sensors

Article

E2DR: A Deep Learning Ensemble-Based Driver Distraction Detection with Recommendations Model

Mustafa Aljasim and Rasha Kashef *

Electrical, Computer, and Biomedical Engineering, Ryerson University, Toronto, ON M5B 2K3, Canada; mustafa.aljasim@ryerson.ca
* Correspondence: rkashef@ryerson.ca

Abstract: The increasing number of car accidents is a significant issue in current transportation systems. According to the World Health Organization (WHO), road accidents are the eighth highest top cause of death around the world. More than 80% of road accidents are caused by distracted driving, such as using a mobile phone, talking to passengers, and smoking. A lot of efforts have been made to tackle the problem of driver distraction; however, no optimal solution is provided. A practical approach to solving this problem is implementing quantitative measures for driver activities and designing a classification system that detects distracting actions. In this paper, we have implemented a portfolio of various ensemble deep learning models that have been proven to efficiently classify driver distracted actions and provide an in-car recommendation to minimize the level of distractions and increase in-car awareness for improved safety. This paper proposes E2DR, a new scalable model that uses stacking ensemble methods to combine two or more deep learning models to improve accuracy, enhance generalization, and reduce overfitting, with real-time recommendations. The highest performing E2DR variant, which included the ResNet50 and VGG16 models, achieved a test accuracy of 92% as applied to state-of-the-art datasets, including the State Farm Distracted Drivers dataset, using novel data splitting strategies.

Keywords: deep learning; stacking; ensemble learning; distracted driving

check for updates

Citation: Aljasim, M.; Kashef, R. E2DR: A Deep Learning Ensemble-Based Driver Distraction Detection with Recommendations Model. *Sensors* **2022**, *22*, 1858. https://doi.org/10.3390/s22051858

Academic Editors: Andrei Velichko, Dmitry Korzun and Alexander Meigal

Received: 25 December 2021
Accepted: 23 February 2022
Published: 26 February 2022

1. Introduction

With the continuous growth of the population, new technologies and transportation methods need to emerge to serve people effectively and efficiently. Building an efficient and safe transportation system can positively affect economies, the environment, and human mental and physical health. The increasing number of accidents is a major issue in our current transportation system. According to the World Health Organization, road accidents are the eighth highest reason for death worldwide [1]. According to [2], 1.35 million people die every year in a car accident, and up to 50 million people are injured. This makes traffic safety a major concern worldwide. More than 80% of road accidents are caused by distracted driving, such as using a mobile phone, talking to passengers, and smoking [2]. Therefore, more attention has been directed to driver action analysis and monitoring. Many efforts have been made to tackle the problem of driver distraction with effective approaches using countermeasures for distracted driver actions [3]. The measures can be divided into three categories: (1) distraction prevention before distraction occurs, (2) distracting action detection (through alertness) after distraction occurs, and (3) collision avoidance when a potential collision is expected [3]. Imposing strict fines, government legislation, and raising public awareness are methods used to decrease the number of accidents caused by distracted driving through preventing distraction sources before it happens. When a potential collision is expected, collision avoidance systems are implemented in most newly manufactured cars through lane control, automatic emergency braking, and forward-collision warning. Distraction alertness is critical and can be more effective in preventing driver distraction; thus, accurate real-time driver action detection methods are essential for this driver distraction category. Distraction alertness can be approached with different modalities. A camera was used to detect distracted driving behavior while driving in many applications. A Controller Area Network Bus can assess the vehicle's performance, such as wheel angle and brake level. Moreover, the system can detect if the driver is distracted or focused on driving based on the in-car collected information. Finally, sensors, such as the electrocardiograph and the electroencephalograph, can be used to estimate the emotional and physiological states of a driver, which can be associated with the level of distraction and fatigue in drivers [4]; however, there is a lack of reflection on the fact that they are invasive sensors for a driver. The driver's action or behavior, such as gaze, head pose, and hand position, can be detected through deep learning models and the analysis of the car information [5]. Existing methods in detecting driver distraction fall short in providing accurate detection and recommendations in real-time. Ensemble learning has shown better classification performance compared to individual models, as it combines the benefits of multiple models while overcoming their drawbacks [6]. While authors in [6] used a fixed architecture of only three deep learning models including the residual network (ResNet), the hierarchical recurrent neural network (HRNN), and the Inception network, there is a research gap in the literature in using a scalable and incremental stacking-based ensemble learning with real-time recommendations to achieve high accuracy in detecting distracted driving activities with minimal computational overhead. Thus, in this paper, we have proposed a novel scalable model that uses ensemble learning, focusing on stacking that combines two or more baseline models and generates an ensemble with better performance than the adopted models. Our method aims to enhance generalization, reduce overfitting, increase performance, and provide real-time recommendations. This paper first examines the performance of several state-of-the-art image deep learning classification methods. An Ensemble-Based Distraction Detection with Recommendations model is designed, namely (E2DR), with the goal of improving the accuracy of distracted behavior detection. In the proposed E2DR model, two or more deep learning models are aggregated in a stacking ensemble. A recommendation layer is also provided for real-time recommendations to drivers in each case of the distracted behaviors to allow drivers or autonomous vehicles to take the best course of action when drivers are detected under distracted behaviors. Experimental results show that state-of-the-art image classification models achieve a test accuracy ranging between 82–88% in detecting driver distraction. Furthermore, results show an average improvement of 5–8% in detection accuracy when the proposed E2DR is used with a real-time data splitting based on the driver IDs. Similar results are obtained for other metrics such as Precision and F1 score. The rest of this paper is as follows: Section 2 discusses deep learning driver distraction detection systems and the related work. Sections 3 and 4 introduce the adopted and proposed models, respectively. Section 5 presents the experimental results and analysis. We conclude the paper with future directions in Section 6.

2. Related Work on Deep Learning Driver Distraction Detection

There are three main types of distracted driving: cognitive, visual, and manual distraction. Cognitive distraction occurs when the driver's mind is not entirely focused on driving. Driver gaze and talking to passengers are examples of cognitive distraction. Even drivers listening to music or the radio are at risk. The audio or music can shift the driver's attention from driving and overall surroundings. Visual distraction is when the driver is not looking at the road ahead. Drivers who observe shop signs and billboards on the side of the road are considered visually distracted. Looking at electronic devices such as GPS devices, smartphones, and digital entertainment devices while driving is under the category of visual distraction. Finally, manual distraction occurs when the driver, for any reason, takes their hands off the steering wheel. Drivers who smoke while driving, eat and drink in the car, or try to get something from anywhere in the vehicle are under the risk of manual distraction. Texting while driving is the most dangerous driver distraction as it combines all three types of distractions. When drivers take their eye off the road to send a message or check a notification, it is long enough to cover the length of a football field at 80 km/h [5]. Various research studies have been proposed to address the drivers' distraction detection problem. This section will survey the most recent state-of-the-art research work using Single-based vs. Hybrid-based deep learning to address this problem.

2.1. Single-Based Deep Learning Models

A gaze estimation model called X-Aware is introduced in [7] to analyze the driver's face along with contextual information. The model visually improves the fusion of the captured environment of the driver's face, where the contextual attention mechanism is directly attached to the output of convolutional layers of the InceptionResNetV2 networks. The accuracy of their best model outperformed the other baseline models in the literature. The dynamics of the driver's gaze and their use to understand other attentional mechanisms are addressed in [8]. The model is built based on two questions, where and what is the driver looking at. The model is trained through coarse-to-fine convolutional networks on short sequences from the DR(eye)VE dataset [8]. Their experiments showed that the driver's gaze could be learned to some extent, considering its highly subjective challenges and the scene's irreproducibility showing the driver's gaze for each sequence. The results showed that the model could achieve accurate results and could be integrated into practical applications. In [9], the authors proposed a deep learning model that detects drivers' behavior and actions during travel. The deep learning model classifies the driver actions into ten classes. The first class represents safe driving, and the other nine classes represent unsafe drivers' actions such as fixing makeup and texting. The driver receives an alert if an unsafe action is detected. They used Convolutional Neural Networks (CNNs) to perform training and detection. The core of the deep learning system is ResNet50. A dense net architecture followed the ResNet50 architecture to make classifications. The dataset used is the State Farm dataset and included images of different drivers' actions that cause distracted driving. The model achieved high accuracy in detecting the driver's actions. A facial expression recognition model in [10] monitors drivers' emotions and operates in low specification devices installed in cars. A Hierarchal Weighted Random Forest Classifier (HRFC) is used and trained on the similarity of sample data. Geometric features and facial landmarks are detected and extracted from input images. The features are vectorized and implemented in the Hierarchal Random-Forest Classifier to detect facial expressions. The method was evaluated on the MMI dataset, the Keimyung University Facial Expression of Drivers (KMU-FED) dataset and the Cohn-Kahnde dataset. The results showed that the proposed model had similar performance to other state-of-the-art methods. The study in [11] introduced a computationally efficient distracted driver detection system based on convolutional neural networks. The authors proposed a new architecture called mobileVGG. The architecture is based on depth-wise separable convolutions. The authors used a simplified version of the VGG16 model, allowing the proposed architecture to be suitable for real-time applications with decent classification accuracy. The datasets used are the State Farm dataset and the American University in Cairo Distracted Driver (AUCDD) detection dataset. The results showed that the proposed architecture outperformed other approaches while being computationally simple. The driver's face pose is detected by training CNNs [12]; the CNNs then identify if the driver's head position is considered under the category of distracted driving. The model consists of five CNNs followed by three fully connected layers. The results showed that the proposed model has better accuracy when compared to non-linear and linear embedding algorithms. In [13], the authors proposed a driver action recognition system called dilated and deformable Faster Region-Based Convolutional Neural Networks (R-CNN). It detects driver actions by detecting motion-specific objects exhibiting inter-class similarity and intra-class differences. The irregular and small features, such as cell phones and cigarettes, are extracted through the dilated and deformable residual block. Then, the region proposal optimization network algorithm decreases the number of features and improves the model's efficiency. Finally, the feature pooling module is replaced with a deformable one, and the R-CNN network is trained as the classifier of the network. The authors established the dataset and contained images of different driver actions. Results showed that the model demonstrates acceptable results. Authors in [14] implemented a driver distraction detection model that uses a light-weight octave-like convolutional neural network. The network consists of octave-like convolutional blocks called OLCMNet. The OCLM block splits the feature map into two branches through point-wise convolution. Average pooling and depth-wise convolution are performed on the feature map. A DC operator captures the fine details in the high-frequency branches. Lastly, the OCLMNet exchanges further information between layers. The model performed well on the Lilong Distracted Driving Behavior dataset while being implemented on a limited computation budget. A unique approach is proposed in [15] that uses both spatial and temporal information of electroencephalography (EEG) signals as an input to a deep learning model. The relationship between the driver distraction and the EEG signal in the time domain is mapped through gated recurrent units (GRUs) and CNNs. Twenty-four volunteers were tested while doing activities that cause a distraction while driving, and their EEG response was recorded. Then, the proposed deep learning network was trained based on the EEG information. The deep learning approach consisted of a temporal–spatial information network (TSIN), combining CNNs and GRUs to better detect spatial and temporal features from EEG signals. The authors of [16] proposed a modifier deep learning approach for distraction detection. They used the OpenPose library for a two-category problem of distraction detection. The library draws 43 points on the facial skeleton to detect the human face. The detection is sent to a deep neural network that uses the ResNet50 model. The results demonstrated good accuracy and outperformed other residual network architectures. The work in [17] introduced a new approach to distracted driver detection using wearable sensing and deep neural networks. The study included information from twenty participants through wearable motion sensors attached to their wrists. The participants performed five distraction activities under instructions in a driving simulator. The captured data were sent to a deep learning model that consisted of recurrent neural networks and long short-term memory (RNN-LSTM) which classified distraction tasks. The results showed a good potential for the wearable proposed sensing approach.

2.2. Hybrid-Based Deep Learning Models

A hybrid model is designed in [18] with an ensemble of weighted CNNs for driver posture classification. The authors proved that using a weighted ensemble classifier using the genetic algorithm resulted in a better confidence score for classification. Additionally, the effect of variable visual elements is analyzed, such as face and hands, in detecting distracted driver action through localization of face and hands. The dataset used is the Distracted Driver dataset, which contains ten classes of driver actions. The best model has an accuracy of 96%, and a smaller version of the ensemble model achieved an accuracy of 94.29%. In [19], a distracted driver detection technique using pose estimation is introduced. The model is an ensemble of ResNets and classifies drivers through pose estimation images. ResNet and HRnet are used to generate pose images. Then, ResNet50 and ResNet101 classify the original and pose estimation images. The grid search method identifies the optimum weight for predictions from both models. Classifying pose estimation images is useful when used with the original image classification model as it increases classification accuracy. The dataset used is the AUCDD dataset. The results showed that the introduced model achieved an accuracy of 94.28%. The study in [20] detected driver–vehicle volatilities using driving data to detect the occurrence of critical events and give appropriate feedback to drivers and surrounding vehicles through analyzing multiple real-time data streams such as vehicle movements, instability of driving, and driver distraction. The deep learning model consisted of a Convolutional Neural Network (CNN) model and a long short-term memory (LSTM) model. The data were collected from more than 3500 drivers and included 1315 severe and 7566 normal events. The model achieved high accuracy and was effective in detecting accidents. A CNNs-based model to identify driver's activities is introduced in [21]. The driving activities are divided into several classes, in which 4 are considered normal driving activities, and the other three are classified as distracted driving. The Gaussian mixture model detects the driver's body from the background before sending the image to the CNN model. The authors used transfer learning to fine-tune three pre-trained state-of-the-art CNN models: Restnet50, AlexNet, and GoogLeNet. The model was trained as a binary classifier to detect whether the driver was distracted or not. The authors collected the data from 10 drivers involved in the most common driving activities. The results showed that the model was effective as a binary classifier. In [22], a model for distracted driving action recognition is proposed using a hybrid of two convolutional neural network architectures, Xception and Inception V3, to detect 10 classes of driver actions. The authors used ImageNet weights for transfer learning. The performance of both models was analyzed under different weighting schemes. Using pre-trained weights helped the network learn basic shapes and edges without starting training from scratch, which allowed the model to achieve good results in under 10 epochs of training, applied to the State Farm Distracted Driver dataset. The results showed that the Inception model had a better performance compared to the Xception model. A distracted driver action recognition system is introduced in [23] based on the Discriminative Scale Space Tracking (DSST) and Deep Predictive Coding Network (PCN) algorithms, dynamic face tracking, location, and face detection. Then, the YOLOv3 object detection model detects distracting behavior around the driver's face, such as phone calls and smoking. The dataset used is a self-built dataset of people making phone calls and smoking. The results demonstrated that the model could detect a driver's behavior with high accuracy. A hybrid driver distraction detection model is presented in [24]. The model uses CNNs and Bidirectional Long Short-Term Memory (BiLSTM). The proposed model captures the spatio-spectral features of the images and consists of two steps: (1) detect the spatial features of different postures using CNNs automatically, and (2) the spectral components from the stacked feature map are extracted through the BiLSTMs. They used the AUCDD dataset. Results showed that their model performed better than most state-of-the-art models. The work in [25] used deep learning to detect driver inattentive and aggressive behavior. They classified inattentive driver behavior into driver fatigue, downiness, driver distraction, and other risky driver behavior such as driving aggressiveness. All these risky driving behaviors are associated with various factors that include driving age, experience, illness, and gender. The authors used CNNs, RNNs and LSTMs. They showed that the CNNs achieved the best performance. The algorithm in [26] detects driver manual distraction using two modules; in the first module, the bounding boxes of the driver's right ear and right hand are detected from RGB images through YOLO, a deep learning object detection model. Then, the bounding boxes are taken as an input by the second module, a multi-layer perceptron, to predict the distraction type. The dataset consisted of 106,677 frames extracted from a video obtained from 20 participants in a driving simulator. The proposed algorithm achieved comparable results with other models in the same field. Table 1 provides a comparative study of the recent work in driver distraction detection systems. There is a research gap in the literature in using stacking-based ensemble learning to achieve high accuracy in detecting distracted driving activities with minimal computational overhead. Thus, in this paper, we have proposed a framework that uses ensemble learning, focusing on stacking that combines two baseline models and generates an ensemble with better performance than the adopted models.

Table 1. Comparative Analysis of Driver Distraction detection systems.

Paper	Model (Type)	Dataset	Validation	Pros	Cons
[7]	Deep learning Gaze estimation system	Driver Gaze in the Wild dataset	Accuracy	High performance	It can be only accurate to an extent
[8]	Deep learning Gaze estimation driver-assistant system	DR(eye)VE dataset	Ground truth	Provide suggestion	Driver gaze is subjective
[9]	Deep learning distracted driver detection	Distracted driver dataset	Accuracy, Recall, Precision, F1 score	Computationally efficient	Few epochs for training
[10]	Hierarchical, weighted random forest (WRF) model	The Keimyung University Facial Expression of Drivers (KMU-FED) and the Cohn-Kahnde datasets	Accuracy	Requires low amount of memory and computing operations	Not accurate when the face is rotated
[11]	Driver distraction detection using CNNs	State farm dataset and the AUC distracted driver dataset	Accuracy, sensitivity	Computationally efficient	Not enough validation metrics
[12]	Deep learning distracted driver detection using pose estimation	AUC Distracted Driver Dataset	Accuracy, Fl score	Pose estimation improves accuracy	Low-resolution images affected training
[13]	Driver action recognition using R-CNN	images of different driver actions	Accuracy and log loss	Effective feature representation	Small dataset
[14]	Driver Distraction recognition Using Octave-Like CNN	Lilong Distracted Driving Behavior data	Accuracy, training duration	Lightweight network	Not enough validation metrics
[15]	Temporal–Spatial Deep Learning driver distraction detection	EEG signals from 24 participants	Precision, Recall, F1 score	Unique approach	Drivers' individual differences need to be considered
[16]	Optimized Residual Neural Network Architecture	The State Farm Distracted Driver dataset	Accuracy, training time	Enhanced model	Only detects head movement
[17]	Wearable sensing and deep learning driver distraction detection	Wearable sensing information from 20 participants	Recall, Precision, F1 score	Good potential	Small dataset
[18]	Hybrid Distraction detection model using deep learning	State farm dataset	Accuracy	Computationally expensive	Not enough validation
[19]	Triple-Wise Multi-Task Learning	AUC Distracted Driver Dataset	Accuracy, sensitivity	High detection accuracy	High computational cost
[20]	Safety-critical events prediction	Driving events from 3500 drivers	Accuracy, Recall, Precision, F1 score	Can detect potential car accidents	Hard to get enough data
[21]	CNN driver action detection system	10 drivers' data with driving activities	Accuracy	Accurate	It does not detect the driver action
[22]	CNN driver action detection system	Distracted driver dataset	Accuracy and loss	Computationally simple	Not enough training
[23]	Distracted driver behavior detection using deep learning	Self-built dataset of drivers making phone calls and smoking	Recall, Precision, Speed	Real-time	Only trained to detect 2 driver actions
[24]	hybrid driver distraction detection model	(AUC) Distracted Driver Dataset	Accuracy and loss	High Performance	Complex
[25]	Driver Inattentiveness detection	NA	Accuracy	Comprehensive analysis of deep learning models	Not effective in detecting aggressive behavior
[26]	Deep learning manual distraction detection model	106,677 frames extracted from a video that was taken from 20 participants in a driving simulator	Accuracy, Recall, Precision, F1 score	Novel approach	Only detects manual distraction

3. Adopted Deep Learning Models

3.1. ResNet50 Model

ResNet50 is a 50-layer deep convolutional neural network (Figure 1) [27]. The pre-trained version of the network can be imported from the ImageNet database [28], trained on over a million photos. The network is trained to classify images into 1,000 different object categories, including pencils, tables, mice, and various animals and objects. As a result, the network has learned a variety of rich feature representations for a large variety of images [29]. In the final classification model, ResNet50 is used as the convolutional base, and the pre-trained model is used to learn the patterns in the data. ResNet50 requires the size of the input images to be 224 × 224 (width, height) [29]. All the experiments were conducted with color images. The dimensions of each image sent to the classification model were (224, 224, 3), where 3 indicates the number of channels in the images. The three channels indicated are color channels composed of Green, blue, and red.

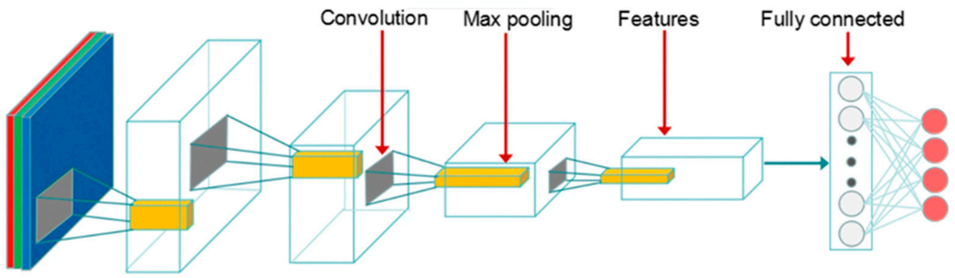


Figure 1. The ResNet50 blocks [27].

3.2. VGG16 Model

Oxford's VGG16 architecture from the Visual Geometry Group (VGG) (Figure 2) has an advantage over AlexNet by replacing large kernel-sized filter 5 and 100 in the second and first convolutional layers, respectively, with several kernel-sized filters of size 3 × 3 one after another [30]. Similar to the ResNet50 model, the input to the network is a 224 × 224 × 3 image, where 3 indicates the RGB channels. The image is passed through a series of convolutional layers with small receptive field filters with a size of 3 × 3 [31]. This filter size is the smallest to capture center, up/down and left/right notions. Additionally, the configuration utilizes 1 × 1 convolutional filters that can be considered a linear transformation of the input channel [32].

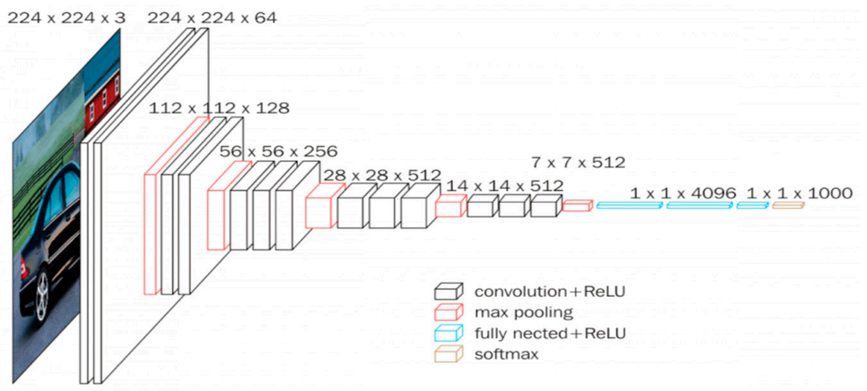


Figure 2. The VGG16 blocks [33].

3.3. Inception Model

The Inception model (Figure 3) is a significant milestone in the evolution of CNN classifiers. Inception changed the traditional approach of adding more and deeper convolutional layers to improve performance [34]. Inception went through several versions and developed with time. Inception V1 [35] uses multiple filters that operate simultaneously, making the network wider rather than deeper. Then, the authors introduced Inception V2 and V3 [36]. Inception V2 significantly improved performance and computational speed by using two 3 × 3 convolutional operations instead of a single 5 × 5 convolution which is 2.78 times more computationally expensive [37]. Inception V3, the model used in the experiment, includes all upgrades in Inception V2 and factorized 7 × 7 convolutions, which improved performance even more, and added Label Smoothing to decrease the chances of overfitting [38]. The main contribution of Inception V4 is adding reduction blocks to change the width and height of the grid [39].

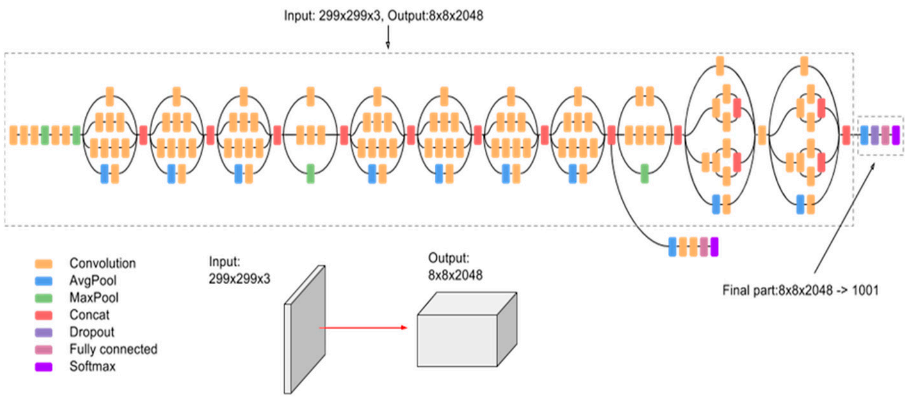


Figure 3. The Inception v3 architecture [39].

3.4. MobileNet Model

MobileNet (Figure 4) is the first mobile computer vision model based on TensorFlow [40]. The name Mobile implies the ability of the model to function in mobile applications [41]. MobileNet is based on separable depth-wise convolutions, which significantly reduces the number of parameters, especially when compared to networks with regular convolutions that have the same depth of the nets. This makes MobileNet a lightweight deep neural network suitable for mobile applications [42]. The depth-wise separable convolution is made from two primary operations: depth-wise convolution and point-wise convolution. The depth-wise convolution came from the idea that the filter's spatial and depth dimensions can be separated. The filter is separated by its height and width dimensions, and then the depth dimension is separated from the horizontal (width×height) dimension. The point-wise convolution is a 1 × 1 convolution that changes the dimension of the previous layer.

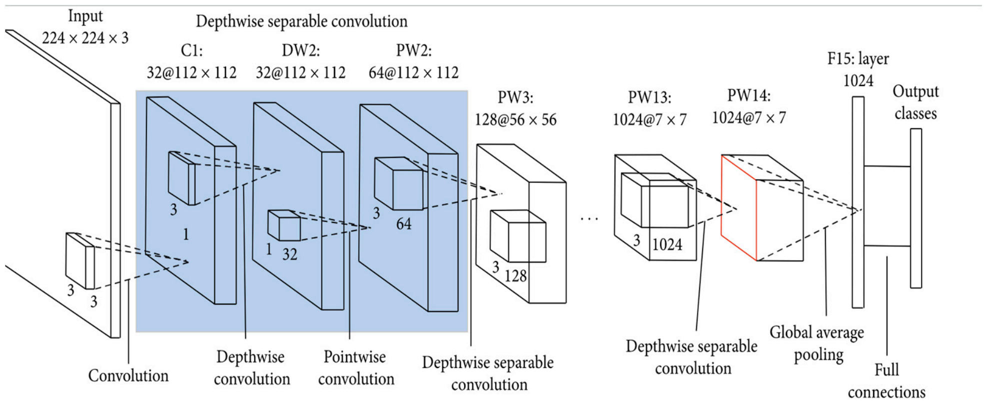


Figure 4. The MobileNet Architecture [43].

4. The Proposed E2DR Model

Existing driver distraction detection systems in the literature only use a single model trained for classification. Moreover, most recent work uses a single state-of-the-art classifier or a network of convolutional neural layers to get the best performance. In this paper, an Ensemble-Based Distraction Detection with Recommendations system is designed, namely

E2DR, to improve the accuracy of driver distraction detection and provide recommendations. In the proposed E2DR model, two deep learning models are aggregated in a stacking ensemble. A recommendation layer is also provided for real-time recommendations in each case of distracted behaviors. The E2DR model enhances the generalization of the detection process and reduces overfitting. The E2DR model allows drivers or autonomous vehicles, depending on the technology in the vehicle, to take the best action when drivers are detected under distracted behaviors.

4.1. The Ensemble-Based Distraction Detection with Recommendations Model (E2DR)

Stacked generalization (SG) was first introduced in [44]; it was shown that stacking reduces the bias of the single model concerning the training set, where bias is the average difference between actual and predicted results [44]. The deduction results from the stacking model's ability to harness the capabilities of more than one well-performing model on a regression or classification task to generate better performance and predictions than base classifiers in the ensemble, which reduces the error and bias. Inspired by the theory of stacking generalization, the E2DR model combines two or more detection models to provide better and more accurate predictions than individual models. The stacking algorithm takes the output of the base model as an input to another model, sometimes called a meta-learner, which learns how to combine the predictions of base models to generate better predictions. In more detail, the stacking architecture has two levels. The first level contains the base models, and the second level includes a meta-learner that concatenates the outcomes of base models to provide final predictions. Figure 5 shows the pair-wise stacking in the E2DR architecture (i.e., only two base models are combined).

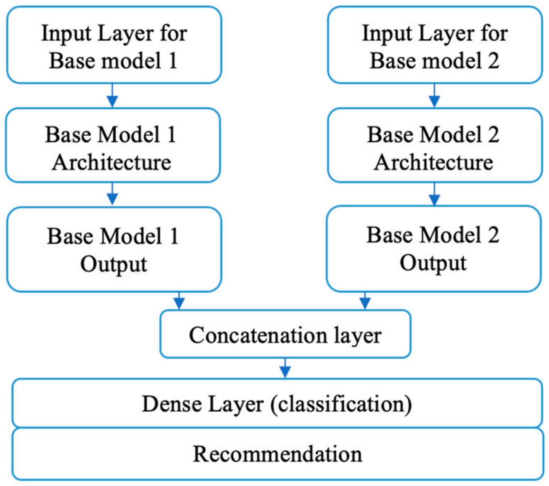

Input Layer for Base model 1
Input Layer for Base model 2
Base Model 1 Architecture
Base Model 2 Architecture
Base Model 1 Output
Base Model 2 Output
Concatenation layer
Dense Layer (classification)
Recommendation

Figure 5. The E2DR (Pair-wise stacking) Architecture.

4.2. E2DR Variants

In this paper, six models are developed using variants of base model 1 and base model 2, such as E2DR (A1, A2) where A1 and A2 $\in$ {ResNet, VGG16, MobileNet, Inception}. The E2DR model combines 2 of the mentioned models using the Stacked Generalization (SG) ensemble method. The SG ensemble method uses the outputs from the pre-trained base models, concatenates them, and sends them to a meta-learner model at level 2 consisting of a dense layer for classification. Once the distracted behavior is classified, a set of recommended actions are provided to ensure safety. We calculate an assessment measure for each E2DR (A1, A2) using Accuracy, Loss, F-measure, Precision, and Recall.

4.3. Computational Complexity

Assume T_{A1} and T_{A2} are the computational time taken by both base models A1 and A2, respectively. Assume the meta learner needs overhead of T_M and the recommendation layer needs O(k) to retrieve recommendations based on the output classification, where k is the number of recommendations. We assume a linear search algorithm for recommendations extraction. The overall computational complexity of the E2DR model is computed as the maximum of T_{A1} and T_{A2} in addition to the overhead of concatenation and recommendation retrieval, as shown in Equation (1).

$$T_{E2DR} = O(Max(T_{A1}, \text{and } T_{A2}) + T_M + O(k) \tag{1}$$

4.4. Adopted Base Model Architectures

All adopted architectures use CNNs, a deep learning model that learns from spatial features of images by creating feature maps using filters and kernels (sliding windows). Many variations of CNNs have been studied recently to detect driving postures and actions. The models adopted in this paper are among the highest-performing CNN models. For all models, we used a learning rate of 0.001 and Categorical Cross Entropy as the loss function. The models are explained in detail in Section 3.

ResNet50 Model: When building the CNN model, the classification top of the ResNet50 Model (Figure 6), which was originally designed to classify 1000 classes [45], was dropped from the network to adapt the new CNN architecture to the dataset used that included only 10 classes. To avoid the problem of overfitting, a drop-out layer was added, and performance on the validation set was observed after every epoch. The hyperparameters, such as batch size, learning rate, and the number of neurons, were adjusted to optimize the model's performance and enhance the model's generalizability. We used different batch sizes: 128, 64, 32, and 16 to understand the significance of selecting the batch size as well as its effect on the network, with 32 as the best performing batch size.

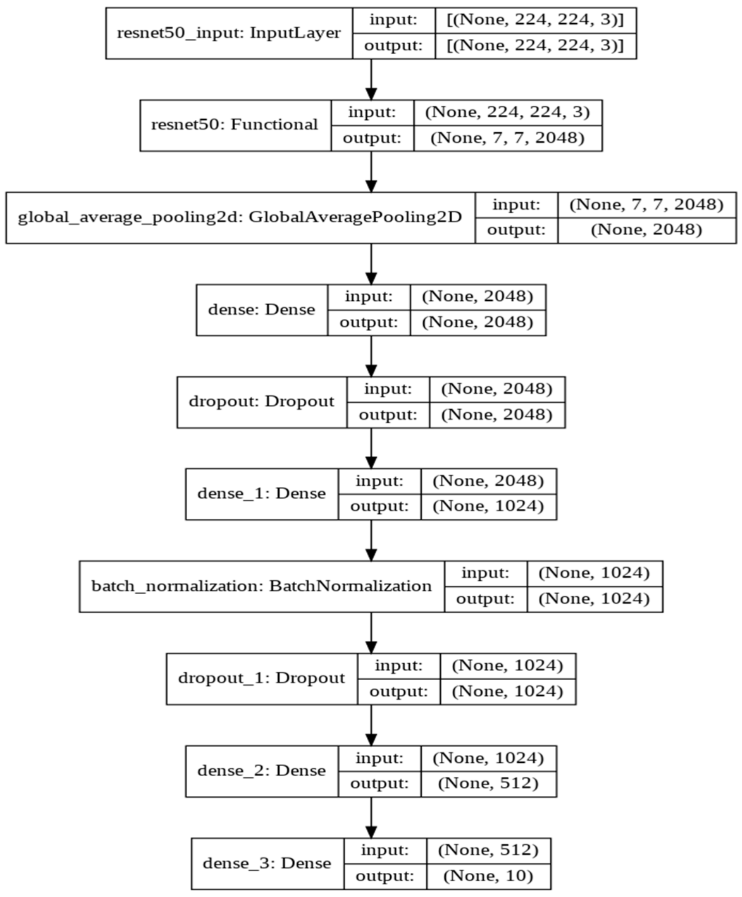


Figure 6. ResNet50 Architecture.

VGG16 Model: The VGG16 model (Figure 7) is among the largest models with many parameters. The padding is 1 pixel for 3 × 3 convolutional layers to preserve the spatial padding after undergoing convolution. The five-max pooling layers carry out the spatial pooling that follows some convolutional layers [46]. The max pooling is applied through a 2 × 2 pixel widow that has a stride value of 2. The same filter size is applied several times, allowing the network to represent complex features [47]. This "blocks" concept became more common after VGG was introduced [46]. As with the ResNet50, the classification layer (top layer) is adapted to the used dataset with 10 classes. The hyperparameter was chosen to optimize performance and training time with a batch size equals to 32.

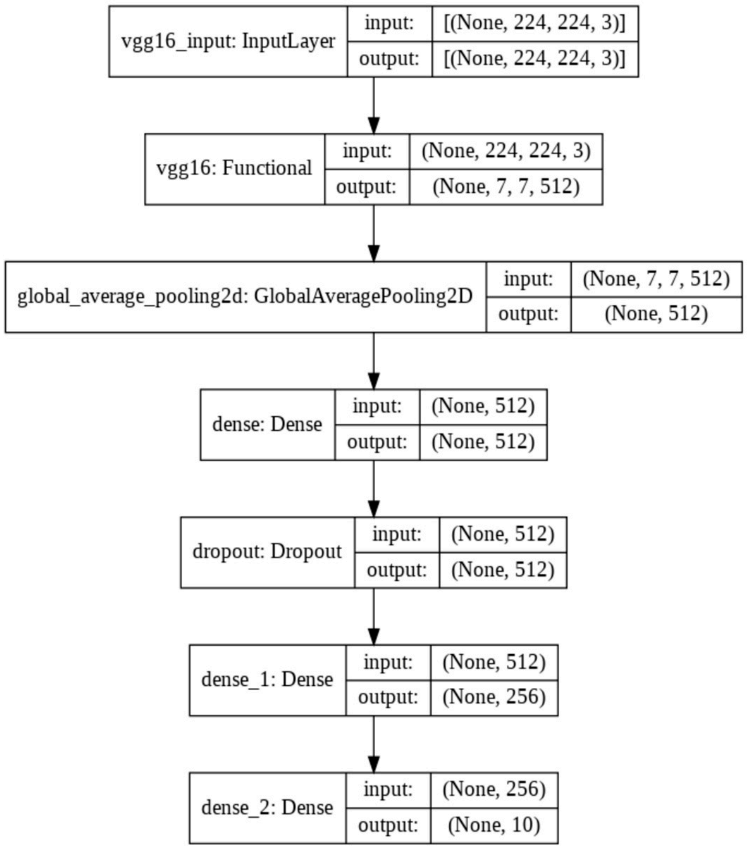


Figure 7. Dense net architecture (VGG16).

Inception model: The Inception model (Figure 8) is a widely-used image classification model with medium complexity. Its continual evolution resulted in the creation of multiple versions of the model. The one used in the experiments is the third version of the network, which has similar parameters to the ResNet50 model discussed earlier. The batch size equals 32, and the model is trained for 5 epochs. The model uses smaller convolutions which can be significant in decreasing computational time. In addition, factorizing convolutions reduces the number of parameters. Therefore, the Inception model has a lower training time compared to the ResNet50 and VGG16 models.

MobileNet model: we used the MobileNet architecture (Figure 9) to classify drivers' actions to examine its performance, considering it is the smallest state-of-the-art image classification network in terms of size and number of parameters. The main difference compared to other models is that MobileNet uses a 3 × 3 depth-wise convolution and 1 × 1 point-wise convolution instead of the traditional 3 × 3 convolution layer in most CNN models. The dense network is similar to the ResNet and Inception model networks discussed earlier. The batch size is 32, and the model was trained for 5 epochs. The top layer was removed from the network as with other models to modify the classifications according to the dataset used.

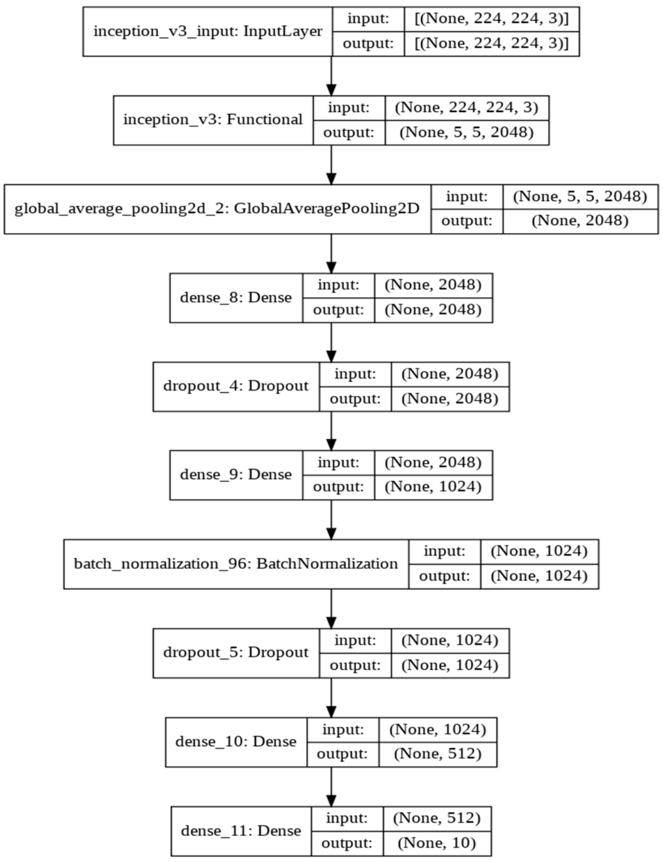


Figure 8. The Inception Model Architecture.

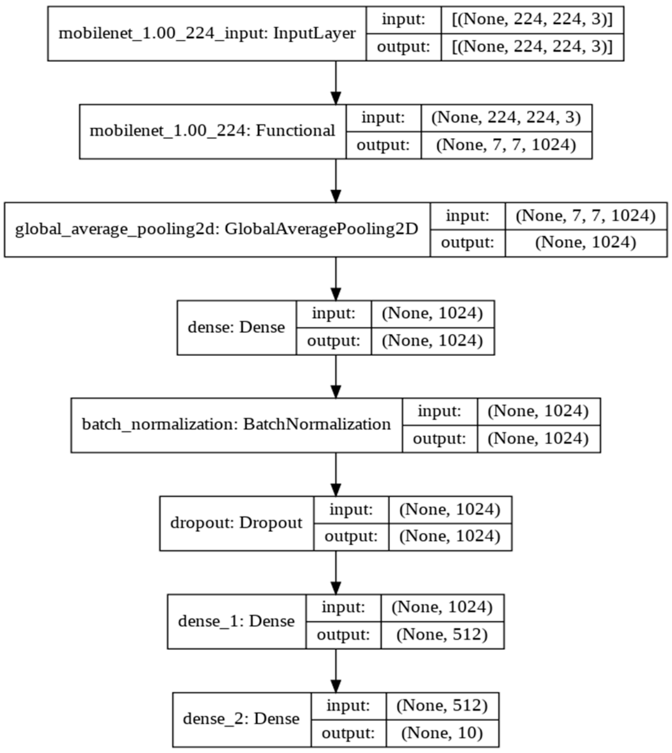

mobilenet_1.00_224_input: InputLayer
input: [(None, 224, 224, 3)]
output: [(None, 224, 224, 3)]
mobilenet_1.00_224: Functional
input: (None, 224, 224, 3)
output: (None, 7, 7, 1024)
global_average_pooling2d: GlobalAveragePooling2D
input: (None, 7, 7, 1024)
output: (None, 1024)
dense: Dense
input: (None, 1024)
output: (None, 1024)
batch_normalization: BatchNormalization
input: (None, 1024)
output: (None, 1024)
dropout: Dropout
input: (None, 1024)
output: (None, 1024)
dense_1: Dense
input: (None, 1024)
output: (None, 512)
dense_2: Dense
input: (None, 512)
output: (None, 10)

Figure 9. Model Architecture (MobileNet).

4.5. Recommendations

Adding a recommendation after the classification of each class can be a great improvement to the distracted driver detection implementations. In most cases, the best action to consider, especially in driving, is as easy as keeping the driver's focus on the road with no distraction. Reminding drivers is the best action to consider avoiding losing their attention during driving. An alert system can be effective with alternatives and actions to remind drivers of options to ensure their safety, especially drivers that need to take an important phone call or send an urgent text. In the case of autonomous vehicles, the recommendations can be sent to the vehicle to perform the best course of action. However, this is limited by the technology available in the vehicle. Table 2 provides the list of recommended actions as alertness signals to the driver based on the class of distracting activities while driving. The category of the distracting activities is retrieved from the ensemble classification layer in the E2DR model.

Table 2. Recommendations for distracted drivers.

Class Number	Class	Recommendation
C0	Safe driving	-
C1	Texting—Right	"Please avoid texting in all cases or make a stop"
C2	Talking on the phone—Right	"Please use a hands-free device"
C3	Texting—Left	"Please avoid texting in all cases or make a stop"
C4	Talking on the phone—Left	"Please use a hands-free device"
C5	Adjusting Radio	"Please use steering control"
C6	Drinking	"Please keep your hands at the steering wheel or make a stop"
C7	Reaching Behind	"Please keep your eyes on the road make a stop"
C8	Hair and Makeup	"Please make a stop"
C9	Talking to passenger	"Please keep your eyes on the road while talking"

5. Experimental Analysis and Results

5.1. Dataset

The dataset used in the experiment is the State Farm Distracted Drivers dataset [48]. There are 22,424 images of drivers in distracted positions that can be used for training and testing. The images in the dataset were taken with the contribution of 26 unique subjects in different cars (random numbering (i.e., not in consecutive order)). The dataset has 10 classes: safe driving, texting—right, talking on the phone—right, texting—left, talking on the phone—left, operating the radio, drinking, reaching behind, hair and makeup, and talking to a passenger. A representation of each class in the dataset is shown in Figure 10.

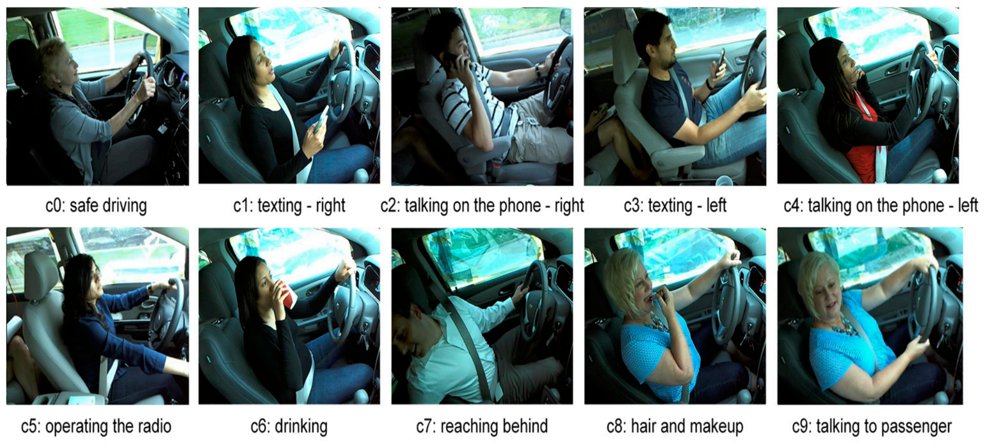


Figure 10. State Farm Distracted Driver dataset class representation [47].

5.2. Experimental Setup

A MacBook Pro and Google collab (Pro) are used to train and test the models.

5.3. Preprocessing and Splitting Strategy

The Distracted Driver dataset is ingested, then preprocessed as follows: (1) the images and the driver_imgs_list.csv file are stored, (2) the images are loaded, converted from BGR to RGB, and resized to 244 × 244 × 3 as this is the size used by most models and architectures for transfer learning, (3) the data are split into training, validation, and test sets; the validation and test set was created based on the subject (Driver ID); the subjects chosen for validation were: p18, p27, and p39, and the subjects chosen for testing were: p015, p022, p050, and p056. Finally, the labels were converted into categorical values. As a result, the training set had 15,963 images, the validation set had 2769 images and the test set had 3692 images.

5.4. Evaluation Metrics

Precision, Recall, F1 score, and Accuracy [49–54] are the most well-known evaluation metrics to assess the performance of a classifier. Precision finds pertinent instances among the gathered instances. It can be defined as the ratio between the True Positives (TP) and the sum of True Positives and False Positives (FP) as shown in Equation (2).

$$\text{Precision} = (\text{TP})/(\text{TP} + \text{FP}) \tag{2}$$

Recall, also known as Sensitivity, is defined as the ratio of True Positives and the sum of True Positives and False Negatives (FN).

$$\text{Recall } = (\text{TP})/(\text{TP} + \text{FN}) \tag{3}$$

F1 score: It is defined as the harmonic mean of Precision and Recall as shown in Equation (4).

$$\text{F1} - \text{Score } = (2 * \text{Precision } \times \text{ Recall})/(\text{Precision} + \text{Recall}) \tag{4}$$

Accuracy finds the correct predictions among the total predictions. It is defined as the ratio between the sum of True Positives and True Negatives and the sum of True Positives, False Negatives, True Negatives, and False Negatives.

$$\text{Accuracy } = (\text{TP} + \text{TN})/(\text{TP} + \text{FP} + \text{TN} + \text{FN}) \tag{5}$$

5.5. Performance Evaluation: Base Models

The performance of all base models is shown in Table 3. ResNet50 performs best with the highest accuracy and recall on the test set. The VGG16 performs just as well as the ResNet50 model, but the training time was significantly longer than ResNet50 and all other tested models since it has many variables. The Inception model had a test accuracy of 0.83, lower than ResNet50 and VGG16. The Inception model had the highest loss, and the training time was close to the ResNet50 model. Finally, the MobileNet had a similar performance to the Inception model with a lower loss and significantly faster training time. This is because MobileNet is a low-power, low-latency, light model parameterized to meet the constraints of computational and time resources of various applications. Choosing which optimum model to use depends on the trade-off between performance and computational complexity. ResNet50 and VGG16 can be used when powerful and robust computational resources and flexible time constraints are available. The MobileNet can be used for faster training and decent accuracy, which is not as good as VGG16 and ResNet50 but is still a good choice for limited computational resources. The Inception model did not perform well and lagged other models in most aspects, making it not favorable compared to the other tested models. Figure 11 shows how each model performed on the training, validation, and test sets. As shown in Figure 11, all models have a gradual increase in validation accuracy with the increasing number of epochs except for the MobileNet model, which has the highest validation accuracy at the third epoch and decreases afterwards. Similarly, the loss of all models decreases as epochs increase except for the MobileNet, which had the lowest loss value at epoch number three, which shows that the model was overfitting after epoch three. We even increased the number of epochs up to one, and we observed that the individual baseline models suffered from overfitting and could not show an increase in performance even when training was continued for a larger number of epochs. The performance report (Figure 12) shows how each model performs for each class. All models perform well for classes 1–5 and perform poorly for class 8, which is "hair and makeup", because it is a challenging class to be detected and usually confused with class 7, as shown in the confusion matrices in Figure 13. Moreover, the data quality for this class might not be on the same level as other classes.

Table 3. Deep learning image classification models performance.

Model	Training Accuracy	Validation Accuracy	Test Accuracy
ResNet	0.89	0.88	0.88
VGG16	0.94	0.86	0.87
Mobile Net	0.88	0.84	0.82
Inception	0.83	0.84	0.83

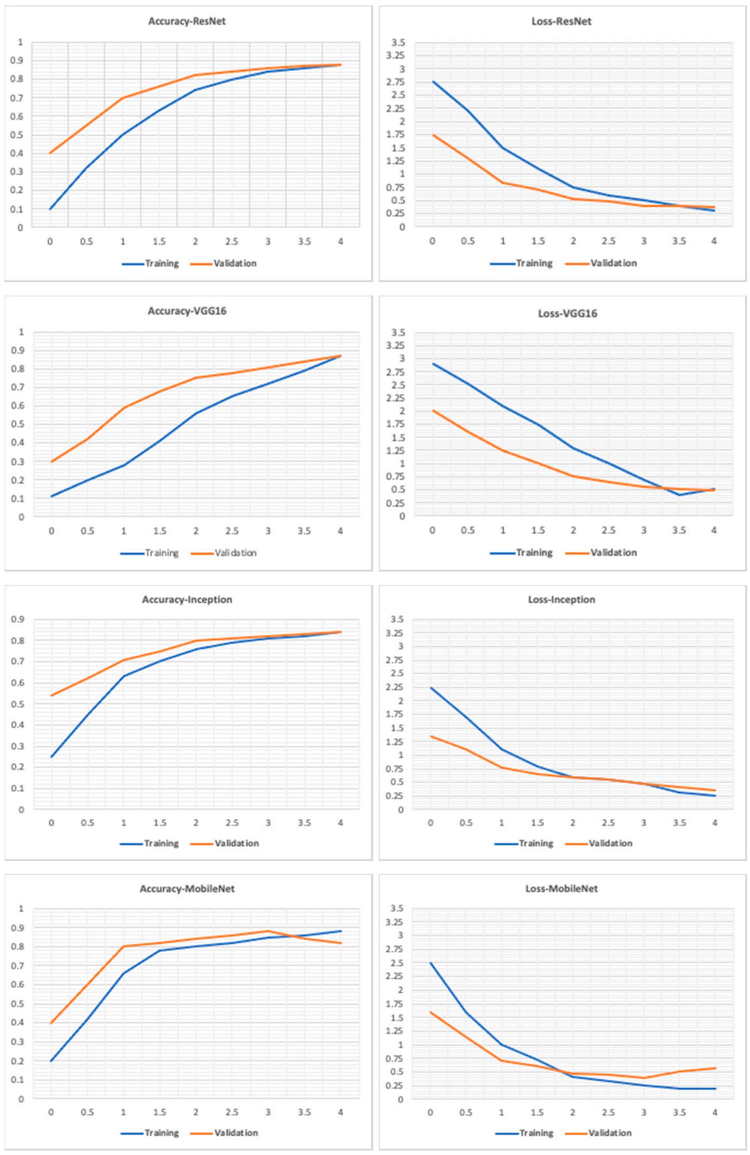
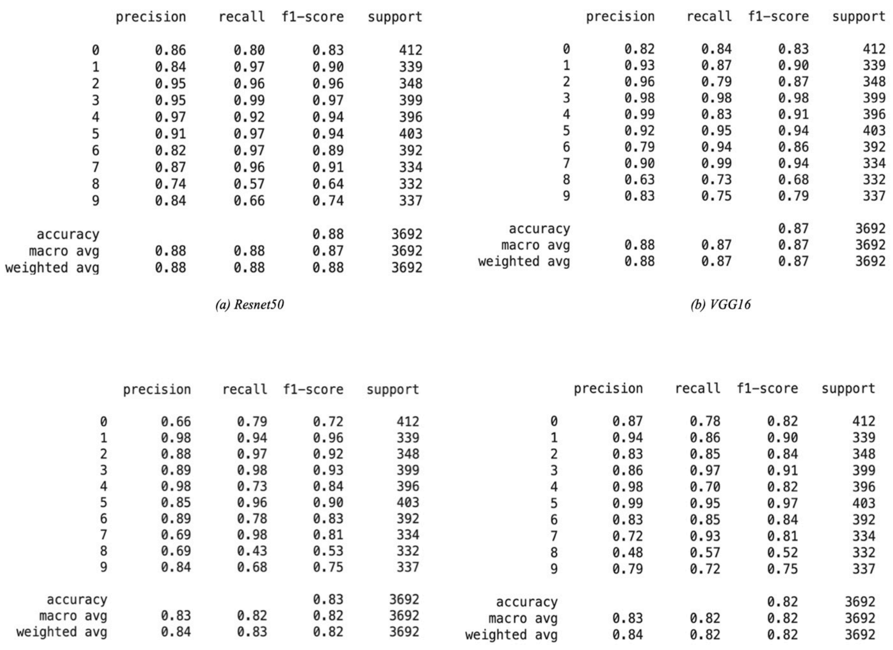


Figure 11. Accuracy and loss graphs for the models on the training and validation sets.



```
              precision    recall  f1-score   support

           0       0.86      0.80      0.83       412
           1       0.84      0.97      0.90       339
           2       0.95      0.96      0.96       348
           3       0.95      0.99      0.97       399
           4       0.97      0.92      0.94       396
           5       0.91      0.97      0.94       403
           6       0.82      0.97      0.89       392
           7       0.87      0.96      0.91       334
           8       0.74      0.57      0.64       332
           9       0.84      0.66      0.74       337

    accuracy                           0.88      3692
   macro avg       0.88      0.88      0.87      3692
weighted avg       0.88      0.88      0.88      3692
```

(a) Resnet50

```
              precision    recall  f1-score   support

           0       0.82      0.84      0.83       412
           1       0.93      0.87      0.90       339
           2       0.96      0.79      0.87       348
           3       0.98      0.98      0.98       399
           4       0.99      0.83      0.91       396
           5       0.92      0.95      0.94       403
           6       0.79      0.94      0.86       392
           7       0.90      0.99      0.94       334
           8       0.63      0.73      0.68       332
           9       0.83      0.75      0.79       337

    accuracy                           0.87      3692
   macro avg       0.88      0.87      0.87      3692
weighted avg       0.88      0.87      0.87      3692
```

(b) VGG16

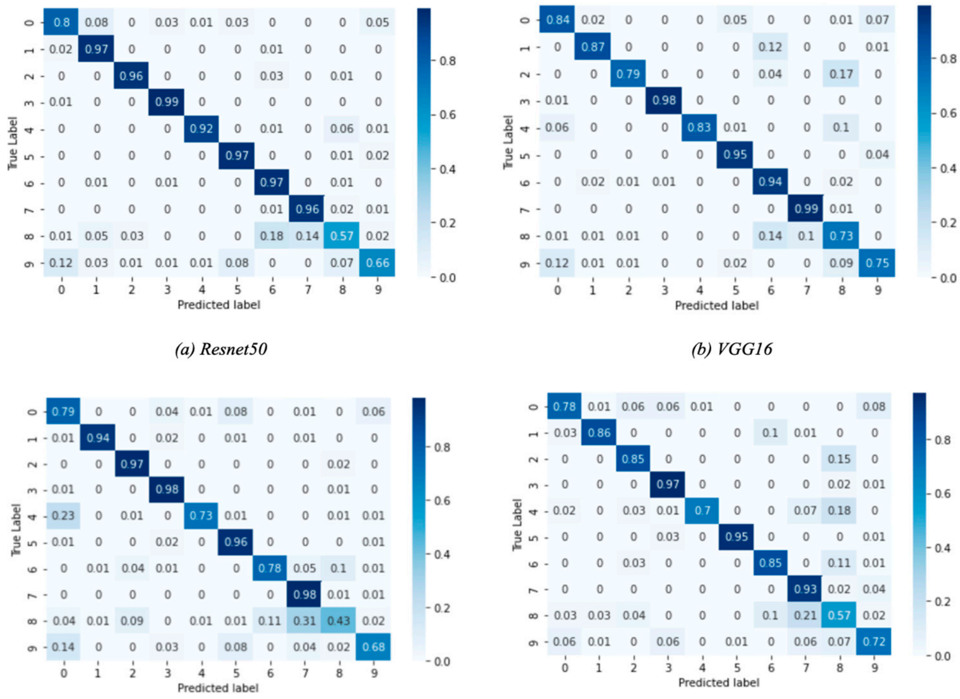
```
              precision    recall  f1-score   support

           0       0.66      0.79      0.72       412
           1       0.98      0.94      0.96       339
           2       0.88      0.97      0.92       348
           3       0.89      0.98      0.93       399
           4       0.98      0.73      0.84       396
           5       0.85      0.96      0.90       403
           6       0.89      0.78      0.83       392
           7       0.69      0.98      0.81       334
           8       0.69      0.43      0.53       332
           9       0.84      0.68      0.75       337

    accuracy                           0.83      3692
   macro avg       0.83      0.82      0.82      3692
weighted avg       0.84      0.83      0.82      3692
```

(c) Inception

```
              precision    recall  f1-score   support

           0       0.87      0.78      0.82       412
           1       0.94      0.86      0.90       339
           2       0.83      0.85      0.84       348
           3       0.86      0.97      0.91       399
           4       0.98      0.70      0.82       396
           5       0.99      0.95      0.97       403
           6       0.83      0.85      0.84       392
           7       0.72      0.93      0.81       334
           8       0.48      0.57      0.52       332
           9       0.79      0.72      0.75       337

    accuracy                           0.82      3692
   macro avg       0.83      0.82      0.82      3692
weighted avg       0.84      0.82      0.82      3692
```

(d) MobileNet

Figure 12. Performance reports for deep learning models on the test set.



	0	1	2	3	4	5	6	7	8	9
0	0.8	0.08	0	0.03	0.01	0.03	0	0	0	0.05
1	0.02	0.97	0	0	0	0	0.01	0	0	0
2	0	0	0.96	0	0	0	0.03	0	0.01	0
3	0.01	0	0	0.99	0	0	0	0	0	0
4	0	0	0	0	0.92	0	0.01	0	0.06	0.01
5	0	0	0	0	0	0.97	0	0	0.01	0.02
6	0	0.01	0	0.01	0	0	0.97	0	0.01	0
7	0	0	0	0	0	0	0.01	0.96	0.02	0.01
8	0.01	0.05	0.03	0	0	0	0.18	0.14	0.57	0.02
9	0.12	0.03	0.01	0.01	0.01	0.08	0	0	0.07	0.66

(a) Resnet50

	0	1	2	3	4	5	6	7	8	9
0	0.84	0.02	0	0	0	0.05	0	0	0.01	0.07
1	0	0.87	0	0	0	0	0.12	0	0	0.01
2	0	0	0.79	0	0	0	0.04	0	0.17	0
3	0.01	0	0	0.98	0	0	0	0	0	0
4	0.06	0	0	0	0.83	0.01	0	0	0.1	0
5	0	0	0	0	0	0.95	0	0	0	0.04
6	0	0.02	0.01	0.01	0	0	0.94	0	0.02	0
7	0	0	0	0	0	0	0	0.99	0.01	0
8	0.01	0.01	0.01	0	0	0	0.14	0.1	0.73	0
9	0.12	0.01	0.01	0	0	0.02	0	0	0.09	0.75

(b) VGG16

	0	1	2	3	4	5	6	7	8	9
0	0.79	0	0	0.04	0.01	0.08	0	0.01	0	0.06
1	0.01	0.94	0	0.02	0	0.01	0	0.01	0	0
2	0	0	0.97	0	0	0	0	0	0.02	0
3	0.01	0	0	0.98	0	0	0	0	0.01	0
4	0.23	0	0.01	0	0.73	0.01	0	0	0.01	0.01
5	0.01	0	0	0.02	0	0.96	0	0	0	0.01
6	0	0.01	0.04	0.01	0	0	0.78	0.05	0.1	0.01
7	0	0	0	0	0	0	0	0.98	0.01	0.01
8	0.04	0.01	0.09	0	0.01	0.01	0.11	0.31	0.43	0.02
9	0.14	0	0	0.03	0	0.08	0	0.04	0.02	0.68

(c) Inception

	0	1	2	3	4	5	6	7	8	9
0	0.78	0.01	0.06	0.06	0.01	0	0	0	0	0.08
1	0.03	0.86	0	0	0	0	0.1	0.01	0	0
2	0	0	0.85	0	0	0	0	0	0.15	0
3	0	0	0	0.97	0	0	0	0	0.02	0.01
4	0.02	0	0.03	0.01	0.7	0	0	0.07	0.18	0
5	0	0	0	0.03	0	0.95	0	0	0	0.01
6	0	0	0.03	0	0	0	0.85	0	0.11	0.01
7	0	0	0	0	0	0	0	0.93	0.02	0.04
8	0.03	0.03	0.04	0	0	0	0.1	0.21	0.57	0.02
9	0.06	0.01	0	0.06	0	0.01	0	0.06	0.07	0.72

(d) MobileNet

Figure 13. Confusion matrices for the deep learning models.

5.6. The E2DR Performance Evaluation

5.6.1. Settings

In the first layer of the E2DR model, the individual models were trained. Their layers are executed in parallel (after optimizations) when loaded in the ensemble, so their weights are not altered during training. Each of the two base models has 10 outputs representing each class in the dataset. The output of the base models is sent to a concatenation layer, which is then sent to a dense layer with 10 neurons (equal to the number of classes).

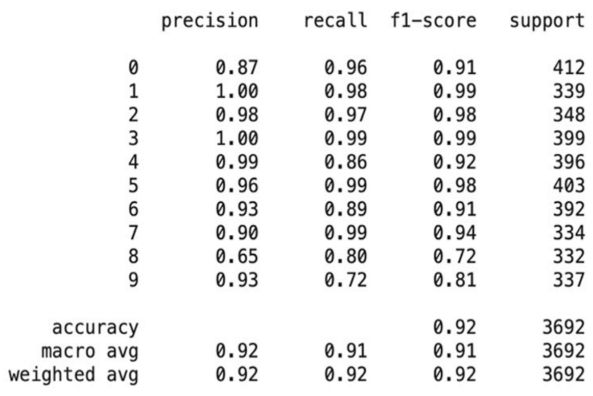
A SoftMax activation function is used to perform classification. A learning rate of 0.001 and Adam optimizer are used to compile the model with a batch size equal to 32. The loss function used is the Categorical Cross Entropy loss function. The E2DR was trained for five epochs similar to base models.

5.6.2. Results and Discussion

The results of the E2DR models showed a remarkable improvement in performance compared to the individual models. The best performing E2DR model was the stacked ensemble combination of ResNet and VGG16 with a test accuracy of 92%. The lowest-performing E2DR model with a test accuracy of 88% was the MobileNet–Inception E2DR variant, which was also expected, as the base models did not perform very well individually. The performance of each variant of the E2DR model is shown in Table 4. The improvement in accuracy was around 5–8%, which is a significant improvement considering that the base models could not exceed the late 80% in their accuracy. The fact that E2DR models reached accuracies exceeding 90% proves that the E2DR models effectively improved generalization compared to the individual base models. Other metrics, such as Precision, Recall, and F1 scores, showed similar results and improvements to accuracy, which further validates the model's performance. The loss function used in all experiments is the Categorical Cross Entropy function, representing the confidence of predictions made by the model. The loss of the base models and the E2DR variants were in the same range with a small improvement in the E2DR models. This is because the classification confidence did not significantly improve, which means that despite making more correct classifications, which led to an increase in accuracy, the model did not have high certainty in making those predictions. Although the ensemble model in [6] achieved a higher performance with the traditional percentage-based data split, our method provides further credibility as it was tested on completely new data, which simulates real-world scenarios. This was performed by choosing subjects (Driver IDs) that are not included in the training set when constructing the validation and test sets, allowing the model to be tested on data it had not seen before. This approach is not followed in other implementations. The batch size used when recording the training duration is 32. The performance and confusion matrix of the highest performing E2DR model, which includes ResNet50 and VGG16, is presented in Figure 14. When looking at the performance report of this E2DR variant, the strongest classes from the ResNet50 model were 0 and 6, while the VGG16 performed best for classes 3 and 4. However, after analyzing the performance report of the E2DR model, our method combined the strong classes from each model in a single robust model. This is one of the most useful advantages of the E2DR model, where the model combines the skills and strong points of different models into one model, allowing the base models to complement each other in terms of performance. Similarly, Figure 15 shows the performance report and the confusion matrix of the lowest-performing E2DR variant that uses MobileNet and Inception as base models. Although it is the lowest-performing variant, it still showed a huge performance boost compared to the base models' performance. The E2DR model effectively addressed the weak points of the Inception model (classes 0 and 7) and MobileNet model (classes 2 and 9) by boosting the classification performance for those classes in the E2DR model. Comparing the confusion matrices of the base models and the E2DR variants also improves classification performance, especially for class 8, where the confusion rate with class 7 has decreased compared to the base models. Figures 16–18 visualize the performance evaluation across different metrics for the base models and the E2DR variants. The E2DR variants outperform the baseline models measured by the test Accuracy, Precision, Recall, F1 score, and Loss value, as shown in Figures 16–18. As recorded in Equation (1), the computational time to fully develop the E2DR models would be the maximum training duration of the combined base models in addition to the overhead of concatenation and recommendation retrieval. The execution of the E2DR models after training can be applied in real time. The additional overhead in training the E2DR models is shown in Figure 19. It can be observed that there is an average overhead of 7% in using the E2DR models, as illustrated in Figure 19. However, due to the limited GPU computational power in the experimental hardware used as discussed in Section 5.2., we anticipate that this overhead will be significantly reduced if additional GPUs are used in the training phase. Using the baseline models, the recognition time of one image is 14.45 ms on average with 15.21 ms (on average) when using the ensemble E2DR models.

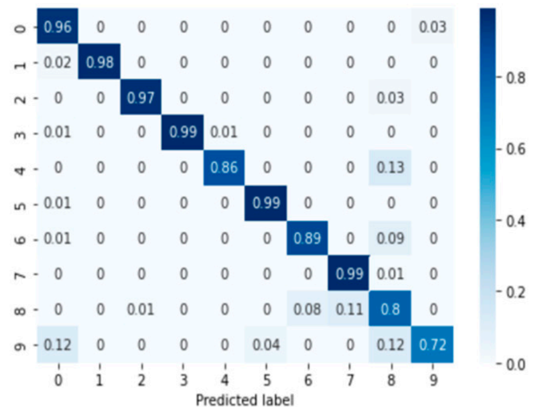
Table 4. E2DR models performance on the test set.

E2DR Model	Accuracy	Precision	Recall	F1 score	Loss
MobileNet–Inception	0.88	0.89	0.88	0.88	0.55
ResNet50–Inception	0.88	0.89	0.88	0.88	0.47
ResNet50–MobileNet	0.90	0.91	0.9	0.9	0.43
VGG16–Inception	0.90	0.91	0.9	0.9	0.39
VGG16–MobileNet	0.91	0.92	0.91	0.91	0.42
ResNet50–VGG16	0.92	0.92	0.92	0.92	0.37



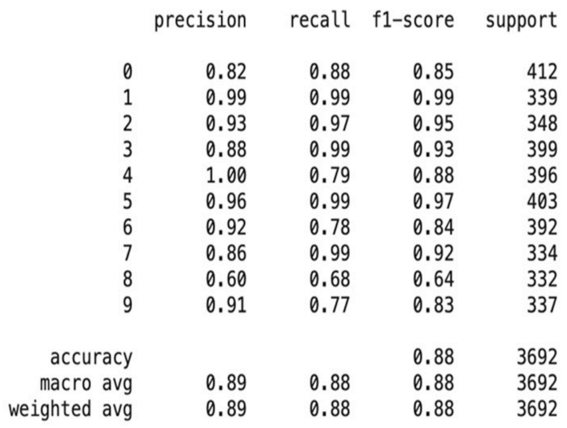
```
              precision    recall  f1-score   support

           0       0.87      0.96      0.91       412
           1       1.00      0.98      0.99       339
           2       0.98      0.97      0.98       348
           3       1.00      0.99      0.99       399
           4       0.99      0.86      0.92       396
           5       0.96      0.99      0.98       403
           6       0.93      0.89      0.91       392
           7       0.90      0.99      0.94       334
           8       0.65      0.80      0.72       332
           9       0.93      0.72      0.81       337

    accuracy                           0.92      3692
   macro avg       0.92      0.91      0.91      3692
weighted avg       0.92      0.92      0.92      3692
```

a) Best performing E2DR model classification report



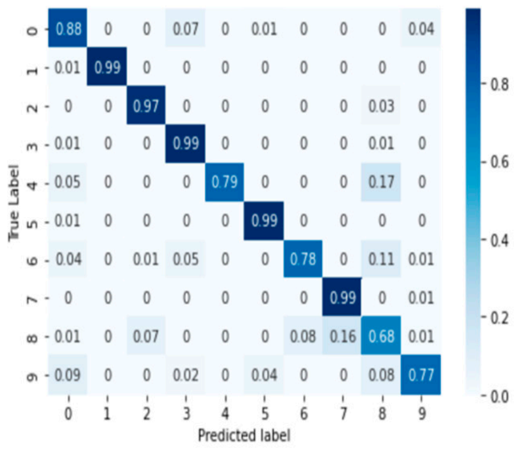
(b) Best performing E2DR model Confusion matrix

Figure 14. Best performing E2DR model classification report and confusion matrix.



```
              precision    recall  f1-score   support

           0       0.82      0.88      0.85       412
           1       0.99      0.99      0.99       339
           2       0.93      0.97      0.95       348
           3       0.88      0.99      0.93       399
           4       1.00      0.79      0.88       396
           5       0.96      0.99      0.97       403
           6       0.92      0.78      0.84       392
           7       0.86      0.99      0.92       334
           8       0.60      0.68      0.64       332
           9       0.91      0.77      0.83       337

    accuracy                           0.88      3692
   macro avg       0.89      0.88      0.88      3692
weighted avg       0.89      0.88      0.88      3692
```

a) Lowest performing E2DR model classification report



(b) Lowest performing E2DR model Confusion matrix

Figure 15. Lowest performing E2DR model classification report and confusion matrix.

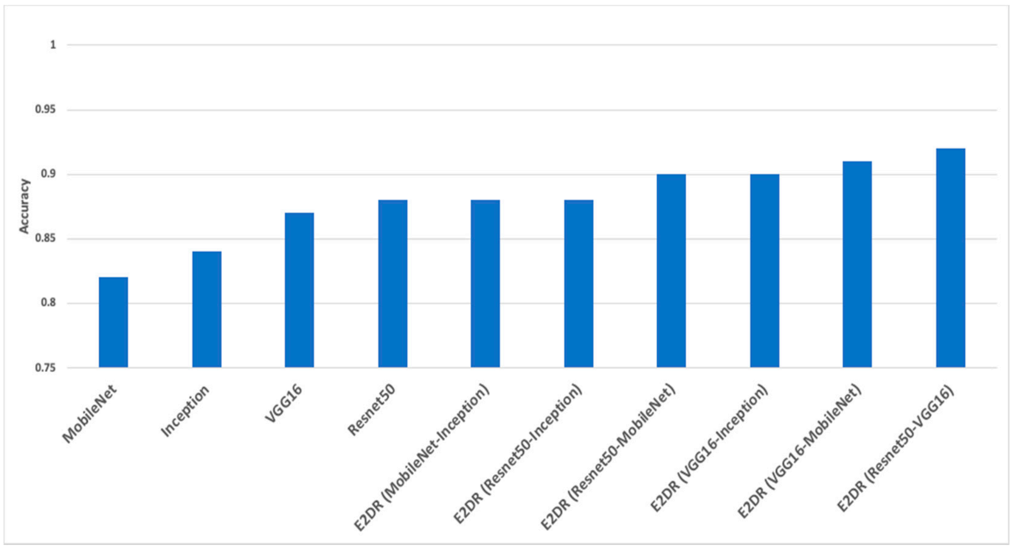


Figure 16. Accuracy of base models and E2DR models.

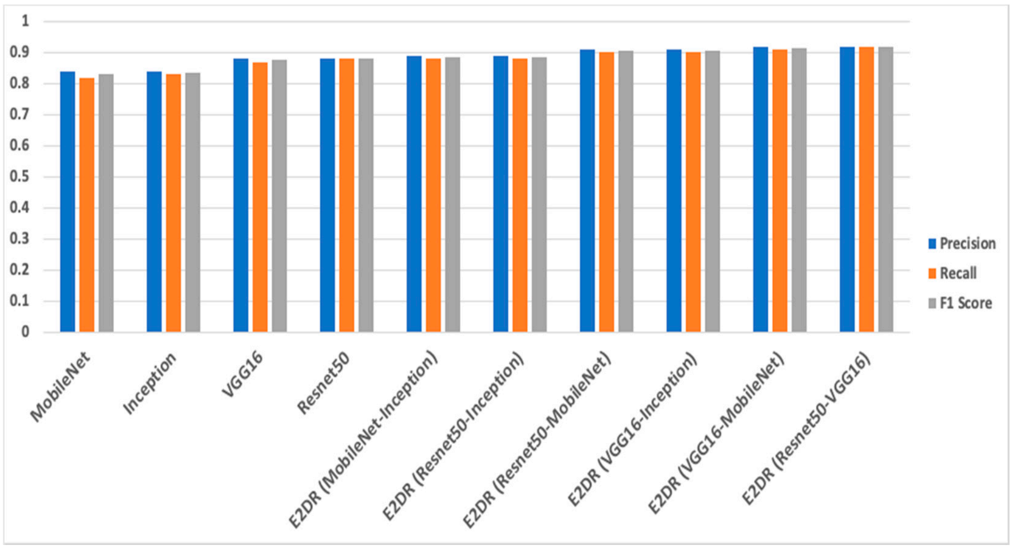


Figure 17. The Precision, Recall, and F1 score of base models and E2DR models.

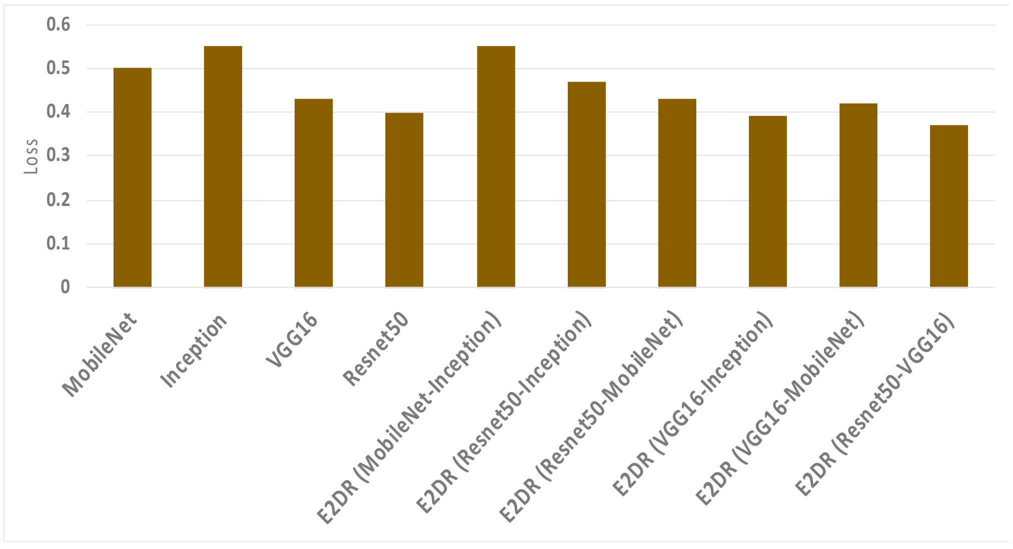


Figure 18. The Loss of base models and E2DR models.

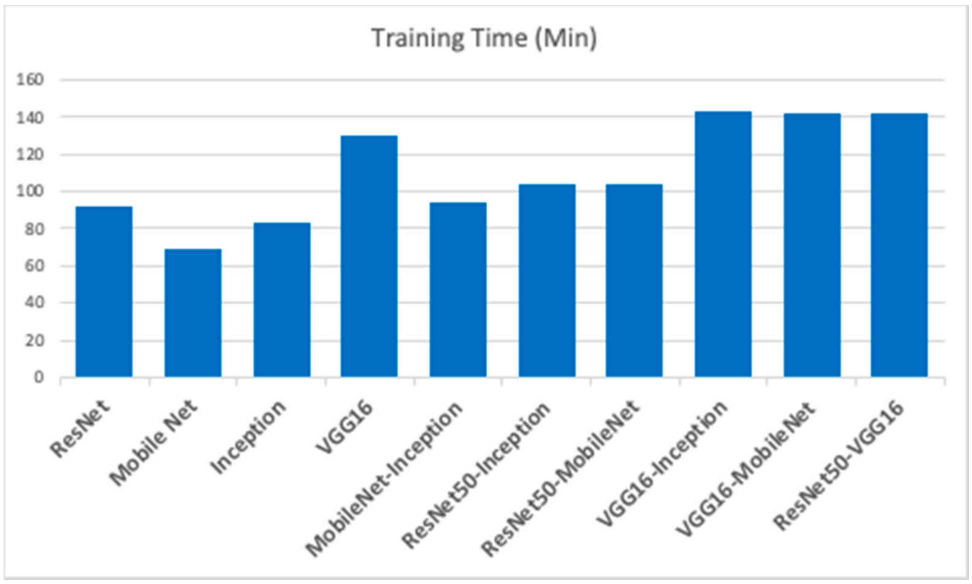


Figure 19. Training Time (Proposed E2DR vs. baseline deep learning models).

6. Conclusions and Future Directions

This paper examines different deep learning classification models for distracted driver classification [55–59] and proposes a model that improves performance and provides recommendations. We explored the performance of different models: ResNet50, VGG16, MobileNet, and Inception. All models provided viable means in detecting distracted driver actions. This paper proposes E2DR, a new model that uses stacking ensemble methods to improve accuracy, enhance generalization and reduce overfitting. Additionally, a set of recommendations are added by the model. The highest performing E2DR variant, which included the ResNet50 and VGG16 models, achieved an accuracy of 92%, while the highest performing single model was the ResNet50 with 88% accuracy. The lowest-performing E2DR model was the MobileNet–Inception variant, which achieved an accuracy of 88%, and the lowest-performing individual model was the MobileNet, with an accuracy of 82%. The accuracy difference between the highest and lowest performing models for the E2DR models and the individual models shows a significant increase in performance when using our proposed E2DR model. Other metrics were recorded and presented to evaluate the classification performance of the tested base models and E2DR variants such as Recall, Precision, and F1 score, which showed a similar increase in performance. Furthermore, the performance reports and confusion matrices showed that the E2DR models effectively addressed the weak points of the base models and boosted their classification performance. The computational complexity when developing the E2DR models from scratch is considered a limitation. Since computational speed is important in real-time applications, a light model such as MobileNet can be integrated with ResNet50 or VGG16, which in our experiment showed a significant boost in performance without adding much computational complexity.

For future work, the performance of more than two models combined in the stacking ensemble method can be examined; it was infeasible to test multiple combinations with the limited computational resources used to conduct the experiments. Furthermore, the model can be associated with the police departments to fine violators and identify drivers' actions in case of accidents. The model can also be integrated with face recognition and alarm systems [60,61] capabilities that can allow the model to be used in a wide range of applications, such as driver authentication and theft prevention. Finally, the model can be developed and used in autonomous vehicles to detect critical conditions or situations that might endanger the driver's health and safety, such as strokes, heart attacks, and other sicknesses. The model can recommend the vehicle to ensure the safety and health of the driver and others.

Author Contributions: Conceptualization, M.A. and R.K.; methodology, M.A. and R.K.; software, M.A.; validation, M.A. and R.K.; formal analysis, M.A. and R.K.; investigation, M.A. and R.K.; resources, M.A.; data curation, M.A.; writing—original draft preparation, M.A. and R.K.; writing—review and editing, M.A. and R.K.; visualization, M.A. and R.K.; supervision, R.K.; project administration, R.K.; funding acquisition, R.K. All authors have read and agreed to the published version of the manuscript.

Funding: This paper was funded by Ryerson University.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The State Farm Distracted Drivers dataset can be accessed in [47].

Conflicts of Interest: The authors declare no conflict of interest.

References

1. The World Health Organization. Global Status Report on Road Safety. 2018. Available online: https://www.who.int/publications/i/item/9789241565684 (accessed on 15 January 2020).
2. Yanbin, Y.; Lijuan, Z.; Mengjun, L.; Ling, S. Early warning of traffic accident in Shanghai based on large data set mining. In Proceedings of the 2016 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS), Changsha, China, 17–18 December 2016; pp. 18–21.
3. Engstörm, J.; Victor, T.W. Real-Time Distraction Countermeasures. In *Driver Distraction: Theory, Effects, and Mitigation*; CRC Press: Boca Raton, FL, USA, 2008; pp. 465–483.
4. Kang, H.B. Various approaches for driver and driving behavior monitoring: A review. In Proceedings of the IEEE International Conference on Computer Vision Workshops, Washington, DC, USA, 2–8 December 2013; pp. 616–623.
5. Krajewski, J.; Trutschel, U.; Golz, M.; Sommer, D.; Edwards, D. Estimating fatigue from predetermined speech samples transmitted by operator communication systems. In Proceedings of the Fifth International Driving Symposium on Human Factors in Driver Assessment, Training and Vehicle Design, Big Sky, MT, USA, 24 June 2009; pp. 468–473.
6. Alotaibi, M.; Alotaibi, B. Distracted driver classification using deep learning. *Signal. Image Video Process.* **2019**, *14*, 617–624. [CrossRef]
7. Stappen, L.; Rizos, G.; Schuller, B. X-aware: Context-aware human-environment attention fusion for driver gaze prediction in the wild. In Proceedings of the 2020 International Conference on Multimodal Interaction, Virtual, 25–29 October 2020; pp. 858–867.
8. Palazzi, A.; Solera, F.; Calderara, S.; Alletto, S.; Cucchiara, R. Learning where to attend like a human driver. *IEEE Int. Veh. Symp.* **2017**, 920–925. [CrossRef]
9. Methuku, J. In-Car driver response classification using Deep Learning (CNN) based Computer Vision. *IEEE Trans. Intell. Veh.* **2020**.
10. Jeong, M.; Ko, B.C. Driver's Facial Expression Recognition in Real-Time for Safe Driving. *Sensors* **2018**, *18*, 4270. [CrossRef] [PubMed]
11. Baheti, B.; Talbar, S.; Gajre, S. Towards Computationally Efficient and Realtime Distracted Driver Detection with Mobile VGG Network. *IEEE Transact. Intell. Veh.* **2020**, *5*, 565–574. [CrossRef]
12. Kumari, M.; Hari, C.; Sankaran, P. Driver Distraction Analysis Using Convolutional Neural Networks. In Proceedings of the 2018 International Conference on Data Science and Engineering (ICDSE), Kochi, India, 7–9 August 2018; pp. 1–5. [CrossRef]
13. Lu, M.; Hu, Y.; Lu, X. Driver action recognition using deformable and dilated faster R-CNN with optimized region proposals. *Appl. Intell.* **2019**, *50*, 1100–1111. [CrossRef]
14. Li, P.; Yang, Y.; Grosu, R.; Wang, G.; Li, R.; Wu, Y.; Huang, Z. Driver Distraction Detection Using Octave-Like Convolutional Neural Network. *IEEE Trans. Intell. Transp. Syst.* **2021**, 1–11. [CrossRef]
15. Li, G.; Yan, W.; Li, S.; Qu, X.; Chu, W.; Cao, D. A Temporal-Spatial Deep Learning Approach for Driver Distraction Detection Based on EEG Signals. *IEEE Trans. Autom. Sci. Eng.* **2021**, 1–13. [CrossRef]
16. Abbas, T.; Ali, S.F.; Khan, A.Z.; Kareem, I. optNet-50: An Optimized Residual Neural Network Architecture of Deep Learning for Driver's Distraction. In Proceedings of the 2020 IEEE 23rd International Multitopic Conference (INMIC), Bahawalpur, Pakistan, 5–7 November 2020. [CrossRef]
17. Xie, Z.; Li, L.; Xu, X. Recognition of driving distraction using driver's motion and deep learning. *IIE Ann. Conf. Proc.* **2020**, 949–954.
18. Abouelnaga, Y.; Eraqi, H.M.; Moustafa, M.N. Real-time distracted driver posture classification. *arXiv* **2017**, arXiv:1706.09498.
19. Koay, H.; Chuah, J.; Chow, C.-O.; Chang, Y.-L.; Rudrusamy, B. Optimally-Weighted Image-Pose Approach (OWIPA) for Distracted Driver Detection and Classification. *Sensors* **2021**, *21*, 4837. [CrossRef]
20. Arvin, R.; Khattak, A.J.; Qi, H. Safety critical event prediction through unified analysis of driver and vehicle volatilities: Application of deep learning methods. *Accid. Anal. Prev.* **2020**, *151*, 105949. [CrossRef] [PubMed]
21. Xing, Y.; Lv, C.; Wang, H.; Cao, D.; Velenis, E.; Wang, F.-Y. Driver Activity Recognition for Intelligent Vehicles: A Deep Learning Approach. *IEEE Trans. Veh. Technol.* **2019**, *68*, 5379–5390. [CrossRef]

22. Varaich, Z.A.; Khalid, S. Recognizing actions of distracted drivers using inception v3 and xception convolutional neural networks. In Proceedings of the 2019 2nd International Conference on Advancements in Computational Sciences (ICACS), Lahore, Pakistan, 18–20 February 2019; pp. 1–8.
23. Mao, P.; Zhang, K.; Liang, D. Driver Distraction Behavior Detection Method Based on Deep Learning. *IOP Conf. Series. Mater. Sci. Eng.* **2020**, *782*, 22012. [CrossRef]
24. Mase, J.M.; Chapman, P.; Figueredo, G.P.; Torres, M.T. A Hybrid Deep Learning Approach for Driver Distraction Detection. In Proceedings of the International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Korea, 21–23 October 2020. [CrossRef]
25. Alkinani, M.H.; Khan, W.Z.; Arshad, Q. Detecting Human Driver Inattentive and Aggressive Driving Behavior Using Deep Learning: Recent Advances, Requirements and Open Challenges. *IEEE Access* **2020**, *8*, 105008–105030. [CrossRef]
26. Li, L.; Zhong, B.; Hutmacher, C., Jr.; Liang, Y.; Horrey, W.J.; Xu, X. Detection of driver manual distraction via image-based hand and ear recognition. *Accid. Anal. Prevent.* **2020**, *137*, 105432. [CrossRef] [PubMed]
27. Springer Link. Available online: https://link.springer.com/article/10.1007/s00330-019-06318-1/figures/1 (accessed on 19 March 2021).
28. ImageNet. Available online: http://www.image-net.org/ (accessed on 17 March 2021).
29. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
30. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2014**, arXiv:1409.1556.
31. Alex, K.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
32. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436. [CrossRef]
33. VGG16—Convolutional Network for Classification and Detection. Available online: https://neurohive.io/en/popular-networks/vgg16/ (accessed on 19 March 2021).
34. A Simple Guide to the Versions of the Inception Network. Available online: https://towardsdatascience.com/a-simple-guide-to-the-versions-of-the-inception-network-7fc52b863202 (accessed on 19 March 2021).
35. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with con-volutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
36. Bose, S.R.; Kumar, V.S. Efficient inception V2 based deep convolutional neural network for real-time hand action recognition. *IET Image Process.* **2020**, *14*, 688–696. [CrossRef]
37. Szegedy, C.; Vincent, V.; Sergey, I.; Jon, S.; Wojna, Z. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2818–2826.
38. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A.A. Inception-v4, inception-resnet and the impact of residual connections on learning. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 12 February 2017.
39. Inception-v3. Available online: https://paperswithcode.com/method/inception-v3 (accessed on 20 March 2021).
40. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861.
41. Image Classification With MobileNet. Available online: https://medium.com/analytics-vidhya/image-classification-with-mobilenet-cc6fbb2cd470 (accessed on 26 March 2021).
42. Kim, W.; Jung, W.-S.; Choi, H.K. Lightweight Driver Monitoring System Based on Multi-Task Mobilenets. *Sensors* **2019**, *19*, 3200. [CrossRef] [PubMed]
43. An Overview on MobileNet: An Efficient Mobile Vision CNN. Available online: https://medium.com/@godeep48/an-overview-on-mobilenet-an-efficient-mobile-vision-cnn-f301141db94d (accessed on 15 July 2021).
44. Wolpert, D.H. Stacked generalization. *Neural Networks* **1992**, *5*, 241–259. [CrossRef]
45. ResNet and ResNetV2. Available online: https://keras.io/api/applications/resnet/#resnet50-function (accessed on 17 July 2021).
46. Ciresan, D.C.; Meier, U.; Masci, J.; Gambardella, L.M.; Schmidhuber, J. Flexible, high performance convolutional neural networks for image classification. In Proceedings of the Twenty-Second international joint conference on Artificial Intelligence, Barcelona, Spain, 16–22 July 2011.
47. He, K.; Sun, J. Convolutional neural networks at constrained time cost. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 5353–5360.
48. State Farm Distracted Driver Detection. Available online: https://www.kaggle.com/c/state-farm-distracted-driver-detection (accessed on 25 July 2021).
49. Close, L.; Kashef, R. Combining Artificial Immune System and Clustering Analysis: A Stock Market Anomaly Detection Model. *J. Intell. Learn. Syst. Appl.* **2020**, *12*, 83–108. [CrossRef]
50. Kashef, R. Enhancing the Role of Large-Scale Recommendation Systems in the IoT Context. *IEEE Access* **2020**, *8*, 178248–178257. [CrossRef]
51. Yeh, T.-Y.; Kashef, R. Trust-Based Collaborative Filtering Recommendation Systems on the Blockchain. *Adv. Int. Things* **2020**, *10*, 37–56. [CrossRef]
52. Ebrahimian, M.; Kashef, R. Detecting Shilling Attacks Using Hybrid Deep Learning Models. *Symmetry* **2020**, *12*, 1805. [CrossRef]

53. Li, M.; Kashef, R.; Ibrahim, A. Multi-Level Clustering-Based Outlier's Detection (MCOD) Using Self-Organizing Maps. *Big Data Cogn. Comput.* **2020**, *4*, 24. [CrossRef]
54. Tobin, T.; Kashef, R. Efficient Prediction of Gold Prices Using Hybrid Deep Learning. In *International Conference on Image Analysis and Recognition*; Springer: Cham, The Netherlands, 2020; pp. 118–129.
55. Ledezma, A.; Zamora, V.; Sipele, O.; Sesmero, M.; Sanchis, A. Implementing a Gaze Tracking Algorithm for Improving Advanced Driver Assistance Systems. *Electronics* **2021**, *10*, 1480. [CrossRef]
56. Liu, D.; Yamasaki, T.; Wang, Y.; Mase, K.; Kato, J. TML: A Triple-Wise Multi-Task Learning Framework for Distracted Driver Recognition. *IEEE Access* **2021**, *9*, 125955–125969. [CrossRef]
57. Kumar, A.; Sangwan, K.S. A Computer Vision Based Approach forDriver Distraction Recognition Using Deep Learning and Genetic Algorithm Based Ensemble. *arXiv* **2021**, arXiv:2107.13355.
58. Eraqi, H.M.; Abouelnaga, Y.; Saad, M.H.; Moustafa, M.N. Driver Distraction Identification with an Ensemble of Convolutional Neural Networks. *J. Adv. Transp.* **2019**, *2019*, 1–12. [CrossRef]
59. Gite, S.; Agrawal, H.; Kotecha, K. Early anticipation of driver's maneuver in semiautonomous vehicles using deep learning. *Prog. Artif. Intell.* **2019**, *8*, 293–305. [CrossRef]
60. Magán, E.; Ledezma, A.; Sesmero, P.; Sanchis, A. Fuzzy Alarm System based on Human-centered Approach. *VEHITS* **2020**, 448–455. [CrossRef]
61. Sipele, O.; Zamora, V.; Ledezma, A.; Sanchis, A. Advanced Driver's Alarms System through Multi-agent Paradigm. In Proceedings of the 2018 3rd IEEE International Conference on Intelligent Transportation Engineering(ICITE), Singapore, 3–5 September 2018; pp. 269–275.