Turk. J. Math. Comput. Sci.
17(1)(2025) 191–211

DOI : 10.47000/tjmcs.1644390

Deep Learning and LSTM Integration for Analyzing Driver Behaviors

Ilkay Cinar

Department of Computer Engineering, Faculty of Technology, Selcuk University, Konya, Türkiye.

Received: 21-02-2025 • Accepted: 28-04-2025

Abstract. Real-time detection of driver behaviors, fundamental for autonomous vehicles, is crucial for preventing accidents and enhancing traffic safety. Traditional methods, relying on manual observations or sensor-based monitoring, are increasingly being replaced by automated solutions using machine learning and computer vision technologies. This study aims to improve the classification of driver behaviors through the integration of deep learning models with LSTM layers. A multi-class driver behavior dataset, including images of safe driving, phone conversations, texting, turning, and other distractions, was used. Data processing involved cross-validation to ensure reliable performance evaluations. Various deep learning models such as VGG19, ResNet50, MobileNetV2, InceptionV3, DenseNet201, and InceptionResNetV2 were employed, each integrated with LSTM layers to create hybrid architecture. LSTM's ability to capture temporal dependencies enabled more accurate behavior classification. Model performances were evaluated using accuracy, precision, recall, F1-Score, Log Loss, and ROC-AUC metrics. Experimental results demonstrated that LSTM integration significantly enhanced classification performance. InceptionResNetV2 and MobileNetV2 also achieved strong results with LSTM, while DenseNet201 was the most accurate at 94.77%. Road safety applications and real-time monitoring systems can benefit from these findings. In addition, this study contributes to the development of driver monitoring systems based on machine learning, which has the potential to enhance safety in autonomous vehicles.

2020 AMS Classification: 68T07

Keywords: Deep learning, driver behavior, driving scenarios, image classification, LSTM.

1. Introduction

Driver behavior detection has become an essential area of research over the past few years, as the fields of autonomous driving and artificial intelligence are progressively being applied such as in Intelligent Transportation Systems (ITS) to safeguard the roads and, by getting rid of human errors in driving, avoid traffic accidents. According to the World Health Organization (WHO) and what it reported in the 2023 Traffic Accidents Report, high-risk causes of traffic accidents are human errors, including distracted driving, fatigue and aggression [39]. Real-time detection of driver behaviors, which is a prerequisite for autonomous car driving is of utmost importance for preventing accidents and making the whole traffic system safer. Driver behavior evaluation by traditional means like manual observations or sensor-based monitoring systems is slowly being replaced by automated machine learning and computer vision technology applications. Such instruments can, in fact, not only sift the big data sets but also catch driver behaviors that the human eye can't, and hence, timely interventions can be done. The key to smart cars, which are low demand human intervention autonomous vehicles, which in turn ensures preventing and decreasing the causes of accidents thereby traffic safety level can be high, is the real-time recognition of drivers' behaviors. Traditional driver observation methods use

Email address: ilkay.cinar@selcuk.edu.tr (I. Cınar)

methods such as sensory measurement or by using observation, but quite unfortunately, these methods are increasingly being replaced by the technology of automated solutions using machine learning and computer vision technologies. Such solutions can process big datasets and identify drivers' behaviors that human vision does not pick up, allowing for on-time interventions.

Machine learning and deep learning algorithms, which are becoming more popular, have shown great potential in traffic classification and detection of driver behavior. Convolutional neural networks (CNN) have really come a long way in the image-based scenario through their ability to endow the system with the opportunity of self-creation of precise feature extraction from visual data, owing to which the performance of the algorithms also increases relative to other methods based on pre-programmed characteristics [19, 42]. CNNs are powerful at learning local relationships within data and making sense of spatial hierarchy, making them ideal for analyzing driver behavior. With these algorithms, it is possible to obtain accurate and faster results from large datasets than with traditional methods [15].

The last few years have seen deep learning models, mainly CNNs, become very popular in classifying driver behaviors such as distraction, fatigue, and aggressive driving. CNNs manage to get around the drawbacks of typical feature engineering methods as they are creating deeper and more complex relationships with their layered structures on the data. As a result, these developments have not only touched the field of driver behavior detection, they have also made it feasible to use such systems more effectively and efficiently in any real-world context [45]. Nowadays, the majority of the driver behavior studies are done on the basis of the immediate, but the temporal nature and the dynamic processes are also coming into consideration. Nonetheless, a milestone representing the super-promising development of the deep learning models is the creation of multi-class classification methodologies referring to the more specific issues of driver behaviors. It is even now possible to get beyond the simple binary classification such as "careful" and "inattentive" and have more precise sense of driver behavior [12]. The implementation of deep learning algorithms, particularly CNNs, has boosted the precision of driver behavior detection and made spectacular changes in the concept of being applied. These developments are not only able to recognize drivers' behavior accurately but also afford safer and more reliable driving systems in essential areas such as autonomous driving and road safety [46].

As the studies already present show, there is a significant challenge in the classification of driver behaviors to a wider range and in getting right the time-varying behaviors of individuals. In some datasets, even when the classifiers are ten and they capture the whole spectrum of behaviors, there are many other behaviors that cannot be categorized as any of the ten provided classes. Another "problem" with the data collected often is focusing on particular driving habits, and the few, if any, available, do not cover the whole range of behaviors. In this work, we set out to examine the driver behaviors more holistically by introducing the "Multi-Class Driver Behavior Image Dataset" which is a newly created public dataset that has not yet been studied which is based on the result of this study [27]. The deep learning models VGG19, ResNet50, MobileNetV2, InceptionV3, DenseNet201, and InceptionResNetV2 were used in the analysis. Moreover, with Long Short-Term Memory (LSTM) integration, they have classified the systems of these models. Thus, it is possible to predict driver's behavioral changes that occur over time, which, adequately embedded into the system, result in more sensitive and accurate predictions in the decision-making process. This research study is yet another pilot to the successful implementation of not only fully-automated vehicles but also the potential of safety systems for driverless cars.

2. Related Works

A deep convolutional neural network model was designed by Yan et al. (2016) to find the driver's behaviors. In the work, a Gaussian Mixture Model was proposed as an enhancement effect to the RCNN architecture. The driver's posture and environmental signs were utilized to categorize actions. A cap of the Southeast University Driving-Posture Dataset (SEU) was used for the identification of six different driving behaviors (e.g., talking on the phone, eating), and the model obtained accuracy with 97.76% rate. According to Yan et al., this model is a good solution for traffic safety [42].

In the course of their research, Huang et al. (2019) studied various methods of abnormal driver recognition through deep learning technologies on videos. They applied the State Farm Distracted Driver Detection dataset to compare the three models: the wide group dense (WGD) model, the wide group residual dense (WGRD) model, and the alternative wide group residual dense (AWGRD) model. The model AWGRD they used was the one that showed the highest accuracy rate, 96.5%, and the highest F1 score, 0.96. They particularly pointed out that this model was outstanding in identifying abnormal behaviors such as distraction among others [19].

In 2020, Zhang et al. proposed an innovative framework referred to as "Interwoven Deep Convolutional Neural Network (InterCNN)," which was intentionally designed to leverage multi-stream data for determining the behaviors of drivers and recognizing distraction. The main objective of the research was to detect driver staff by the means of the equipment inside the car and the information found in the optical flow. They carried out the study on a dataset that involved 50 volunteers and identified the performance of the model on five classes such as overall activities which achieved an accuracy rate of 81.66% [45].

According to Huang et al. (2020), a novel approach called the Hybrid CNN Framework (HCF) was being created to determine the behaviors of those who are distracted while driving. They conducted the research on the State Farm Distracted Driver Detection dataset, which was composed of 22,424 above images and which showed 10 out of 10 different distracted driver behaviors (for example, talking on the phone, eating, and hands on the steering wheel). Features of HCF are ResNet50, InceptionV3, and Xception with an accuracy rate of almost 97%, which makes it efficient to use in real-time situations of distracted driving detection [15].

Chen et al. (2020) made progress in creating something newer while using deep learning classification techniques, which in this case is the two-stream Convolutional Neural Network. More specifically, such techniques have not yet been exploited on the analysis of the driving behaviors of certain drivers. The authors of this paper, as a result, came up with the idea of using spatial and temporal aspects jointly. Finally, they have trained and tested the system on 10 different driving activities (e.g., texting with the right hand) classified. The model achieved the intended task, resulting in an accuracy of 68.25% [8].

The authors, Al doori et al. (2021) took an initiative to provide safety on the roads by resolving the driver errors in their experiments through the image classification models. They used a 10-class dataset that had a record of 22,424 images regarding different driver errors in the training and evaluation of their model. The SqueezeNet CNN architecture was trained with transfer learning, and the CNN was configured to complete the feature extraction task followed by the classification layer. The three machine learning algorithms: k-Nearest Neighbors (k-NN), Support Vector Machine (SVM), and Random Forest (RF) were used in the classification of the features extracted. From the results of the classification, k-NN had the best performance with an accuracy of 98.1%, followed by SVM on 95.8% and RF scoring 88.7% [2].

Huang and Chen (2021) proposed a new autonomous driving model (DDBD). They claimed a 94.72% accuracy rate was obtained, using the Kaggle Driving Dataset, which was the study's source. This dataset includes 10 driver behaviors (e.g., texting with the right hand, calling with the left hand, consuming drinks). It was further explained that DDBD can visualize the points of interest which allowed a better performance in comparison to the other existing methods [18].

Zhao et al. (2021) conceived a driver behavior detection system by utilizing an adaptive spatial attention mechanism. They assessed it on a dataset made up of various cars and real driving situations. The dataset contains photos of 44 different drivers and is categorized into 10 behavior groups (e.g., fingering with the right hand, calling with the left hand, adjusting the radio). In the tests involving ResNeXt50 and ResNeXt101 architectures, both models using the multi-scale fusion method achieved an accuracy of 97.19%, a sensitivity of 97.32%, and an F1 score of 97.16% [46].

The recent advancement in the field of machine learning proposed by Xiao et al. (2024) is the incorporation of fuzzy logic and attention mechanisms, which is the Fuzzy Deep Attention Network (FDAN) model, for evaluating and detecting driver behavior. The research was conducted using the Hunan University (HNU) dataset and the American University in Cairo (AUC) datasets, achieving 90.81% Top-1 accuracy and 98.48% Top-5 accuracy rates. They believe that FDAN is the one that guarantees high accuracy by managing data uncertainties [40].

Al Ali et al. (2024) developed a system to monitor attentive driving using the YOLOv8 object detection algorithm. For the training of the model, a dataset consisting of 8,440 images including different lighting conditions, ethnicities, eye colors and facial angles was created, and this dataset was divided into seven classes (smoking, using the phone, eating, drinking, not wearing a seat belt, yawning and eyes closed). 80% of the images were used for training, 10% for validation and 10% for testing. During the training process, the YOLOv8 model was trained for 170 epochs, and the average accuracy (mAP) was reported as 93.1%, precision as 89.2% and recall as 90.1%. In real-time performance tests, it was stated that the average accuracy rates were achieved as 95.7% and 96.2%, respectively [1].

Lai (2024) integrated attention mechanisms in driver behavior detection using MobileNetV2, InceptionV3 and ResNet50V2 models. In the study, AUC Distracted Driver Dataset with ten different driver behaviors was used and they stated that they achieved the highest rate of 98.8% accuracy with ResNet50V2 [26].

Alzeari and Becerikli (2024) classified 15 different driver behaviors such as distraction, aggression, and fatigue using machine learning methods. Methods such as Histogram of Oriented Gradients (HOG), Local Binary Patterns (LBP), RGB, KAZE, and Color Histogram were applied for feature extraction; these features were subjected to dimensionality reduction with PCA and LDA techniques. In the classification phase, 10 different algorithms such as Logistic Regression (LR), Support Vector Machines (SVM), K-Nearest Neighbors (KNN), Decision Trees (DT), Random Forest (RF), Bagging, AdaBoost, Stochastic Gradient Descent (SGD), and Extreme Gradient Boosting (XGB) were used. SVM and LR algorithms achieved the highest accuracy rate with 99.47% and 98.82% F1 score [3].

Kang et al. (2024) classified driver behaviors using a deep learning method based on ResNet. In the study, driver behaviors are classified using three different models (IRDMDB-1, IRDMDB-2, IRDMDB-3) and this method is tested on The State Farm and Driver Drowsiness datasets. It was stated that the IRDMDB-3 model detected various driving behaviors with the highest accuracy rate of 99.79% [21].

Reflecting on the literature, various studies have been conducted on the classification of driver behaviors. However, since the dataset used in this study has not been used in any previous study, it is not possible to make an effective comparison. In this study, the hybrid LSTM+CNN models were systematically compared with different configurations, and it is possible to say that the 94.77% accuracy rate obtained with all the models used, especially the DenseNet201 model, exhibited a competitive performance when compared to similar models in literature.

3. Material and Methods

This section encompasses the research that follows a well-organized and systematic method for identifying driver behavior through deep learning models along with LSTM. Initially, a Multi-Class Driver Behavior Image Dataset is used which consists of various images of drivers involved in numerous activities like safe driving, texting, and phone chatting. The dataset is then further processed using cross-validation thereby rendering a systematic analysis through various iterations by the training and the test data for evaluation purposes. In the subsequent phases of the study, a series of deep learning models such as VGG19, ResNet50, MobileNetV2, InceptionV3, DenseNet201, and InceptionResNetV2 were employed in classifying driver behavior. Besides stand-alone deep learning cases, the study also explores hybrid models where LSTM is combined with each of the models to utilize LSTM's potential of capturing the temporal dependencies present in the data.

Finally, the results from both deep learning models and deep learning + LSTM models are compared to assess their classification performance. This comparison helps determine whether adding LSTM enhances accuracy and robustness in recognizing different driver behaviors. The overall study aims to develop an effective model for detecting distracted driving, which can be useful for road safety applications and real-time monitoring systems. The flow diagram of the study is given in Figure 1.

3.1. **Dataset Description.**

Distracted driving accidents are becoming a global problem, especially as road traffic increases in densely populated areas. To address this issue, a new dataset is presented by Sahin Afridi et al. to support the development of real-time monitoring and detection systems by capturing actual driver behaviors [27]. This dataset, collected in October 2024 in Ashulia district of Dhaka, Bangladesh, consists of 7286 high-resolution images taken under real-world driving conditions in both private vehicles and public transport buses. However, since a total of 42 images in all classes were corrupted, 7244 images were used in this study. The images were captured using personal mobile phones to provide a realistic and diverse visual dataset. This dataset includes a wide range of driving behaviors, including potentially risky behaviors such as safe driving, turning, texting, talking on the phone, and fatigued driving. By depicting these behaviors in everyday driving scenarios, the dataset constitutes a valuable resource for training and evaluating models developed to detect unsafe driving practices in real time. Information about the classes in the dataset and sample images are provided in Table 1.

3.2. **Model Architecture.**

The project would be impressed by the development of various deep learning algorithms to enable the proper identification of driver behaviors. Each model utilized is vital to the recognition of driver behaviors. Below is a detailed breakdown of the architecture that were selected, and their procedures for integration.

3.2.1. ***Convolutional Neural Networks (CNNs).***

CNN were used as the basis for accurate classification of driver behaviors. CNNs are a highly effective structure for learning features from visual data and are widely used in deep learning applications [6, 36]. In this study, the following CNN-based models were used to detect driver behaviors:

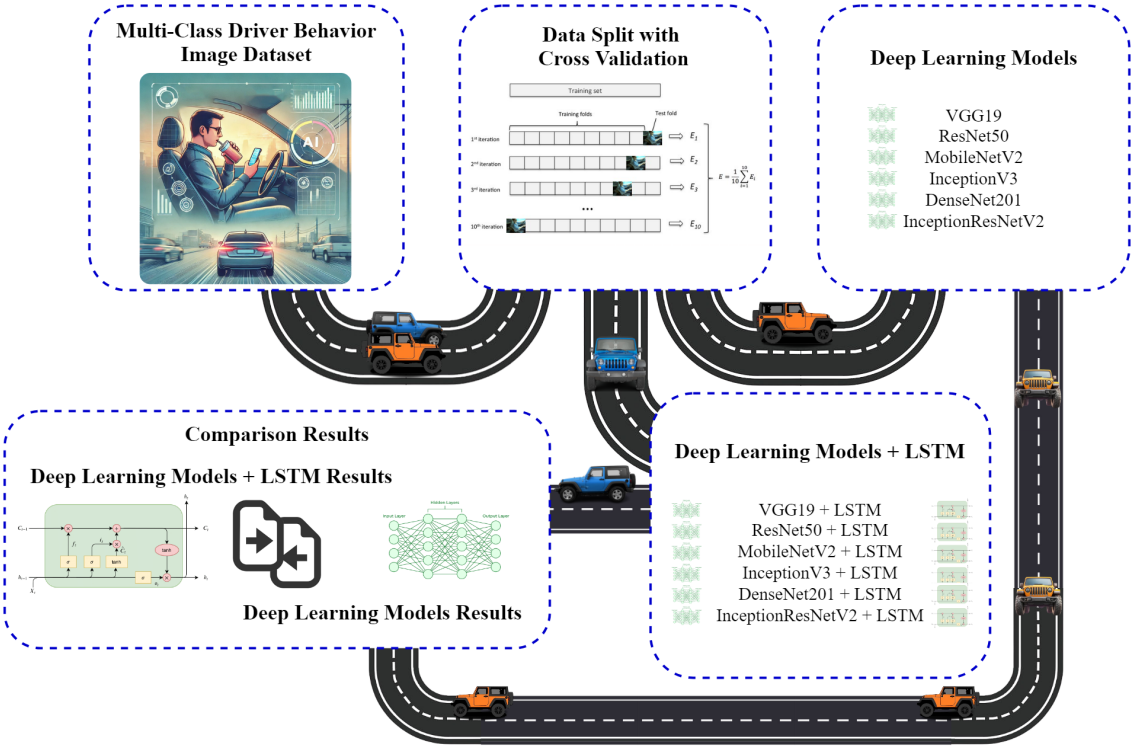

FIGURE 1. Flow diagram of the study

VGG19: The CNN VGG19 is one of the most popular deep learning algorithms. Its multi-layered structure gives it the ability to learn deep features. The architecture of this network includes 19 layers, which allow to take image data and extract high-level features. The design of VGG19 is simple but capable of learning much deeper features from data. Such structure enables the model to recognize distinct features in the image owing to its exceptional efficiency. Every layer processes the inputs from the previous layers by applying increasingly detailed operations, thus learning more complex features [29, 38, 44].

ResNet50: ResNet50, which integrates residual connections to the back-propagation issue of deep neural networks, is very popular due to its extremely high accuracy rates and deep networks can be more productive through its application when training it. ResNet50 is a fifty-layer architecture that enjoys the process of learning through direct connections. The biggest benefit of this model is its residual structure that assures that the deeper layers do not skip the learning step. In ResNet50, the layers augment the information coming from the previous layers in the way of direct connections, a method that allows the model to learn quicker and more effectively [11, 13, 41].

MobileNetV2: A model that has been optimized for use in mobile devices is capable of executing all operations without compromising on the accuracy of the model while being faster-in computational power consumption than other standard convolutional neural networks by using depthwise separable convolutions [14, 30].

InceptionV3: InceptionV3 is a model that is capable of extensive feature extraction in deep networks by means of filters of different sizes between layers. Thus, it can be virtually considered as a learning tool used for a wide range of objects that filter from a large one to actual separate pixels. In such a way, the model is allowed to simultaneously feature even the smallest detail of the large structure in the image. Versatility in feature extraction as well as the learning of different sizes of the filter in parallel between layers are the main characteristics of InceptionV3 [33, 34, 43].

DenseNet201: Each layer can take the output of the previous layers, and it is this ability to process the information that is learned in the previous steps that the layers can learn more about the data. The architecture of DenseNet201 lays out the structure for each layer to learn more features in a dense way by making the connections between each of them. This type of network design makes it possible for a large number of connections between layers, smooths the

TABLE 1. Information about the dataset

Class Information	Description	Number of Images	Sample Images
Safe Driving	Images of the driver looking at the road with both hands on the wheel, one hand on the steering wheel and the other on the gear lever. This is an ideal example of distraction-free driving.	1679	
Turning	Images of the driver changing direction by moving his/her head or whole-body during turns. This behaviour provides an important indication of how much the driver focuses on daily tasks such as turning the steering wheel.	1343	
Texting Phone	Photos of the operator texting or engaging with the display through the phone. Texting is particularly well-known to be a problem in driving. It is critical to know this action in the study of distraction.	1561	
Talking Phones	A driver's photo could be of him/her speaking on the phone or holding the phone to his/her ear. Another major source of distraction is talking on the cell phone.	1513	
Others	It covers any other actions that go against safe driving practices while driving, such as drinking water, sleeping, or talking to someone behind you.	1190	

information delivery, and thus makes it possible for the model to learn better. The interdependence of features is better understood through DenseNet201, which allows for a more thorough classification [16, 17].

InceptionResNetV2: The Inception and ResNet models are combined to constitute a single framework. The integration of both architectures yields a wide set of benefits. The model exploits Inception's multi-scale feature extraction along with ResNet's residual connections thus enabling the more effective learning of deep networks. InceptionResNetV2 is particularly designed for deeper networks which leads to a more efficient learning process and consequently better results. The use of the residual connections in the decrease of the selectivity of the deep layer's activity is increased on the one hand while the on the other hand Inception framework is able to perform more feature extraction by operating multiple kinds of filters in parallel [28, 32].

Table 2 presents a comparison of the deep learning models used in the study in terms of number of layers, parameter size and processing complexity (FLOPs).

This table shows that the models offer different advantages with respect to computational complexity (FLOPs), depth (number of layers) and parameter sizes. The DenseNet201 model stands out for its acceptable parameter size, while MobileNetV2 is ideal for real-time applications with its low FLOPs and small number of parameters. Deep

TABLE 2. Number of layers and architectural characteristics of the deep learning models used

Model	Number of Layers	Parameter Size (Million)	FLOPs (Billion)
VGG19	19	143	19.6
ResNet50	50	25.6	3.8
MobileNetV2	53	3.4	0.3
InceptionV3	48	23.8	5.7
DenseNet201	201	20.2	4.3
InceptionResNetV2	572	55.9	13.1

models such as ResNet50 may perform poorly when used alone due to its inability to model temporal patterns well, but its performance improves significantly with LSTM. VGG19 has a large number of parameters, so training time and hardware requirements can be high. These differences clearly demonstrate the suitability of each model for different application scenarios.

3.2.2. ***Long Short-Term Memory (LSTM)***. Long Short-Term Memory (LSTM) is a feedback neural network model that was specially created to deal with time series data and features that change over time. LSTM is a subtype of the Recurrent Neural Networks (RNN) family, but it is also intended for the specific task of solving RNNs problems which include components like gradient vanishing and gradient explosion. The key to the success of LSTM is that it can maintain information for an extended period based on what it has learned over time while at the same time being able to purge any unnecessary information selectively by forgetting what is not significant to retain the important ones. This feature makes LSTM very effective for working with changing data in order and allows it to accurately model dynamic changes, especially driver behavior [4, 22].

LSTM Structure: LSTM has three main components in each cell: the forget gate (f_t), the input gate (i_t), and the output gate (O_t). These components provide control over how the network stores past information, which information it forgets, and which information it decides to discard.

In this way, LSTM can keep past data in its long-term memory and use it for future predictions [4, 22]. The LSTM structure is given in Figure 2.

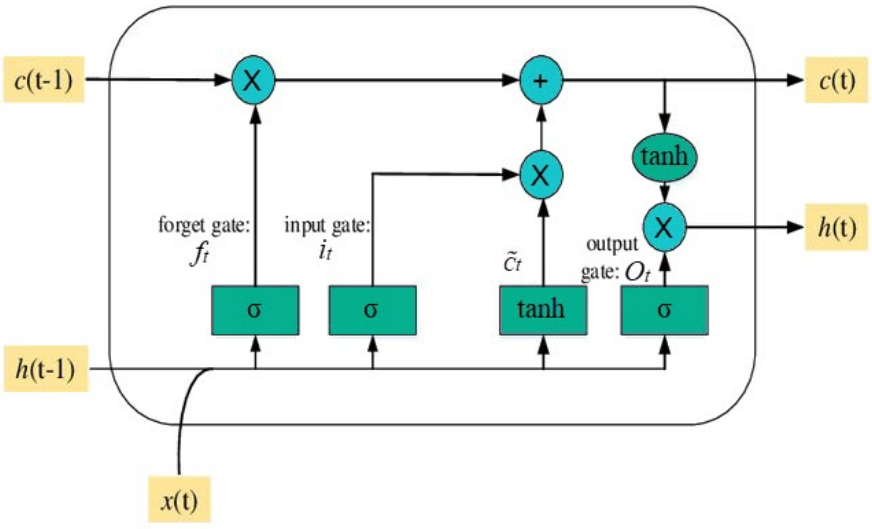

FIGURE 2. LSTM Structure

1. Forgetting Gate: This gate determines what information the network should forget at each time step. It decides how much of the information from the previous time step (C_{t-1}) will be stored and how much will be discarded. The sigmoid activation function σ is used because it returns a value between 0 and 1 [4].

$$f_t = \sigma\left(W_f \cdot \left[h_{t-1}, x_t\right] + b_f\right)$$

Here:
W_f : Weight matrix of forget gate
b_f : Bias term of the forgetting gate
h_{t-1} : Hidden state in previous time step
x_t : Current input data
If the f_t output is close to 0, it tends to forget, if it is close to 1, it tends to store.

2. Input Gate (i_t) and Candidate Cell State ($\tilde{C}_t$): This gate determines how much information the network will receive at each time step. Before adding new information to the cell, it is decided which data will be included in the network's memory cell [4].

$$i_t = \sigma\left(W_i \cdot \left[h_{t-1}, x_t\right] + b_i\right)$$

$$\tilde{C}_t = \tanh\left(W_c \cdot \left[h_{t-1}, x_t\right] + b_c\right)$$

Here:
i_t : Determines which information should be included (between 0 and 1).
$\tilde{C}_t$: The new candidate cell status takes a value between -1 and 1 using the tanh function.

3. New Cell State (C_t): The new cell state is calculated by combining the old cell state and the new candidate cell information [4].

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t$$

Here $\odot$, Represents element-wise multiplication operation.
$f_t \odot C_{t-1}$: Stores a certain part of the previous cell state.
$i_t \odot \tilde{C}_t$: Adds new information.
This will update the cell status.
4. Output Gate (O_t), and Hidden State (h_t): This gate determines what information the network will give out, i.e. the output. This gate converts the current information in the cell into output. Finally, the output gate determines what will be transferred from the cell state to the hidden state [4].

$$O_t = \sigma\left(W_o \cdot \left[h_{t-1}, x_t\right] + b_o\right)$$

$$h_t = O_t \odot \tanh(C_t)$$

Here:
O_t : The part determined by the output gate.
$\tanh(C_t)$: The cell state is compressed between -1 and 1.
h_t : The hidden state that is the output of the network.

3.2.3. ***Integration of LSTM with CNN***. In this study, LSTM is integrated with CNN models. CNNs are extremely effective in extracting features from image data, but they do not take into account time dependencies. LSTM is used to process time-varying information by taking feature maps obtained from CNNs. Owing to this integration, CNNs recognize basic features in the image, while LSTM understands the dynamic change of these features over time. A representative explanation of LSTM integration to pre-trained CNN models is given in Figure 3. Here, the reshape layer represents the layer where the CNN outputs are transformed into a suitable format for the LSTM input.

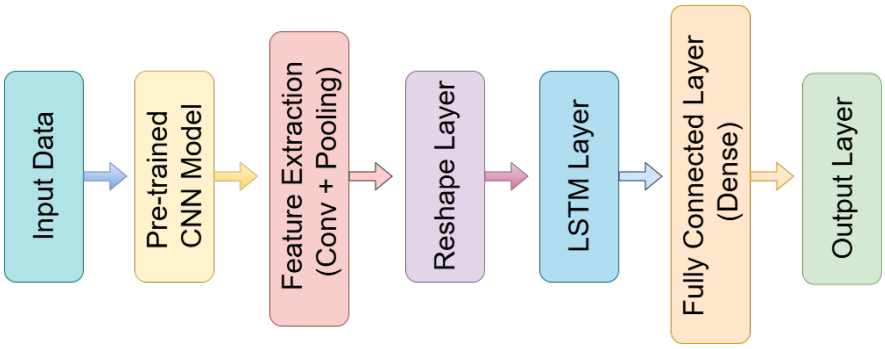

FIGURE 3. Representative Illustration of LSTM Integration into Pre-trained CNN Models

3.3. **Training and Evaluation.** In this study, the hyperparameters used in training the models include basic parameters such as batch size, epoch number, learning rate and optimizer. Batch size was selected as 16 and 32 in all models because this value allows the model to be updated more frequently and allows for good performance with high accuracy while keeping memory consumption at a reasonable level. Epoch number was determined as 20, thus ensuring that the model receives sufficient training and preventing possible overfitting. Learning rates were selected as 0.001 and 0.0001 because a low learning rate allows the model to learn more slowly but accurately and prevents large steps at the beginning of training, allowing it to approach the minimum loss in a more stable manner. Adam was selected as the optimizer, which can provide a faster and more efficient training process by adapting the learning rate to each parameter. In addition, faster and more effective results were obtained with the benefits of pre-trained models using transfer learning. These hyperparameters were selected considering the characteristics of the dataset and the overall performance of the model, and ensured that the model was trained faster, more accurately and more efficiently.

In the training process of the models, the data was divided differently in each fold using the 10-fold cross-validation method and the training and validation processes were performed. The method under discussion involves splitting the data into 10 clusters of equal size. Each of the clusters serves as a validation set once, while the model is trained using the other nine clusters. The process is then repeated, and the results of each iteration are compiled for an assessment of the model's total performance [10, 23, 37]. Model training and testing was performed on a server with a NVIDIA GPU with 12 GB RAM with 10240 CUDA cores and a processor with 12 cores and 64 GB RAM.

During the evaluation phase the models used are examined through Accuracy, Precision, Recall, F1-Score, Log Loss, ROC-AUC metrics. These metrics are the most widely used for measuring the model's accuracy, reliability and overall performance.

Confusion Matrix: A tool that is utilized for comprehending the in-depth operational character of classification models. This matrix, which is the crux of the confusion matrix consists of a table that shows the correct and incorrect predictions for each class of the model. The confusion matrix serves to indicate where the model is failing regarding class and assess its performance with more detail [5, 20, 31, 35]. The two-class confusion matrix representation is given in Figure 4, and the multi-class confusion matrix is given in Figure 5.

The basic components in Figure 4 are explained below on the safe driving class [5, 7, 20, 31]:

1. True Positive (TP): Cases where the model correctly predicts the positive class (e.g. safe driving). That is, if the true label is positive (safe driving) and the model's prediction is also positive (safe driving), this is a TP.
2. True Negative (TN): Cases where the model correctly predicted the negative class (e.g. unsafe driving). That is, if the true label is negative (unsafe driving) and the model's prediction is also negative (unsafe driving), this is a TN.
3. False Positive (FP): Cases where the model incorrectly predicted the positive class. That is, if the true label is negative (unsafe driving) but the model predicted the positive (safe driving), this is an FP. The model incorrectly predicted safe driving.
4. False Negative (FN): Cases where the model incorrectly predicted the negative class. That is, if the true label is positive (safe driving) but the model predicts the negative (unsafe driving), this is a FN. The model incorrectly predicted unsafe driving.

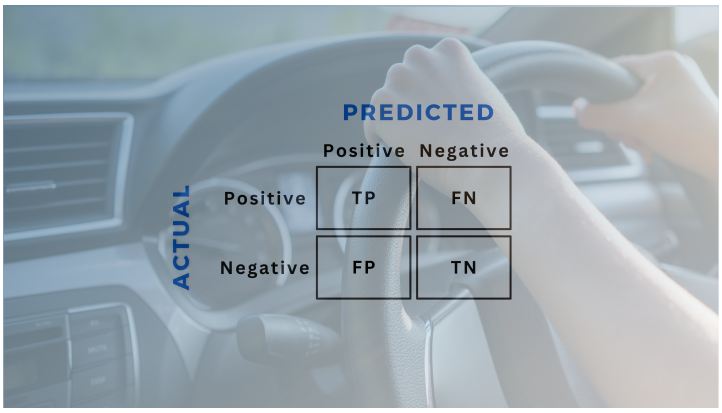

FIGURE 4. Binary Class Confusion Matrix

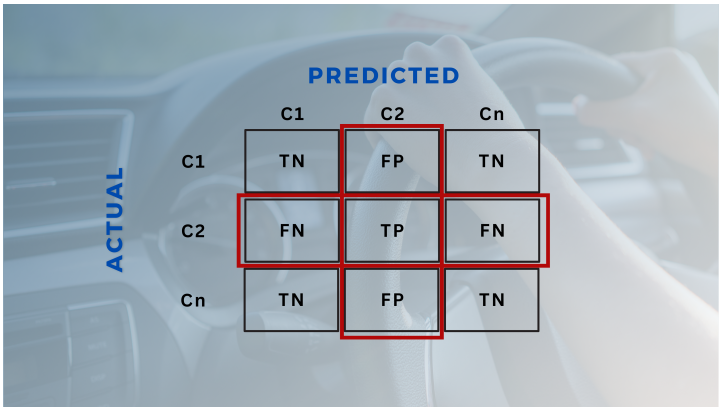

FIGURE 5. Multi-Class Confusion Matrix

Accuracy is the ratio of the number of examples that the model correctly classified to the total number of examples. This metric is one of the most widely used performance measures that shows the overall success of the model. The formula for Accuracy is given in Eq. (3.1) [5, 20, 24, 31].

$$Accuracy = \frac{\sum TP_i}{Total\ Number\ of\ Samples} \tag{3.1}$$

Precision is the proportion of instances that the model predicted as positive that are actually positive. This metric indicates how accurately the model predicted positive classes. The precision formula is given in Eq. (3.2) [5, 20, 24, 31].

$$Precision = \frac{\sum TP_i}{\sum(TP_i + FP_i)} \tag{3.2}$$

Recall measures how many of the true positive examples were correctly predicted as positive. That is, it shows how well the model detects true positive examples. The recall formula is given in Eq. (3.3) [5, 20, 24, 31].

$$Recall = \frac{\sum TP_i}{\sum(TP_i + FN_i)} \tag{3.3}$$

F1-Score is the harmonic mean of precision and recall metrics. F1-Score is used to evaluate precision and recall in a balanced manner, especially in cases where classes are unbalanced. The F1-Score formula is given in Eq. (3.4) [5, 20, 24, 31].

$$F1\text{-}Score = \sum 2 \times \frac{Precision_i \times Recall_i}{Precision_i + Recall_i} \tag{3.4}$$

Log Loss (or Logarithmic Loss) is a loss function that measures how close the model's predictions are to the true classes. Log Loss shows that the performance of the model improves as it takes smaller values. The formula for Log Loss is given in Eq. (3.5) [9].

$$LogLoss = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(p_i) + (1 - y_i)\log(1 - p_i)] \tag{3.5}$$

Here:

- y_i actual class (0 or 1).
- p_i the probability value predicted by the model.

ROC Curve (Receiver Operating Characteristic Curve) is a tool that visually shows the classification accuracy of the model for all classes. This curve is a graph showing the False Positive Rate (FPR) and True Positive Rate (TPR) [19, 25].

AUC (Area Under the Curve), It refers to the area under the ROC curve. The AUC value shows the overall performance of the model. As the AUC value approaches 1, the performance of the model increases. An AUC value of 0.5 indicates that the model makes random assessments [9, 25].

4. Experiments and Results

The application of deep learning, particularly CNNs, to the detection and classification of driver behaviors while driving, is the primary focus of this research. The identification of driver's behaviors accurately is an essential parameter for improving the safety of our roadways and reducing the risk of accidents. In this study, a number of effective CNN architectures were employed in an effort to enhance classification performance and provide a strategy for behavior prediction.

The current experiment implements CNN networks of the following types: DenseNet201, InceptionResNetV2, InceptionV3, MobileNetV2, ResNet50, and VGG19. These architectures were selected due to the recognition of their efficiency in image classification, which is based on deep extraction features. The models used for training and testing consist of the images showcasing various driver-related activities that helped to assess the models' classification power. The goal of the classification was to segregate five driver actions: safe driving, turning, texting on the phone, talking on the phone, and other activities. Each of these behaviors offers a different degree of risk and precisely detecting them is vital for the proper design of advanced driver monitoring systems. To figure out how effective different CNNs are, various performance metrics were used. These include accuracy, precision, recall, F1-score, log loss, and ROC-AUC. These metrics also represent the broader capabilities of models in classifying events and hence are useful for a more detailed strength-weakness comparison of driver behavior predictions.

The results obtained from the image classification process are analyzed based on performance metrics. The findings highlight the effectiveness of CNNs in recognizing and predicting driver behaviors while also identifying the most suitable architectures for this task. This study contributes to the growing body of research on intelligent driver behavior monitoring systems and underscores the potential of deep learning in enhancing road safety through automated behavior detection.

Figure 6(a) illustrated confusion matrix that evaluates the performance of a classification model using LSTM and DenseNet201 with parameters 16, 0.0001. The model demonstrates strong classification accuracy, as most predictions align correctly with the actual classes. The highest values along the diagonal indicate successful classifications, with 106 correctly identified instances of "Other Activities," 157 for "Safe Driving," 146 for "Talking Phone," 150 for "Texting Phone," and 126 for "Turning." While the model performs well overall, 30 total misclassifications are present.

The ROC curve demonstrates the excellent performance of the LSTM-DenseNet201 model with 10-fold cross-validation. The mean ROC (blue line) is close to the top-left corner, with an AUC of 0.99 ± 0.00, indicating near-perfect classification. The minimal variance (shaded area) suggests consistency across folds. Since the curve is well above the random classifier baseline (red dashed line), the model significantly outperforms random guessing (Figure-6b).

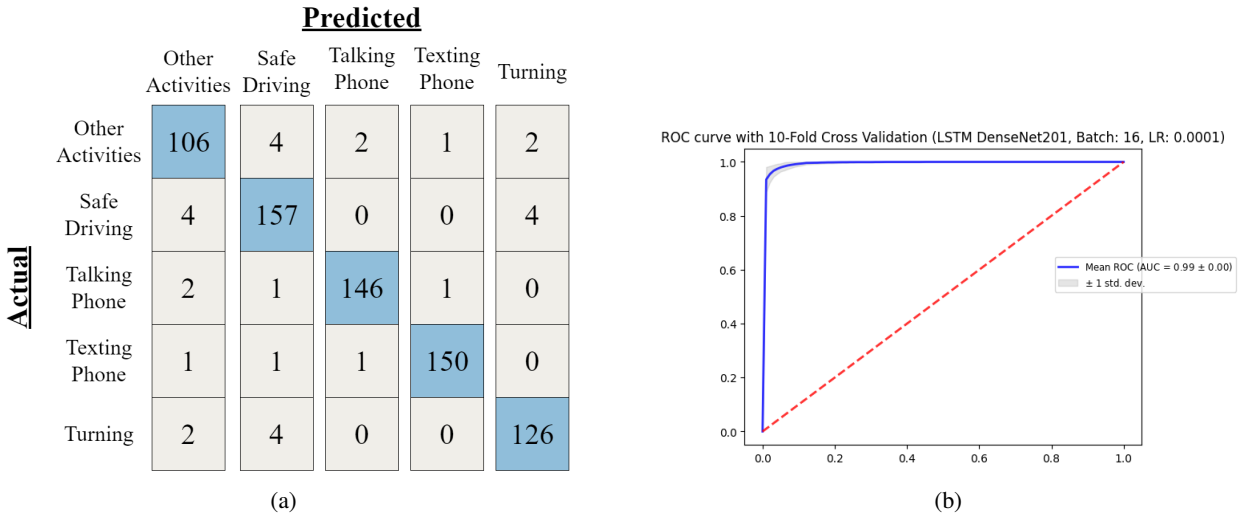

FIGURE 6. (a) Confusion matrix and (b) ROC curve for DenseNet201 model with LSTM with 16 batch size and 0.0001

The confusion matrix represented in Figure 7(a) showed the performance of a classification model using MobileNetV2 without LSTM, with parameters 16, 0.0001. The model generally performs well, as most predictions align correctly with the actual classes, reflected by the high values along the diagonal. Specifically, 102 instances of "Other Activities," 155 of "Safe Driving," 144 of "Talking Phone," 150 of "Texting Phone," and 124 of "Turning" were correctly classified. However, some misclassifications are present, 37 misclassifications were presented which remains minimal error. The ROC curve represents the performance of the MobileNetV2 model with 10-fold cross-validation without LSTM shown in Figure 7(b). Compared to the LSTM-integrated model, the classification accuracy appears slightly lower, with a few more misclassifications in each category. This suggests that LSTM may improve sequential pattern recognition, reducing classification errors in activities with subtle temporal differences.

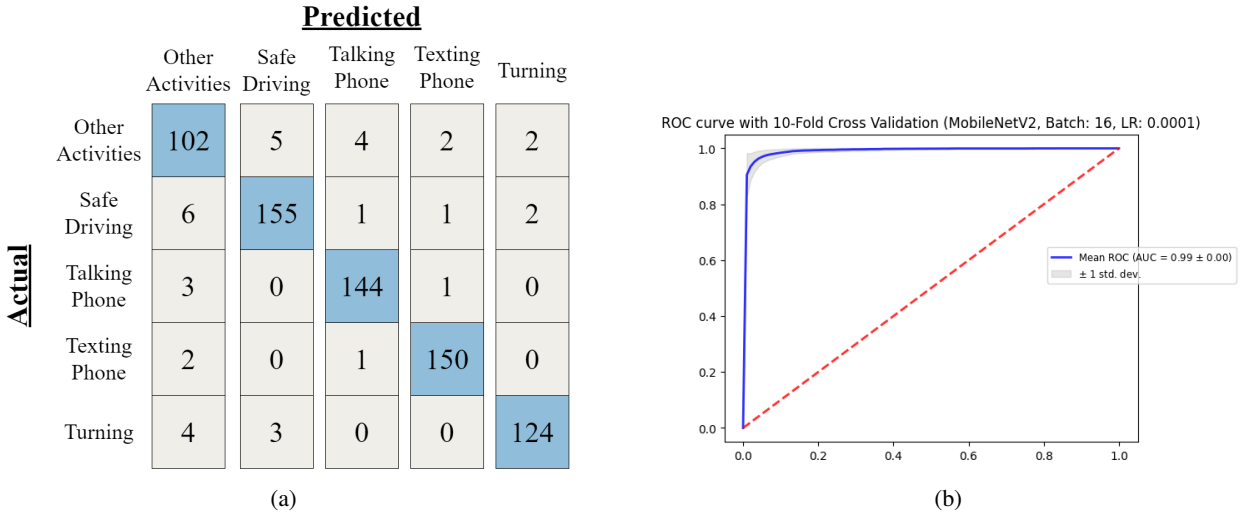

FIGURE 7. (a) Confusion matrix and (b) ROC curve for MobilenetV2 model without LSTM with 16 batch size and 0.0001

The performance of six CNN architectures was evaluated for classifying driver behaviors. The models were trained using an LSTM layer with 16 units and a learning rate of 0.001 to enhance temporal feature extraction and improve classification accuracy. Table 3 shows the highest accuracy of 93.53% was achieved by DenseNet201, Additionally, a high ROC-AUC score of 0.9939 was recorded, indicating strong performance in distinguishing between different classes. A comparable accuracy of 92.79% was obtained by VGG19, with a similar ROC-AUC score of 0.9937. An accuracy of 92.70% was recorded by MobileNetV2, highlighting its efficiency despite being a lightweight model designed for mobile applications. This result suggests that MobileNetV2 could be an effective choice for real-time driver monitoring systems where computational efficiency is crucial. Slightly lower accuracies of 91.94% and 90.85% were obtained by InceptionResNetV2 and InceptionV3, respectively. The lowest accuracy of 87.78% was observed in ResNet50, along with a relatively lower F1-score of 0.8776 and a ROC-AUC of 0.9830. These results suggest that while ResNet50 has been effective in general image classification tasks, it may not have been the most suitable architecture for detecting driver behaviors in this dataset.

Table 3. Evaluation Results for CNNs with LSTM: Batch Size 16, Learning Rate 0.001

Models	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Log_Loss**	**ROC_AUC**
DenseNet201	93.53	0.9367	0.9353	0.9355	0.3167	0.9939
InceptionResNetV2	91.94	0.9212	0.9194	0.9195	0.3583	0.9917
InceptionV3	90.85	0.9095	0.9085	0.9080	0.4476	0.9893
MobileNetV2	92.70	0.9279	0.9270	0.9269	0.3757	0.9922
ResNet50	87.78	0.8794	0.8778	0.8776	0.3604	0.9830
VGG19	92.79	0.9302	0.9279	0.9277	0.3242	0.9937

The performance of the CNN models without the LSTM layer (16 units, 0.001 learning rate) was evaluated, revealing variations in classification effectiveness. DenseNet201 achieved the highest accuracy of 91.44% presented in Table 4, along with a strong ROC-AUC score of 0.9933, indicating its superior classification ability. MobileNetV2 and InceptionResNetV2 followed closely, with accuracies of 91.05% and 90.79%, respectively, demonstrating their efficiency in identifying driver behaviors. InceptionV3 and VGG19 exhibited slightly lower accuracies of 89.22% and 87.48%, respectively, suggesting that while they remained effective, their performance was slightly inferior compared to the top models. ResNet50, however, showed the weakest performance, with an accuracy of only 73.96% and a lower ROC-AUC score of 0.9439, indicating significant difficulty in correctly classifying certain behaviors.

Table 4. Evaluation Results for CNNs Without LSTM: Batch Size 16, Learning Rate 0.001

Models	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Log_Loss**	**ROC_AUC**
DenseNet201	91.44	0.9209	0.9144	0.9146	0.3171	0.9933
InceptionResNetV2	90.79	0.9119	0.9079	0.9079	0.2819	0.9911
InceptionV3	89.22	0.9002	0.8922	0.8926	0.4321	0.9885
MobileNetV2	91.05	0.9138	0.9105	0.9103	0.3636	0.9918
ResNet50	73.96	0.7685	0.7396	0.7325	0.7063	0.9439
VGG19	87.48	0.8851	0.8748	0.8756	0.3413	0.9863

The performance of the CNN models was further evaluated with LSTM layer size (32 units, 0.001 learning rate), Table 5 showing overall improvements in classification accuracy. DenseNet201 achieved the highest accuracy of 94.02%, reinforcing its effectiveness in extracting deep spatial features for driver behavior classification. A strong

ROC-AUC score of 0.9948 further confirmed its reliability in distinguishing between different behaviors. MobileNetV2 and VGG19 followed closely, with accuracies of 93.82% and 93.48%, respectively, demonstrating their robust classification capabilities with minimal loss. InceptionResNetV2 and InceptionV3 recorded slightly lower accuracies of 92.84% and 91.32%, respectively, maintaining competitive performance but lagging slightly behind the top models. ResNet50, however, showed the lowest accuracy at 87.24%, with a comparatively lower recall and F1-score, suggesting it struggled more with classification even with the LSTM enhancement.

TABLE 5. Evaluation Results for CNNs with LSTM: Batch Size 32, Learning Rate 0.001

Models	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Log_Loss**	**ROC_AUC**
DenseNet201	94.02	0.9417	0.9402	0.9404	0.2891	0.9948
InceptionResNetV2	92.84	0.9293	0.9284	0.9283	0.3447	0.9928
InceptionV3	91.32	0.9160	0.9132	0.9131	0.4227	0.9909
MobileNetV2	93.82	0.9392	0.9382	0.9382	0.3205	0.9949
ResNet50	87.24	0.8805	0.8724	0.8722	0.3829	0.9840
VGG19	93.48	0.9368	0.9348	0.9351	0.2975	0.9943

CNN models without the LSTM layer (32 units, 0.001 learning rate) was analyzed, showing noticeable differences compared to the LSTM-enhanced versions in Table 6. DenseNet201 achieved the highest accuracy of 92.49%, maintaining its strong classification ability with a high ROC-AUC score of 0.9937. MobileNetV2 followed closely with an accuracy of 91.87%, demonstrating its efficiency in identifying driver behaviors. InceptionV3 and InceptionResNetV2 recorded accuracies of 90.88% and 89.74%, respectively, indicating competitive performance but slightly lower effectiveness than the top-performing models. VGG19 achieved an accuracy of 87.33%, while ResNet50 showed the weakest performance at 71.49%, with significantly lower recall and F1-score, suggesting difficulties in correctly classifying driver behaviors. The high Log Loss value of 0.7558 for ResNet50 further indicated greater uncertainty in its predictions.

TABLE 6. Evaluation Results for CNNs without LSTM: Batch Size 32, Learning Rate 0.001

Models	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Log_Loss**	**ROC_AUC**
DenseNet201	92.49	0.9275	0.9249	0.9250	0.2741	0.9937
InceptionResNetV2	89.74	0.9047	0.8974	0.8964	0.2994	0.9904
InceptionV3	90.88	0.9111	0.9088	0.9090	0.3567	0.9897
MobileNetV2	91.87	0.9223	0.9187	0.9176	0.3149	0.9930
ResNet50	71.49	0.7592	0.7149	0.7031	0.7558	0.9416
VGG19	87.33	0.8824	0.8733	0.8731	0.3586	0.9850

The CNN models was evaluated with the inclusion of an LSTM layer (16 units, 0.0001 learning rate), which led to notable improvements across most models. DenseNet201 emerged as the top performer, achieving an accuracy of 94.77% and a high ROC-AUC score of 0.9964, indicating its exceptional capability to classify driver behaviors. InceptionResNetV2 also demonstrated strong performance with an accuracy of 94.15%, supported by a robust ROC-AUC of 0.9953. Table 7 presents that MobileNetV2, despite being a lightweight architecture, secured an impressive accuracy of 94.01% and displayed a high precision of 0.9407, showing its efficiency for real-time applications. Similarly, InceptionV3 achieved an accuracy of 92.82%, maintaining solid performance but slightly lagging behind the top contenders. VGG19, with an accuracy of 93.08%, performed well, achieving a balanced precision of 0.9325 and a low Log Loss

of 0.2483, which reflected its stable and reliable predictions. However, ResNet50 lagged significantly with the lowest accuracy of 86.32%, and its higher Log Loss of 0.3704 indicated more uncertainty in its predictions, marking it as less effective for this particular classification task.

Table 7. Evaluation Results for CNNs with LSTM: Batch Size 16, Learning Rate 0.0001

Models	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Log_Loss**	**ROC_AUC**
DenseNet201	94.77	0.9481	0.9477	0.9478	0.2520	0.9964
InceptionResNetV2	94.15	0.9423	0.9415	0.9414	0.2545	0.9953
InceptionV3	92.82	0.9283	0.9282	0.9279	0.3278	0.9941
MobileNetV2	94.01	0.9407	0.9401	0.9402	0.2853	0.9960
ResNet50	86.32	0.8734	0.8632	0.8626	0.3704	0.9836
VGG19	93.08	0.9325	0.9308	0.9310	0.2483	0.9944

The performance of the CNN models without the LSTM layer (16 units, 0.0001 learning rate) varied across the models. MobileNetV2 achieved the highest accuracy of 93.43%, with strong precision (0.9362) and recall (0.9343). DenseNet201 closely followed with an accuracy of 93.13% and a high ROC-AUC score of 0.9928. InceptionV3 achieved 90.57% accuracy, with solid precision and recall, while InceptionResNetV2 had an accuracy of 89.62%, with slightly lower precision and recall. VGG19 reached an accuracy of 83.24%, though its higher Log Loss (0.5590) indicated more uncertainty in its predictions. ResNet50 performed the weakest, with only 62.96% accuracy and lower precision, recall, and ROC-AUC scores. Table 8 shows the mentioned results.

Table 8. Evaluation Results for CNNs without LSTM: Batch Size 16, Learning Rate 0.0001

Models	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Log_Loss**	**ROC_AUC**
DenseNet201	93.13	0.9326	0.9313	0.9312	0.2323	0.9928
InceptionResNetV2	89.62	0.8992	0.8962	0.8949	0.3218	0.9854
InceptionV3	90.57	0.9108	0.9057	0.9056	0.3418	0.9886
MobileNetV2	93.43	0.9362	0.9343	0.9345	0.2401	0.9932
ResNet50	62.96	0.6782	0.6296	0.6159	0.9896	0.8972
VGG19	83.24	0.8344	0.8324	0.8308	0.5590	0.9630

The performance of the CNN models with and without the LSTM layer (32 units, 0.0001 learning rate) showed some differences. With LSTM (Table 9), DenseNet201 achieved the highest accuracy of 94.26%, closely followed by InceptionResNetV2 at 94.24%, both demonstrating excellent precision, recall, and ROC-AUC scores. MobileNetV2 performed well with an accuracy of 93.35%, while InceptionV3 reached an accuracy of 92.08%. VGG19 and ResNet50 showed weaker performance, with accuracies of 92.56% and 86.40%, respectively. Without the LSTM (Table 10), DenseNet201 again performed well with an accuracy of 92.86%, though slightly lower than with LSTM, while MobileNetV2 achieved 93.15%. InceptionV3 performed similarly to its LSTM counterpart with a 91.33% accuracy. InceptionResNetV2 showed a decrease in accuracy to 88.93%, while VGG19 and ResNet50 saw even greater declines, with VGG19 reaching only 79.61% and ResNet50 60.51%, indicating significant performance drops without the LSTM layer.

Table 11 showed the study's accuracy summary reveals, the impact of LSTM layers (16 and 32 units with varying learning rates) on the performance of different CNN models. DenseNet201 consistently achieved the highest accuracy across configurations, with its best performance observed with LSTM 16, 0.001 (93.53%) and LSTM 32,

TABLE 9. Evaluation Results for CNNs with LSTM: Batch Size 32, Learning Rate 0.0001

Models	Accuracy	Precision	Recall	F1-Score	Log_Loss	ROC_AUC
DenseNet201	94.26	0.9431	0.9426	0.9426	0.2525	0.9958
InceptionResNetV2	94.24	0.9427	0.9424	0.9423	0.2424	0.9959
InceptionV3	92.08	0.9208	0.9208	0.9204	0.3245	0.9924
MobileNetV2	93.35	0.9340	0.9335	0.9336	0.3033	0.9947
ResNet50	86.40	0.8744	0.8640	0.8638	0.3808	0.9819
VGG19	92.56	0.9276	0.9256	0.9257	0.2531	0.9940

TABLE 10. Evaluation Results for CNNs without LSTM: Batch Size 32, Learning Rate 0.0001

Models	Accuracy	Precision	Recall	F1-Score	Log_Loss	ROC_AUC
DenseNet201	92.86	0.9296	0.9286	0.9284	0.2262	0.9925
InceptionResNetV2	88.93	0.8921	0.8893	0.8887	0.3719	0.9807
InceptionV3	91.33	0.9138	0.9133	0.9126	0.2821	0.9896
MobileNetV2	93.15	0.9325	0.9315	0.9315	0.2201	0.9930
ResNet50	60.51	0.6280	0.6051	0.5812	1.0783	0.8710
VGG19	79.61	0.7968	0.7961	0.7904	0.6641	0.9510

0.0001 (94.77%). InceptionResNetV2 showed a significant boost when LSTM was included, especially with LSTM 32, 0.0001, reaching 94.24%, compared to 89.74% without LSTM 16, 0.001. InceptionV3 demonstrated a relatively stable performance, with its highest accuracy of 92.82% achieved with LSTM 32, 0.0001, while its accuracy ranged between 89.22% and 91.33% across other configurations. MobileNetV2 also showed robust results, reaching an accuracy of 93.82% with LSTM 16, 0.001 and 94.01% with LSTM 32, 0.0001, which outperformed its performance without LSTM (91.87% to 93.15%). ResNet50 consistently showed the lowest accuracy, with a peak of 87.78% when paired with LSTM 16, 0.001, but dropping to 60.51% without LSTM 32, 0.0001. VGG19 followed a similar trend, showing higher accuracy with LSTM, particularly with LSTM 32, 0.0001 (93.08%), but lower performance without LSTM, ranging from 87.33% to 79.61%.

The highest accuracies for the models with and without LSTM were observed as follows. With LSTM 16, 0.001, DenseNet201 achieved the highest accuracy of 93.53%, and with LSTM 32, 0.0001, DenseNet201 again reached the highest accuracy of 94.77%. Without the LSTM layers, MobileNetV2 delivered the highest accuracy of 93.43% with LSTM 16, 0.001, while DenseNet201 achieved the highest accuracy of 94.26% with LSTM 32, 0.0001.

Graphs presenting comparison accuracy across different architectures with and without LSTM and different batch sizes (16, 32) with learning rates (0.001, 0.0001) were shown in Figure 8 (a-d).

5. Discussion and Conclusion

This study comprehensively evaluated the performance of various CNN architectures, both with and without the integration of LSTM layers, for the classification of driver behaviors. The primary objective was to assess the impact of incorporating LSTM on enhancing the accuracy and robustness of driver behavior classification models, which is crucial for developing intelligent driver monitoring systems aimed at improving road safety. The experimental results clearly demonstrate the significant contribution of LSTM layers to the performance of CNN models. The CNN models employed in this study are primarily DenseNet201, InceptionResNetV2, InceptionV3, MobileNetV2, ResNet50, and

TABLE 11. Accuracy Summary of CNN Models with and without LSTM Integration

Models	With LSTM				Without LSTM			
	16		32		16		32	
	0.001	0.0001	0.001	0.0001	0.001	0.0001	0.001	0.0001
DenseNet201	**93.53**	**94.77**	**94.02**	**94.26**	**91.44**	93.13	**92.49**	92.86
InceptionResNetV2	91.94	94.15	92.84	94.24	90.79	89.62	89.74	88.93
InceptionV3	90.85	92.82	91.32	92.08	89.22	90.57	90.88	91.33
MobileNetV2	92.70	94.01	93.82	93.35	91.05	**93.43**	91.87	**93.15**
ResNet50	87.78	86.32	87.24	86.40	73.96	62.96	71.49	60.51
VGG19	92.79	93.08	93.48	92.56	87.48	83.24	87.33	79.61

(a) (b)

(c) (d)

FIGURE 8. Impact of LSTM on Model Accuracy Across Different Architectures

VGG19 which provide satisfactory results in classifying images, thus were leveraged. The selection of these architectures was made based on their validated ability to be effective in image classification tasks, and further they were employed as tools for distinguishing various driving behaviors by means of deep feature extraction. Available drivers' images representing different activities were used to train and test the models for a thorough evaluation of classification

performance. The classification task was to identify five specific driver behaviors which are: Safe Driving, Turning, Texting on the Phone, Talking on the Phone, and Other Activities. Each of these behaviors is characterized by a different level of risk, which makes accurate detection necessary for the creation of intelligent driver-monitoring systems. The performance of each CNN model was evaluated using multiple performance metrics, including but not limited to Accuracy, Precision, Recall, F1-Score, Log Loss, and the ROC-AUC to examine the drivers' behavior forecasting quality of the models. The metrics are intended to deliver a complete understanding of the models' classification capacities and allow for a detailed comparison of strengths and weaknesses of the models in driver behavior prediction. The LSTM architecture has been identified as a way to harness temporal correlations within sequential data, its exceptional delivery of accurate prediction of the whole range of models that have been tested demonstrates how vital LSTM was for the classifying accuracy of the models. Temporal patterns that show driver's behaviors as the time dimension e.g. texting, turning, or talking on the phone are developed. LSTM layers were added to let such temporal dynamism be mastered and thus enhance the efficiency of the models. The accuracy improvements observed after adding LSTM layers were substantial. DenseNet201, which constantly proved to be the best-performing model, scored 94.77% accuracy with LSTM (32 batch size, learning rate 0.0001) versus 92.86% without LSTM having similar conditions. This pattern was corroborated by other models as well. For example, InceptionResNetV2's accuracy grew from 88.93% (without LSTM) to 94.24% (with LSTM), and MobileNetV2's performance leaped from 93.15% to 94.01% following the LSTM addition. The research also looked into how the variation in the batch sizes (16 and 32) as well as learning rates (0.001 and 0.0001) affected the performance of the models. Results were that combining LSTM levels with a little (16) and a slow (0.0001) batch size were the best combinations. In addition, one explanation for such a standout result is the finer gradient updates and more frequent weight adjustments during training time can be achieved with the combination of the LSTM layers and smaller batch sizes. This is where the information that users have to put across feedback through smaller batch sizes is taken into account. Also, they needed to be 1 or 2 in number and the learning rate to be 0.0001. For the duration of this study, mobile phone users texting Gestures and ResNet50 could not be used in the same connotation. Its best result among all such models was 87.78% which is hugely lower than those of DenseNet201 and MobileNetV2. It's possible that certain parts of ResNet50's structure may not be flexible enough to correctly determine slight temporal variance patterns in driver behavior data thus, classifying it in the same manner as the others. To conclude, this research pointed out how critical LSTM applied to CNN-based models to enhance the performance of driver behavior classification is. The LSTM functionalities which can correlate with time on the data are a major contributing factor to the advanced accuracy of classifications and it means that it is a must-have element for smart driver monitoring systems. Out of all the models assessed, DenseNet201 was able to get a description of being any of the consistent performers making it an interesting selection for real-world applications. It was noticed trough the results that using different deep learning methodologies combined with temporal modeling techniques could be a way of automated detection of driver behavior in safety through the roads realization. For real-time driver behavior monitoring systems, the MobileNetV2 + LSTM model is recommended due to its low computational demand and high classification accuracy.

Data Availability

The dataset can be accessed through the following link: https://data.mendeley.com/datasets/mzb4b6dff3/1

Conflicts of Interest

It is stated that the authors have no known financial conflicts of interest or personal relationships that could have influenced the work reported here.

Ethical Approval

The data used in this paper is a public dataset.

Funding

No funding was received for this study.

References

[1] Al Ali, F.M., Alnuaimi, M.J., Alawadhi, S. A., Abdallah, S., *Abnormal Driver Behavior Detection Using Deep Learning* in 2024 7th International Conference on Signal Processing and Information Security (ICSPIS), IEEE, (2024), 1-4.

[2] Al doori, S.K.S., Taspinar, Y.S., Koklu, M., *Distracted driving detection with machine learning methods by cnn based feature extraction*, International Journal of Applied Mathematics Electronics and Computers, **9**(4)(2021), 116-121.

[3] Alzebari, N.A.M., Becerikli, Y., *Driver behavior detection using intelligent algorithms*, Journal of Millimeter-wave Communication, Optimization and Modelling, **4**(2)(2024), 39-51.

[4] Butt, F.M., Hussain, L., Mahmood, A., Lone, K.J., *Artificial Intelligence based accurately load forecasting system to forecast short and medium-term load demands*, Mathematical Biosciences and Engineering, **18**(1)(2021), 400-425.

[5] Cengel, T.A., Gencturk, B., Yasin, E.T., Yildiz, M.B., Cinar, I. et al., *Apple (Malus domestica) Quality Evaluation Based on Analysis of Features Using Machine Learning Techniques*, Applied Fruit Science, **66**(2024), 2123–2133.

[6] Cengel, T.A., Gencturk, B., Yasin, E.T., Yildiz, M.B., Cinar, I. et al., *Automating egg damage detection for improved quality control in the food industry using deep learning*, Journal of Food Science, **90**(1)(2025), e17553.

[7] Cengel, T.A., Gencturk, B., Yasin, E.T., Yildiz, M.B., Cinar, I. et al., *Classification of Orange Features for Quality Assessment Using Machine Learning Methods*, Selcuk Journal of Agriculture & Food Sciences/Selcuk Tarim ve Gida Bilimleri Dergisi, **38**(3)(2024).

[8] Chen, J.-C., Lee, C.-Y., Huang, P.-Y., Lin, C.-R., *Driver behavior analysis via two-stream deep convolutional neural network*, Applied Sciences, **10**(6)(2020), 1-14.

[9] Cinar, I., Kaya, F.F., *Application of ConvNeXt Models for Indian Spices Classification*, in Proceedings of International Conference, Abu Dhabi, BAE, (2024), 36-47.

[10] Erdem, K., Yasin, E., Yıldız, M.B., Koklu, M., *Classification of Heart Diseases with Ensemble Learning Algorithms*, Sinop Üniversitesi Fen Bilimleri Dergisi, **9**(2)(2024), 369-387.

[11] Gencer, K., Gencer, G., Cizmeci, İ.H., *Deep learning approaches for retinal image classification: a comparative study of GoogLeNet and ResNet architectures*, International Scientific and Vocational Studies Journal, **8**(2)(2024), 123-128.

[12] Gong, Y., Lu, J., Liu, W., Li, Z., Jiang, X. et al., *Sifdrivenet: Speed and image fusion for driving behavior classification network*, IEEE Transactions on Computational Social Systems, **11**(1)(2023).

[13] He, K., Zhang, X., Ren, S.,d Sun, J., *Deep residual learning for image recognition*, in Proceedings of the IEEE conference on computer vision and pattern recognition, (2016), 770-778.

[14] Howard, A.G., *Mobilenets: Efficient convolutional neural networks for mobile vision applications*, arXiv preprint arXiv:1704.04861, (2017).

[15] Huang, C., Wang, X., Cao, J., Wang, S., Zhang, Y., *HCF: A hybrid CNN framework for behavior detection of distracted drivers*, IEEE access, **8**(2020), 109335-109349.

[16] Huang, G., Liu, Z., Pleiss, G., Van Der Maaten, L., Weinberger, K.Q., *Convolutional networks with dense connectivity*, IEEE transactions on pattern analysis and machine intelligence, **44**(12)(2019), 8704-8716.

[17] Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q., *Densely connected convolutional networks*, in Proceedings of the IEEE conference on computer vision and pattern recognition, (2017), 4700-4708.

[18] Huang, T., Fu, R., Chen, Y., *Deep driver behavior detection model based on human brain consolidated learning for shared autonomy systems*, Measurement, **179**(2021), 109463.

[19] Huang, W., Liu, X., Luo, M., Zhang, P., Wang, W., Wang, J., *Video-based abnormal driving behavior detection via deep learning fusions*, IEEE Access, **7**(2019), 64571-64582.

[20] Isik, H., Tasdemir, S., Taspinar, Y.S., Kursun, R., Cinar, I. et al., *Maize seeds forecasting with hybrid directional and bi-directional long short-term memory models*, Food Science & Nutrition, **12**(2)(2024), 786-803.

[21] Kang, H., Zhang, C., Jiang, H., *Advancing Driver Behavior Recognition: An Intelligent Approach Utilizing ResNet*, Automatic Control and Computer Sciences, **58**(5)(2024), 555-568.

[22] Koklu, M., Cinar, I., Taspinar, Y.S., *CNN-based bi-directional and directional long-short term memory network for determination of face mask*, Biomedical signal processing and control, **71**(2022), 103216.

[23] Koklu, M., Kursun, R., Taspinar, Y.S., Cinar, I., *Classification of date fruits into genetic varieties using image analysis*, Mathematical Problems in Engineering, **1**(2021), 4793293.

[24] Koklu, N., Sulak, S.A., *The Systematic Analysis of Adults' Environmental Sensory Tendencies Dataset*, Data in Brief, **55**(2024), 110640.

[25] Koklu, N., Sulak, S.A., *Using Artificial Intelligence Techniques for the Analysis of Obesity Status According to the Individuals' Social and Physical Activities*, Sinop Üniversitesi Fen Bilimleri Dergisi, **9**(1)(2024), 217-239.

[26] Lai, Z., *Driver Behavior and Action Prediction in Human-Computer Interaction*, in 2024 3rd International Conference on Artificial Intelligence, Internet of Things and Cloud Computing Technology (AIoTC), IEEE, (2024), 97-101.

[27] Sahin Afridi, A., Kafy, A., Nessa Moon, M. N., Shakil, M.S., *Multi-Class Driver Behavior Image Dataset*, Mendeley Data, (2024).

[28] Seo, H., Hwang, J., Jeong, T., Shin, J., *Comparison of deep learning models for cervical vertebral maturation stage classification on lateral cephalometric radiographs*, *Journal of Clinical Medicine*, **10**(16)(2021), 3591.

[29] Simonyan, K., Zisserman, A., *Very deep convolutional networks for large-scale image recognition*, arXiv preprint arXiv:1409.1556 (2014).

[30] Sinha, D.,El-Sharkawy, M., *Thin mobilenet: An enhanced mobilenet architecture*, in 2019 IEEE 10th annual ubiquitous computing, electronics & mobile communication conference (UEMCON), IEEE, (2019) , 0280-0285.

[31] Sulak, S.A., Koklu, N., *Analysis of Depression, Anxiety, Stress Scale (DASS-42) With Methods of Data Mining*, European Journal of Education, **59**(4)(2024), e12778.

[32] Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A., *Inception-v4, inception-resnet and the impact of residual connections on learning*, in Proceedings of the AAAI conference on artificial intelligence, **31**(1)(2017).

[33] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. et al., *Going deeper with convolutions*, in Proceedings of the IEEE conference on computer vision and pattern recognition, (2015), 1-9.

[34] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z., *Rethinking the inception architecture for computer vision*, in Proceedings of the IEEE conference on computer vision and pattern recognition, (2016), 2818-2826.

[35] Taspinar, Y.S., Koklu, M., Altin, M., *Acoustic-driven airflow flame extinguishing system design and analysis of capabilities of low frequency in different fuels*, Fire technology, **58**(3)(2022), 1579-1597.

[36] Taspinar, Y.S., Selek, M., *Object recognition with hybrid deep learning methods and testing on embedded systems*, International Journal of Intelligent Systems and Applications in Engineering, (2020).

[37] Tutuncu, K., Cinar, I., Kursun, R., Koklu, M., *Edible and poisonous mushrooms classification by machine learning algorithms*, in 2022 11th Mediterranean Conference on Embedded Computing (MECO), Budva, Montenegro, 07-10 June 2022: IEEE, (2022), 1-4.

[38] Wen, L., Li, X., Li, X., Gao, L., *A new transfer learning based on VGG-19 network for fault diagnosis*, in 2019 IEEE 23rd international conference on computer supported cooperative work in design (CSCWD), IEEE, (2019), 205-209.

[39] WHO. *Road traffic injuries*. https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries (accessed 20.01.2025, 2025).

[40] Xiao, W., Xie, G., Liu, H., Chen, W., Li, R., *FDAN: Fuzzy deep attention networks for driver behavior recognition*, Journal of Systems Architecture, **147**(2024), 103063.

[41] Xie, L., Xiang, X., Xu, H., Wang, L., Lin, L.et al., *FFCNN: A deep neural network for surface defect detection of magnetic tile*, IEEE Transactions on Industrial Electronics, **68**(4)(2020), 3506-3516.

[42] Yan, S., Teng, Y., Smith, J.S., Zhang, B., *Driver behavior recognition based on deep convolutional neural networks*, in 2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), IEEE, (2016), 636-641.

[43] Yasin, E., Koklu, M., *A comparative analysis of machine learning algorithms for waste classification: inceptionv3 and chi-square features*, International Journal of Environmental Science and Technology **22**(2024), 9415–9428.

[44] Yurttakal, A.H., Erbay, H., Çinarer, G., Bas, H., *Classification of Diabetic Rat Histopathology Images Using Convolutional Neural Networks*, International Journal of Computational Intelligence Systems, **14**(1)(2021), 715-722.

[45] Zhang, C., Li, R., Kim, W., Yoon, D., Patras, P., *Driver behavior recognition via interwoven deep convolutional neural nets with multi-stream inputs*, IEEE Access, **8**(2020), 191138-191151.

[46] Zhao, L., Yang, F., Bu, L., Han, S., Zhang, G. et al., *Driver behavior detection via adaptive spatial attention mechanism*, Advanced Engineering Informatics, **48**(2021), 101280.