JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 2, 2022, p. 2006 - 2012
https://publishoa.com
ISSN: 1309-3452

Real Time Driver Drowsiness Detection Based on Convolution Neural Network

M.A. Ahmed[1], Harith A. Hussein[2], Mohammed Basim Omar[3], Qabas A Hameed[4],

[1,2,3,4] College of Computer Science and Mathematics, Department of computer science, Tikrit University, Saladin, (Tikrit), 34001, Iraq
[1] Mohamed.aktham@tu.edu.iq
[2] Harith_abd1981@tu.edu.iq
[3] mahammed.b.omer35524@st.tu.edu.iq
[4] qabas.a.hameed@tu.edu.iq

***Abstract* __** Fatigue, in reality, is a severe threat on the road since it affects a driver's ability to react and process information. Fatigue is the primary cause of the rising frequency of road accidents. The power of these algorithms to accommodate variance in human face and lightning conditions is one of their biggest hurdles. We want to implement an advanced processing system that will dramatically minimize traffic accidents. We may use this method to determine face traits in drivers, such as the proportion of time their eyes are closed. We offer an effective and non-intrusive approach for detecting driver weariness in this paper using eye extraction. A webcam is used to observe the driver in this system continuously. Haar cascade classifiers are used to recognize the driver's face and eye. Eye pictures are collected and sent into a custom-built Convolutional Neural Network to determine if the left and right eyes are closed. The eye closure score is derived based on the classification. An alert will sound if the driver is judged asleep.

Keywords – Convolutional neural network; Drowsiness, Driver monitoring, Deep Learning

I- INTRODUCTION

Autonomous driving automobiles are becoming increasingly popular in today's cutting-edge automotive technology, but they still have a long way to go in terms of safety and reliability. Hybrid automobiles are currently the most popular, and they rely entirely on human skills for speedy travel and safety, and anyone can be a victim of driving while asleep. The circumstance is usually caused by insufficient night sleep or an unaltered physical condition, with the driving environment being taken into mind. Drowsiness detectors are divided into three categories: vehicle-based, signal-based, and facial feature-based. Vehicle-based approaches attempt to infer tiredness from changes in steering wheel angle, acceleration, lateral position, and other factors. These methods, on the other hand, are too slow for real-time jobs. Drowsiness is inferred from psychophysiological characteristics using signal-based approaches. Several research based on these methodologies has been conducted in recent years [1]. Based on a review of the literature. Still there need for a reliable intelligent driver tiredness detection system. The goal is to develop an intelligent processing technique that will help prevent traffic accidents. This can be accomplished by continuously detecting tiredness and alerting the driver of inattention to avoid accidents.

II-LITERATURE SURVEY

A straightforward method for detecting eyeballs, the length of the intensity change in the eye region is calculated using a primary method for recognizing eyes. Other physiological parameters such as blink rate, yawning, head movement, and so on are not considered. It is not suitable for processing in real-time [2]. The representation of movement vectors was incorporated into spatial and impermanent information [3]. Only yawning is the focus of Google Net [4], which employs frame-by-frame prediction. As a result, it necessitates a large amount of memory. The technique for extracting a feature is not properly defined. For all electrodes and changes in spatial input activity, the relevance of temporal dependency learning was associated [5]. Computational techniques for fatigue identification have been difficult to develop using electroencephalography (EEG) signals. When compared to prior algorithms, CNN with increased information and connected datasets has slightly poorer accuracy [6]. In another study for detecting driver drowsiness with the ability to aggregate frame-level CNN outputs into video-level, predictions in order to extract spatial picture data and a Long Short Term Memory Network to analyze temporal aspects. However, we should point out that they lack fine distinction and can make mistakes in some situations, such as when there isn't enough light [7]. A driving simulator capable of creating realistic landscapes and driving experiences while allowing for both autonomous and manual driving. The driver's state is constantly monitored and the driver's drowsy is assessed. However, it did not account for changing lighting conditions or collision avoidance in autonomous driving mode when the driver becomes fatigued [8].

III. PROPOSED SYSTEM

Our proposed method will offer a way to keep track of a driver's drowsiness. The present system's drawbacks of extracting only selected hand-crafted characteristics are overcome by implementing a custom-designed CNN and providing an input driver image. A webcam will be used to observe the driver continuously. The collected footage is transformed into a frame sequence. The face and eye are recognized in each frame using predefined classifiers termed haar cascade classifiers, available in OpenCV. Eye pictures are retrieved and routed to a sequence of 2D CNN layers (3x3 kernel valid padding), max-pooling layers (2x2), and the fully connected dense layer, which determines whether or not the eyes are closed. Eye closure is used to calculate a score. If both eyes are closed for 15 frames in a row, the system interprets this as drowsy and sounds an alarm to notify the driver. By employing custom-designed CNN, the categorization of driver drowsiness is done appropriately, and the normalization concerns are eliminated.

A. System Architecture

The flow of the system to be constructed is depicted in Figure 1. A webcam is used to watch the driver in the first stage. The video input is transformed into a frame sequence. Haar cascade classifiers recognize the

driver's face and eyes in each frame. The images of the detected eyeballs are saved as a data set for CNN. Following that, the pictures of both eyes are subjected to several image preprocessing operations such as

JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 2, 2022, p. 2006 - 2012
https://publishoa.com
ISSN: 1309-3452

grayscale conversion, resizing, and normalizing, among others. The method also allows for preparing the ocular data set to train the CNN model. Data augmentation is used to train the image and expand the number of data sets. It is fed into a pre-trained CNN model with convolution layers, max-pooling layers, and dense layers to predict eye closure. A score is calculated based on the forecast. If the system determines that the driver is drowsy, an alarm will notify the driver.

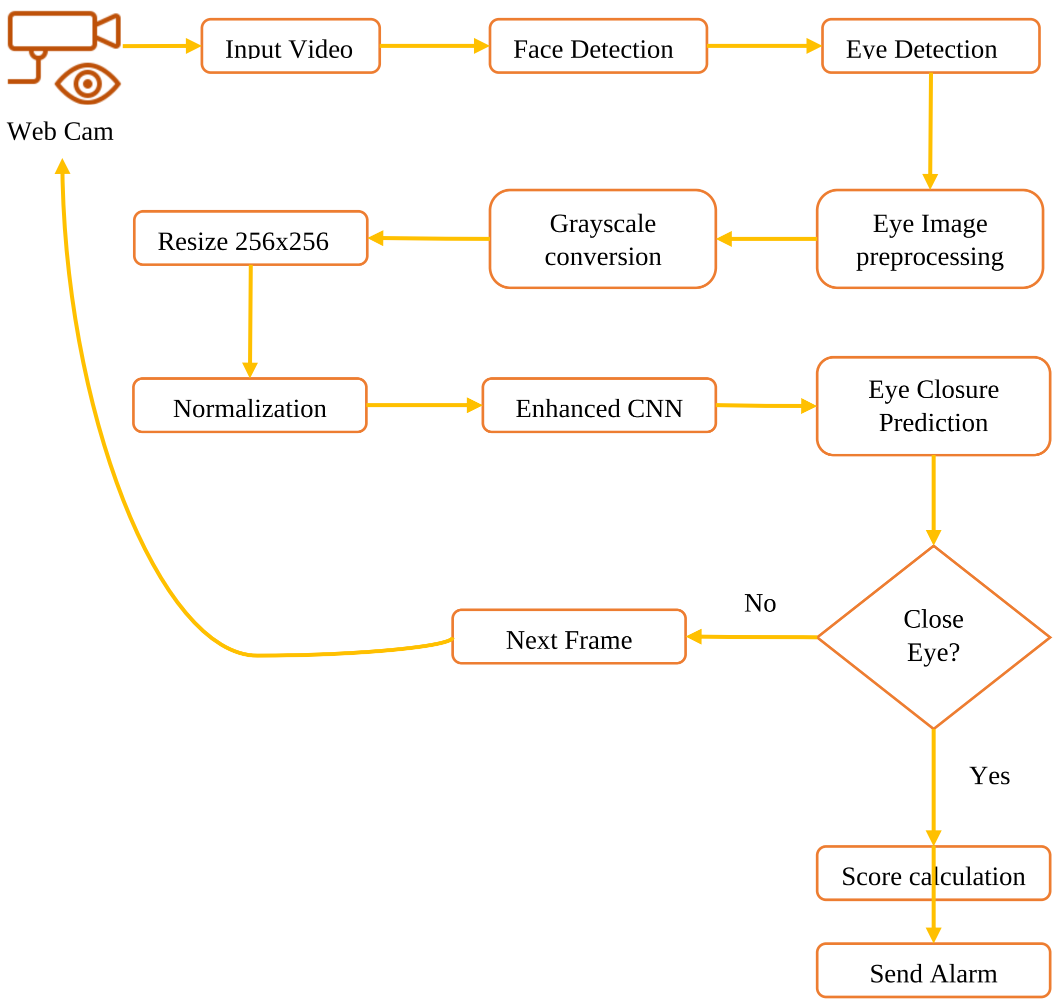


Fig. 1 System Architecture of CNN

B. Implementation Description

Face and eye detection, preprocessing and labeling, data augmentation, enhanced CNN, and alarm triggering are components. The Face and eye identification module in OpenCV uses a loading technique to supply pre-trained models.

The OpenCV information folder contains the pre-trained forms. To examine the image of eyeballs, the system will employ previously trained Haar models. Cascade Classifier is created and uses its technique to load the necessary XML file. The multiscale approach then generates a bound rectangle for the observed eyes. We preprocessed and labeled the eyes images received from a webcam as open and closed eyes to prepare our dataset. The visual images were extracted and transformed to grayscale images, then scaled to 256x256 pixels and normalized. For the Data Augmentation module, we did not acquire new data; instead, we changed existing data. We achieve data augmentation with Keras, which accepts rotation, brightness, shear, zoom, and other ranges as parameters. As illustrated in Figure 2, each CNN layer includes multiple arguments that can be adjusted and used to perform various tasks on the input data. In convolutional layers with a kernel size of 3x3, we employed the relu activation function. In completely connected layers, the Softmax activation function is employed to produce either open or closed eyes. The adam optimizer is used to train CNN.

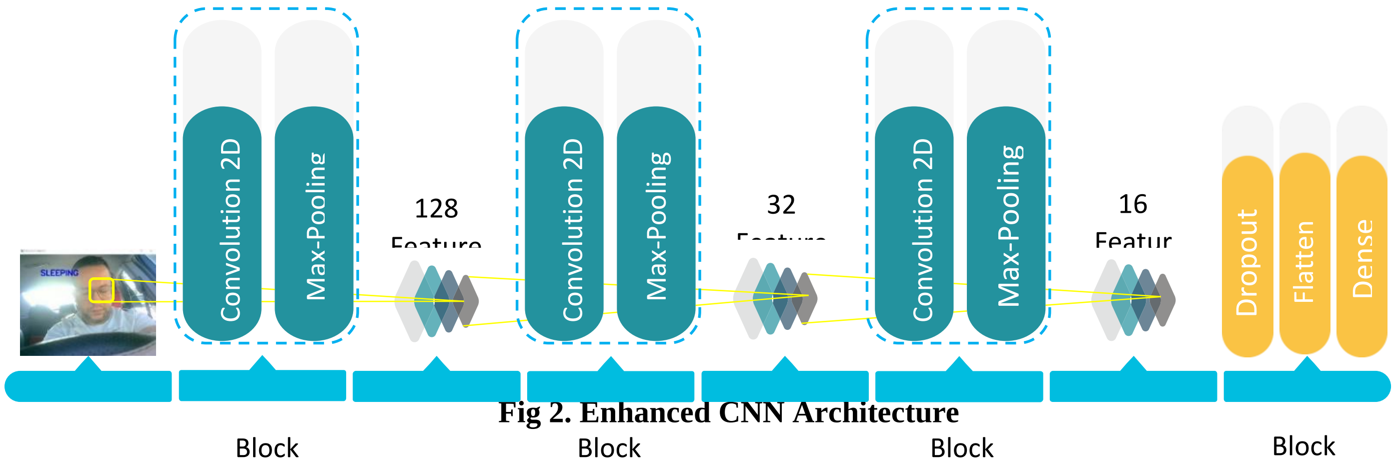
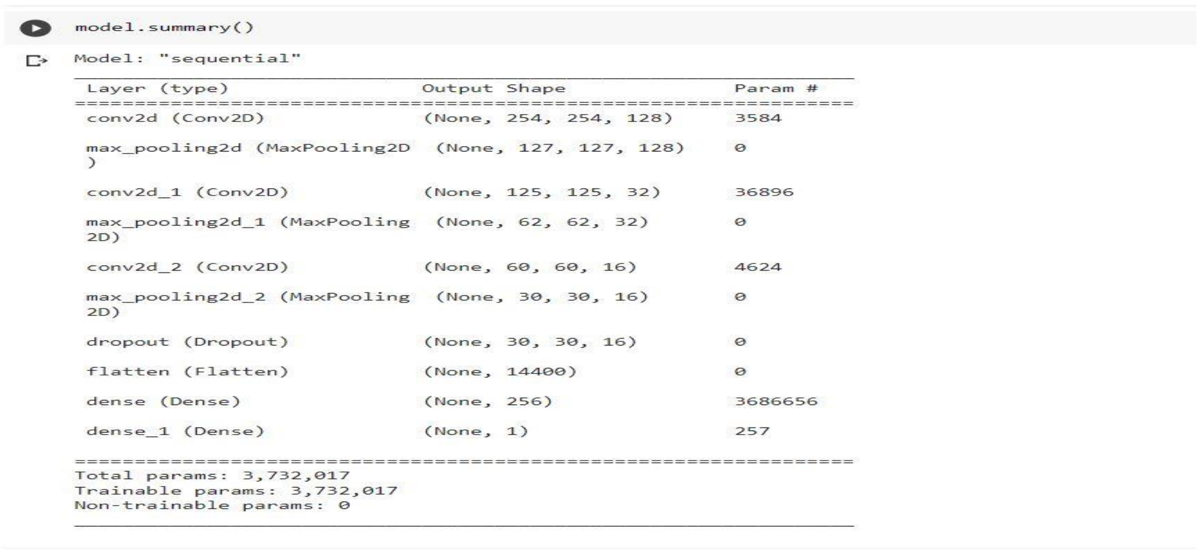


Fig 2. Enhanced CNN Architecture

To notify the driver before going to sleep, we use the Pygame library. The score value is used to calculate how long the motorist has been driving with his or her eyes closed. We raise the score when both eyes are closed, and we lower the score when both eyes are open. We're working on a result that will show the driver's current time situation.

the model summary explains the layers involved in CNN, the input and output shapes, and many trainable parameters. Extra layers such as dropout, dense, and flatten are added to improve learning efficacy. As shown in Figure 3.

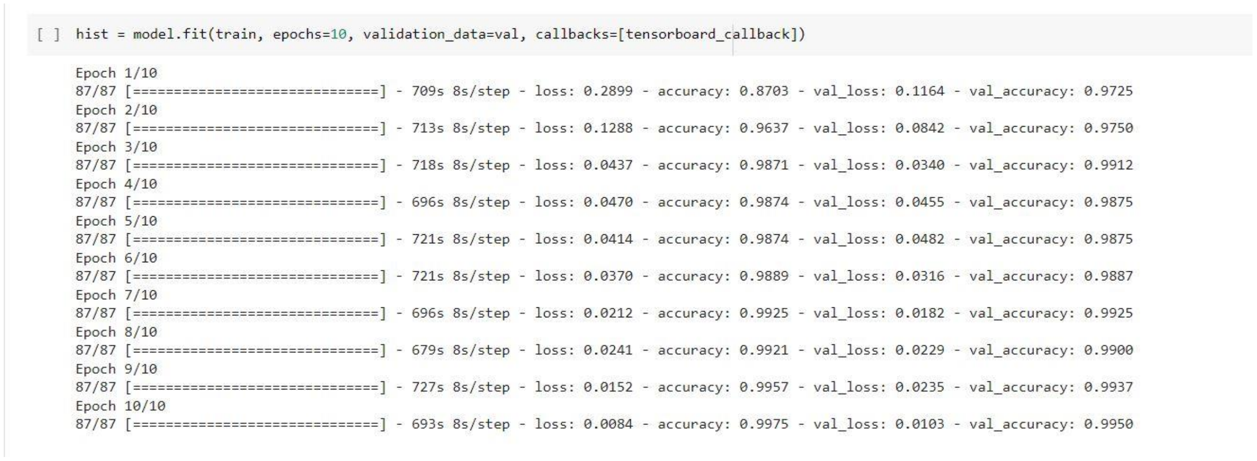
As shown in Figure 4, this system takes ten epochs to complete, with training accuracy and losses.



```
model.summary()

Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 254, 254, 128)     3584

 max_pooling2d (MaxPooling2D  (None, 127, 127, 128)    0
 )

 conv2d_1 (Conv2D)           (None, 125, 125, 32)      36896

 max_pooling2d_1 (MaxPooling  (None, 62, 62, 32)       0
 2D)

 conv2d_2 (Conv2D)           (None, 60, 60, 16)        4624

 max_pooling2d_2 (MaxPooling  (None, 30, 30, 16)       0
 2D)

 dropout (Dropout)           (None, 30, 30, 16)        0

 flatten (Flatten)           (None, 14400)             0

 dense (Dense)               (None, 256)               3686656

 dense_1 (Dense)             (None, 1)                 257

=================================================================
Total params: 3,732,017
Trainable params: 3,732,017
Non-trainable params: 0
_________________________________________________________________
```

Fig 3. Model summary of Enhanced CNN

Fig. 4. Output of Enhanced CNN



```
[ ]  hist = model.fit(train, epochs=10, validation_data=val, callbacks=[tensorboard_callback])

     Epoch 1/10
     87/87 [==============================] - 709s 8s/step - loss: 0.2899 - accuracy: 0.8703 - val_loss: 0.1164 - val_accuracy: 0.9725
     Epoch 2/10
     87/87 [==============================] - 713s 8s/step - loss: 0.1288 - accuracy: 0.9637 - val_loss: 0.0842 - val_accuracy: 0.9750
     Epoch 3/10
     87/87 [==============================] - 718s 8s/step - loss: 0.0437 - accuracy: 0.9871 - val_loss: 0.0340 - val_accuracy: 0.9912
     Epoch 4/10
     87/87 [==============================] - 696s 8s/step - loss: 0.0470 - accuracy: 0.9874 - val_loss: 0.0455 - val_accuracy: 0.9875
     Epoch 5/10
     87/87 [==============================] - 721s 8s/step - loss: 0.0414 - accuracy: 0.9874 - val_loss: 0.0482 - val_accuracy: 0.9875
     Epoch 6/10
     87/87 [==============================] - 721s 8s/step - loss: 0.0370 - accuracy: 0.9889 - val_loss: 0.0316 - val_accuracy: 0.9887
     Epoch 7/10
     87/87 [==============================] - 696s 8s/step - loss: 0.0212 - accuracy: 0.9925 - val_loss: 0.0182 - val_accuracy: 0.9925
     Epoch 8/10
     87/87 [==============================] - 679s 8s/step - loss: 0.0241 - accuracy: 0.9921 - val_loss: 0.0229 - val_accuracy: 0.9900
     Epoch 9/10
     87/87 [==============================] - 727s 8s/step - loss: 0.0152 - accuracy: 0.9957 - val_loss: 0.0235 - val_accuracy: 0.9937
     Epoch 10/10
     87/87 [==============================] - 693s 8s/step - loss: 0.0084 - accuracy: 0.9975 - val_loss: 0.0103 - val_accuracy: 0.9950
```

IV. PERFORMANCE MEASURES

As indicated in figures 5 and 6 show accuracy and loss speeds that are faster than predicted by training. To apply open and closed eye images, we employed the upgraded database for training and the original database for testing.

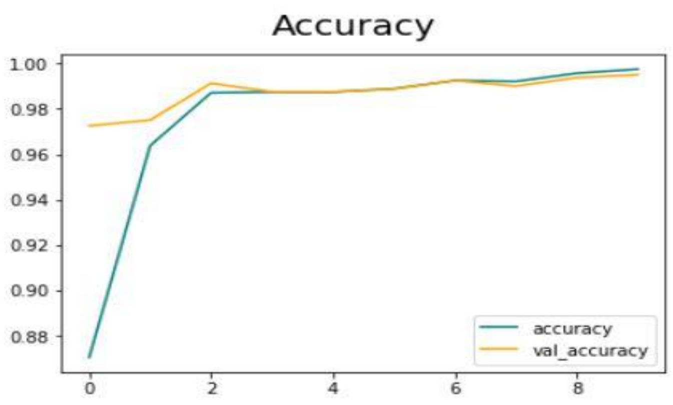


Fig. 5. Model accuracy of Enhanced CNN

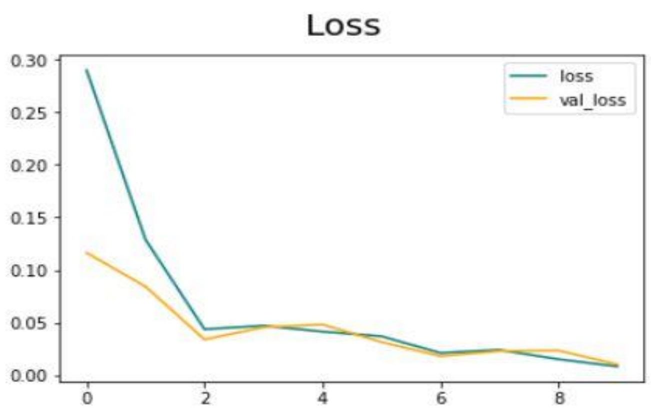


Fig. 6. Model loss of Enhanced CNN

Table 1. shows that the number of images closing the eye is least than the number of images opening the eye.

JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 2, 2022, p. 2006 - 2012
https://publishoa.com
ISSN: 1309-3452

TABLE 1. DATABASES - ORIGINAL AND AUGMENTED IMAGES

Image Type	Original Dataset	Expanded dataset
Eye opened	4337	17675
Eye closed	3765	15340

V. Result

Static photographs of the eye in various illumination settings, varied pupil directions, and eye attributes make up our dataset, which is utilized to evaluate the model. In the training process, the eye is divided into two groups (open and closed, regardless it is the right or left eye).

Because earlier work in this sector only used train and validation data, we only used train and validation data in our dataset to compare our networks to other research. 70% of the data was used for training, 20% for validation, and 10% for testing. The network's performance on our dataset, as shown in Table 3, indicates that our model has an image accuracy of 0.998. This value indicates that the chosen convolutional neural network method and architecture can adequately satisfy the accuracy parameter. The True Positive (TP), False Positive (FP), True Negative (TN), False Negative (FN), recall, and precision of our model are also presented in Table 3 to provide a thorough assessment of this network. As can be seen, our model has a precision of 0.998 and a recall of 0.997.

TABLE 2. Training results of Drowsiness.

Accuracy	Precision	Recall	TrueNegatives	FalseNegatives	TruePositives	FalsePositives
0.998	**0.998**	**0.997**	**0.992**	**0.008**	**0.71**	**0.29**

To summarize, the implementation of real-time performance in Adaptable real-time systems is required. We will put a camera in front of the driver that checks a live video feed for faces, and if one is found, we will turn off the facial contour detection and only take the eye field. To see if the eyes are closed, we can utilize the eye field to calculate the aspect ratio of the eye. The images in figure 7 were captured in real-time from the frame video of the training datasets.

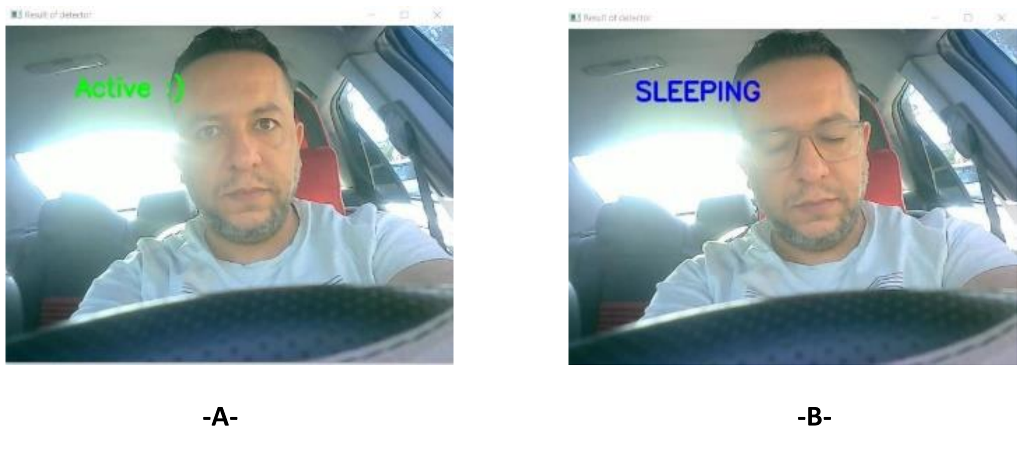


Fig. 7. A- Active state, B- Drowsy state, in a real-time prediction system.

VI. CONCLUSION

A model for detecting drowsiness is based on a CNN architecture that is designed to detect tiredness based on eye closure. The process began with the preparation of image collections for both open and closed eyes. The custom-designed CNN training uses 70% of the data set, while the rest is used for testing and validation reasons. The information video is first converted into frames, and then the face and eyes are detected in each frame. The improved CNN provided us with an automated and effective learned characteristic that helps us categorize eye-opening and closure. An alarm is activated to inform the driver if the driver's eyes close for 15 frames in a row.

The proposed CNN has a 99 percent training accuracy. Extra facial attributes could be introduced in future works to improve detection accuracy. We may also integrate the retrieved facial features with vehicle driving pattern data obtained from On-Board Diagnostics sensors.

REFERENCES

[1] M. Hashemi, A. Mirrashid, and A. J. S. C. S. Beheshti Shirazi, "Driver safety development: Real-time driver drowsiness detection system based on convolutional neural network," vol. 1, no. 5, pp. 1-10, 2020.

[2] L. Celona, L. Mammana, S. Bianco, and R. Schettini, "A multi-task CNN framework for driver face monitoring," in *2018 IEEE 8th International Conference on Consumer Electronics-Berlin (ICCE-Berlin)*, 2018, pp. 1-4: IEEE.

[3] T. U. Ahmed, S. Hossain, M. S. Hossain, R. ul Islam, and K. Andersson, "Facial expression recognition using convolutional neural network with data augmentation," in *2019 Joint 8th International Conference on Informatics, Electronics & Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*, 2019, pp. 336-341: IEEE.

[4] U. Sinha, K. K. Mehta, and A. J. I. J. o. C. A. Shrivastava, "Real Time Implementation for Monitoring Drowsiness Condition of a Train Driver using Brain Wave Sensor," vol. 139, no. 9, pp. 25-30, 2016.

[5] X. Ma, L.-P. Chau, and K.-H. Yap, "Depth video-based two-stream convolutional neural networks for driver fatigue detection," in *2017 International Conference on Orange Technologies (ICOT)*, 2017, pp. 155-158: IEEE.

[6] S. Park, F. Pan, S. Kang, and C. D. Yoo, "Driver drowsiness detection system based on feature representation learning using various deep networks," in *Asian Conference on Computer Vision*, 2016, pp. 154-164: Springer.

[7] W. Zhang and J. Su, "Driver yawning detection based on long short term memory networks," in *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2017, pp. 1-5: IEEE.

[8] D. Tran, E. Tadesse, W. Sheng, Y. Sun, M. Liu, and S. Zhang, "A driver assistance framework based on driver drowsiness detection," in *2016 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, 2016, pp. 173-178: IEEE.