International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 04 | April - 2024 SJIF Rating: 8.448 ISSN: 2582-3930

COGNITIVE DRIVER ACTION RECOGNITION

Aryan Gaikwad[1], Aditya Adhane[2], Vishal Sangole[3], Vedant More[4]

Prof. S.P. Jadhav[5]

Department Of Information Technology, Sinhgad College Of Engineering, Pune, India

---***---

Abstract - Operating an automobile is a multifaceted endeavor, demanding unwavering focus and attention from the driver. Distracted driving, encompassing any behavior that diverts the driver's concentration away from the road, poses a grave threat to road safety. Alarming statistics reveal that approximately 1.35 million lives are tragically lost each year due to road traffic accidents, inflicting significant economic ramifications, with road traffic crashes costing most nations an estimated 3% of their gross domestic product. The primary objective of our project is to put in place an exhaustive process for identifying potentially dangerous driving behaviors and discerning appropriate driving practices in light of this disappointing reality. By utilizing a wide range of machine learning models, we aim to correctly classify the given photos into discrete groups that correlate to various types of driver distraction. Furthermore, our work goes beyond simple classification; it also aims to perform a comparison analysis of different Machine Learning Models in order to determine how well they perform and how accurate they are in the context of cognitive driver action detection. This all-encompassing strategy demonstrates our dedication to improving traffic safety and lowering the possibility of collisions and injuries to other drivers.

Key Words: Transfer learning, Deep learning, Image classification, Distracted driving, TensorFlow.

1.INTRODUCTION

The goal of this research is to use a range of advanced deep-learning techniques to detect and classify driver activities based on photographs. Fundamentally, the goal of this project is to create a highly precise model that can classify driver behaviors into separate predetermined classifications. The main objective is to use the categorization capacity to determine if a driver is voluntarily following recommended safety procedures or, unfortunately, is not.

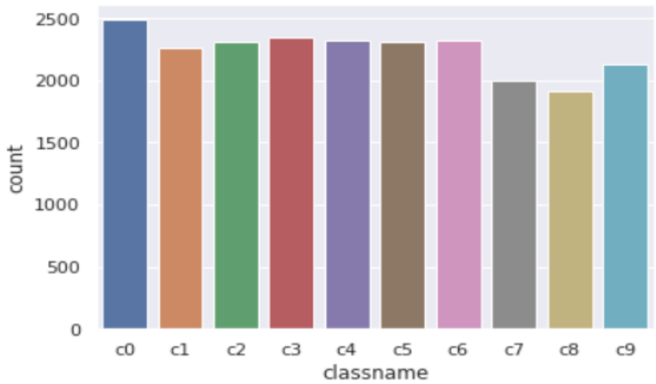
Many states have implemented strict laws prohibiting actions like texting, talking on the phone, and other distracted driving behaviors in the current context of road safety. We are convinced that by strategically implementing the increased safety measures that are urgently needed, using machine learning techniques, we can categorize driver photos into distinct distracted classes and so accurately determine the likelihood of future incidents. One essential component of this endeavor is the careful gathering of data for additional in-depth study.

The capacity to promptly identify and signal instances of distracted driving in the context of a smart car has great potential to reduce the number of traffic accidents. With the use of cutting-edge computer vision and machine learning algorithms, we want to precisely recognize and understand a variety of activities and actions taken by drivers. Our ultimate goal in doing this is to proactively lower the frequency of accidents by promptly identifying cognitive driver behaviors that depart from safe driving standards. Road safety is crucial in today's dynamic and changing transportation sector, and a considerable reduction in accidents is attainable with the effective and real-time use of this technology.

2. METHODOLOGIES

1.Data Preparation:

The training data is loaded and pre-processed by iterating through each class in the provided classes list. For each class, images in the corresponding directory are read using OpenCV (cv2) and converted to RGB format to ensure consistency. Subsequently, each image is resized to a specified size (img_size). The resized image, along with its class label, is then appended to the training_data list. Similarly, for the testing data, images in the test directory are iterated through, read, converted to RGB, and resized. These resized images are stored in the test_array for further processing.



International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 04 | April - 2024 SJIF Rating: 8.448 ISSN: 2582-3930

2.MODELS

a) Traditional CNN : The provided code utilizes Matplotlib.pyplot for visualization, scikit-learn for data splitting and evaluation, and TensorFlow and Keras for model building and training. It employs layers such as Conv2D, MaxPooling2D, Dense, Dropout, Flatten, Activation, and BatchNormalization from TensorFlow.keras.layers to construct a convolutional neural network (CNN) architecture. The model is trained using techniques like train_test_split for data splitting, classification_report for evaluation, and callbacks for monitoring training progress. This comprehensive approach enables the creation and assessment of a CNN model for image classification tasks.

b) DenseNet : The DenseNet code initializes a DenseNet-121 model pre-trained on ImageNet, known for its dense connections between layers, facilitating feature reuse and gradient flow. It then freezes the base layers to prevent weight updates during training, leveraging pre-trained features while training only the classifier layers. The existing classifier is replaced with a new one, including a dropout layer for regularization and a fully connected linear layer for mapping features to the output classes determined by the `class_names` variable. For training, the code initializes a CrossEntropyLoss function for multi-class classification, employs Stochastic Gradient Descent (SGD) optimizer with a learning rate of 0.005 and momentum of 0.9, and uses a step scheduler to decrease the learning rate by a factor of 0.1 every 7 epochs, aiming to improve convergence and achieve better performance.

c) EfficientNet : The EfficientNet code likely imports necessary libraries including `torch`, `torch.nn`, `optim`, `lr_scheduler`, and `torchinfo.summary`. It initializes the EfficientNet-B0 model using pre-trained weights from `torchvision.models.efficientnet_b0` and moves it to a specified device for computation. Base layers are frozen to retain pre-trained knowledge and reduce computational cost during training. The existing classifier is replaced with a new one, featuring a dropout layer for regularization and a linear layer mapping features to the output classes. Loss function initialization utilizes `CrossEntropyLoss` for multi-class classification, while optimization is performed with Stochastic Gradient Descent (SGD) with specified parameters. Additionally, a step scheduler is employed to adjust the learning rate during training for improved convergence and potentially better performance.

d) MobileNet : The MobileNet code likely imports necessary libraries including `torch`, `torch.nn`, `optim`, `lr_scheduler`, and `torchinfo.summary`, alongside importing MobileNetV3 from `torchvision.models`. It initializes the MobileNetV3 model with pre-trained weights and moves it to a specified device for computation. Base layers are frozen to retain pre-trained knowledge and reduce computational cost during training. The existing classifier is replaced with a new one, featuring a dropout layer for regularization and a linear layer mapping features to the output classes. Loss function initialization utilizes `CrossEntropyLoss` for multi-class classification, while optimization is performed with SGD. Additionally, a step scheduler is employed to adjust the learning rate during training for improved convergence.

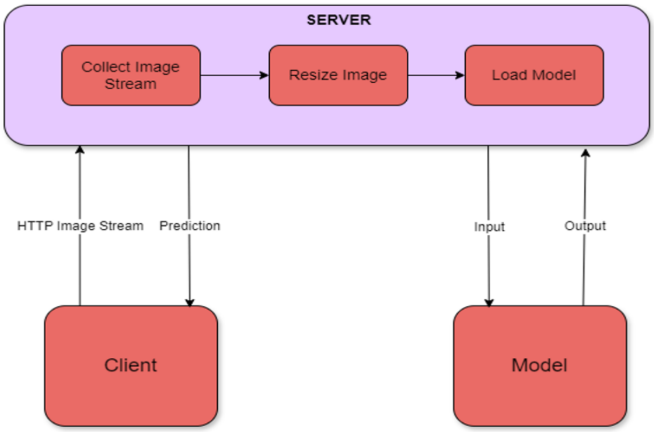
d) Ensembler : The Ensembler code likely imports necessary libraries including `torch`, `torch.nn`,optim`, `lr_scheduler`, and `torchinfo.summary`, alongside importing MobileNetV3 from `torchvision.models`. It initializes the MobileNetV3 model with pre-trained weights and moves it to a specified device for computation. Base layers are frozen to retain pre-trained knowledge and reduce computational cost during training. The existing classifier is replaced with a new one, featuring a dropout layer for regularization and a linear layer mapping features to the output classes. Loss function initialization utilizes `CrossEntropyLoss` for multi-class classification, while optimization is performed with SGD. Additionally, a step scheduler is employed to adjust the learning rate during training for improved convergence.

Table -1: Accuracy Report

	precision	recall	f1-score	support
0	0.94	0.82	0.88	74
1	0.91	1.00	0.95	61
2	0.89	0.99	0.94	68
3	0.93	0.93	0.93	76
4	0.84	0.99	0.90	67
5	0.92	0.94	0.93	63
6	0.97	0.90	0.93	77
7	1.00	0.97	0.99	68
8	0.86	0.85	0.85	52
9	1.00	0.88	0.94	67
accuracy			0.93	673
macro avg	0.93	0.93	0.92	673
weighted avg	0.93	0.93	0.93	673

3.BACKEND

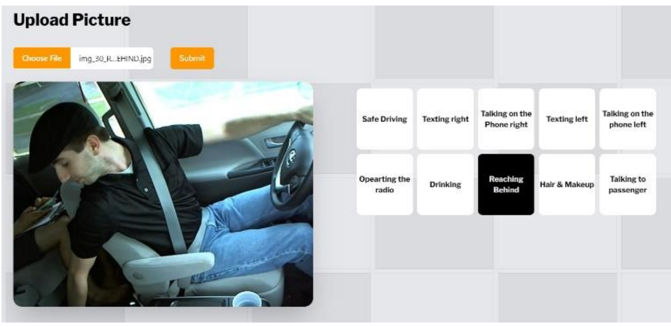
The backend API, implemented in Node.js, utilizes Express and TensorFlow.js to create an endpoint for image classification. The app loads a pre-trained model asynchronously using TensorFlow.js's loadLayersModel function. Multer is configured for handling file uploads with in-memory storage. The route **/classify** handles POST requests, expecting a single file upload named 'image'. If no file is uploaded, it returns a 400 status code with an error message. Otherwise, it decodes and resizes the image, preprocesses it, makes predictions using the loaded model, and sends the predicted class in the response.



4.CLIENT

On the right side, the team members' names and the coordinator's name are displayed. In the middle section, the project title is highlighted alongside an upload file option, accompanied by a box for displaying uploaded images. Users have the option to process uploaded images, and subsequently, the system converts the processed results from text to speech. The backend code features conditional statements designed for metadata and a trained traditional model. This integrated setup ensures efficient processing and conversion of data.

Fig -1: Figure



5. CONCLUSION

This Over the course of this study, a number of important successes and discoveries have been made, which taken together have cleared the path for a noteworthy progress in the field of cognitive driver action detection. With the help of this research, action recognition accuracy has significantly increased, achieving real-time efficiency as well as maximizing the potential of transfer learning and improving datasets. Taken as a whole, these accomplishments show off the amazing potential of deep learning and computer vision techniques to transform traffic safety and reduce the dangers of distracted driving. The project's progress has reached a significant milestone with the improved accuracy in action recognition. We have improved the accuracy of distinguishing different driver actions and greatly decreased false positives by fine-tuning algorithms and models. This increased precision is essential for guaranteeing that the system can respond to urgent circumstances promptly and consistently, which will improve overall traffic safety. Another remarkable project accomplishment is its real-time efficiency. Real-time processing and recognition of driver actions is essential for quickly warning drivers of possible dangers and enabling quick reactions. Real-time efficiency increases the system's usefulness and applicability in real-world situations, in addition to increasing its efficacy. The accomplishment of this project has been made possible in large part by transfer learning and dataset improvement. By utilizing pre-trained models and augmenting the dataset with varied and demanding scenarios, we have improved the system's capacity to identify an extensive array of driver actions. This method has sped up the development process and increased the system's flexibility to accommodate various driving scenarios and settings. To sum up, this work is an impressive step forward in the field of cognitive driver action identification. It has demonstrated the potential to significantly improve road safety by pushing the limits of accuracy, efficiency, and adaptability. The project is evidence of the revolutionary potential of deep learning. Using computer vision approaches in tackling the crucial
problem of driving while distracted, and it emphasizes the need to allocate significant hardware resources in order to properly fulfill the project's potential influence on road safety.

6. FUTURE SCOPE

The system's ability to classify driver actions in real time will be further enhanced by the continued development of progressively more complex and precise models made possible by advances in computer vision and machine learning. Furthermore, the combination of edge computing with IoT technologies could encourage the deployment of these systems on a larger scale. facilitating their smooth interaction with other smart car parts for increased safety. Moreover, the integration of multi-modal data sources, such sensors and in-car cameras, may offer a comprehensive comprehension of the driver's actions. With the advancement of technology, it is possible that distracted driving detection and accident prediction will be included, providing preventative safety measures. These devices could soon have a significant impact on how road safety. is shaped and help in a new era of preventive driver aid technologies.

ACKNOWLEDGEMENT

It gives me great pleasure to convey my profound gratitude and appreciation to my guide, Prof. S. P. Jadhav, for his invaluable direction throughout the job, without which it would have been an extremely challenging undertaking. There are no enough words to convey how much I appreciate your invaluable advice, tremendous help, and collaboration. extended to every employee in my department. Without giving special gratitude to Dr. S.R. Ganorkar, Head of the Department of Information Technology, for his assistance during the task, this acknowledgement would not be complete.

REFERENCES

[1] Chen Huang, Xiaochen Wang, Jiannong Cao, Shihui Wang, Yan Zhang, "HCF: A Hybrid CNN Framework for Behaviour Detection of Distracted Drivers", 2020, IEEE.
[2] Vanessa Buhrmester, David Münch, Michael Arens, "Analysis of Explainers of Black Box Deep Neural Networks for Computer Vision: A Survey ", 2021, MDPI.
[3] Sadbhavana Babar, Sukhendu Das, " Where to Look? Mining Complementary Image Regions for Weakly Supervised Object Localization ", 2021, IEEE.
[4] Sai Yeshwanth Chaganti, Ipseeta Nanda, Koteswara Rao Pandi, Tavva G.N.R.S.N. Prudhvith, Niraj Kumar, " Image Classification using SVM and CNN ", 2020, IEEE.
[5] Ding-Xuan Zhou, " Theory of deep convolutional neural networks: Down sampling", 2020, IEEE.