Distracted Driver Detection

*Detecting distracted drivers using deep learning models

1st Dr. Anurag Goel
CSE Department
Delhi Technological University
New Delhi, India
anurag@dtu.ac.in

2nd Aastha Ahlawat
24/AFI/01
Delhi Technological University
New Delhi, India
aasthaahlawat_24afi01@dtu.ac.in

3rd Riya
24/AFI/14
Delhi Technological University
New Delhi, India
riya_24afi14@dtu.ac.in

***Abstract*—Distracted driving has serious repercussions for public safety since it contributes significantly to traffic accidents. In order to detect instances of driver's distractions in real time, this project aims to construct a Distracted Driver Detection System that employs computer vision, deep learning and machine learning. The system's goal is to categorize typical/general distractions like using a phone, eating, drinking, or turning to talk to other passengers.**

Images of drivers performing a variety of tasks are included in annotated datasets that are used to train the detection model. By using methods such as convolutional neural networks (CNNs), the system is able to detect whether a driver is distracted or not. Using images, the implementation incorporates the model to detect and predict for a driver accurately if he/she is distracted or not.

This project demonstrates how different neural network or deep learning models can be implemented on a dataset made entirely of images. Various models are trained on the same dataset and their training and validation accuracy can be easily compared to illustrate how different neural networks perform in detecting distracted drivers. By addressing the critical issue of distracted driving, this system has the potential to significantly reduce accidents and save lives, contributing to the advancement of intelligent transportation systems.

***Index Terms*—Distracted Driver, CNN, RESNET, MOBILENET**

I. Introduction

According to the World Health Organization (WHO) Global Status Report on Road Safety 2023, approximately 1.19 million people lose their lives annually due to road traffic accidents.An additional 20–50 million people are injured or incapacitated. Road crashes remain a leading cause of death globally, particularly among young people aged 5–29 years. Vulnerable road users, such as pedestrians, cyclists, and motorcyclists, account for a significant proportion of these fatalities.

The National Crime Research Bureau (NCRB), Government of India, reports that the biggest number of fatalities worldwide occur on Indian highways. Since 2006, India has seen a steady rise in the number of fatal traffic accidents. According to the latest data, India still has a serious problem with road safety. In 2021, traffic accidents claimed the lives of almost 1.55 lakh people. Pedestrians, cyclists, and motorcyclists were the most susceptible populations, and road traffic injuries continued to rank among the nation's major causes of death and health loss by 2024. Almost half of all deaths occur in states like Tamil Nadu, Maharashtra, and Uttar Pradesh.

The goal of continuous safety initiatives, such as "zero fatality corridors" and improved urban road designs, is to reduce fatalities by 50% by 2030. Nonetheless, the rise in fatalities highlights the urgent need for enhanced road infrastructure, more traffic law enforcement, and greater driver education. Driver errors are largely to blame for India's one of the highest road death rates in the world, according to recent data from the International Traffic Safety Data and Analysis Group (IRTAD). Despite advancements worldwide, many low- and middle-income nations, including India, continue to face dire circumstances.

Recent events have brought to light persistent worries around driver distraction in driverless vehicles and advanced driver assistance systems (ADAS). Investigations of multiple crashes involving Tesla's Autopilot in 2024 found that driver inattention was a common contributing cause. The National Highway Traffic Safety Administration (NHTSA) examined more than 700 ADAS-related crashes, including Tesla, in which there was clear evidence of misuse or careless conduct. For example, a Tesla driver in Washington State was allegedly distracted and using their phone when they were involved in a tragic incident. In a similar view, Uber's self-driving cars or autonomous systems are still under investigation following previous fatalities, highlighting the necessity of strong driver monitoring technology at all automation levels.

Artificial intelligence and computer vision are utilized to recognize and categorize distracted driving practices. This technology can identify risky habits like eating, using a mobile phone, or adopting an inattentive posture by utilizing real-time video feeds and machine learning models. It can then promptly alert automated systems or drivers to these actions. Such a system's deployment could improve road safety while also supporting the larger objectives of autonomous driving and intelligent transportation networks.

This research investigates the development, training, and implementation of a distraction detection model, emphasizing its applicability, difficulties, and possible contribution to a decrease in human error-related traffic accidents.

II. LITERATURE SURVEY

In order to identify distracted behaviors, early detection techniques mostly depended on human monitoring or on-road observations. These methods were labor-intensive, subjective, and prone to human error, though. To measure facial directions and eye movements, sensor-based systems were devised, including infrared sensors and eye-tracking devices. A disadvantage of these systems was that they were costly, invasive, and needed specialist technology, which limited their scalability in practical settings.

Modern advancements in artificial intelligence (AI) and computer vision have enabled robust solutions for detecting distracted driving behaviors. Convolutional Neural Networks (CNNs) have shown significant promise in analyzing images to identify activities such as phone usage, eating, or conversing while driving. CNN-based models process visual input efficiently, providing high accuracy in classifying driver behavior. Models were trained to detect actions such as texting, chatting on the phone, or grasping for things using tagged driver photos. Video-Based Approaches incorporating Spatial and temporal activities, including continuous phone use or eating, can be captured by time-series analysis with Recurrent Neural Networks (RNNs) or 3D-CNNs.

To address real-world limitations, lightweight neural network architectures like MobileNet have been developed. MobileNet delivers improved computational efficiency, enabling real-time performance even on mobile and edge devices. The MobileNetV2 model further optimizes accuracy with reduced complexity, making it highly suitable for distracted driver detection in intelligent transportation systems.

Despite these advancements, challenges persist. Public datasets for training CNN models are often limited, and there is a trade-off between achieving high accuracy and maintaining real-time detection capabilities. Addressing these challenges is essential for widespread adoption of AI-driven distracted driving detection systems.

The proposed study aims to leverage MobileNetV2 for real-time detection of distracted driving behaviors, contributing to improved road safety and intelligent transportation solutions.

III. DEEP LEARNING MODELS

The term ”Deep Learning” describes a subset of machine learning methods that process vast volumes of data using artificial neural networks. With the help of their many levels—an input layer, hidden layers, and an output layer—these networks are able to automatically extract hierarchical patterns from text, audio, and image data. Because these networks can automatically learn properties from unprocessed data and replicate the neuron-based structure of the brain, they are extremely task-adaptive.

Although deep learning models are computationally demanding, they have revolutionized sectors by facilitating the development of AI-driven solutions in fields like entertainment, healthcare, and transportation. Transformer models for natural language, Recurrent Neural Networks (RNNs) for sequential data, and Convolutional Neural Networks (CNNs) for visual data are important categories of deep learning models. Some of the models that have been used in our research are as follows:

A. CNN

Images and other grid-like data are processed using Convolutional Neural Networks (CNNs), a form of deep learning model. They employ filters that identify patterns such as edges, textures, and forms in order to automatically extract features from the input data using convolutional layers. Following these layers are pooling layers, which lower the spatial dimensions and aid in the model's effective capture of significant features. CNNs are especially good at tasks like segmenting images, classifying images, and detecting objects.

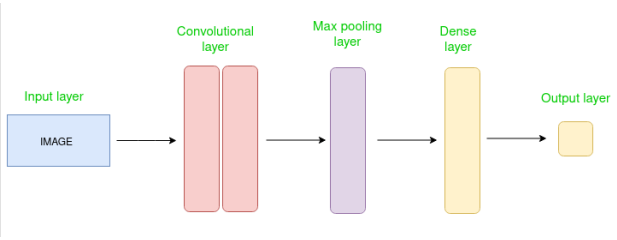

Fig. 1. Simple CNN Architecture

B. RESNET

A deep learning architecture called ResNet (Residual Neural Network) was created to address the issue of vanishing gradients in extremely deep networks. Data can immediately skip one or more layers thanks to the use of skip connections or residual connections. The model can learn more intricate hierarchical features thanks to its architecture, which makes it possible to train incredibly deep networks without suffering a noticeable performance loss. ResNet's performance on benchmarks like ImageNet shows how good it is at picture recognition tasks.

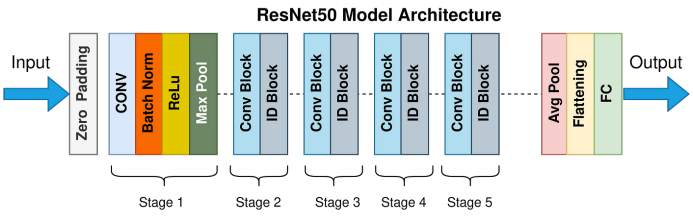

Fig. 2. ResNet50 Architecture

The image depicts the architecture of a ResNet (Residual Neural Network). It consists of several key components:

- Zero Padding: Prepares the input image by adding a border of zeros to maintain spatial dimensions after convolution.
- Convolution (Conv): Applies convolution operations to the input image to extract feature maps.
- Batch Normalization: Normalizes the output of each layer to improve convergence during training.

- ReLU Activation: Introduces non-linearity to the network, allowing it to model complex patterns.
- Max Pooling: Downsamples the data by selecting the maximum value in a pooling window, reducing spatial dimensions.
- Residual Block (ID Block): Consists of multiple convolutional layers, but skips connections (shortcuts) allow data to bypass some layers, improving training efficiency and performance. This structure helps the network learn residual mappings, which are differences between input and output features.

C. MOBILENET

A lightweight deep learning model architecture called MobileNet was created to operate effectively on mobile and edge devices. Because of its cheap computational cost and speed optimization, it is perfect for resource-constrained applications. In order to drastically reduce computational complexity, MobileNet employs Depthwise Separable Convolutions, which split conventional convolutions into two stages: depthwise convolution (filtering input channels) and pointwise convolution (combining filtered outputs).

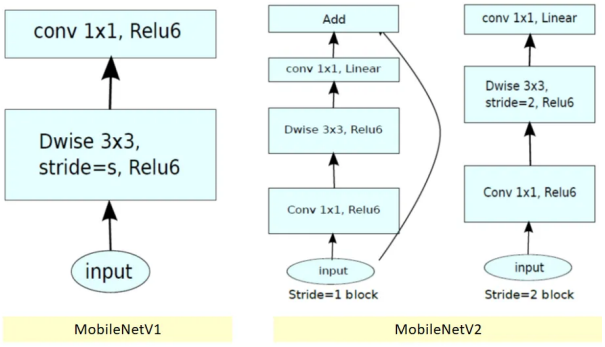

Fig. 3. MobileNet Architecture

In MobileNetV1, the architecture uses depthwise separable convolutions, splitting standard convolution into two parts:

- Depthwise Convolution: Applies a single convolutional filter per input channel, performing spatial filtering efficiently.
- Pointwise Convolution: A 1x1 convolution that combines the outputs of depthwise convolution to create new features by linearly combining input channels.

The model uses ReLU6 (min(max(x, 0), 6)) activation to ensure stability, particularly in low-precision computations, making it efficient for mobile and embedded devices.

MobileNetV2 is an enhanced version of MobileNetV1. It introduces inverted residual blocks with linear bottlenecks, which extend the input for convolution after it has been compressed, and then project it back to a low-dimensional representation, building on the success of MobileNetV1. This invention preserves important functionalities while cutting down on computational costs. In addition, MobileNetV2 makes use of ReLU6 activation for stability in low-precision calculations and separable depthwise convolutions to maximize performance. Image classification, object detection, and segmentation are among the tasks that MobileNetV2 excels at because of its effective 17-block design and enhanced accuracy-to-latency trade-off. It is the model of choice for real-time AI applications on edge devices due to its harmonious combination of high accuracy, low latency, and low memory needs.

IV. DATASET DESCRIPTION

A carefully chosen set of tagged photos, the Distracted Driver Detection Dataset is intended to aid in the study and creation of computer vision models for identifying distracted driving. The data set is ideal for classification tasks and object recognition, as it contains high-quality photos of drivers doing a variety of safe and risky driving actions.

In order to train machine learning models to identify risky or distracted driving, the data set consists mainly of front-facing photos of the driver in an automobile. Each behavior is labeled. For the development of computer vision technologies that aim to improve traffic safety and reduce distracted driving-related collisions, this data set is essential. Ten behavior classes, designated c0 through c9, are included in the distracted driver detection dataset; each class corresponds to a unique action taken by the driver.

- c0: Driving safely (the driver is paying attention).
- c1: Using the right-hand hand to text.
- c2: Using the right-hand to talk on the phone.
- c3: Using the left-hand hand to text.
- c4: Using the left-hand to talk on the phone.
- c5: Using the automobile radio.
- c6: Drinking (drinks are held and consumed).
- c7: Reaching back (driver swivels to retrieve).
- c8: Hair and makeup (applying makeup or grooming).
- c9: Engage in conversation with a passenger (driver turning).

V. METHODOLOGY

Any deep learning method for image categorization targeted at distracted driver detection is described in the form of its components and methodology:

A. Data Set Loading and Preparation

The notebook utilizes kagglehub to download a dataset from Kaggle. Images from this dataset are probably categorized according to the various degrees of driver distraction (e.g., gazing at the phone, altering settings, etc.).After that, torchvision datasets are used to load and convert the dataset. This entails resizing pictures to a standard size. Model performance is enhanced by normalizing pixel data to a standard range (0–1) or mean/std normalization. To allow for model evaluation and avoid overfitting, the dataset is divided into training and validation sets.

To make the model more resilient, a variety of data augmentation techniques are used, such as random rotations and horizontal flips. These changes improve the model's ability to

generalize to various image kinds and increase its conditional invariance. Even when the input photos are not precisely centered or have different scales, the model can still produce accurate predictions thanks to the application of random cropping and padding.

B. Model Definition

Different models are being trained on the dataset with different architectures and definitions.

- CNN:
  The convolutional blocks in this CNN architecture gradually capture hierarchical spatial information in input images. Fully connected layers then translate these features to class labels. The last softmax layer guarantees that the output is a probability distribution over the predicted classes, and dropout is utilized to enhance generalization. Tasks like distracted driver identification, which call for precise image classification into various states (distracted or not), are a good fit for this architecture.
  Three different archtectures are used:
  1) Three convolution blocks with TanH activation function abbreviated as CNN1.
  2) A single convolution block with ReLU activation function abbreviated as CNN2.
  3) A single convolution block with TanH activation function abbreviated as CNN3.
- ResNet:
  A pre-trained ResNet50 model with pre-trained weights is loaded by the code. In order to match a new classification job with ten classes, it then adjusts the last completely connected layer to contain ten output neurons. By doing this, the model is modified for the new task. Lastly, to expedite calculations, the model is transferred to the GPU (if available). ResNet50 may be effectively adjusted for the new work with this configuration.
- MobileNet:
  A pre-trained MobileNetV2 model is loaded by the code snippet. A solid starting point is provided by the pretrained flag, which loads the model's weights that were trained on a sizable dataset such as ImageNet.
  The MobileNetV2 classifier is then altered. The number of classes in the ImageNet dataset is reflected in the single output layer of the original classifier. Here, a new fully connected layer that produces 10 classes in accordance with the new task at hand takes its place.
  Lastly, for effective calculation, the model is sent to the designated device (GPU if available). In order to speed up the process, this configuration modifies MobileNetV2 for the new classification task with the proper output layer and GPU support.

C. Training the Model

- The CrossEntropyLoss loss function, which is appropriate for multi-class classification applications, is used. It creates a single loss value by fusing the negative log-likelihood and softmax output.
- Adam, which is effective at managing big models and datasets, is used to train the model. During training, it dynamically modifies the learning rate, which may hasten convergence.
- To enable the model to discover patterns in the data, the training loop iterates over a number of epochs, usually ten or more.
- Every epoch involves the model processing the training data, forward propagating the data through the network to generate predictions, and backpropaging the loss to adjust the model's parameters.
- The accuracy on both the training and validation sets is monitored to ensure the model is learning effectively and not overfitting.

D. Model Evaluation and Prediction

Prior to final deployment, the trained model's performance (accuracy, loss) is assessed using the validation set. To make sure the model performs effectively when applied to unknown data, this step is essential. It produces a classification report that assists in understanding the model's performance by providing precision, recall, and F1-score for each class. Furthermore, a confusion matrix is calculated to show the proportion of accurate and inaccurate predictions for each class, providing information about the model's accuracy across classes. By assessing the pre-trained model's performance on unseen data, this configuration provides a clear picture of its advantages and disadvantages.

To make predictions on fresh, unobserved data, a prediction function (predict) is defined. It entails loading and altering the picture. Class scores are obtained by feeding the image through the model. giving back the class that was anticipated based on the top score.

E. Saving The Model

Python objects can be serialized (pickled) and deserialized (unpickled) using the pickle package. Any Python object, including trained models, can be saved to a file and then loaded for later use without requiring a complete rebuild or retraining. This is especially helpful for storing model setups, states, and outcomes for later use or sharing in various contexts. The model can be loaded later to make predictions on fresh data after it has been trained and saved to a file. When the model must be applied in real-time situations, this is helpful.

VI. RESULTS AND DISCUSSION

The different models are trained and their avarage loss, training and validation accuracy across different epocs are calculated. The Table I shows that the performance of the various models varies significantly:

- Poor performance is demonstrated by CNN2 and CNN1, which have very low accuracies (10.19% and 11.61% training, 10.91% and 12.76% validation) and significant average losses (2.36 and 2.30 respectively).

- Although not as bad as CNN2 and CNN1, CNN3 still has a moderate accuracy (69.45% training, 74.06% validation) and average loss (1.76), which could indicate that the data is too complicated for the model or that it is over-regularized.
- At an average loss of 0.52 and high training and validation accuracy of 83.14% and 93.53%, respectively, ResNet exhibits a better performance.
- Additionally, MobileNet performs well with a high accuracy (97.41% training, 95.38% validation) and an average loss of 0.08.

TABLE I
MODEL PERFORMANCE SUMMARY

Model	Average Loss	Training Accuracy	Validation Accuracy
CNN1	2.3010	11.61%	12.76%
CNN2	2.3593	10.19%	10.91%
CNN3	1.7647	69.45%	74.06%
ResNet	0.5216	83.14%	93.53%
MobileNet	0.0828	97.41%	95.38%

The Potential Causes for the performance and results observed are:

- ResNet and MobileNet are more effective models that use depthwise separable convolutions to retain high accuracy and minimal complexity.
- On the other hand, underfitting appears to be the problem of CNN2 and CNN1 being out of alignment with the data distribution. This suggests that these models either overfit or fail to capture significant data properties. Their high average losses (2.35 and 2.30) indicate underfitting, possibly due to insufficient depth, poor initialization, or ineffective learning rate settings.
- Low overfitting and efficient generalization are shown by ResNet. The accuracy is only for one epoch which means that is we increase the number of epocs to 5 like in the other models, it can provide more significant results. ResNet's skip connections likely help in reducing overfitting and improving generalization
- The CNN3 provides the best result out of the three CNN architectures but the perfromance is still not satifactory enough showing a validation accuracy of approximately 75%. Its higher loss (1.7647) suggests room for optimization, possibly due to a simpler architecture compared to ResNet or MobileNet.
- MobileNet's great accuracy, despite its lesser complexity, may be explained by its design being more appropriate for simpler tasks, indicating good performance especially where efficiency and speed are important considerations. This indicates that MobileNet effectively balances accuracy and computational efficiency, potentially benefiting from its lightweight depthwise separable convolutions.

To sum up, the top-performing models are MobileNet and ResNet. While CNN3 achieves moderate success but could use more improvement, CNN1 and CNN2 may perform worse due to architectural constraints. These variations most likely result from each architecture's innate capacity to identify and extrapolate intricate patterns in the data.

TABLE II
MOBILENET PERFORMANCE SUMMARY

Epoch	Average Loss	Training Accuracy	Validation Accuracy
1	0.3329	89.38%	96.07%
2	0.1478	95.48%	96.28%
3	0.1226	96.24%	97.86%
4	0.0979	96.99%	93.78%
5	0.0828	97.41%	95.38%

According to the Table II, the MobileNet model performs well on the training data, demonstrating a very high training accuracy of 97.41%. Nevertheless, its 95.38% validation accuracy indicates a minor decline when tested on fresh, untested data, which might be a sign of a little overfitting, but the performance is still excellent.

TABLE III
MOBILENET PERFORMANCE METRICS

Class	Precision	Recall	F1-Score	Support
0	0.97	0.96	0.96	472
1	0.84	1.00	0.91	465
2	0.94	0.99	0.97	485
3	0.98	0.99	0.98	476
4	0.99	0.99	0.99	517
5	0.99	0.99	0.99	435
6	0.96	1.00	0.98	449
7	0.99	0.96	0.98	387
8	0.98	0.78	0.87	372
9	0.95	0.87	0.91	426
Accuracy	-	-	-	96%
Macro Avg	0.96	0.95	0.95	-
Weighted Avg	0.96	0.96	0.96	-

According to the confusion matrix given in Table III, in the majority of classes, the model MobileNet obtains good F1-score, precision, and recall. Class 1 (Label 1) exhibits nearly zero misclassifications with a flawless recall of 1.00. The model appears to perform well in classes 4, 5, and 6, as seen by their extremely high precision, recall, and F1-scores. There is potential for improvement as Class 8 (Label 8) has a lower recall (0.78) and F1-score (0.87). With a weighted average F1-score of 0.96 and a macro average F1-score of 0.95, the overall accuracy is 0.96, demonstrating strong model performance across all classes. Direct consultation of the confusion matrix or performance metrics might yield additional insights for more thorough analysis or comparison.

A confusion matrix that evaluates a classification model's performance is shown in the figure 4. The number of times the real class (True) was predicted as a different class (Predicted) is displayed in each cell of the matrix. For example, 452 instances of the true class c0 were properly predicted as c0, as indicated by the cell at row c0 and column c0 with the number 452. Higher numbers are represented visually by darker hues, which also show misclassifications and prediction accuracy. This matrix aids in determining which classes the model has trouble accurately predicting.

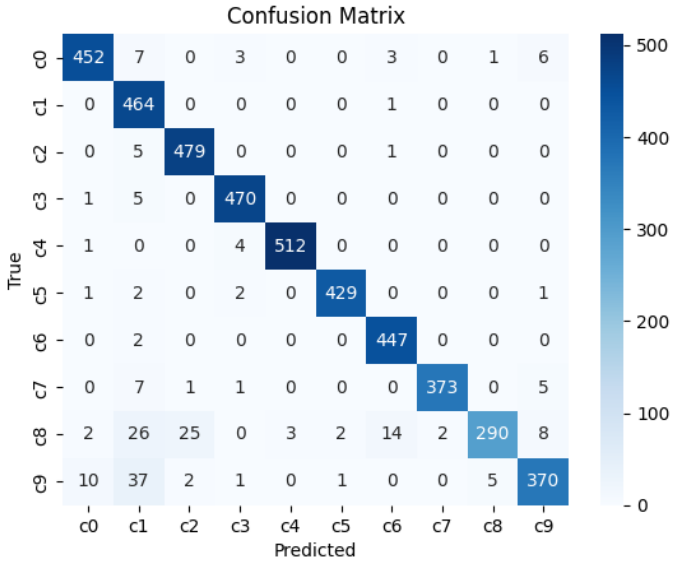

Fig. 4. MobileNet Architecture Confusion Matrix

A line plot of the training loss over epochs is displayed in the figure 5. It shows how the model's performance becomes better over time as the number of epochs increases and the loss metric falls. As training goes on, the training loss progressively drops from its initial high value, indicating successful optimization and model convergence. The plot offers information on how well the model was trained.

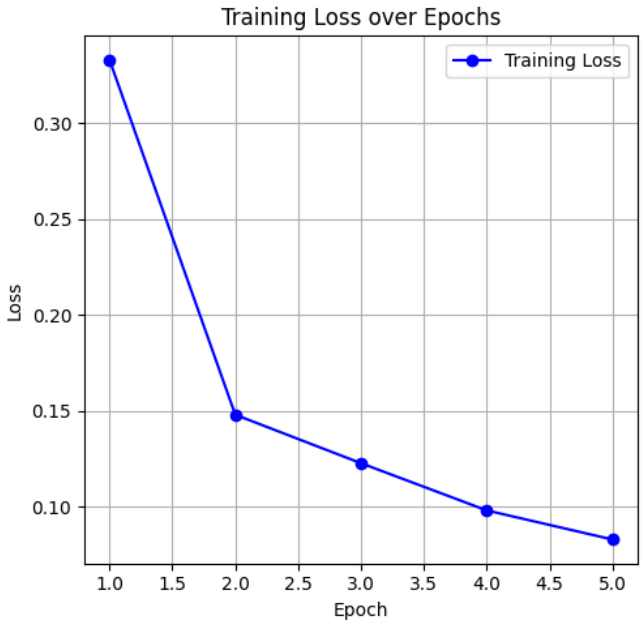

Fig. 5. MobileNet Architecture Loss

A line plot of the training and validation over epochs is displayed in the figure 6. It shows how the model's performance becomes better over time as the number of epochs increases,

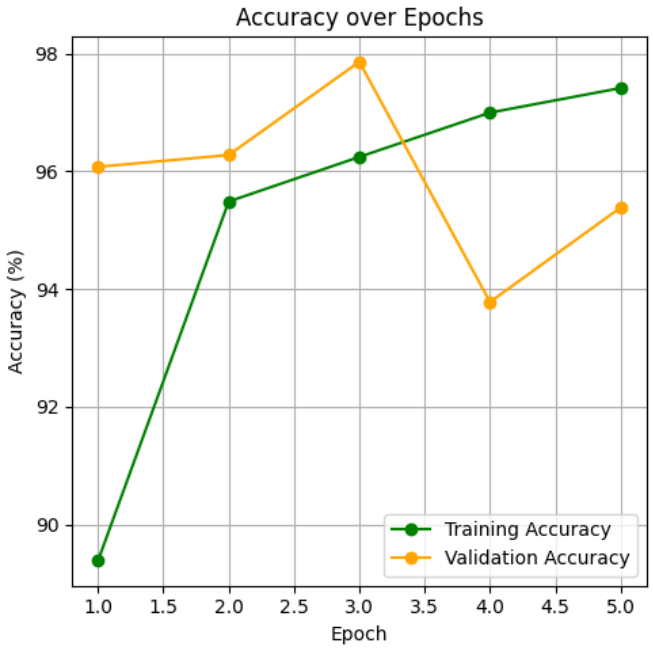

Fig. 6. MobileNet Architecture Accuracy

the training accuracy increases and the validation accuracy is also shown across different epocs.

VII. CONCLUSION AND FUTURE SCOPE

Using computer vision and deep learning techniques, the Distracted Driver Detection System shows great promise in tackling the pervasive problem of distracted driving. When comparing the CNN, ResNet, and MobileNet models, it is evident that MobileNet provides better accuracy (95.38% validation accuracy) and efficiency, which makes it perfect for real-time applications, particularly on devices with limited resources. The system's capacity to precisely categorize driver distractions, such talking on the phone, eating, or socializing with other passengers, can be extremely important in improving road safety.

Notwithstanding the encouraging outcomes, there are still issues with some classes showing lower recall scores, which points to the possibility of further strengthening the model's resilience. Implementing such systems in practical settings may help support intelligent transportation systems and lower accident rates.

Future research can explore additional data augmentation, advanced model architectures, and integration with autonomous driving technologies to enhance overall performance. Integrating distracted driver detection systems with Advanced Driver Assistance Systems (ADAS) would enable proactive notifications and remedial measures, including automated lane correction or braking, when preoccupation is identified. For smooth and effective monitoring, lightweight models will be deployed in real time on edge devices, such as smartphones, in-car cameras, and Internet of Things-based systems. Combining multi-modal data—such as eye-tracking

devices, accelerometers, steering wheel sensors, and visual inputs—can increase accuracy by providing a more comprehensive picture of driver behavior.

Personalized driver monitoring systems can create individual behavioral profiles to detect anomalies like fatigue or stress. Incorporating facial recognition and emotion analysis will further enhance detection by identifying emotional states linked to distraction.

REFERENCES

[1] Kaggle, "Distracted Driver Dataset," Kaggle.
[2] BuiltIn, "MobileNet: First Mobile Computer Vision Model," BuiltIn.
[3] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," Towards Data Science.
[4] GeeksforGeeks, "What is MobileNet V2?," GeeksforGeeks.
[5] GeeksforGeeks, "Introduction to Convolution Neural Network," GeeksforGeeks.
[6] ScienceDirect, "Residual Neural Networks (ResNet)," ScienceDirect.
[7] J. Doe, "Driver Behavior Detection Using CNN," ScienceDirect.
[8] S. Kumar, P. Sharma, "Detection of Distracted Driver Using CNN," ResearchGate.
[9] ImageVision, "Distracted Driver Detection Applications," ImageVision.AI.
[10] IEEE Xplore, "Distracted Driving Detection Using Deep Learning," IEEE Xplore.