Synapse Task 2 Synopsis 60004200157

Kush Maniar

Topic:- Computer Vision

Introduction:-

This synopsis is about utilizing OpenCV and take the pictures from dashboard webcam and feed them into a Deep Learning CNN model which will group whether the individual's eyes are 'Open' or ‘Closed’ . It checks the score of the eye utilizing whether eyes are shut or opened if the eyes are shut completely it identify that the driver is drowsing off so it begins playing an alarm which was set as sound. It stops the caution when again the driver opens their eyes.

This undertaking can be utilized in each vehicle as of now on street to guarantee security and diminish the odds of a mishap because of sleepiness or interruption of the driver. This task can be executed as a versatile application to decrease the expense. It can be coordinated with the vehicle, so programmed speed control can be bestowed if the driver is discovered dozing.

Aim and Objective:-

This can be an important safety implementation as studies suggest that accidents due to drivers getting drowsy or sleepy account for around 20% of all accidents and on certain long journey roads it’s up to 50%. It is a serious issue and most people that have driven for long hours at night can relate to the fact that fatigue and slight brief state of unconsciousness can happen to anyone and everyone.

Overview:-

Step 1: Take frame by frame input from camera using openCV
Step 2:Detect the face and create an ROI (Region of Interest)
Step 3:Detect the eyes and feed it to the haar classifier.
Step 4: Predict the state of the eyes (Closed/Open) using the CNN model.
Step 5: Calculate score to check whether the user is drowsy or no.

Techniques:

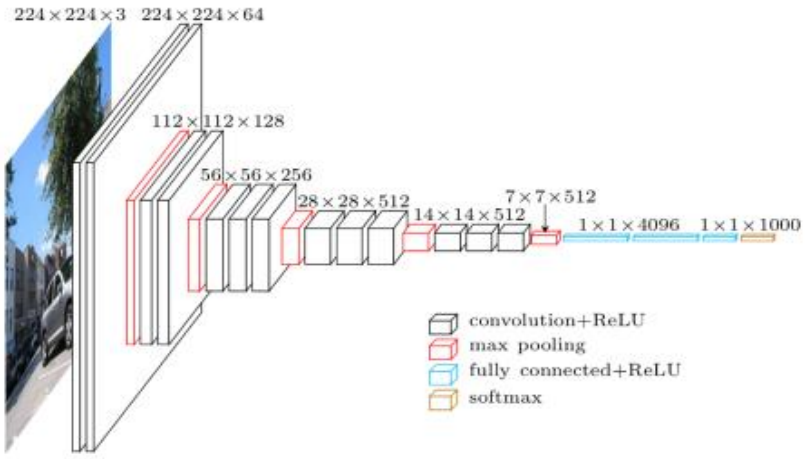
Drowsiness can be detected by many human signs like eyes shutting , yawning, wobbly head, etc. The most accurate way of detecting would be using Biosensors but that require physical contact on the driver's body , therefore the next best approach is to detect the state of the eyes using the trained model and haar classifiers,
With a webcam, we will take images as input. So to access the webcam, we use an infinite loop that will capture each frame, using the method provided by OpenCV to access the camera and set the capture object (cap) which will read each frame and we store the image in a frame variable.

```
cap = cv2.VideoCapture(0)

while(True):
    ret, frame = cap.read()
    height,width = frame.shape[:2]
```

Operational Definition:-

1)CNN(Convulutional Neural Network):

The model we utilized is worked with Keras (bundle) utilizing Convolution Neural Networks (CNN). Convolution neural system is a unique kind of profound neural system which performs incredibly well for picture order purposes used to identify the eyes from a camera and to arrange the picture.The data comprises around 7000 images of people's eyes under different lighting conditions. After training the model, the dataset has been prepared with the final weights. Now this model can be used to classify a person's eyes to be open or closed. The model architecture is designed with Input /Output layer, hidden inner layers and a fully connected layer. Convolution kernels or filters perform performs 2D matrix multiplication on the layer



2)Keras , TensorFlow, Pygame:

Used to construct the grouping model and pygame is used to play the alarm sound. A convolution activity is performed on these layers utilizing a channel that performs 2D framework augmentation on the layer and channel. The Sequential keras class helps us train the model using filters like Conv2D, MaxPooling2D, Dropout, Flatten, Dense. A Relu activation function is used in all the layers except the output layer in which we used Softmax.

```
model = Sequential(
    [
        Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(24,24,1)),
        MaxPooling2D(pool_size=(1,1)),
        Conv2D(32,(3,3),activation='relu'),
        MaxPooling2D(pool_size=(1,1)),
        Conv2D(64, (3, 3), activation='relu'),
        MaxPooling2D(pool_size=(1,1)),
        Dropout(0.25),
        Flatten(),
        Dense(128, activation='relu'),
        Dropout(0.5),
        Dense(2, activation='softmax')
    ]
)
```

3)Haar Cascade File:

To recognize the eyes from the face in the picture, we have to initially change over the picture into grayscale as the OpenCV calculation for object recognition takes dark pictures in the info. We needn't bother with shading data to recognize the items. We will utilize a haar course classifier to identify faces.

The “haar cascade files” consists of the xml files like

- haarcascade_frontalface_alt.xml,
- haarcascade_lefteye_2splits.xml,
- haarcascade_righteye_2splits.xml

that are needed to detect objects from the image. In this case, we are using the haar cascade classifier to detect the face and eyes of the person. The below lines are used to set the classifier.

```
face = cv2.CascadeClassifier('haar cascade files\haarcascade_frontalface_alt.xml')
leye = cv2.CascadeClassifier('haar cascade files\haarcascade_lefteye_2splits.xml')
reye = cv2.CascadeClassifier('haar cascade files\haarcascade_righteye_2splits.xml')
```

Then a rectangle frame is created to make a Region of Interest (ROI). At that point, we utilize faces = face.detectMultiScale. It returns an array of detections

with x,y coordinates, and height, the width of the boundary box of the object. Now we can iterate over the faces and draw boundary boxes for each face.

```
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
faces = face.detectMultiScale(gray,minNeighbors=5,scaleFactor=1.1,minSize=(25,25))
left_eye = leye.detectMultiScale(gray)
right_eye =  reye.detectMultiScale(gray)
cv2.rectangle(frame, (0,height-50) , (200,height) , (0,0,0) , thickness=cv2.FILLED )
for (x,y,w,h) in faces:
    cv2.rectangle(frame, (x,y) , (x+w,y+h) , (100,100,100) , 1 )
```

Methods and Working Procedure:-

The same procedure to detect faces is used to detect eyes. First, we set the cascade classifier for the eyes. Now we can extract the eye from the image by making a boundary box. This will be fed into our CNN classifier which will predict if eyes are open or closed.

```
for (x,y,w,h) in right_eye:
    r_eye=frame[y:y+h,x:x+w]
```

```
for (x,y,w,h) in left_eye:
    l_eye=frame[y:y+h,x:x+w]
```

To feed our image into the model, we need to perform certain operations because the model needs the correct dimensions to start with First, we convert the coloured image into grayscale. Then, we resize the image to 24*24 pixels as our model was trained on 24*24 pixel. We normalize our data for better convergence (All values will be between 0-1). Expand the dimensions to feed into our classifier. We loaded our model using . Now we predict each eye with our model. If the value of predict_r[0] = 1, it states that eyes are open, if value of predict_r[0] = 0 then, it states that eyes are closed.

1.

```
r_eye = cv2.cvtColor(r_eye,cv2.COLOR_BGR2GRAY)
r_eye = cv2.resize(r_eye,(24,24))
r_eye= r_eye/255
r_eye=  r_eye.reshape(24,24,-1)
r_eye = np.expand_dims(r_eye,axis=0)
rpred = model.predict(r_eye)
predict_r = np.argmax(rpred, axis=1)
if(predict_r[0]==1):
    lbl='Open'
if(predict_r[0]==0):
    lbl='Closed'
break
```

2.

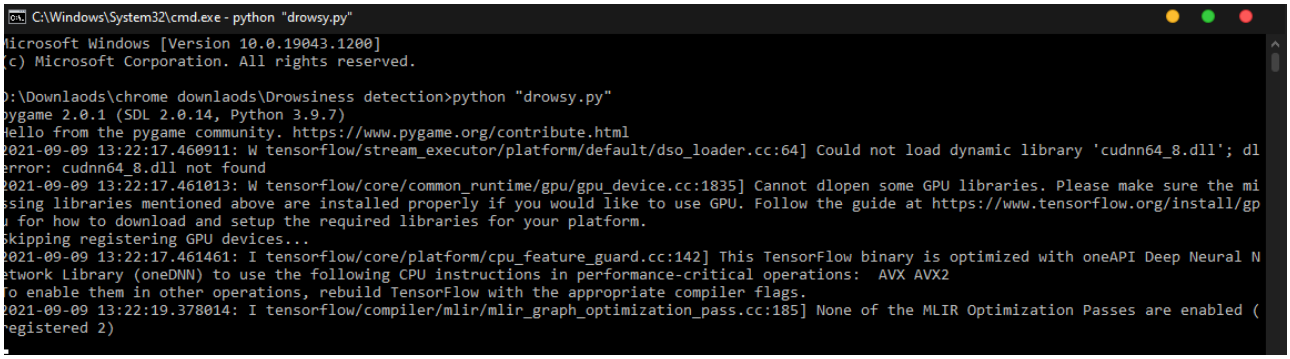
```
l_eye = cv2.cvtColor(l_eye,cv2.COLOR_BGR2GRAY)
l_eye = cv2.resize(l_eye,(24,24))
l_eye= l_eye/255
l_eye=l_eye.reshape(24,24,-1)
l_eye = np.expand_dims(l_eye,axis=0)
lpred = model.predict(l_eye)
predict_l = np.argmax(lpred, axis=1)
if(predict_l[0]==1):
    lbl='Open'
if(predict_l[0]==0):
    lbl='Closed'
break
```

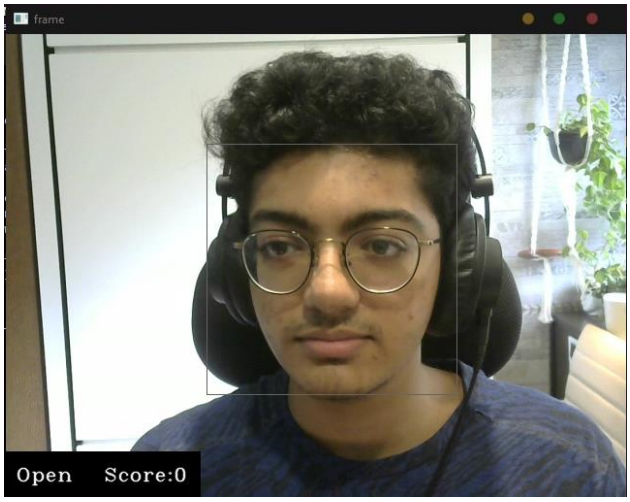
As explained above, a threshold is defined to track the amount of time the eyes were closed. For example if score becomes greater than 15 that means the person's eyes are closed for a long period of time.
If the predict_r[0] and predict_l[0]==0 , it means that the eyes are closed and the score keeps increasing till it crosses 15, this is when the alarm is played using **sound.play()** (roblox death OOF.wav) and the frame is given a red border.
If the predict_r[0] and predict_l[0]==1 it detects the eyes to be open the score decrements back to 0 and then the alarm stops.

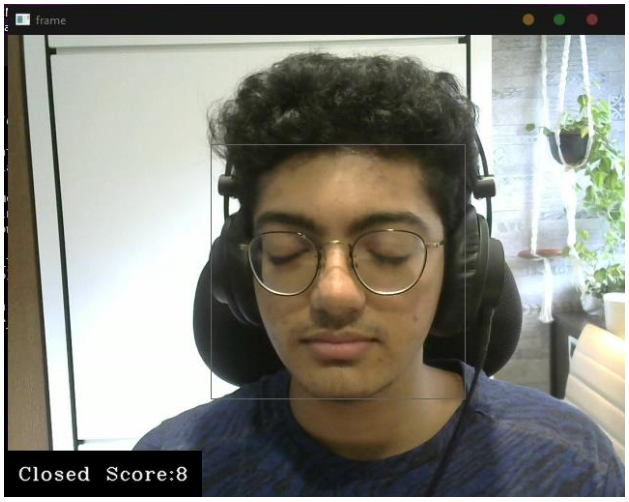
```
if(predict_r[0]==0 and predict_l[0]==0):
    score=score+1
    cv2.putText(frame,"Eyes are Closed",(10,height-20), font, 1,(255,255,255),1,cv2.LINE_AA)

else:
    score=score-1
    cv2.putText(frame,"Eyss are Open",(10,height-20), font, 1,(255,255,255),1,cv2.LINE_AA)


if(score<0):
    score=0
cv2.putText(frame,'Score:'+str(score),(100,height-20),x font, 1,(255,255,255),1,cv2.LINE_AA)
if(score>15):

    cv2.imwrite(os.path.join(path,'image.jpg'),frame)
    try:
        sound.play()

    except:
        pass
    if(thick<16):
        thick= thick+2
    else:
        thick=thick-2
        if(thick<2):
            thick=2
    cv2.rectangle(frame,(0,0),(width,height),(0,0,255),thick)
cv2.imshow('frame',frame)
```

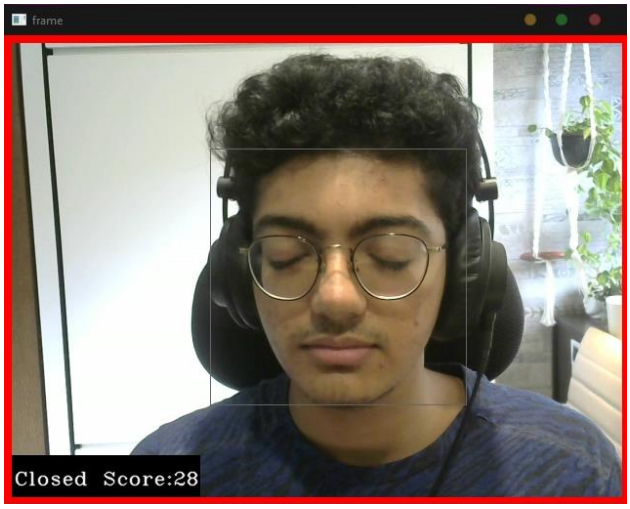
Output:-

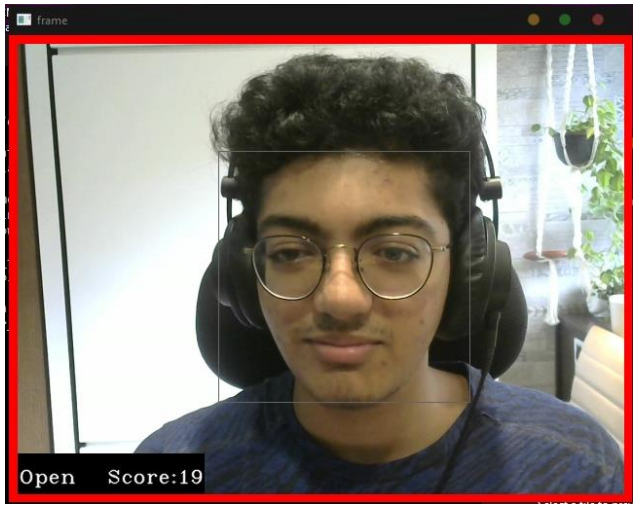
To start the project, you need to open a command prompt, go to the directory where our main file "drowsy.py" exists. Run the script with this command.
python "drowsy.py"
It may take a few seconds to open the webcam and start detection.

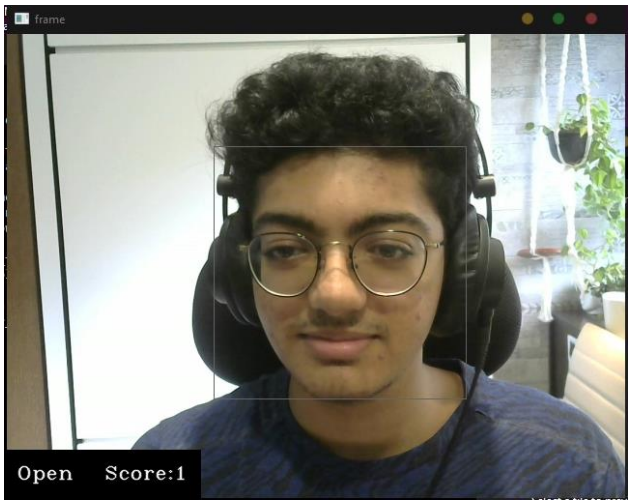

C:\Windows\System32\cmd.exe - python "drowsy.py"
Microsoft Windows [Version 10.0.19043.1200]
(c) Microsoft Corporation. All rights reserved.
D:\Downlaods\chrome downlaods\Drowsiness detection>python "drowsy.py"
pygame 2.0.1 (SDL 2.0.14, Python 3.9.7)
Hello from the pygame community. https://www.pygame.org/contribute.html
2021-09-09 13:22:17.460911: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cudnn64_8.dll'; dl
error: cudnn64_8.dll not found
2021-09-09 13:22:17.461013: W tensorflow/core/common_runtime/gpu/gpu_device.cc:1835] Cannot dlopen some GPU libraries. Please make sure the mi
ssing libraries mentioned above are installed properly if you would like to use GPU. Follow the guide at https://www.tensorflow.org/install/gp
u for how to download and setup the required libraries for your platform.
Skipping registering GPU devices...
2021-09-09 13:22:17.461461: I tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow binary is optimized with oneAPI Deep Neural N
etwork Library (oneDNN) to use the following CPU instructions in performance-critical operations: AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
2021-09-09 13:22:19.378014: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:185] None of the MLIR Optimization Passes are enabled (
registered 2)


frame
Open Score:0


frame
Closed Score:8


frame
Closed Score:28


frame
Open Score:19


frame
Open Score:1


frame
Open Score:0

Conclusion:-

In this project ,I used OpenCV to detect faces and eyes using a haar cascade classifier and then a CNN model , i.e. a highly optimized deep neural network was used to predict the driver's drowsiness state, it was designed and compressed for embedded systems. The minimum inputs to detect driver's drowsiness and a compression technique of knowledge distillation are applied to be implemented on a real-time embedded system. The result of the project is used to avoid accidents that are made by the drowsiness of the driver it avoided by using this project by alerting the driver when they feels sleepy.

References:-

https://towardsdatascience.com/dont-sleep-building-your-first-drowsiness-detection-system-28a9903015f3

https://www.youtube.com/watch?v=zfiSAzpy9NM

https://www.pygame.org/docs/ref/mixer.html

https://machinelearningmastery.com/tutorial-first-neural-network-python-keras/

https://docs.opencv.org/3.4/d1/de5/classcv_1_1CascadeClassifier.html#ab3e572643114c43b21074df48c565a27

https://www.youtube.com/watch?v=j-3vuBynnOE