25 [==============================] - 9s 349ms/step - loss: 0.4605 - accuracy: 0.8081 - val_loss: 0.3570 - val_accuracy: 0.8973
Epoch 8/30
25/25 [==============================] - 8s 331ms/step - loss: 0.4341 - accuracy: 0.8229 - val_loss: 0.3025 - val_accuracy: 0.9241
Epoch 9/30
25/25 [==============================] - 8s 328ms/step - loss: 0.4098 - accuracy: 0.8357 - val_loss: 0.2564 - val_accuracy: 0.9286
Epoch 10/30
25/25 [==============================] - 8s 330ms/step - loss: 0.3307 - accuracy: 0.8725 - val_loss: 0.1783 - val_accuracy: 0.9554
Epoch 11/30
25/25 [==============================] - 8s 315ms/step - loss: 0.2969 - accuracy: 0.8881 - val_loss: 0.1229 - val_accuracy: 0.9911
Epoch 12/30
25/25 [==============================] - 9s 341ms/step - loss: 0.3004 - accuracy: 0.8800 - val_loss: 0.1377 - val_accuracy: 0.9688
Epoch 13/30
25/25 [==============================] - 8s 331ms/step - loss: 0.2647 - accuracy: 0.8990 - val_loss: 0.0890 - val_accuracy: 0.9911
Epoch 14/30
25/25 [==============================] - 8s 331ms/step - loss: 0.1983 - accuracy: 0.9322 - val_loss: 0.0621 - val_accuracy: 1.0000
Epoch 15/30
25/25 [==============================] - 8s 336ms/step - loss: 0.2196 - accuracy: 0.9182 - val_loss: 0.0656 - val_accuracy: 0.9955
```

```
Epoch 16/30
25/25 [==============================] - 9s 340ms/step - loss: 0.1804 - accuracy: 0.9335 - val_loss: 0.0483 - val_accuracy: 0.9911
Epoch 17/30
25/25 [==============================] - 9s 346ms/step - loss: 0.1446 - accuracy: 0.9488 - val_loss: 0.0336 - val_accuracy: 0.9955
Epoch 18/30
25/25 [==============================] - 8s 332ms/step - loss: 0.1233 - accuracy: 0.9584 - val_loss: 0.0631 - val_accuracy: 0.9777
Epoch 19/30
25/25 [==============================] - 8s 328ms/step - loss: 0.1510 - accuracy: 0.9444 - val_loss: 0.0382 - val_accuracy: 0.9911
Epoch 20/30
25/25 [==============================] - 8s 327ms/step - loss: 0.1186 - accuracy: 0.9600 - val_loss: 0.0292 - val_accuracy: 0.9955
Epoch 21/30
25/25 [==============================] - 8s 329ms/step - loss: 0.0950 - accuracy: 0.9656 - val_loss: 0.0192 - val_accuracy: 0.9955
Epoch 22/30
25/25 [==============================] - 8s 330ms/step - loss: 0.0997 - accuracy: 0.9680 - val_loss: 0.0267 - val_accuracy: 0.9955
Epoch 23/30
25/25 [==============================] - 9s 354ms/step - loss: 0.0691 - accuracy: 0.9757 - val_loss: 0.0025 - val_accuracy: 1.0000
Epoch 24/30
25/25 [==============================] - 9s 373ms/step - loss: 0.0779 - accuracy: 0.9693 - val_loss: 0.0045 - val_accuracy: 1.0000
Epoch 25/30
25/25 [==============================] - 8s 328ms/step - loss: 0.1137 - accuracy: 0.9687 - val_loss: 0.0183 - val_accuracy: 1.0000
Epoch 26/30
25/25 [==============================] - 8s 340ms/step - loss: 0.1436 - accuracy: 0.9482 - val_loss: 0.0122 - val_accuracy: 1.0000
Epoch 27/30
25/25 [==============================] - 9s 348ms/step - loss: 0.0439 - accuracy: 0.9859 - val_loss: 0.0031 - val_accuracy: 1.0000
Epoch 28/30
25/25 [==============================] - 9s 357ms/step - loss: 0.0488 - accuracy: 0.9840 - val_loss: 0.0063 - val_accuracy: 1.0000
Epoch 29/30
25/25 [==============================] - 9s 353ms/step - loss: 0.0518 - accuracy: 0.9834 - val_loss: 0.0095 - val_accuracy: 1.0000
Epoch 30/30
25/25 [==============================] - 8s 335ms/step - loss: 0.0458 - accuracy: 0.9869 - val_loss: 0.0071 - val_accuracy: 1.0000
```

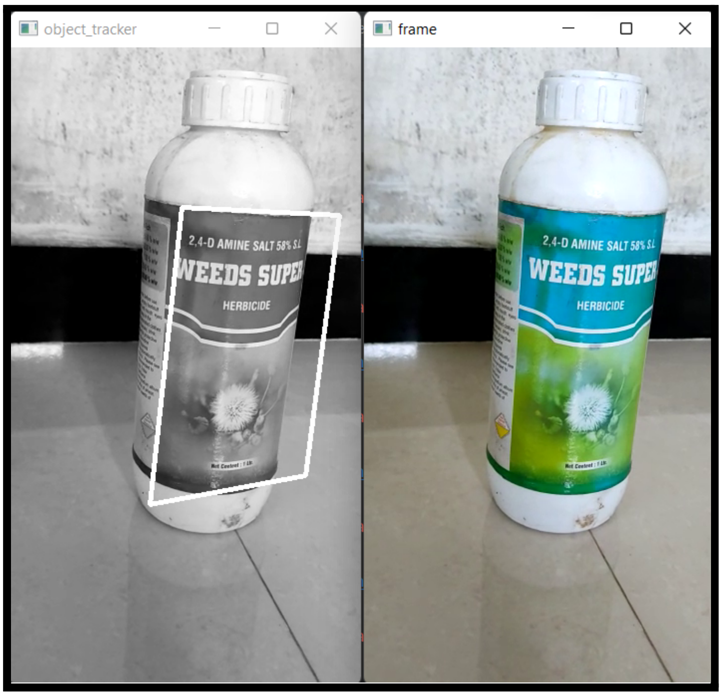
c) Loss graph while training iterations :

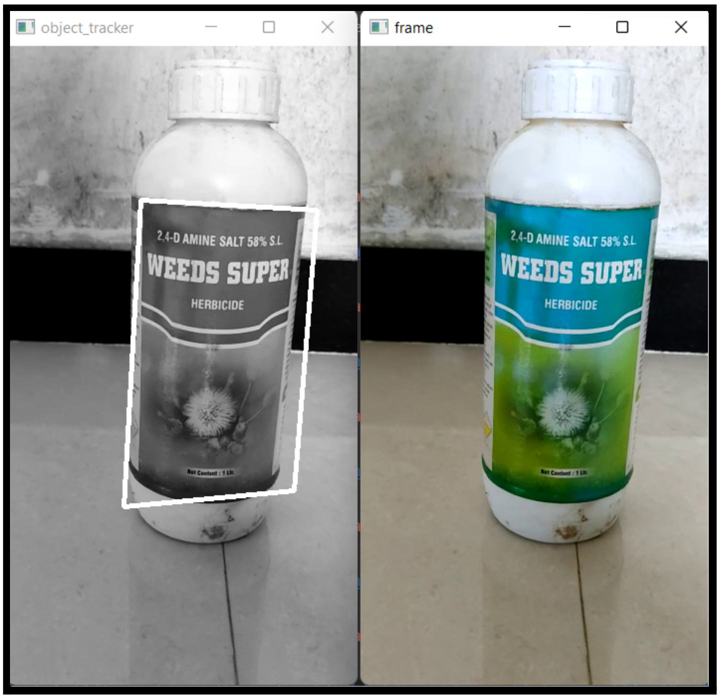


d) Accuracy graph while training :



e) Object detection and bounding box making result :





f) Test scores and accuracy :

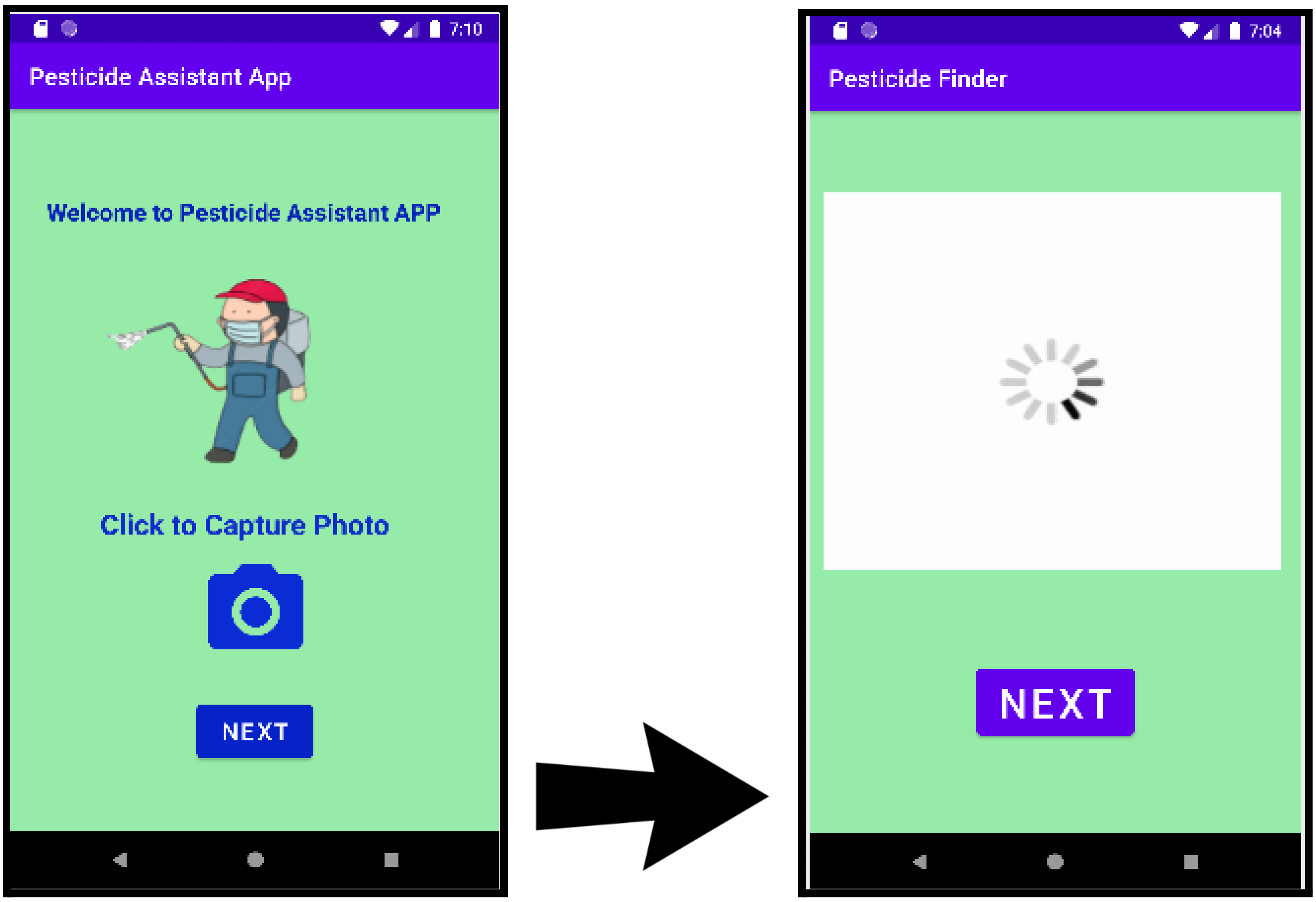
```
test score is =  0.005705494433641434
test accuracy is =  1.0
```

g) Pesticide class prediction :

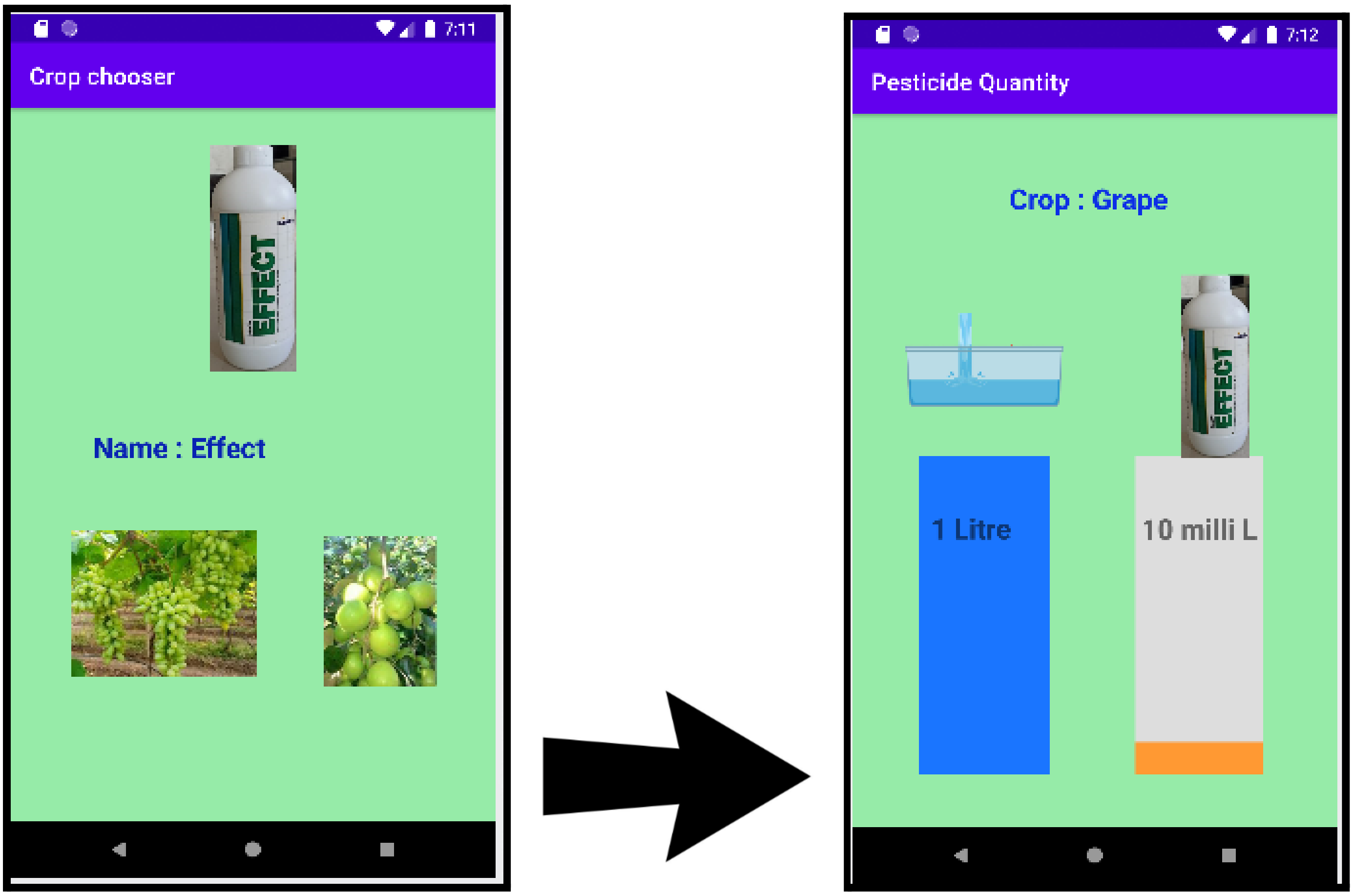
Chapter
4

Final Product :

Our final product will have four different activities whose working is described in detail in the App layout under the software section.
So the final product will be look as the images of different activities as listed below:



First Activity **Second Activity**



Third Activity **Fourth Activity**

As per the activities shown the app will work finely without crashing even if activities are randomly navigated.

We tested the final app apk on a smartphone with the help of a real pesticide bottle having the “Effect” pesticide image on it. It works perfectly and shows the required result.

Chapter

5

Future Scope

- **Image import from Gallery**

  In future, we will be trying to add an option of importing the image from the user's gallery so as to facilitate smooth service.

- **More crops options**

  We are also thinking of providing more crop options so that all types of farmers will get covered.

- **Option of verifying on Internet**

  In case the user didn't get the match with the image that they are trying to find, then they will be directly redirected to the internet and will try to find the best match.

- **Detecting the type of pest on the crop**

  Also will try to add a facility to detect the pest as same as detecting bottle with suggesting the required pesticide.

- **Calling facility for faster processing**

  If in future this app gets lots of customers will try to add the voice calling facility to assist farmers based on needs.

Chapter

6

Conclusion

1. This App is free of any human intervention and thus makes the control of the system simple, fast and accurate. The use of Mobile as its hardware component helps for a handy approach to users.

2.Class prediction accuracy of close to 90-95% makes the app to predict the pesticide correctly so there will not be any mispredictions.

3. The App can be used to prevent excessive and reduce pesticide consumption rate in farm, and thus improving the yield of specific crops.

Chapter 7

References

1. https://neptune.ai/blog/data-augmentation-in-python
2. https://www.geeksforgeeks.org/learning-model-building-scikit-learn-python-machine-learning-library/
3. Feature Detection and Matching + Image Classifier Project | OPENCV PYTHON
4. How To Train Deep Learning Models In Google Colab- Must For Everyone
5. Text Detection using Neural Networks | OPENCV Python
6. https://developer.android.com/reference/android/widget/ProgressBar
7. https://developer.android.com/guide/topics/media/camera
8. https://stackoverflow.com/questions/3004713/get-content-uri-from-file-path-in-android
9. Android Development Tutorial For Beginners In Hindi (With Notes) 🔥
10. https://www.tensorflow.org/lite/android/quickstart