Drowsiness Detection System using Deep Learning- DDS-101

Abdallah Abdelnaby & Youssef Hussien
The American University in Cairo, Computer Engineering Department
Computer Engineering Department, CSCE 4604

I- Abstract

Drowsiness detection is a safety technology that can prevent accidents caused by drowsy -sleepy- drivers while driving. This project aims to build a drowsiness detection system that will detect if a driver's eyes are closed for a few seconds. The system detects this state of drowsiness and shall alert the driver whenever this happens. If the driver's drowsiness can be predicted at the initial stages, and the driver can be alerted, then the number of accidents can be reduced. We built this system using deep learning techniques which best fit the problem's requirements. The data set we use is available online by Kaggle platform and contains images of many people's closed and open eyes and includes images of people yawning versus non-yawning people. This dataset is best suited for detecting whether a person is drowsy or not. The validation accuracy of our system reached 95%, while the testing accuracy reached 95.15%. The system can detect drowsiness in both well-lit environments and dark ones. Additionally, our system is able to efficiently detect multiple persons at once and detect the drowsiness state of the left-most person assuming this is the driving side.

II- Introduction

Driving drowsiness or driving sleepiness is the state of the person when he/she feels sleepy while driving. Many people drive on the highway, day and night, whether they are bus drivers, truck drivers, or even people traveling long-distance. All of these people suffer from the common problem of feeling sleepy while driving. This problem is becoming very dangerous and results in many accidents globally. According to [1], the National Highway Safety Traffic Administration (NHTSA) found that between 2013 and 2017, there were a total of 4,111 fatalities that involved drowsy driving. In 2017, 91,000 police-reported crashes involved drowsy drivers. Those crashes led to about 50,000 people being injured. This and other studies show the dangerousness of this problem, and various techniques are being implemented to minimize this phenomenon.

Drowsiness detection is a safety technology that can prevent accidents caused by drowsy drivers while driving. This paper aims to build a drowsiness detection system that will detect if a driver's eyes are closed for a few seconds. The system detects this state and shall alert the driver whenever this happens. If the driver's drowsiness can be predicted at the initial stages, and the driver can be alerted, then the number of accidents can be reduced. We built this project using deep learning techniques which best fit the problem's requirements. The data set we use is available online by Kaggle [5] and contains images of many people's closed and open eyes and includes images of people yawning versus no-yawning people. This dataset is best suited for detecting whether a person is drowsy or not. Afterward, the system will alert the drowsy driver based on his state.

Section-II in this paper includes a comparative study summarizing the current solutions to this problem implemented by other researchers. Then in section III, we discuss our solution in detail, including how the proposed solution will be different from other solutions. Afterward, in section IV, we discuss details about the dataset used in the experiments, outline the experiments' setup, and compare these results to our benchmark. Finally, in section V we conclude our study with several concluding points and suggest several directions for further extensibility of this project.

III- Review of Existing Solutions

There is an increasing number of solutions that are implemented to approach this problem. In [2] Avasare and Khanna design a machine learning approach to detect drivers' drowsiness. They use OpenCV and Keras, and place a camera in front of the driver to provide live feed of data. If the eyes of the driver are closed for a specific amount of time, the proposed framework draws a conclusion that the driver is feeling drowsy and an alarm for safety is sound. Their system initially works on detecting the driver's face and then works on detecting his eyes. Their detection system has a detection accuracy that reaches 87.5% in general and when the face is kept aligned to the webcam and there is good lighting the accuracy can reach around 95% accuracy. They use the same dataset that we are working on, which will provide a common baseline.

The second related work is [3] a research done by Mazloumi, et al. in which they use Voilla-Jones algorithm to detect facial expressions as well as the location of the eyes. They define a criteria for detecting drivers' levels of drowsiness by eye tracking including eye blink duration, blink frequency and PERCLOS that was used to confirm the results. They conclude that eye closure duration and blink frequency have a direct ratio of drivers' levels of drowsiness. Meanwhile, the percentage of accuracy of the detection of their system is 93%.

The third and final research [4] we find most relevant to our project is the work done by Singh, et al. where they also built a machine learning system to detect drivers' drowsiness. They also used OpenCV and Keras in their system and they converted the captured images to grayscale. They detect the driver's face after capturing the image using a camera placed in front of the driver generating continuous video. Then they detect the left and right eyes of the driver, after which they are classified as either open or closed eyes. Their work is similar to the first research we discussed. When the score of drowsiness is over 15, an alarm is fired to alert the driver. In their model they first, two hidden layers were chosen which provided the accuracy of around 50%, which is not quite satisfactory. Then, in order to improve their accuracy, they added another hidden layer which increased the accuracy almost twice (from around 50% to around 95%). Thus, from the previous research we decided to make the model accuracy our benchmark and we aim to increase the accuracy to either overcome the 95%

as other solutions or simplify the model while obtaining an approximate accuracy.

IV- Proposed Solution

In this section we explain our solution and how it is different from the existing solutions.

After reading through the literature, we had a solid idea of where to begin. We saw that there are very good results in the literature that reach 95%, so our solution was aiming at even higher accuracy. Our solution is a drowsiness detection system using deep learning. When the driver is feeling drowsy the system will detect this, and if this state of sleepiness continues for a number of seconds an alarm is issued to the driver. The alarm will keep ringing until the system detects that the user is not sleepy any more. To reach this end goal, we will train a Convolutional Neural Network model (CNN) on a data set consisting of images of closed and open eyes as well as images of people yawning and not yawning. After the model is trained on this data set and we reach our desired accuracy, the best model's weights are saved. Then we will capture a live stream of image frames of the user using OpenCV. From these continuous frames, in each frame the image is fed to the trained model which will predict if the driver is drowsy or not. When the model detects drowsiness of the driver for a number of continuous frames, i.e seconds, this implies that the driver is feeling drowsy and that this is not a one second thing. Then the agent alarms the driver by initiating an alarm which will keep ringing until the driver's drowsiness score is reduced again.

There are five major steps that each image undergoes until a score is given to it. Firstly, we take an image as an input from a camera, then we detect the face in the image and create a Region of Interest (ROI). Afterwards, we detect the eyes from ROI and feed it to the classifier. The classifier will classify whether the eyes are open or closed.Finally, the model calculates a score to check whether the person is drowsy. A score of negative one represents a non-drowsy driver while a score of positive one represents a drowsy driver. This process continues for a

continuous number of frames until the cumulative score reaches 15 or higher. If this is the case then an alarm is fired, and keeps firing until the cumulative score is less than 15 again.

The CNN-model we built is using TensorFlow and Keras. We built a CNN as the classification model, to classify if the eyes of a person are closed or not. Our CNN model architecture is a sequential model that consists of three 2d convolution layers, three max-pooling layers, two dense layers, two dropout layers, one flattening layer and one output layer. The full model details are shown in the below figure.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 22, 22, 32)	320
max_pooling2d (MaxPooling2D)	(None, 22, 22, 32)	0
conv2d_1 (Conv2D)	(None, 20, 20, 32)	9248
max_pooling2d_1 (MaxPooling2D)	(None, 20, 20, 32)	0
conv2d_2 (Conv2D)	(None, 18, 18, 64)	18496
max_pooling2d_2 (MaxPooling2D)	(None, 18, 18, 64)	0
dropout (Dropout)	(None, 18, 18, 64)	0
flatten (Flatten)	(None, 20736)	0
dense (Dense)	(None, 128)	2654336
dropout_1 (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 4)	516

After capturing the image from the camera, we convert it to grayscale as the OpenCV algorithm for object detection takes gray images in the input. We don't need color information to detect the objects. Then we use the HaarCascade classifier to detect faces in the image, and draw the respective ROI. The HaarCascade returns an array of detections with x,y coordinates, and height, and

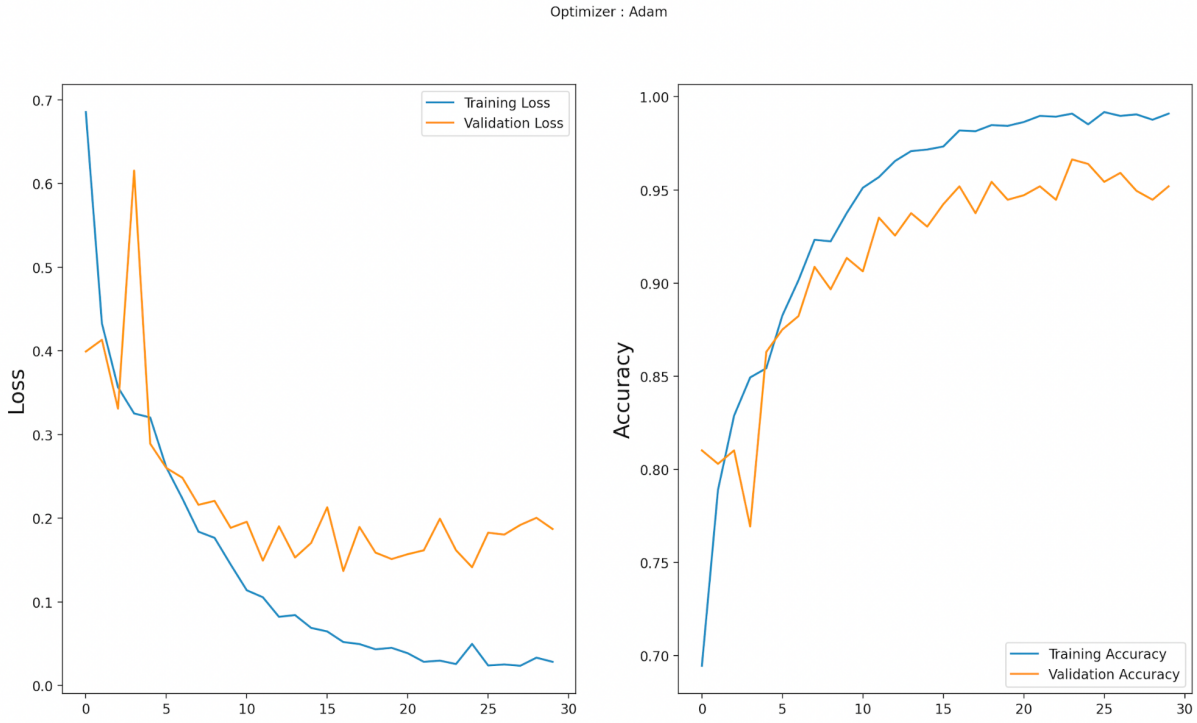
the width of the boundary box of the object. Then we iterate over the faces and draw boundary boxes for each face.If we have multiple faces detected, they are aligned on the x-axis and we only care about the face at the left. We do this as an assumption that the direction of driving the car is to the left. After that, we do the same procedure for the left eye and the right eye detection. For each eye in the two eyes we feed the image into the CNN classifier which will predict if this eye is open or closed.

Before feeding the eyes' images into the model, we perform certain image processing steps. First, we convert the color image into grayscale using, then we resize the image to 24*24 pixels as our model was trained on 24*24 pixel images. Afterwards, we normalize the images and expand the dimensions to feed into our classifier. Then we load our pre-trained CNN model and predict each eye. If the value of the prediction is 1, it states that eyes are open, if the value of is 0 then, it states that eyes are closed. To sum up, this is the model architecture we designed to address the problem.

V- Experiments and Results

V.1 Training Vs Validation Accuracy

The first results we report are the training vs validation accuracy and loss. We trained different models with different architectures, and the results obtained in the below figure represent the best results. The validation accuracy exceeded the 95% as shown in the graph, while the model could reach more than 97% training accuracy. The model started to over-fit starting from the 25th epoch, so we stopped the training after 30 epochs as there is no need to keep training the model.



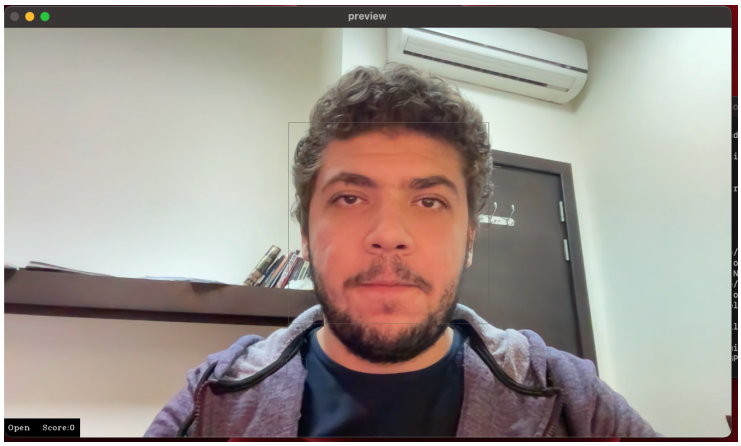
Model ACCR:

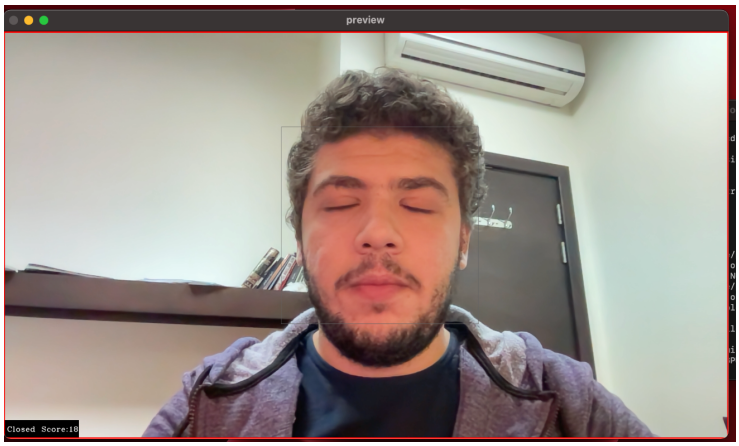
After picking the best model based on the previous results, we tested the model on the testing set. The overall accuracy -ACCR- of the model is 95.15% -Figure3- which is slightly better than the research papers in the literature.

```
Model Overall ACCR is 95.15151977539062% and model overall Loss is 0.23065192997
455597
(env_tensorflow) abdallahmohamed@Abdallahs-MacBook-Pro models %
```

V.2 Well Lit Environment

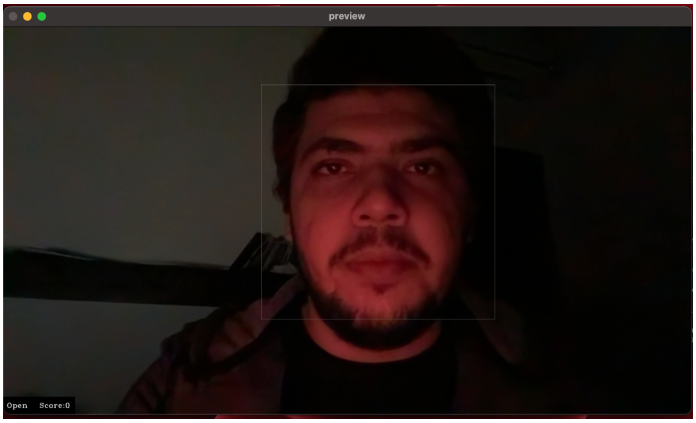
In this experiment we test the correctness of the system by running it in a well-lit environment. As shown in Figure4, the system outputs a score of zero when both eyes are opened. Once the driver closes both left and right eyes, the score keeps going up and fires the alarm when it exceeds 15, representing a possible drowsy driver.

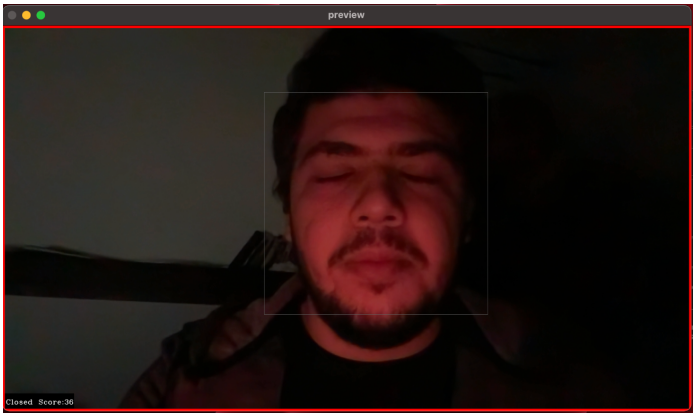




V.3 Dark Environment

In this case we are testing the detection system in a low light (dark) environment. The system is successfully able to detect drowsy people in dark environments as the case in the well-lit environment. However, when the light gets very low, it will not be able to detect the face.





V.4 Multiple Persons

In this case, we are assuming that there are multiple faces in the frame. Considering the driver has his friend sitting on the seat next to him, the camera will capture both faces. That is why we designed the system to process the face of the person sitting on the wheel on the left. The results showed that the system is successfully able to detect multiple persons in the image and only work on the drowsiness state of the person on the left, Figure5. That is why the score interacts only with the driver, not other people in the frame.

VI- Conclusion

The driver drowsiness model is built on capturing a video stream of the driver, and alerting the user if detected drowsy for more than 15 consecutive frames. The data set used consists of four classes: closed eyes, open eyes, yawning, and not yawning and data augmentation is used to increase the data. The model is trained on the training set and gets the best model according to the best validation accuracy and validation loss using the validation set. The overall ACCR accuracy reported by the model is 95.15% The cascade classifier is used to detect the face of the driver and his eyes and draw a ROI for each object. The detected objects are fed to the trained classification model to predict the drowsiness of the driver. The detection system passed all experiments successfully by successfully detecting the drowsiness of the user most of the time and alerting them. The system works correctly both in well-lit environments and dark environments. The system when faced with multiple persons in the same video, it picks the leftmost person assuming that he is the driver.

VII- References

[1] Stanley Coren. Daylight savings time and traffic accidents. New England Journal of Medicine, 334(14):924–925, Apr 1996

[2] Avasare, P., Khanna, K.; Chawan, P. (2021). Driver Drowsiness Detection System using openCV and keras. International Research Journal of Engineering and Technology (IRJET), 8(8).

[3] Poursadeghiyan, M., Mazloumi, A., Nasl Saraji, G., Baneshi, M. M., Khammar, A., & Ebrahimi, M. H. (2018). Using Image Processing in the Proposed Drowsiness Detection System Design. Iranian journal of public health, 47(9), 1371–1378.

[4] Kaushish, V., Singh, N., Maggo, K., Nagrath, P., & Jain, R. (2021). Driver drowsiness detection system with opencv and keras. *SSRN Electronic Journal*.

[5] S. Rajubla, ”yawn eye data set”, kaggle. https://www.kaggle.com/serenaraju/yawn-eye-dataset-new. Accessed: 2021-12-01.