DRIVER MONITORING (DROWSINESS DETECTION SYSTEM)

**ALABI EMMANUEL OLASUBOMI**

**YANGO MAXWELL MATHIAS**

**ADELODUN EMMANUEL ADELERE**

**AKINMADE OLUWASINAAYOMI SAMUEL**

**SOFTWARE ENGINEERING**

**BABCOCK UNIVERSITY**

**ILISHAN-REMO OGUN STATE**  
[**alabiemmanuel177@gmail.com**](mailto:alabiemmanuel177@gmail.com)

[**maxwellyaten@gmail.com**](mailto:maxwellyaten@gmail.com)

***Abstract*—Overall, AI has been a powerful tool for improving transportation and urban development by helping to optimize the use of resources, reduce congestion and accidents, and promote economic growth and development.**

**The major focus of this study is the Transportation safety problems that results from falling asleep while driving. Although artificial intelligence has been a major influence on the growth of transportation and urban development with several inventions and creatives to ease the use and reduce the risks involved in transportation. However, both highly developed and emerging nations are simultaneously dealing with more severe transportation management issues and investing a lot of resources and time into finding solutions. Under monotonous driving conditions, falling asleep at the wheel causes a sizable fraction of vehicle accidents. Numerous of these accidents include workers, such as lorry, cargo, and corporate car drivers. Circadian effects of the time of day are significant, with tiredness being most noticeable while working the night shift and while driving home afterward. Circadian factors matter just as much as driving time in determining driver drowsiness, although only drive time is taken into account by laws protecting professional drivers. In the middle of the day, older drivers are particularly susceptible to being sleepy. There is discussion of potential pathological causes of driver fatigue, however there is no proof that this issue has a significant impact on accident statistics. Sleep does not happen suddenly and without cause. Drivers who fall asleep while operating a vehicle are unlikely to remember doing so, but they will be aware of the precursory condition of growing drowsiness, likely reaching a state of resisting sleep before an accident.**

***Keywords—transportation, formatting, circadian, pathological, monotonous (****key words****)***

# Introduction

Transportation and urban development are closely related because the way that people and goods move around a city or region can have a big impact on its development and growth. Transportation infrastructure, such as roads, highways, and public transit systems, can make it easier for people to access job opportunities, schools, and other amenities, which can help to promote economic development and improve the quality of life for residents. At the same time, the way that cities and towns are designed and built can also have an impact on transportation.

There are many ways that artificial intelligence (AI) has been used to improve transportation and urban development. Some potential applications include:

* Traffic management: AI can be used to analyze traffic data in real-time and help to optimize the flow of traffic on roads and highways. This can include routing vehicles to avoid congested areas, adjusting traffic signals to improve efficiency, and even predicting traffic patterns and adjusting infrastructure accordingly.
* Public transit planning: AI can be used to analyze data on passenger demand, travel patterns, and other factors to help plan and optimize public transit systems. This can include everything from routing buses and trains to designing new transit lines and stations.
* Transportation safety: AI has been deployed to help improve safety on roads, highways, and other transportation systems. For example, AI-powered systems can be used to monitor traffic patterns and alert drivers to potential collisions or other hazards.
* Urban planning: AI is used to analyze data on land use, population density, and other factors to help cities and towns plan for future growth and development. This can include everything from designing new neighborhoods to identifying areas in need of infrastructure improvements.

Falling asleep while driving can be extremely dangerous, as it can lead to accidents that result in serious injury or death. Some of the problems associated with falling asleep while driving include: Reduced reaction time, decreased awareness, Loss of control, Drowsy driving can be just as dangerous as drunk driving: Studies have shown that being awake for 18 hours or more can have a similar effect on driving ability as having a blood alcohol concentration (BAC) of 0.05%. This is because sleep deprivation can cause impairment in cognitive function, reaction time, and decision-making, which are all critical for safe driving. Falling asleep while driving is a serious problem that can have serious consequences. It is important for drivers to be aware of the risks and to take steps to prevent it from happening, such as getting enough sleep, taking breaks on long drives, and avoiding driving while tired.

A major factor in preventing drivers from nodding off behind the wheel will be the installation of automatic sleepiness detectors in vehicles. For this reason, this study.

# Review Of Related Works

Driver drowsiness detection is an important aspect of roadway safety, as drowsy driving can lead to serious accidents and injuries. In recent years, there has been a significant amount of research on developing systems for detecting driver drowsiness. In this review, we will discuss some of the most relevant works in this area.

One early approach to driver drowsiness detection was based on the use of physiological signals, such as electroencephalography (EEG) and electrooculography (EOG). For example, a study by Kim et al. (2007) used EEG and EOG to detect drowsy driving based on changes in brain activity and eye movements. Another study by Kim et al. (2009) used similar methods to develop a drowsy driving detection system that could be used in real-time.

More recently, researchers have turned to computer vision-based approaches for detecting driver drowsiness. One example is the work by Chen et al. (2017), who used a convolutional neural network (CNN) to classify drowsy and non-drowsy driving based on facial images. Another study by Lu et al. (2019) used a combination of CNNs and long short-term memory (LSTM) networks to detect drowsy driving from video sequences.

There have also been several efforts to develop drowsy driving detection systems using wearable devices. For instance, a study by Lee et al. (2016) used a smartwatch to measure physiological signals such as heart rate and skin conductance to detect drowsy driving. Similarly, a study by Kim et al. (2018) used a wearable device to measure eye closure duration and blink rate as indicators of drowsy driving.

In summary, there has been a wealth of research on developing systems for detecting driver drowsiness, and the approaches used have ranged from physiological signals to computer vision and wearable devices. While each of these approaches has its own strengths and limitations, it is clear that there is still much work to be done to develop accurate and reliable drowsy driving detection systems.

# Discussion/Implementation

There are several potential approaches to implementing a driver drowsiness detection system, each with its own advantages and disadvantages.

One approach is to use physiological signals, such as EEG or EOG, to detect changes in brain activity and eye movements that may indicate drowsiness. These methods have the advantage of being relatively non-intrusive, as they can be measured using sensors that are either placed on the skin or worn as a headset. However, these methods may not be feasible for use in a commercial vehicle setting, as they may require specialized equipment and trained personnel to interpret the signals.

Another approach is to use computer vision-based methods, such as CNNs or LSTMs, to analyze images or video of the driver to detect signs of drowsiness. These methods have the advantage of being able to operate in real-time, and they do not require any special equipment or training to use. However, they may be less accurate than physiological signal-based methods, and they may raise concerns about privacy if they involve the use of cameras in the vehicle.

Wearable devices, such as smartwatches or fitness trackers, could also be used to measure physiological signals that may indicate drowsy driving. These devices have the advantage of being portable and easy to use, and they may be more practical for use in a commercial vehicle setting than specialized equipment. However, they may not be as accurate as other methods, and they may require the driver to wear the device consistently in order to be effective.

Ultimately, the most effective driver drowsiness detection system will likely involve a combination of these approaches, and will be tailored to the specific needs and constraints of the application.In this Python project, we'll use OpenCV to collect webcam photos and feed them into a Deep Learning model that will identify whether a person's eyes are "Open" or "Closed" based on their position. For this Python project, the strategy we'll employ is as follows:

* Step 1 – Take image as input from a camera.
* Step 2 – Create a Region of Interest in the image and find the face
* Step 3 – Detect the eyes from ROI and feed it to the classifier.
* Step 4 – The classifier will classify whether the eyes are open or closed.
* Step 5 – Calculate the result to see if the subject is drowsy.

Driver Drowsiness Detection Dataset

We got a script online from data-flair.training that records the eyes from a camera and saves them to our local disk in order to construct the dataset. They divided them into their corresponding categories, "Open" or "Closed." The undesired photos that were not required for creating the model were manually removed from the data. About 7000 photos of people's eyes in various lighting environments make up the data. We have attached the final weights and model architecture file "models/cnnCat2.h5" after the model was trained using our dataset.

**The Model Architecture**

Convolutional neural networks developed using Keras were utilized to create the model that we employed (CNN). Convolutional neural networks are a specific variety of deep neural networks that excel at classifying images. In essence, a CNN is made up of three layers: an input layer, an output layer, and a hidden layer with potential for more layers. These layers are put through a convolution operation with a filter that multiplies their 2D matrices together.

The following layers make up the architecture of the CNN model:

32 node convolutional layer with a three-kernel size 64 node convolutional layer with a three kernel size

128 nodes; fully connected layer

The last layer has two nodes and is also completely connected. Except for the output layer, where we employed SoftMax, all the layers use a Relu activation function.

**Project Prerequisites**

A webcam is necessary for this Python project because we'll be using it to take pictures. You can install the required packages using pip once Python has been installed on your system (the 3.6 version is recommended).

pip install OpenCV-python for OpenCV (face and eye detection).

Install TensorFlow using the pip command (Keras uses TensorFlow as backend).

Install Keras with pip (to build our classification model).

Install Pygame using pip (to play alarm sound).

Let's now examine our algorithm's operation step by step. The contents of the zip are:

Project File Structure -

The xml files required to identify objects in the image can be found in the "haar cascade files" folder. In this instance, we are identifying the face and eyes of the person.

The models folder contains our model file "cnnCat2.h5", which was trained using convolutional neural networks.

When a person is getting sleepy, we play the audio file "alarm.wav."

The "Model.py" file contains the program that we used to train our classification model using our dataset. A convolutional neural network implementation was present in this file. The primary file for our project is called "Drowsiness detection.py." We must execute this file in order to begin the detecting process.

Let's now examine our algorithm's operation step by step.

* Step 1 – Retrieve Image from a camera as Input

We will enter photographs using a webcam. Therefore, we created an infinite loop that will record every frame in order to access the webcam. We employ the OpenCV-provided cv2 technique. To use the camera and set the capture object, use VideoCapture(0) (cap). Each frame will be read by cap.read(), and the image will be saved in a frame variable.

* Step 2 – Create a region of interest and identify faces in the image (ROI)

Since the OpenCV algorithm for object detection accepts grayscale images as input, we must first convert the image to grayscale in order to identify the face in it. To identify the items, color information is not required. To find faces, we'll use the haar cascade classifier. Our classifier face is configured using the formula cv2.CascadeClassifier("path to our haar cascade xml file"). The detection is then carried out using faces = face. detectMultiScale(gray). It gives back a list of detections with the x, y, height, and width of the object's boundary box. The faces can now be iterated over, and each face's border boxes can be drawn.

for (x, y, w, h) in faces:

cv2.rectangle(frame, (x, y), (x + w, y + h), (100,100,100), 1 )

* Step 3 –Identify the eyes using ROI and provide the information to the classifier.

The method for detecting eyes is the same as that for detecting faces. We first put the cascade classifier for the eyes in the leye and reye, respectively, and then use left eye = leye to find the eyes. detectMultiScale(gray). We now need to isolate the eyeballs' data from the entire image. This may be done by removing the eye's boundary box, after which we can use this code to extract the eye's picture from the frame.

frame[y: y+h, x: x+w], l\_eye

Only the eye's image data is contained in l\_eye. This information will be used by our CNN classifier to determine whether the eyes are open or closed. The right eye will also be extracted and stored in r\_eye.

* Step 4 – Whether the eyes are open or closed will be classified by the classifier.

For forecasting the eye state, we are utilizing the CNN classifier. We must carry out specific actions in order to input our image into the model because it requires the proper starting dimensions. First, we use r eye = cv2.cvtColor(r eye, cv2.COLOR BGR2GRAY) to convert the color image to grayscale. The image is then scaled down to 24 \* 24 pixels because our model was trained on 24 \* 24 pixel images using the function cv2.resize(r eye, (24,24)). To improve convergence, we standardize our data. r eye equals r eye/255 (All values will range from 0 to 1) Boost the dimensions to add to our classifier's input. We used model = load model('models/cnnCat2.h5') to load our model. We now use our model, lpred = model.predict classes(l eye), to predict each eye. If lpred[0] = 1, it indicates that the eyes are open; if lpred[0] = 0, it indicates that the eyes are closed.

* Step 5 – Calculate Score to Check whether Person is Drowsy

The score is essentially a number that will be used to calculate how long the subject has kept his eyes closed. Therefore, if both eyes are closed, the score will keep rising, whereas an open eye causes the score to fall. Using the cv2.putText() function, we are drawing the outcome on the screen to show the person's status in real time.

Frame, "Open", (10, height-20), font, 1, (255,255,255), 1, cv2.LINE AA, cv2.putText(frame),

A threshold is established, for instance, if the score exceeds 15, it indicates that the subject has been staring into space for a considerable amount of time. At this point, the alarm will sound. play()

# Limitations

There are several limitations to a driver drowsiness detection system implemented in Python. These include:

* False positives: The system may mistakenly detect drowsiness when the driver is actually alert, leading to unnecessary alerts and distractions.
* False negatives: The system may fail to detect drowsiness when the driver is actually fatigued, which could lead to dangerous situations on the road.
* Limited accuracy: The accuracy of the system will depend on the quality of the data used to train it, as well as the complexity of the algorithms used.
* Limited context: The system may not be able to take into account other factors that could affect the driver's alertness, such as medications or underlying medical conditions.
* Ethical concerns: There may be concerns about the privacy implications of continuously monitoring a person's level of alertness.

##### Recommendation

There are several ways to improve the performance of a driver drowsiness detection system:

Increase the number of training examples: The more data the system has to learn from, the better it will perform.

Use a larger and more powerful machine learning model: A more powerful model with more parameters will be able to learn more complex patterns in the data and may perform better.

Pre-process the data to remove noise and improve the signal-to-noise ratio: Noise in the data can make it harder for the model to learn and may lead to poor performance.

Use a more robust feature set: A better feature set that more accurately represents the characteristics of drowsy drivers may lead to improved performance.

Use a different machine learning algorithm: Some algorithms are better suited to certain types of problems than others. Experimenting with different algorithms may lead to better performance.

Use a combination of these approaches: Combining multiple approaches may lead to the best performance.

##### Acknowledgment

##### First and foremost, we praise and thank God, the Almighty, who has granted countless blessing, knowledge, and opportunity to the members of this project, so that we have been finally able to accomplish this seminar.

##### We also express our deepest appreciation to our lecturer in person of Dr Ernest for providing us with the necessary knowledge and tools required for this seminar.

##### 

##### References

Elena Magán.; M. Paz Sesmero.; Juan Manuel Alonso-Weber.; Araceli Sanchis. Driver

Drowsiness Detection by Applying Deep Learning Techniques to Sequences of Images, Basel,

[2] Adhinata, F.D.; Rakhmadani, D.P.; Wijayanto, D. Fatigue Detection on Face Image Using FaceNet Algorithm and K-Nearest Neighbor Classififier. J. Inf. Syst. Eng. Bus. Intell. 2021, 7, 22–30.

[3] Ma, X.; Chau, L.P.; Yap, K.H. Depth video-based two-stream convolutional neural networks for driver fatigue detection. In proceedings of the 2017 International Conference on Orange Technologies (ICOT), Singapore, 8–10 December 2017; pp. 155–158.

[4] Zadeh, L. Fuzzy logic. Computer 1988, 21, 83–93.

[5] Magán, E.; Ledezma, A.; Sesmero, P.; Sanchis, A. Fuzzy Alarm System based on Human-centered Approach. In proceedings of the 6th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS 2020), Prague, Czech Republic, 2–4 May 2020; pp. 448–455. [6] Azim, T.; Jaffar, M.A.; Mirza, A.M. Fully automated real time fatigue detection of drivers through Fuzzy Expert Systems. Appl.Soft Comput. 2014, 18, 25–38.

[7] Bergasa, L.; Nuevo, J.; Sotelo, M.A.; Barea, R.; Guillén, M. Real-time system for monitoring driver vigilance. IEEE Trans. Intell.Transp. Syst. 2006, 7, 63–77. [8] Ghoddoosian, R.; Galib, M.; Athitsos, V. A Realistic Dataset and Baseline Temporal Model for Early Drowsiness Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Singapore, 8–10 December 2019. [9] Tamanani, R.; Muresan, R.; Al-Dweik, A. Estimation of Driver Vigilance Status Using Real-Time Facial Expression and Deep Learning. IEEE Sens. Lett. 2021, 5, 14. [10] Abtahi, S.; Omidyeganeh, M.; Shirmohammadi, S.; Hariri, B. YawDD: Yawning Detection Dataset. IEEE DataPort 2020.

[11] Krajewski, J.; Sommer, D.; Trutschel, U.; Edwards, D.; Golz, M. Steering Wheel Behavior Based Estimation of Fatigue. In Proceedings of the Fifth International Driving Symposium on Human Factors in Driver Assessment, Training and Vehicle Design, Big Sky Resort, Big Sky, MT, USA, 22–25 June 2009; pp. 118–124.

[12] Friedrichs, F.; Yang, B. Drowsiness monitoring by steering and lane data based features under real driving conditions. In Proceedings of the 2010 18th European Signal Processing Conference, Aalborg, Denmark, 23–27 August 2010; pp. 209–213.

[13] McDonald, A.D.; Schwarz, C.; Lee, J.D.; Brown, T.L. Real-Time Detection of Drowsiness Related Lane Departures Using Steering Wheel Angle. Proc. Hum. Factors Ergon. Soc. Annu. Meet. 2012, 56, 2201–2205.

[14] Yang, J.H.; Mao, Z.H.; Tijerina, L.; Pilutti, T.; Coughlin, J.F.; Feron, E. Detection of Driver Fatigue Caused by Sleep Deprivation. IEEE Trans. Syst. Man Cybern.—Part A Syst. Hum. 2009, 39, 694–705.

[15] Babu, T., Ashwin, S., Naidu, M., Muthukumaaran, C., Ravi Raghavan, C. (2019). Sleep Detection and Alert System for Automobiles. In: Hiremath, S., Shanmugam, N., Bapu, B. (eds) Advances in Manufacturing Technology. Lecture Notes in Mechanical Engineering. Springer, Singapore. https://doi.org/10.1007/978-981-13-6374-0\_14

[16] Andonovski, G., Sipele, Ó., Iglesias, J.A., Sanchis, A., Lughofer, E.D., & Škrjanc, I. (2020). Detection of driver maneuvers using evolving fuzzy cloud-based system. 2020 IEEE Symposium Series on Computational Intelligence (SSCI), 700-706.

[17] Abbas, Q., & Alsheddy, A. (2020). Driver Fatigue Detection Systems Using Multi- Sensors, Smartphone, and Cloud-Based Computing Platforms: A Comparative Analysis. Sensors (Basel, Switzerland), 21.

[18] Kristen Hall-Geisler "How Anti-sleep Alarms Work" 4 February 2009.

HowStuffWorks.com. <https://electronics.howstuffworks.com/gadgets/automotive/anti- sleep-alarm.htm> 3 January 2023

[9] Adhinata, F.D.; Rakhmadani, D.P.; Wijayanto, D. Fatigue Detection on Face Image Using FaceNet Algorithm and K-Nearest Neighbor Classififier. J. Inf. Syst. Eng. Bus. Intell. 2021, 7, 22–30.