Project Development Phase

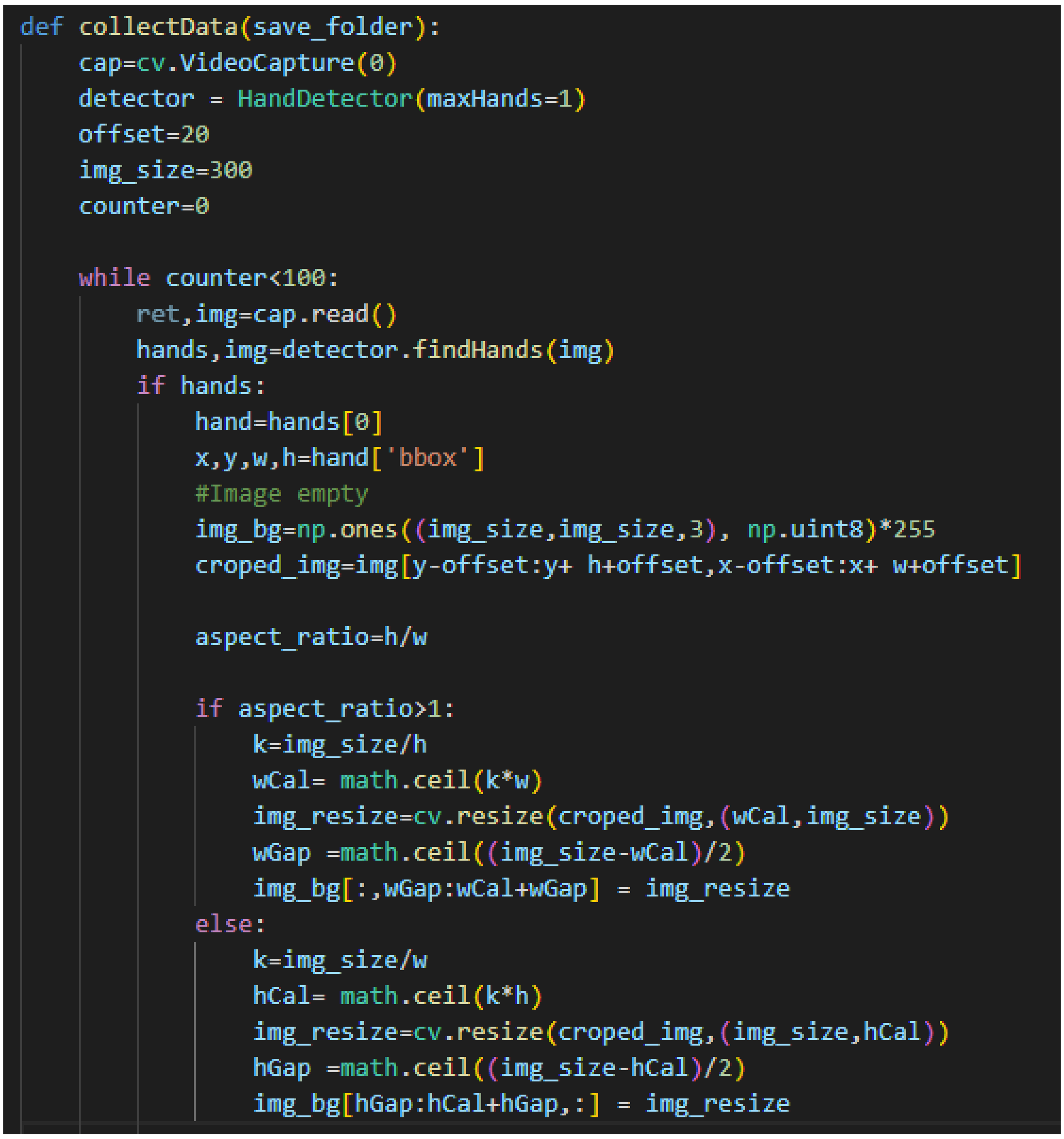
Sprint – 2

Date	5-11-2022
Team ID	PNT2022TMID22290
Project Title	Real-Time Communication System Powered by AI for Specially Abled

Creating CNN Model and Predication Base programs :

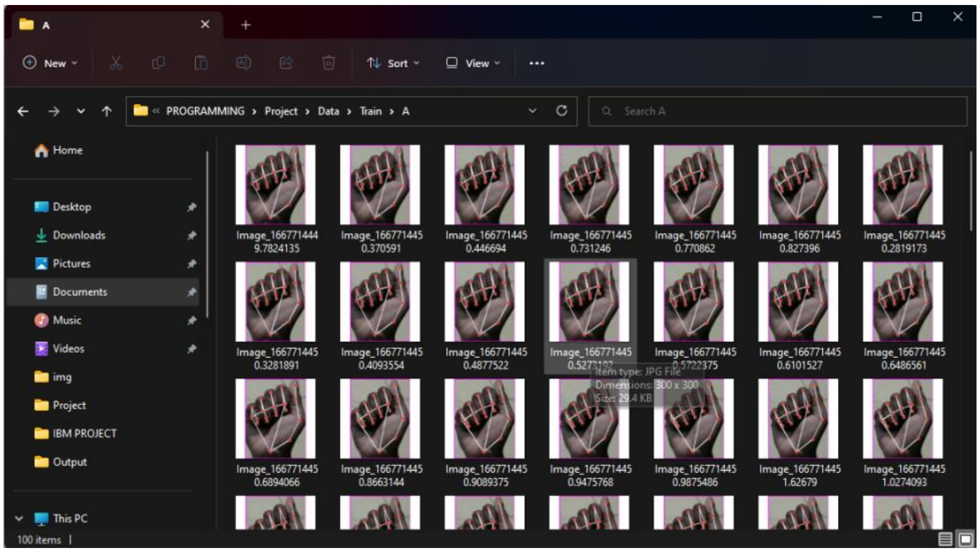
1. **Collecting Datasets.**

   a. Creating datacollection program to collect data from the user and detect the hands in the image frames add crop the detected hands and adding a background of size 300 x 300 px.



```
def collectData(save_folder):
    cap=cv.VideoCapture(0)
    detector = HandDetector(maxHands=1)
    offset=20
    img_size=300
    counter=0

    while counter<100:
        ret,img=cap.read()
        hands,img=detector.findHands(img)
        if hands:
            hand=hands[0]
            x,y,w,h=hand['bbox']
            #Image empty
            img_bg=np.ones((img_size,img_size,3), np.uint8)*255
            croped_img=img[y-offset:y+ h+offset,x-offset:x+ w+offset]

            aspect_ratio=h/w

            if aspect_ratio>1:
                k=img_size/h
                wCal= math.ceil(k*w)
                img_resize=cv.resize(croped_img,(wCal,img_size))
                wGap =math.ceil((img_size-wCal)/2)
                img_bg[:,wGap:wCal+wGap] = img_resize
            else:
                k=img_size/w
                hCal= math.ceil(k*h)
                img_resize=cv.resize(croped_img,(img_size,hCal))
                hGap =math.ceil((img_size-hCal)/2)
                img_bg[hGap:hCal+hGap,:] = img_resize
```

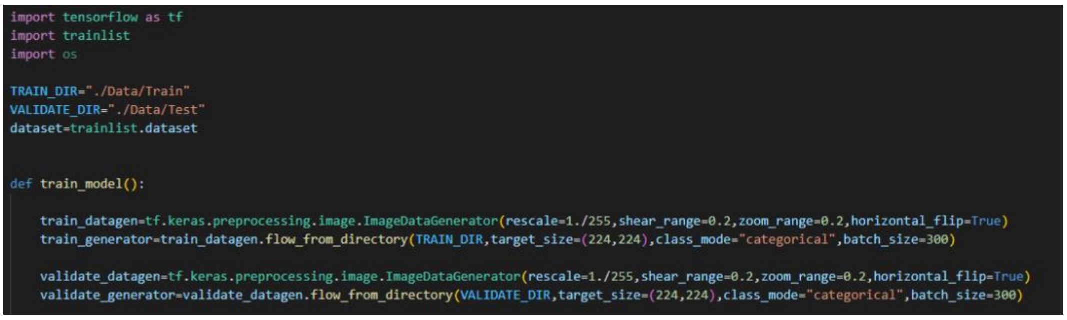
b. Saving the collected image in a Data directory which contain Train and Test folder.



c. Do this for all the dataset labels.

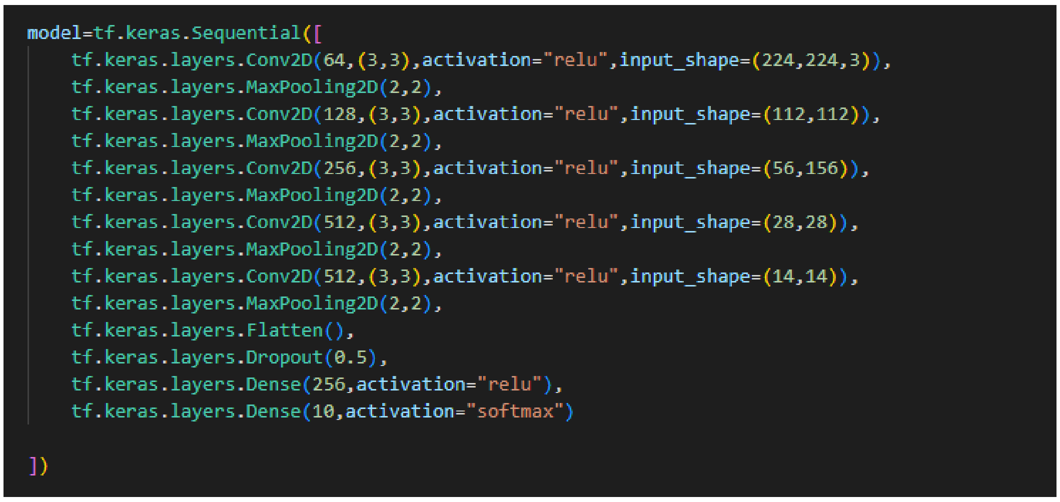
2. Creating CNN model:

a. Import the Tensorflow library to the python program for using the keras module to create CNN.



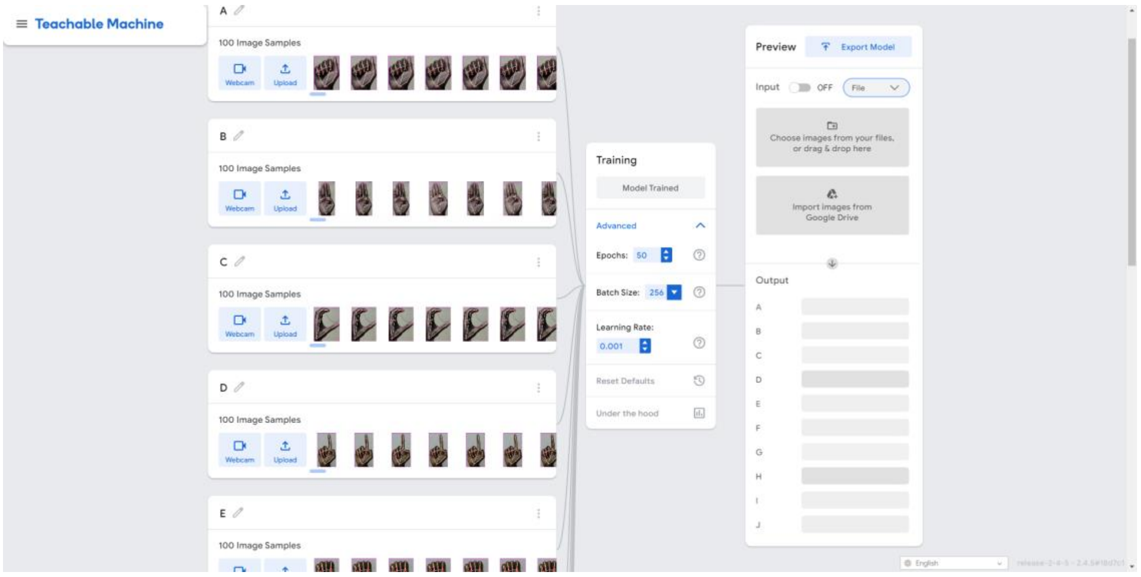
```
import tensorflow as tf
import trainlist
import os

TRAIN_DIR="./Data/Train"
VALIDATE_DIR="./Data/Test"
dataset=trainlist.dataset


def train_model():

    train_datagen=tf.keras.preprocessing.image.ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True)
    train_generator=train_datagen.flow_from_directory(TRAIN_DIR,target_size=(224,224),class_mode="categorical",batch_size=300)

    validate_datagen=tf.keras.preprocessing.image.ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True)
    validate_generator=validate_datagen.flow_from_directory(VALIDATE_DIR,target_size=(224,224),class_mode="categorical",batch_size=300)
```

b. Use Sequential module to build a CNN model and add the required layers such as Conv2D , Maxpool , Flatten , Dense layers.



```
model=tf.keras.Sequential([
    tf.keras.layers.Conv2D(64,(3,3),activation="relu",input_shape=(224,224,3)),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(128,(3,3),activation="relu",input_shape=(112,112)),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(256,(3,3),activation="relu",input_shape=(56,156)),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(512,(3,3),activation="relu",input_shape=(28,28)),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(512,(3,3),activation="relu",input_shape=(14,14)),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.Dense(256,activation="relu"),
    tf.keras.layers.Dense(10,activation="softmax")

])
```

c. To increase the accuracy and reduce the size of our model we use the TEACHABLE MACHINE webapp of google to train our model.



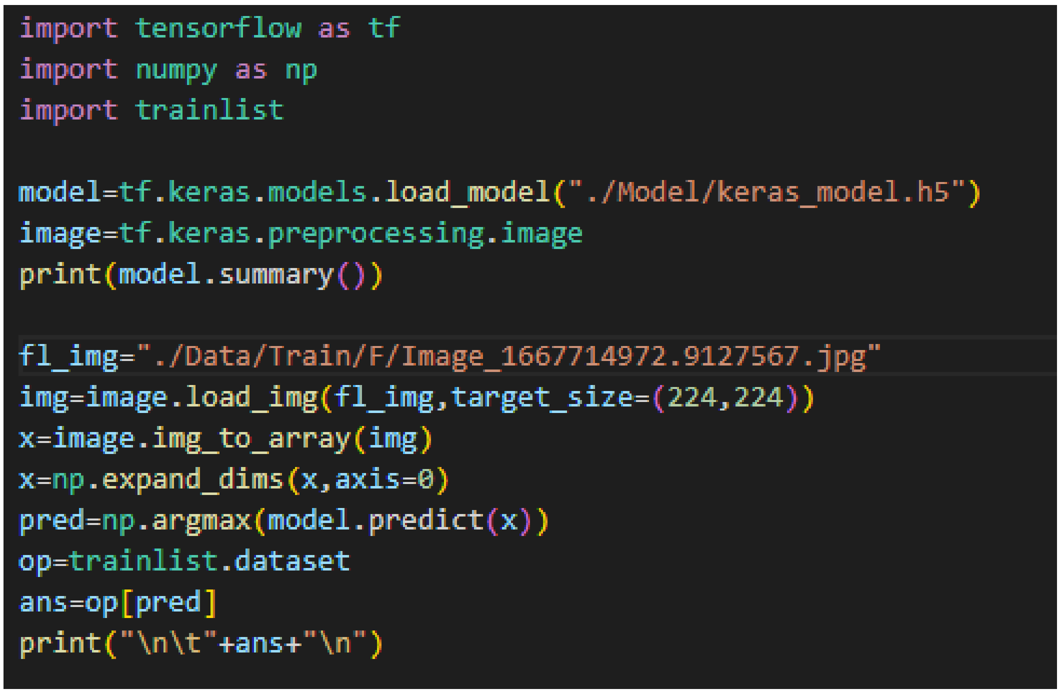
Epochs: 50

Batch Size: 256

Learning Rate: 0.001

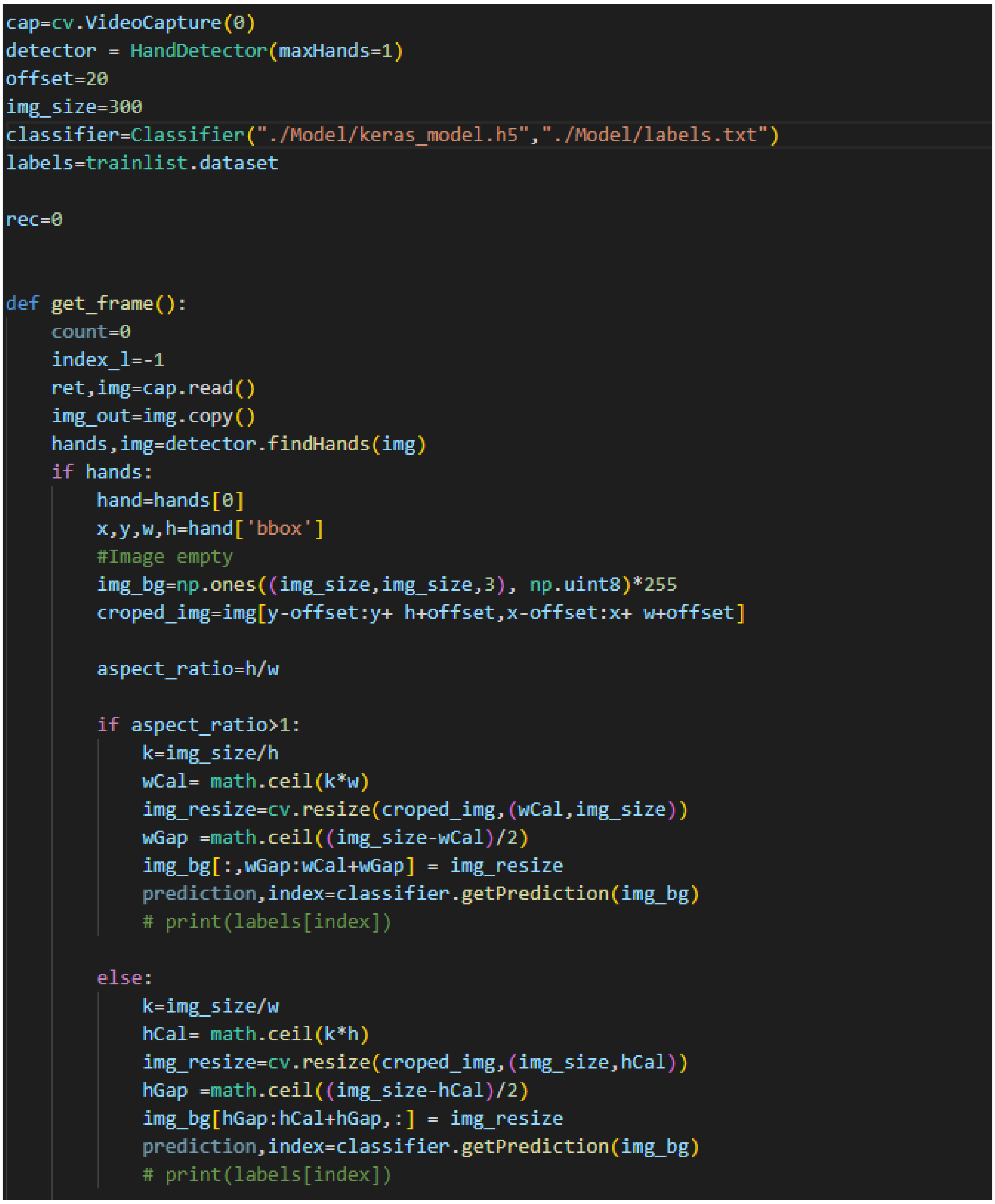
3. **Testing CNN model :**

a. By using the Tensorflow model we can predict the output of our project.



```
import tensorflow as tf
import numpy as np
import trainlist

model=tf.keras.models.load_model("./Model/keras_model.h5")
image=tf.keras.preprocessing.image
print(model.summary())

fl_img="./Data/Train/F/Image_1667714972.9127567.jpg"
img=image.load_img(fl_img,target_size=(224,224))
x=image.img_to_array(img)
x=np.expand_dims(x,axis=0)
pred=np.argmax(model.predict(x))
op=trainlist.dataset
ans=op[pred]
print("\n\t"+ans+"\n")
```

b. We use the cvzone library to predict the Hand Gestures and we also use this module to detect the hand in our image frame.



```
cap=cv.VideoCapture(0)
detector = HandDetector(maxHands=1)
offset=20
img_size=300
classifier=Classifier("./Model/keras_model.h5","./Model/labels.txt")
labels=trainlist.dataset

rec=0


def get_frame():
    count=0
    index_l=-1
    ret,img=cap.read()
    img_out=img.copy()
    hands,img=detector.findHands(img)
    if hands:
        hand=hands[0]
        x,y,w,h=hand['bbox']
        #Image empty
        img_bg=np.ones((img_size,img_size,3), np.uint8)*255
        croped_img=img[y-offset:y+ h+offset,x-offset:x+ w+offset]

        aspect_ratio=h/w

        if aspect_ratio>1:
            k=img_size/h
            wCal= math.ceil(k*w)
            img_resize=cv.resize(croped_img,(wCal,img_size))
            wGap =math.ceil((img_size-wCal)/2)
            img_bg[:,wGap:wCal+wGap] = img_resize
            prediction,index=classifier.getPrediction(img_bg)
            # print(labels[index])

        else:
            k=img_size/w
            hCal= math.ceil(k*h)
            img_resize=cv.resize(croped_img,(img_size,hCal))
            hGap =math.ceil((img_size-hCal)/2)
            img_bg[hGap:hCal+hGap,:] = img_resize
            prediction,index=classifier.getPrediction(img_bg)
            # print(labels[index])
```