Karadeniz Fen Bilimleri Dergisi, 15(1), 319-340, 2025. DOI: 10.31466/kfbd.1548673

KFBD

Karadeniz Fen Bilimleri Dergisi
The Black Sea Journal of Sciences
ISSN (Online): 2564-7377 https://dergipark.org.tr/tr/pub/kfbd

Araştırma Makalesi / Research Article

An Intelligent Driver Tracking and Driving Analysis Reporting System

Ferhat AKALAN[1], Betül MUMCU[2], Erdinç ŞAHİN[3*]

Abstract

As the number of vehicles on the roads increases day by day, the number of traffic accidents also increases. According to Turkish Statistical Institute (TÜİK) 2023 data, 88.9% of these accidents are caused by driver errors. These mistakes of drivers usually occur because they are careless, tired, sleepy, or busy with a different task while driving. At this point, the need for the use of smart systems in vehicles has begun. Intelligent vehicle systems have been developed to warn drivers during these behaviors. These systems aim to provide the driver with a safer driving experience. Looking at the literature, studies have been done on this subject and embedded systems have been developed. The general purpose of these studies is to ensure safe driving by warning the driver while driving. The proposed intelligent driver tracking and driving analysis reporting system in this study is designed mostly for companies in the freight and passenger transportation sector. In order to prevent traffic accidents caused by driver error, the aim is to conduct a driving analysis of commercial vehicle drivers and report them to the transportation company, thus minimizing traffic accidents by choosing more careful drivers or ensuring that the drivers are more careful. The proposed system consists of two cameras placed inside the vehicle, facing with the driver and traffic, recording video throughout the drive. The video recording is uploaded to the driving analysis reporting software, two separate videos are passed through two separate models and the results are displayed in a meaningful format. While carrying out the project, two different models are created. While the first of these models detects the driver's behavior such as sleepiness, making a phone call, or smoking, the second analyzes whether the vehicle complies with the speed limits and traffic lights. These analyses are presented in the form of reports with a user-friendly interface. In this way, it is aimed to ensure a safer traffic flow by preventing accidents caused by drivers.

Keywords: Driver Behavior Tracking, Machine Learning, Driving Analysis.

Akıllı Sürücü Takip ve Sürüş Analizi Raporlama Sistemi

Öz

Her geçen gün yollardaki araç sayısının artmasıyla beraber trafik kazalarının sayısı da artmaktadır. Türkiye İstatistik Kurumu (TÜİK) 2023 verilerine göre, bu kazaların %88,9'u sürücü hatalarından kaynaklanmaktadır. Sürücülerin bu hataları genellikle sürüş anında dikkatsiz, yorgun, uykulu ya da farklı bir işle meşgul olmalarından dolayı kaynaklanmaktadır. Bu noktada araçlarda akıllı sistemlerin kullanılmasına ihtiyaç duyulmaya başlanmıştır. Sürücüleri bu davranışları sırasında onları uyaran akıllı araç sistemleri geliştirilmiştir. Bu sistemlerle sürücünün daha güvenli bir sürüş sağlaması hedeflenmiştir. Literatüre bakıldığında bu konuda çalışmalar yapılmış ve gömülü sistemler geliştirilmiştir. Bu çalışmaların genel amacı sürücüyü sürüş sırasında uyararak güvenli bir sürüş sağlamaktır. Bu çalışmada önerilen akıllı sürücü takip ve sürüş analizi raporlama sistemi daha çok yük ve yolcu taşımacılığı sektöründe bulunan şirketler için tasarlanmıştır. Sürücü hatasından kaynaklanan trafik kazalarının önüne geçebilmek için ticari araç sürücülerinin sürüş analizi yapılarak şirkete raporlanması ve bu sayede daha dikkatli sürücülerin tercih edilmesini ya da mevcut sürücülerin daha dikkatli olmasını sağlayarak trafik kazalarını en aza indirgenmesi hedeflenmektedir. Sistem araç içerisine yerleştirilen sürücüye ve trafiğe bakan sürüş boyunca video kaydını alan iki kameradan oluşmaktadır. Video kaydı sürüş analizi raporlama yazılımına yüklenerek iki ayrı video iki ayrı modelden geçirilerek sonuçları anlamlı bir formatta görüntülemektedir. Projeyi gerçekleştirirken iki farklı model oluşturulmuştur. Bu modellerden ilki sürücünün uykulu olma, telefonla ilgilenme, sigara içme gibi davranışlarını tespit ederken ikincisi aracın trafikte hız sınırlarına ve trafik ışıklarına uyup uymadığı konusunda analizler yapmaktadır. Bu analizler kullanıcı dostu bir ara yüz ile rapor şeklinde sunulmaktadır. Bu sayede sürücü nedenli kazaların önüne geçilerek daha güvenli bir trafik akışının sağlanması hedeflenmektedir.

Anahtar Kelimeler: Sürücü Davranış Takibi, Makine Öğrenmesi, Sürüş Analizi.

[1,2,3]Giresun Üniversitesi, Bilgisayar Mühendisliği, Giresun, Türkiye,
ferhatakalann@outlook.com betuullm01@gmail.com erdinc.sahin@giresun.edu.tr

[*]Sorumlu Yazar/Corresponding Author

Geliş/Received: 11.09.2024 **Kabul/Accepted:** 19.02.2025 **Yayın/Published:** 15.03.2025

Akalan, F., Mumcu, B., Şahin, E. (2025). An Intelligent Driver Tracking and Driving Analysis Reporting System. *Karadeniz Fen Bilimleri Dergisi*, 15(1), 319-340.

1. Introduction

Nowadays, cargo and passenger transportation on highways is of great importance. With the increasing number of vehicles on the roads daily, traffic safety is also becoming critical. The slightest mistake in traffic can cause loss of life and property.

According to Turkish Statistical Institute (TÜİK) data, 1.314.136 traffic accidents occurred on highways in 2023. Of these accidents, 235.071 resulted in death and 1.079.065 resulted in financial damage. As a result of these accidents, 350.855 people were injured and 6.548 people lost their lives (Table 1.1). According to the statistics, it is clear that the majority of traffic accidents are caused by driver errors. Analyzing the causes of traffic accidents involving death and injury, it was observed that 249.856 of the 281.054 accidents were caused by drivers. The fault rate of drivers in these traffic accidents is 88.9% (Table 1.2).

Table 1. Traffic Accident Statistics between the years of 2009-2023. (Turkish Statistical Institute (TÜİK), 2023, Number of vehicles registered, accident, persons killed and injured, 2009-2023)

Year	Number of vehicles registered to traffic	Total number of accidents	Number of accidents involving death or injury	Number of accidents involving material loss only	Number of persons killed			Number of persons injured
					Total	At accident scene	Accident follow-up(1)	
2009	14.316.700	1.053.346	111.121	942.225	4.324	4.324	-	201.380
2010	15.095.603	1.106.201	116.804	989.397	4.045	4.045	-	211.496
2011	16.089.528	1.228.928	131.845	1.097.083	3.835	3.835	-	238.074
2012	17.033.413	1.296.634	153.552	1.143.082	3.750	3.750	-	268.079
2013	17.939.447	1.207.354	161.306	1.046.048	3.685	3.685	-	274.829
2014	18.828.721	1.199.010	168.512	1.030.498	3.524	3.524	-	285.059
2015	19.994.472	1.313.359	183.011	1.130.348	7.530	3.831	3.699	304.421
2016	21.090.424	1.182.491	185.128	997.363	7.300	3.493	3.807	303.812
2017	22.218.945	1.202.716	182.669	1.020.047	7.427	3.534	3.893	300.383
2018	22.865.921	1.229.364	186.532	1.042.832	6.675	3.368	3.307	307.071
2019	23.156.975	1.168.144	174.896	993.248	5.473	2.524	2.949	283.234
2020	24.144.857	983.808	150.275	833.533	4.866	2.197	2.669	226.266
2021	25.249.119	1.186.353	187.963	998.390	5.362	2.421	2.941	274.615
2022	26.482.847	1.232.957	197.261	1.035.696	5.229	2.282	2.947	288.696
2023	28.740.492	1.314.136	235.071	1.079.065	6.548	2.984	3.564	350.855

TurkStat, Road Traffic Accident Statistics, 2023, (1) Includes the deaths within 30 days after the traffic accidents due to related accidents and their impacts on people who were injured and sent to health facilities. (-) denotes no information available

Table 2. Traffic Accident Fault Rates (Turkish Statistical Institute (TÜİK), 2023, Number of faults causing traffic accidents involving death or injury, 2009-2023)

Year	Total faults		Driver faults		Passenger fault		Pedestrian faults		Road faults		Vehicle faults	
	Number	(%)	Number	(%)	Number	(%)	Number	(%)	Number	(%)	Number	(%)
2009	155 982	100	139 758	89.6	640	0.4	14 181	9.1	958	0.6	445	0.3
2010	157 970	100	141 728	89.7	564	0.4	14 171	9.0	992	0.6	515	0.3
2011	174 605	100	157 494	90.2	677	0.4	14 860	8.5	1 044	0.6	530	0.3
2012	181 266	100	161 076	88.9	797	0.4	17 672	9.7	1 124	0.6	597	0.3
2013	183 030	100	162 327	88.7	774	0.4	16 458	9.0	1 913	1.0	1 558	0.9
2014	193 215	100	171 236	88.6	901	0.5	18 115	9.4	1 841	1.0	1 122	0.6
2015	210 498	100	187 980	89.3	915	0.4	18 522	8.8	1 916	0.9	1 165	0.6
2016	213 149	100	190 954	89.6	869	0.4	18 612	8.7	1 717	0.8	997	0.5
2017	213 325	100	191 717	89.9	782	0.4	18 095	8.5	1 619	0.7	1 112	0.5
2018	217 898	100	194 928	89.5	1 916	0.9	18 394	8.4	1 300	0.6	1 360	0.6
2019	204 538	100	180 042	88.0	2 572	1.3	16 726	8.2	1 045	0.5	4 153	2.0
2020	177 867	100	157 128	88.3	2 577	1.4	12 520	7.0	897	0.5	4 745	2.7
2021	224 418	100	195 382	87.1	3 941	1.8	18 398	8.2	936	0.4	5 761	2.6
2022	235 176	100	204 233	86.8	2 753	1.2	22 234	9.5	902	0.4	5 054	2.1
2023	281 054	100	249 856	88.9	1 754	0.6	25 355	9.0	940	0.3	3 149	1.1

In traffic, accidents occur due to the driver's careless behavior, failure to comply with traffic rules, and actions that endanger driving safety while driving. One of the reasons why drivers are so often at fault in traffic accidents is the distraction of the driver while driving. Distraction is the inability to concentrate on one action while mentally engaged in a different task. Examples of distracted driving behaviors include using a cell phone, arguing on the phone, being deep in thought, taking one's eyes off the road, sleepiness, being tired, eating, drinking, or smoking. Driver awareness can be increased by monitoring in-vehicle and out-of-vehicle driver behaviors and analyzing the data obtained as a result of this monitoring. In this way, traffic accidents can be significantly reduced. A system designed using image processing and deep learning methods can classify driver behaviors and track defective behaviors by processing these data. By using deep learning methods, a model can be created and the model can be trained to classify driver behavior. Passenger and cargo transportation companies will want to prefer drivers who have merit and obey the rules instead of drivers who misbehave. In this way, both financial and emotional damages can be prevented.

1.1. Literature Review

Advanced driver assistance systems (ADASs) help drivers recognize road hazards while driving. These systems include features such as automatic emergency braking, driver drowsiness display, lane-keeping support, blind spot monitoring, etc.

In the literature, systems similar to the proposed one in this study have been designed. The majority of these systems are based on driver fatigue detection. In 2014, Golgiyaz and his colleagues developed a video-based sleepy driver detection system (Golgiyaz et al., 2014).

In 2017, Aki developed a real-time drowsy driver warning system with J48 decision tree algorithm in his PhD study (Aki, 2017). In 2018, Vural et al. used Raspberry Pi 3 to detect driver fatigue with PERCLOS metric and built a system that warns with an alarm system and sends a message to the selected target (Vural et al., 2018). In 2020, Şahin et al. developed a driver fatigue detection and warning system with Raspberry Pi 4 (Şahin et al., 2020). Also in 2020, a system that classifies four different situations using the driver's eye and mouth regions on the Nvidia Jetson Nano embedded device was developed (Çivik, 2020). In 2021, a real-time ADAS prototype was designed by Güney. He developed a system that detects traffic signs, pedestrians, and objects outside the vehicle and warns the driver by detecting fatigue and sleep by tracking phone and cigarette use and eye tracking inside the vehicle (Güney, 2021). In 2022, Emre Şafak and his colleagues developed a model with an accuracy rate of 95.65% for driver fatigue detection on mobile devices using Evolutionary Neural Networks (Şafak et al., 2022). Considering these studies, most researchers have focused on driver fatigue. According to the results of the research conducted by the authors, there is no study that creates a reporting system by considering both in-vehicle and out-of-vehicle data.

In recent years, deep learning-based image processing methods have been widely used to monitor driver behavior and improve traffic safety. In a study by Xie, a YOLOv8-based model for driver fatigue detection has been developed (Xie et al., 2023). In the study, a system that analyzes the facial features of the driver is designed and eye closure rate (PERCLOS) and mouth opening percentage (POM) indicators are used for fatigue detection. YawDD is used as the dataset and a new dataset is labeled through the Roboflow platform. The YOLOv8 model is trained with this dataset and achieved 96.9% accuracy in detecting facial features. The average processing time of the model is 11.4 ms/square and it is found to provide higher speed and accuracy compared to previous YOLO versions and traditional methods. This study makes an important contribution to traffic safety by showing that AI-based solutions can be effectively used in driver fatigue detection.

In another study, an improved method based on the YOLOv8 algorithm for detecting drivers' distracting behaviors and emotional states is presented (Ma et al., 2024). In the study, the detection accuracy of the model is improved by combining a multi-headed self-attention mechanism and

convolutional neural networks (CNN). The proposed method is trained on the FER2013 dataset and a specially collected dataset. The test results show that the proposed model provides high accuracy in real-time applications and can successfully classify driver behaviors. The model was optimized for the Jetson Nano platform with TensorRT and DeepStream methods, improving speed and efficiency. This research makes significant contributions to the integration of advanced image-processing techniques into driver safety systems.

1.2. Contributions of the study

The difference between the proposed system from the studies in the literature is its method and intended use. The differences and innovations of this system from existing systems are presented below.

- It creates a cost-effective system by using a video-based system instead of a real-time base.
- The designed system will be mostly used for the analysis of commercial vehicle drivers. By reporting the driving analysis in desired time periods, financial and emotional damages can be prevented by enabling companies to prefer meritorious drivers instead of drivers who misbehave.
- In addition to the existing systems, it is aimed to create a detailed driver driving profile in the reporting system.

2. Materials and Methods

2.1. System components

The Intelligent Driver Tracking and Driving Analysis Reporting System consists of two different stages. These are the software and hardware parts. After training the deep learning models, the software to which the model will be integrated is coded. Afterward, a hardware system is installed for this system. The main components of this hardware system are as follows.

Raspberry Pi-4: Raspberry Pi is a low-cost, credit card-sized computer. One of the hardware components, Raspberry Pi-4, is used to process images and speed data from the vehicle with two camera modules and an OBD (On-Board Diagnostics) device.

Webcam: The webcam is the component that receives the video recording that will be used to analyze the driver's driving. This camera is placed at the front of the vehicle facing the traffic.

Raspberry Pi infrared camera module: Raspberry Pi infrared camera module collects data inside the vehicle. This camera records video to detect dangerous behaviors of the driver while

driving.

OBD-II: OBD-II provides speed and other data from inside the vehicle. The speed data is used in the off-vehicle video to determine whether the detected speed limits are being respected.

Other components: Other components of the system include an SD card for using the Raspberry operating system and a Power bank for the Raspberry Pi 4's in-vehicle power connection.

2.2. Data Collection and Labeling

Since the existing datasets in this field are not suitable for the targeted study both in terms of the camera capture of the image data and the classes of the data, it is necessary to create a unique dataset. The original dataset consists of four different classes: smoking, cell phone use, eating and drinking, and sleepiness. The dataset is created in a single session with three distinct drivers. A camera is placed inside the vehicle in a position that allows a clear view of the driver, and video recordings are taken for an average of 6 minutes for each driver. During the recording, each driver repeated the movements of four different classes, and an average of 1.5 minutes of video samples are collected per driver for each class. On average, 4.5 minutes of recordings are obtained for each class. When the videos are converted into image datasets, they are divided every 20 frames to avoid too many of the same frames.

Table 3. Video Dataset: Recording Times per Class

	Smoking	Drinking-or-eating	Drowsiness	Using-phone	Total
Driver 1	1 min 05 sec	1 min 01 sec	1 min 02 sec	1 min 38 sec	4 min 46 sec
Driver 2	1 min 34 sec	1 min 40 sec	1 min 30 sec	2 min 05 sec	6 min 49 sec
Driver 3	1 min 03 sec	1 min 50 sec	1 min 03 sec	1 min 56 sec	5 min 52 sec
Total	3 min 42 sec	4 min 31 sec	3 min 35 sec	5 min 39 sec	17 min 27 sec

Table 4. Image Dataset: Number of Samples per Class

	Smoking	Drinking-or-eating	Drowsiness	Using-phone	Total
Driver 1	206	196	224	400	1026
Driver 2	240	237	250	250	977
Driver 3	255	249	276	250	1030
Total	701	682	750	900	3033

After removing unnecessary data, a total of 3033 images have been obtained. It is observed that the data collected for some classes is insufficient for model accuracy. To overcome this, extra samples are collected from two drivers for each class using the same technique, to make the total number of images 4000. These images have a resolution of 1280x720. The data are labeled in bounding boxes according to their class as shown in Figure 1.

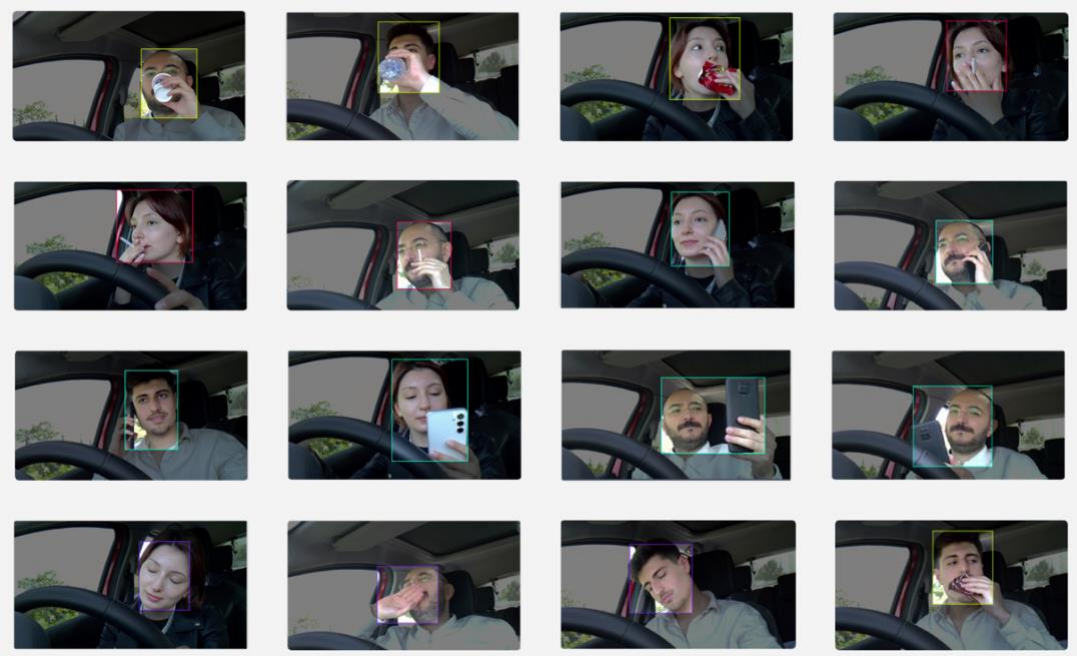

Figure 1. Labeled in-vehicle data

All of the data is acquired in daylight, so there are no examples from different lights and angles. To overcome this shortcoming, 9 different data augmentation techniques are used. The image data is augmented with techniques such as noise, grayscale, blur, brightness, rotation, and cropping from different lighting and angles to reach 9000 image data. Before data augmentation, the data set was divided into 70% training, 15% validation, and 15% test data. Data augmentation techniques are applied only to the training dataset. The off-vehicle data set is taken from the Roboflow platform. This dataset includes images of traffic speed limit signs, traffic lights, and stop signs. (SelfDriving Car, 2023).

2.3. Model Training with YOLO Algorithm

YOLO is an algorithm that performs object detection with convolutional neural networks. YOLO algorithm is much more efficient and faster than most algorithms. Considering the studies, it is seen that the YOLO algorithm is very successful in terms of detection and recognition speed. For this reason, the YOLO algorithm has been utilized in this project.

The YOLO (You Only Look One) algorithm was introduced in 2016 by Joseph Redmon. "You Only Look" means you only look once. The reason for this is that object detection can be done quickly and in one go. The YOLO algorithm works faster and more successfully than other algorithms used in real-time object detection. Algorithms such as R-CNN initially find possible locations where

objects can be found in the given image. The places it finds are called region suggestions. YOLO, on the other hand, does not make location predictions, it runs the received image through an artificial neural network at once processes it, and gives the location and class of the objects. (Redmon et al., 2016)

2.3.1 Operating Procedure of the YOLO Algorithm

The YOLO algorithm divides the received image into grids of size (AxA). Each grid is passed through a convolutional neural network. The grid is surrounded by a bounding box to detect the object inside the grid. As the grids are passed through the network, it is checked to see if there is an object inside the grid. If the center of the object is in the grid, it calculates the width, height, class, and confidence score of the detected object (Redmon et al., 2016). To explain the working principle of the YOLO algorithm clearly, the figures related to the algorithm output and prediction mechanism are given below.

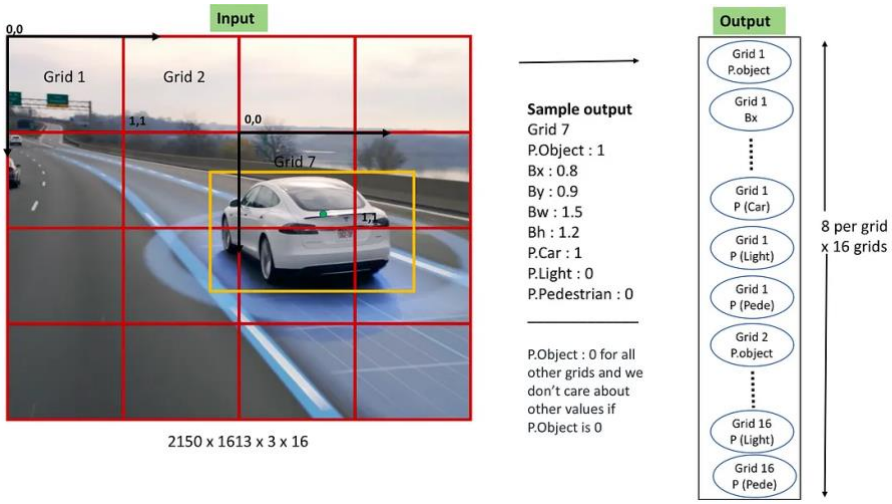

Figure 2. YOLO algorithm output (Grover, 2018)

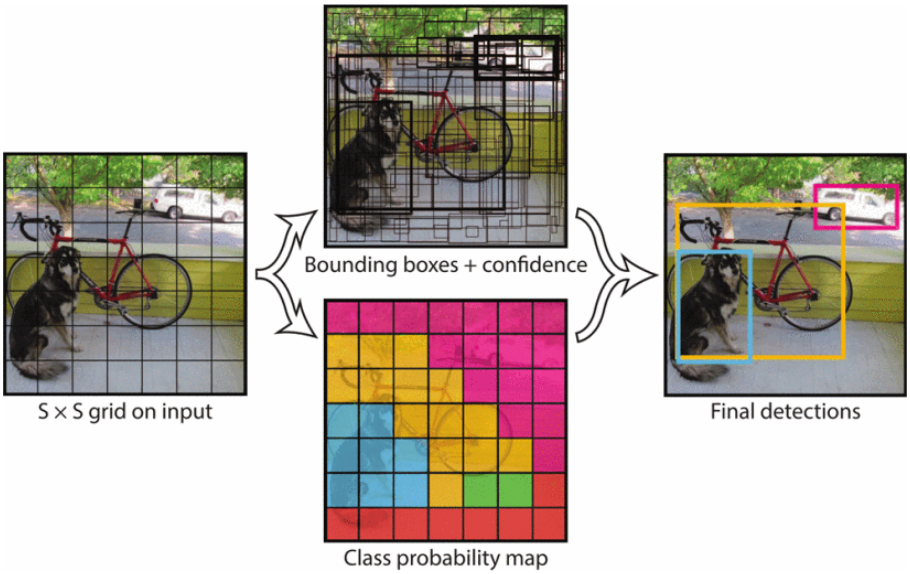

Figure 3. YOLO algorithm prediction mechanism (Redmon et al., 2016)

Bounding boxes with a confidence score below a certain value do not appear in the output. YOLO uses the Non-Max Suppression algorithm for this. The efficiency graphs of the algorithm are shown below.

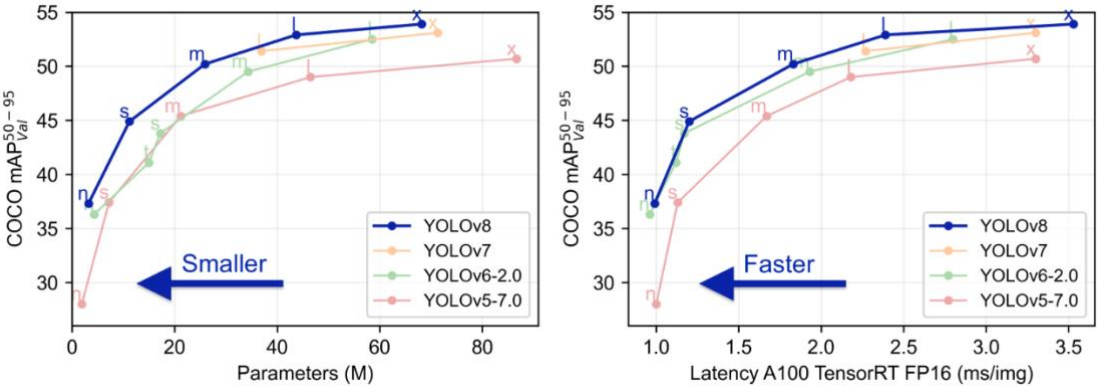

Figure 4. Graph showing the efficiency of the YOLOv8 Algorithm (Jocher et al, 2023)

2.3.2 Training Stage

For model training, YOLOv8 is preferred due to its high accuracy and performance advantages. The model is trained with the object detection model YOLOv8s of the YOLOv8 algorithm. The labeled datasets are used for training the models with the YOLOv8 algorithm in the Google Colab notebook environment. The Colab offers GPU (Graphics Processing Unit) and TPU (Tensor Processing Unit). The model is trained using the NVIDIA A100 GPU in the Google Colab environment. The A100 GPU has a VRAM capacity of 40 GB and enables fast training of deep

learning models. This speed up computations when working with large data sets and complex models.

During the model training process, the labeled data is divided into training, validation, and test sets. This separation is important to assess the generalization ability of the model. Then, preprocessing is applied to the data. The image data is resized and normalized according to the requirements of the model.

The YOLOv8 model is trained on training data with a given number of epoch values. The epoch value determines how many times the entire training data passes through the model. More epochs allow the model to learn more about the data, but too many epochs can lead to overfitting. Overfitting occurs when the model memorizes the training data and gives incorrect results on different data sets. When the in-vehicle model is trained with 100 epochs, overfitting is resulted. A more successful result is obtained when the model is retrained with an epoch value of 50. During training, object detection accuracy is improved by minimizing the loss function of the model, and the performance of the model is continuously monitored using a validation set. The hyperparameters are adjusted when it is deemed necessary. The default learning rate for YOLOv8 is 0.01 with a batch size of 16. For the other hyperparameters, the default values of YOLOv8 are used.

2.4. Software

2.4.1. Data collecting software

The camera and OBD software run on the Raspberry Pi-4 to collect data inside and outside the vehicle. This software performs tasks such as data collection and data storage through various components and modules. It collects video streams from two cameras inside and outside the vehicle, as well as speed and other driving data from the OBD device. The collected data is temporarily stored in a specific format to be used for analysis. Some technologies and languages are used to accomplish these tasks. These are Python, OpenCV, and OBD-Pi. Python is the software language used for the majority of image processing and data collection. OpenCV is used to take images from the second camera installed on the Raspberry Pi. OBD-Pi is the Python library used to get data from the OBD device.

The operating principle of the software is to first check the connections of the devices plugged into the Raspberry Pi, such as OBD, webcam, and pi-camera. Once the device connections are confirmed, recording can be started. Video and OBD data recording continue simultaneously. When the end recording button is clicked, it saves the recorded files to the desired file. The flow chart of the software is given in Figure 5.

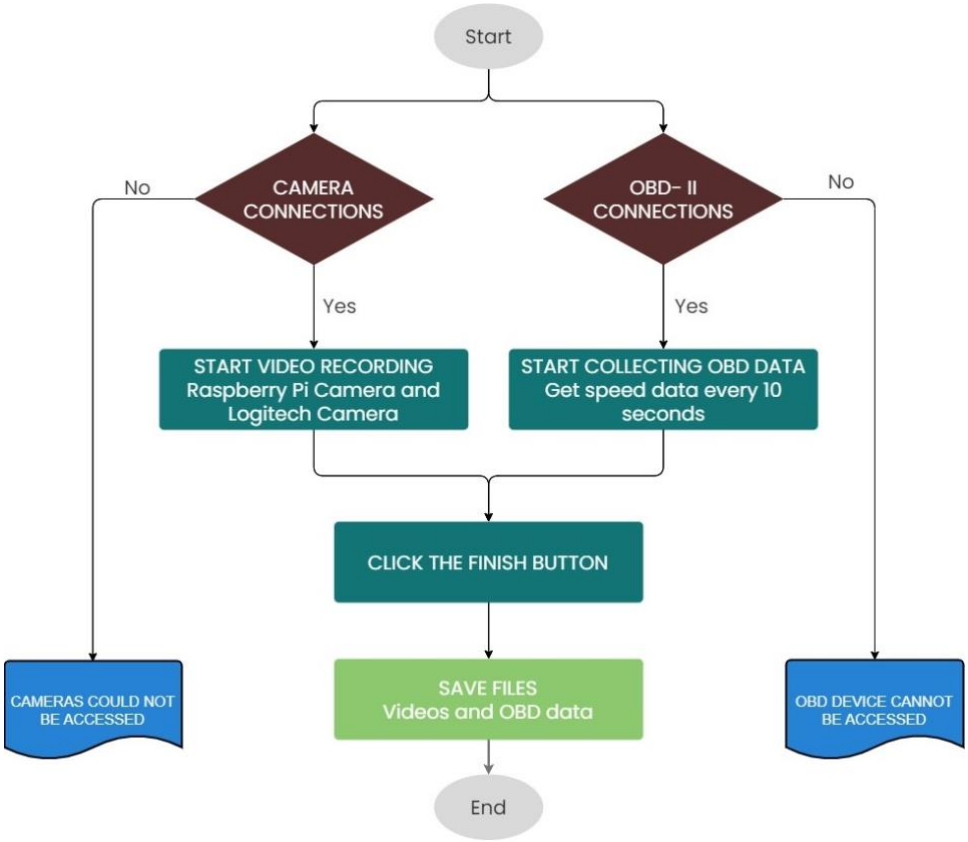

Figure 5. Flow chart of the data collection and OBD software.

2.4.2. Analysis and Reporting Software

The analysis and reporting software enables users to analyze driving and view reports through a web-based interface. This software offers a user-friendly interface and performs features such as data analysis, reporting, user and driver management. The flowchart of this software is shown in Figure 6.

The data collected by the software is analyzed using trained machine-learning models. These analyses include the detection of driver behavior (smoking, talking on the phone, eating, drinking, sleeping, etc.) and obeying traffic rules (speed limits, traffic lights). In addition, during the data upload phase, the speed data from the OBD device is uploaded to the system as a ".csv" file. Speed violation analysis is then performed with this speed data. If a speed sign is detected in the off-vehicle video, the speed value on the sign is compared with the speed data received from the OBD-II device. If the speed of the vehicle is higher than the value read on the signboard, a speed violation is reported. The data obtained after the analysis is saved in the database.

The results of the analysis are visualized with statistical graphs. These reports are used to evaluate driver performance and make necessary improvements. This makes it easier for the user to understand the data.

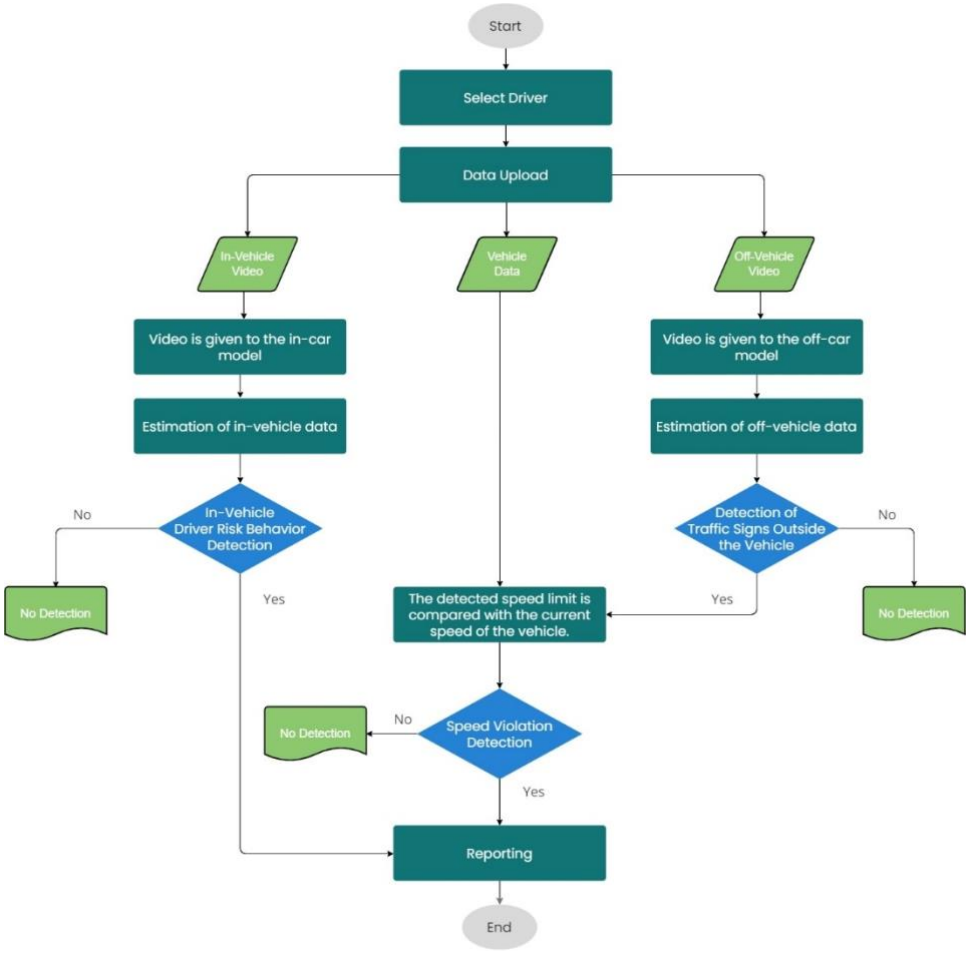

Figure 6. Flow chart for the reporting system.

2.5. Integration of Software and Hardware Equipment

Raspberry Pi is used as the central processing unit for data collection. All components used in data collection are connected to Raspberry Pi 4. There are two cameras for the proposed system. One is positioned to collect images from inside the vehicle and the other is positioned to record images from outside the vehicle. An OBD-II device is used to collect vehicle data. Since there is an incompatibility problem with the Raspberry Pi-4, the data collection process is carried out through the phone by connecting the OBD device to the phone. A power bank with a capacity of 10,000 mAh is used to power the Raspberry Pi-4. The image of the system is shown in Figure 7.

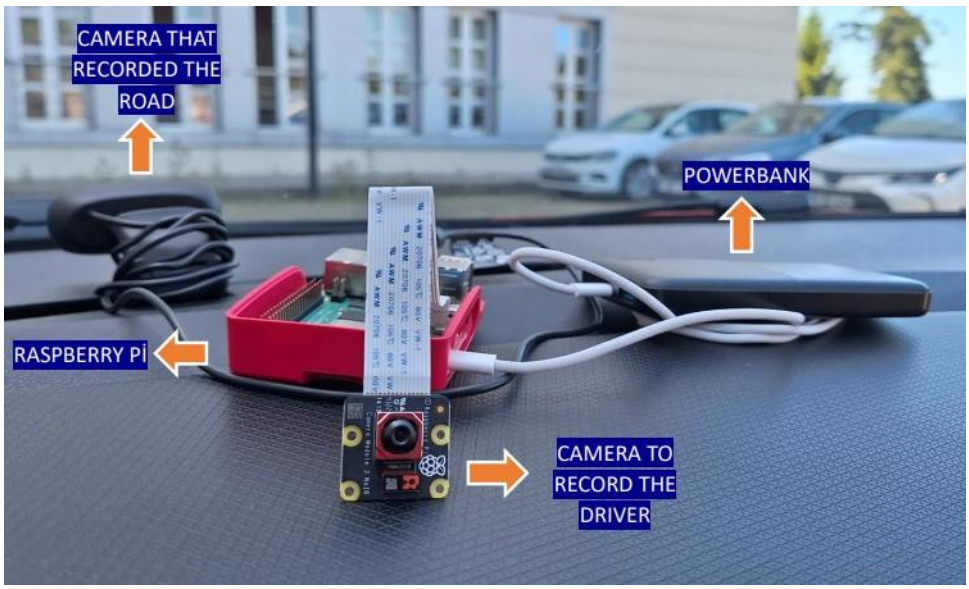

Figure 7. Data collection system

2.6. Implementation of Interface Elements

The realization of the interface elements in the proposed software is an important step in making the system user-friendly and functional. The interface of the system is designed in a modern and user-friendly structure using the "Django" web development framework (Django, 2024). The designed web page shown in Figure 8 is used to upload and analyze data, which is the main function of the system. It is used to analyze the collected data and generate reports. The desired confidence value for the models can be determined from the developed interface. Since long videos may take more time to be processed in the model, a feature that can determine the number of frame skips has been added.

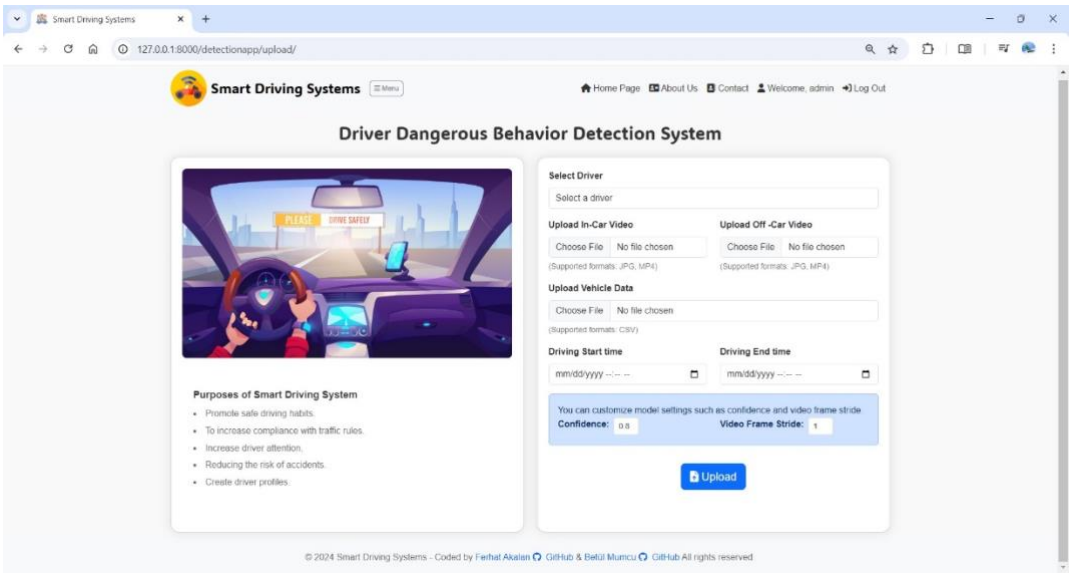

Figure 8. Designed web page for data upload and analysis.

Figure 9 represents the page where the data obtained from the models after the upload process is displayed in graphs and tables. On this web page, information such as in-vehicle driving behaviors and speed violations detected by our model are presented in detail.

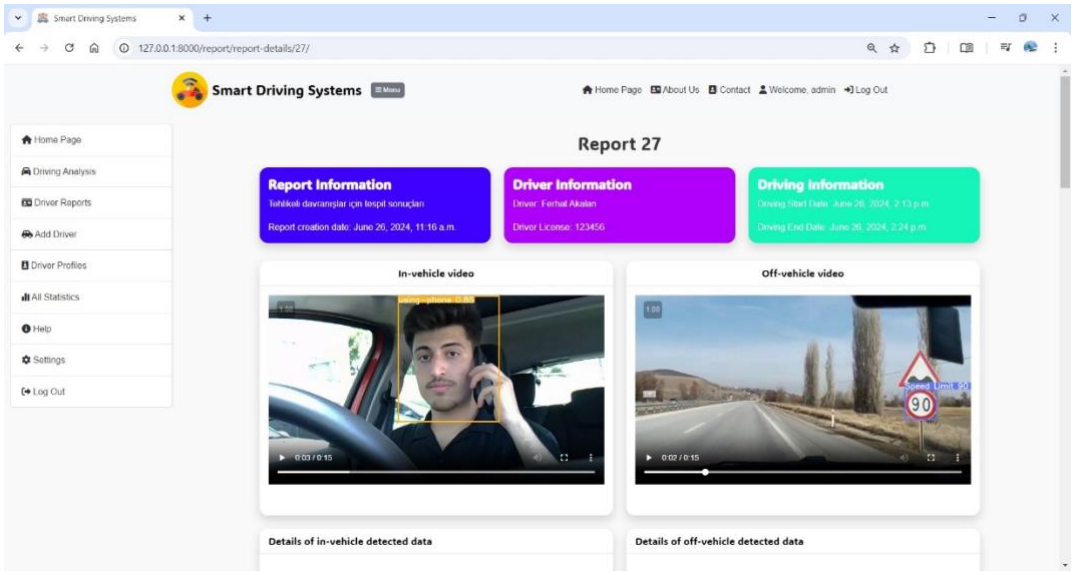

Figure 9. Web page for reporting.

The labels of the outputs obtained from the models as a result of the analysis are shown on the reporting page. A sample demonstration is shown in Figure 10.

In-vehicle labels

Label	Total
Drinking or eating	130
Drowsiness	4
Smoking	77
Using phone	113

Off-vehicle labels

Label	Total
Speed Limit 70	1
Speed Limit 90	2

Figure 10. A sample table of the detected labels.

The detected labels are visualized with graphs. Figure 11 shows the pie charts of the detected labels. Tags detected in off-vehicle tags are shown in the graphs after the speed violation is detected.

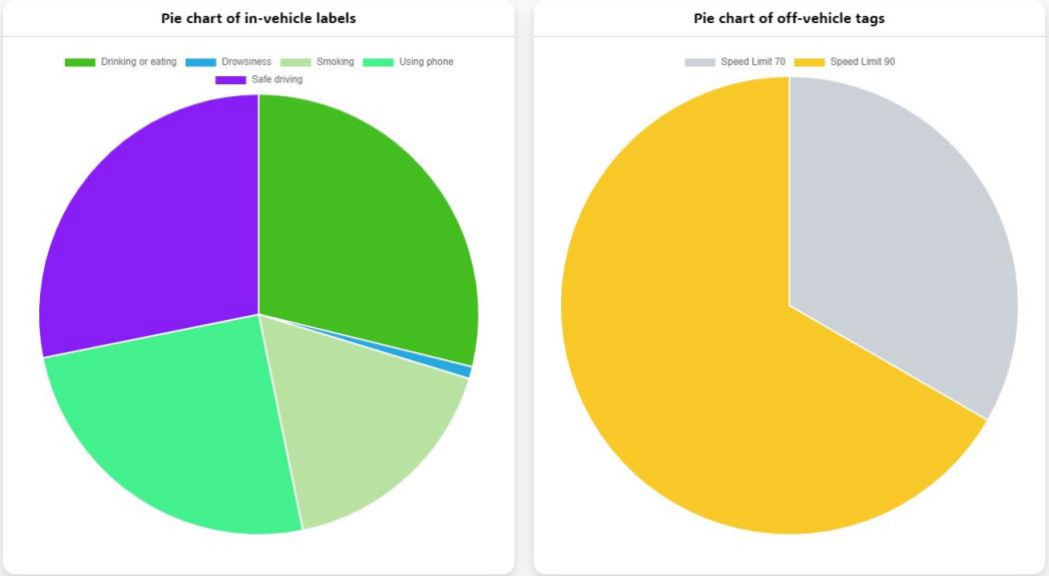

Figure 11. Pie charts representation of the labeled data.

The graph of speed violations shows the comparison between the speed read from the OBD-II device and the detected speed limits. A sample presentation is shown in Figure 12. The points shown in red on the graph indicate speed violations.

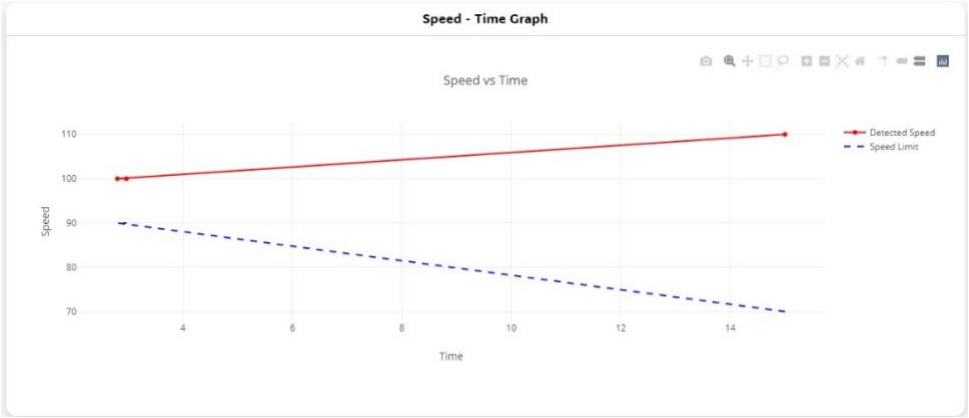

Figure 12. A sample of the speed violation graph.

3. Findings and Discussion

The performance of the trained models is evaluated on the test set and the results are analyzed. The results show the confidence values, complexity matrix, and loss and performance metrics of the test data. The confidence values of the out-of-vehicle model with the test data set are shown in Figure 13. In general, if we check for the predictions of the model on the test data, it is seen that the model gives reasonable results when evaluated with a confidence value of 0.8, although it gives incorrect values for “Green Light” and “Red Light”.

Figure 13. Confidence values for the test data of the off-vehicle model (SelfDriving Car, 2023)

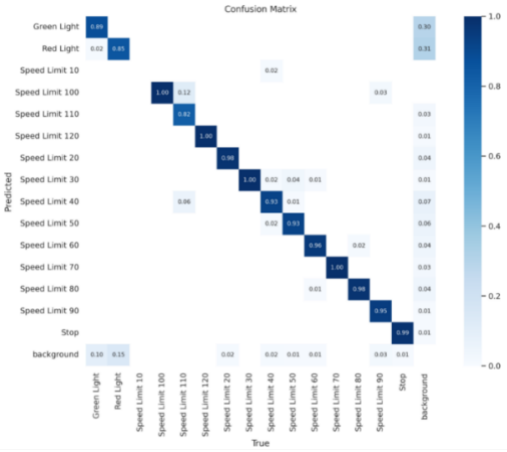

Figure 14. Out-of-vehicle confusion matrix

Looking at the confusion matrix in Figure 14, it can be observed that the model predicts “Green Light” with 89% accuracy, but incorrectly predicts as it is “Red Light” with 2% accuracy. Similarly, the model predicted “Red Light” with 85% accuracy. In general, the model predicted the speed limit classes with high accuracy. This shows that the model is able to distinguish these classes well.

The graphs of the loss and success metrics for the training process of the model are shown in Figure 15. The graphs show the loss functions and metrics based on the training and validation data. These graphs show how the model learns over time and how the accuracy rates increase.

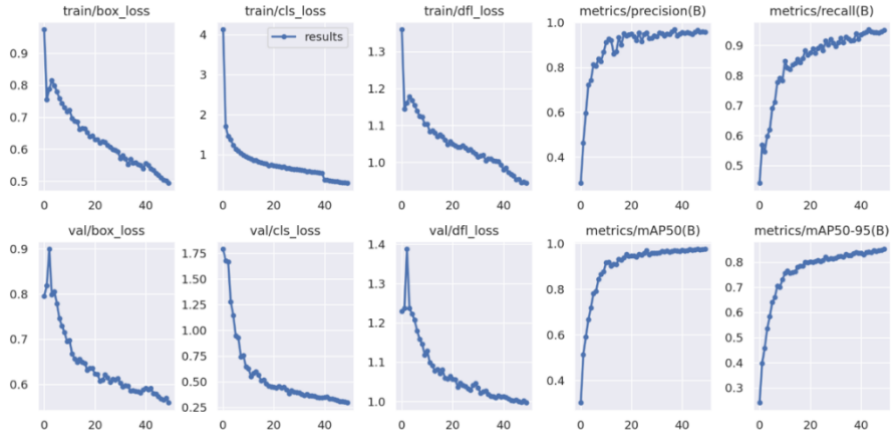

Figure 15. Out-of-vehicle loss and performance metrics.

The losses on both training and validation data decrease over time. This indicates that the model optimizes better and reduces its losses over time. Precision, recall, mAP50, and mAP50-95 metrics increase over time. This indicates that the model's accuracy and recall rates improve, and its overall performance improves. Training and validation losses decrease similarly, indicating that the model is not overfitting. The confidence values of the in-vehicle model calculated with the test data are given in Figure 16. Considering the test result of the in-vehicle model, it is seen that the rate of correctly recognizing the “drinking-or-eating” situations is 90%. Likewise, “drowsiness” is correctly predicted with a confidence value of 0.9.

Figure 16. In-vehicle confidence values.

In the complexity matrix seen from Figure 17, “smoking” and “using a phone” are 100% correctly predicted. However, in the cases of “drowsiness” and “drinking or eating,” there is a 1% misprediction rate.

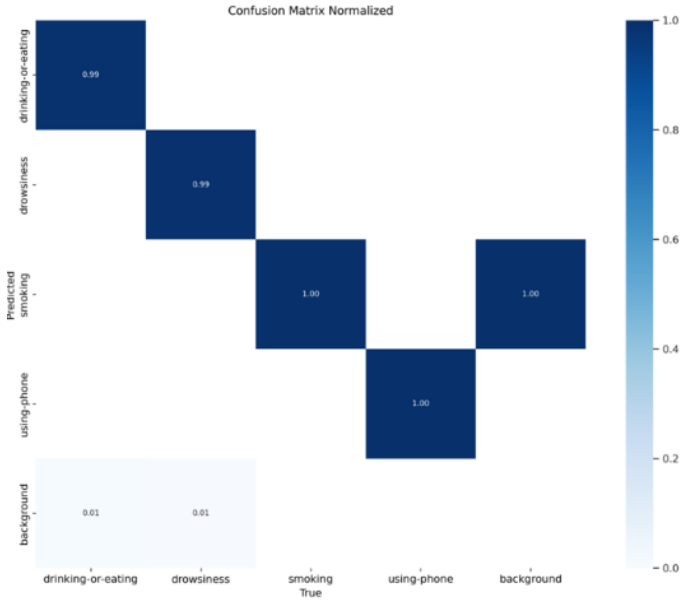

Figure 17. In-vehicle confusion matrix.

Considering the loss and performance metrics given in Figure 18, it can be seen that the values of the loss functions decrease over time, and the other metrics increase. Loss functions are a measure of the success of artificial neural network models in their predictions. If these function values decrease over time, it means that the model is successful in its predictions. In other words, the model has been successfully trained.

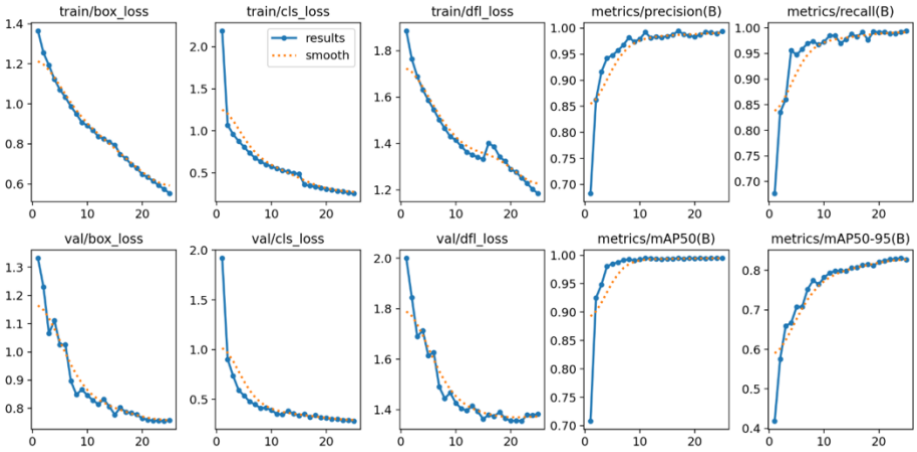

Figure 18. In-vehicle loss and performance metrics.

This study develops a system that can detect both in-vehicle and out-of-vehicle driver misconduct such as smoking, talking/texting on the phone, sleepiness, eating and drinking, speeding, and traffic light violations. Both the data collection software and hardware are tested. Two cameras in the system record driving while the OBD-II device simultaneously records the speed of the vehicle. These recorded data are uploaded to the reporting software by the user. These data are applied to the YOLOv8 models integrated into the software to detect the driver's in-vehicle and out-of-vehicle defective behaviors. These defective behaviors can be used to create a driving analysis and a driver profile can be created based on this data. In this way, fleet managers will be able to prefer drivers with merit by viewing the driver's driving profile.

4. Conclusions and Recommendations

As a result of the study presented in this paper, an intelligent driver monitoring and analysis system has been developed to detect and analyze dangerous drivers' behaviors. The system consists of two cameras and an OBD-II device integrated into a Raspberry Pi-4 board. Two cameras are used to obtain in-vehicle and out-of-vehicle video image data, while the OBD device is integrated to provide speed data. The data collected throughout the drive is evaluated through a web-based application with two different models trained to detect both in-vehicle and out-of-vehicle dangerous driver behaviors and a report is provided to the user.

In the system, two different deep learning models have been created for in-vehicle and out-of-vehicle situations using the YOLOv8 algorithm. A unique dataset has been created for the in-vehicle model, which can detect driver behaviors such as talking on the phone, being sleepy, smoking, eating, and drinking. The out-of-vehicle model can detect red light and speed limit signs. The tests showed that both models are able to make successful predictions with an 80% confidence level. When YOLOv8 and YOLOv11 models are compared, very similar results are obtained in terms of mAP50-95 values. Although YOLOv8 is a previously developed model, it performed competitively compared to YOLOv11. This shows that YOLOv8 can still be a strong alternative.

Table 5 Comparison of the model performance with other YOLO versions.

Algorithm	mAP50-95 All	mAP50-95 Drinking-or-eating	mAP50-95 Drowsiness	mAP50-95 Smoking	mAP50-95 Using-phone	Number of parameters
YOLOv8s	0.829	0.799	0.845	0.855	0.817	11M
YOLOv8m	0.827	0.802	0.839	0.850	0.815	25M
YOLOv11n	0.825	0.793	0.840	0.850	0.816	2.5M
YOLOv11s	0.824	0.796	0.836	0.846	0.819	9M

A web application has been developed to integrate the models and analyze the data, and algorithms are written using the Django framework to process and analyze the model outputs. In addition, speed violations can be successfully detected by simultaneously comparing the speed data obtained from the OBD device with the readings from the speed limit signs detected in the off-vehicle driving video recording. As a result, a reporting system that visualizes the data has been created, and an interface that allows users to analyze and create statistical graphs has been designed. Some technical problems have been encountered during the study. Due to the incompatibility between the OBD-II device and Raspberry Pi, a Bluetooth connection could not be achieved. Therefore, the OBD-II device is connected to a mobile device, and the data is recorded through a mobile application.

The in-vehicle data set has been created from a single angle and only in daylight. Although data augmentation techniques are applied to diversify the data from different lighting and angles, this method alone may not be sufficient. Additional data should be collected from different angles and under various lighting and weather conditions to create a broader and more balanced dataset. Similarly, the off-vehicle data set was insufficient in terms of different weather conditions and lighting. In order to increase data diversity, it is recommended to collect additional data at different time periods and under various weather and lighting conditions. In addition, a cooling fan is used to prevent the Raspberry Pi 4 hardware from overheating, but this solution may be insufficient for long-term use.

Diversifying the data sets for future studies can increase the generalization capability of the model. Real-time data processing can be added to provide instant feedback to drivers and thus increase the effectiveness of the overall system. An advanced user interface could be designed to provide fleet managers with more detailed and user-friendly reports. Furthermore, driver analysis can be made more comprehensive with additional parameters such as fuel consumption, not just limited to existing assessments. The integration of advanced sensors such as Lidar can make the data collection process more precise and comprehensive. In order to develop a low-cost system, Raspberry Pi is employed and the system is designed to be video-based. However, in future studies, more powerful hardware options such as NVIDIA Jetson Nano may be preferred instead of Raspberry Pi to provide better performance and real-time processing support.

Acknowledgments

This study has been entitled to receive support with the application number 1919B012305483 within the scope of the 2209-A University Students Research Projects Support Program 2023 by the Scientific and Technological Research Council of Türkiye (TÜBİTAK). We would like to thank TÜBİTAK for their valuable support.

Authors' Contributions

This study is realized by Ferhat AKALAN and Betül MUMCU under the supervision of Assoc. Prof. Dr. Erdinç ŞAHİN from the undergraduate graduation thesis of Giresun University, Department of Computer Engineering. All authors contributed equally to the work.

Statement of Conflicts of Interest

There is no conflict of interest between the authors.

Statement of Research and Publication Ethics

The author declares that this study complies with Research and Publication Ethics.

References

Acar Vural, R., Sert, M. Y., & Karaköse, B. (2018). Gerçek Zamanlı Sürücü Yorgunluk Tespit Sistemi. Marmara Fen Bilimleri Dergisi, 30(3), 249-259. https://doi.org/10.7240/marufbd.417915

Aki, M. O. (2017). Sürücü uykululuğunun gerçek zamanlı görüntü işleme ve makine öğrenmesi teknikleri ile tespitine yönelik bir sistem tasarımı ve uygulaması, Doktora Tezi, Trakya Üniversitesi, Fen Bilimleri Enstitüsü.

Çivik, E. (2020). Gömülü sistem üzerinde derin öğrenme bazlı sürücü yorgunluk tespiti, Master's thesis, Bilecik Şeyh Edebali Üniversitesi, Fen Bilimleri Enstitüsü.

Django (Version 5.1) (2024). [Computer Software]. https://www.djangoproject.com (Last Access Date: 04.03.2025)

Grover, P. (2018). Evolution of object detection and localization algorithms. Towards Data Science. https://medium.com/towards-data-science/evolution-of-object-detection-and-localization-algorithms-e241021d8bad (Last Access Date: 04.03.2025)

Golgiyaz, S., Kocamaz, A. F., & Okumuş, F. (2017). Video Tabanlı Uykulu Sürücü Algılama Sistemi. Journal of Safety Research, 7(1), 1-12.

Güney, E. (2021). Sürücü asistan sistemleri için mobil GPU tabanlı gerçek zamanlı durum analizi ve tespit uygulamaları, Master's thesis, Sakarya Üniversitesi, Fen Bilimleri Enstitüsü.

Jocher, G., Chaurasia, A., & Qiu, J. (2023). Ultralytics YOLO (Version 8.0.0) [Computer software]. https://github.com/ultralytics/ultralytics (Last Access Date: 04.03.2025)

Ma, B., Fu, Z., Rakheja, S., Zhao, D., He, W., Ming, W., & Zhang, Z. (2024). Distracted Driving Behavior and Driver's Emotion Detection Based on Improved YOLOv8 with Attention Mechanism. IEEE Access. https://doi.org/10.1109/ACCESS.2024.3374726

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 779-788. https://doi.org/10.1109/CVPR.2016.91.

Şafak, E., Dogru, İ., Barışçı, N., & Toklu, S. (2022). Derin öğrenme kullanılarak nesnelerin interneti tabanlı mobil sürücü yorgunluk tespiti. Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi, 37(4), 1869-1882. https://doi.org/10.17341/gazimmfd.999527

Şahin, A., Çil, S., & İstanbullu, A. (2020). Raspberry Pi 4 ile Sürücü Yorgunluk Tespiti ve Uyarı Sistemi. Journal of Science, Technology and Engineering Research, 1(1), 13-18. https://doi.org/10.5281/zenodo.3902907

SelfDriving Car. (2023). Self-Driving Cars Dataset [Open Source Dataset]. Roboflow Universe. Roboflow. https://universe.roboflow.com/selfdriving-car-qtywx/self-driving-cars-lfjou (Last Access Date: 04.03.2025)

Türkiye İstatistik Kurumu. (2023). Karayolu Trafik Kaza İstatistikleri. https://data.tuik.gov.tr/Bulten/Index?p=Karayolu-Trafik-Kaza-Istatistikleri-2023-5347 (Last Access Date: 04.03.2025)

Ultralytics. (2024). Documentation. https://docs.ultralytics.com/tr (Last Access Date: 04.03.2025)

Xie, S., Chuah, J. H., & Chai, G. M. T. (2023). Revolutionizing Road Safety: YOLOv8-Powered Driver Fatigue Detection. (2023) IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE), 1-6. https://doi.org/10.1109/CSDE59766.2023.10487765