

An approach to classify distraction driver detection system by using mining techniques

Reddy Shiva Shankar, Pilli Neelima, Voosala Priyadarshini, Swaroop Ravi Chigurupati

Department of Computer Science and Engineering, Sagi RamaKrishnam Raju Engineering College, Andhrapradesh, India

Article Info

Article history:

Received May 15, 2022

Revised Jun 14, 2022

Accepted Jul 1, 2022

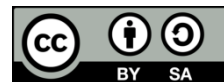
Keywords:

Artificial intelligence
Computer vision
Convolutional neural network
Distraction driver detection
Machine learning

ABSTRACT

According to the motor vehicle safety division, over the past 5-10 years, usage of motor vehicles has rapidly increased, in that specific usage of cars has grown tremendously. The major contribution of this paper is a systematic evaluation of the scholarly literature on driver distraction detection techniques. Our driver distraction detection framework offers a systematic overview of evaluated methodologies for detecting driver attention. So, we need to develop a model that classifies each driver's behaviour and determines its corresponding class name. To overcome this dispute, we have attained an appreciable number of deep learning algorithms on the dataset like convolutional neural network (CNN) and VGG16 to detect what the driver is doing in the car as given in the driver images. This process can be done by predicting the likelihood of the driver's actions in each picture. Of all models, we distinguished that the VGG16 Algorithm has conquered CNN with a loss of 0.298 and an Accuracy of 91.7%.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Reddy Shiva Shankar

Department of Computer Science and Engineering, Sagi RamaKrishnam Raju Engineering College

Bhimavaram, West Godavari District, Andhrapradesh, India

Email: shiva.shankar591@gmail.com

1. INTRODUCTION

Computer vision is an area of artificial intelligence (AI) that offers how computers can gain a higher level of cognition for digital vision. Computers are trained to perform all the easily performed actions by the human visual system. The basic working steps of computer vision are: i) Obtaining an image, ii) Image processing, and iii) Recognizing the image. Computer vision can solve more complex facial recognition and complex image analysis problems. Computer vision applications are objected to recognizing and detection, self-driving cars, and face recognition, as shown in Figure 1.

Examples of computer vision tasks are:

- Image classification: Classifies the images into various classes (safe driving, texting, talking on the phone).
- Image detection: Used to identify particular distractions inside the images.

Many people die yearly in car accidents because drivers aren't paying attention or are distracted while driving. Car manufacturers, suppliers, start-ups, and academics are investing more and more resources to understand better and evaluate driver distraction and inattention. Thus, they create methods to alert and prevent drivers from being distracted, such as automated driving features, or improve the vehicle's degree of automation. Driving weariness and other forms of driver attention may already be detected and signalled to the driver in certain contemporary cars over a specific class or level of equipment. Driver assistance systems, including vehicle automation systems like adaptive distance keeping or lane-keeping, often refer to these technologies. As the driving function becomes more automated, more sophisticated technology will be

integrated into automobiles to achieve completely autonomous or automated driving. Research on driver distraction detection and related themes has increased in recent years.

Nowadays, the primary cause of accidents is distracted drivers. Over 1.3-1.35 million people are dying due to this distraction. The driver's inattention may be texting, answering a call, operating the radio, looking behind, adjusting hair, applying makeup, or talking to passengers, as shown in Figure 2. To detect the distraction, we use machine learning techniques like CNN so that when an image is given as input, it provides the corresponding detection causes output.



Figure 1. Computer vureision

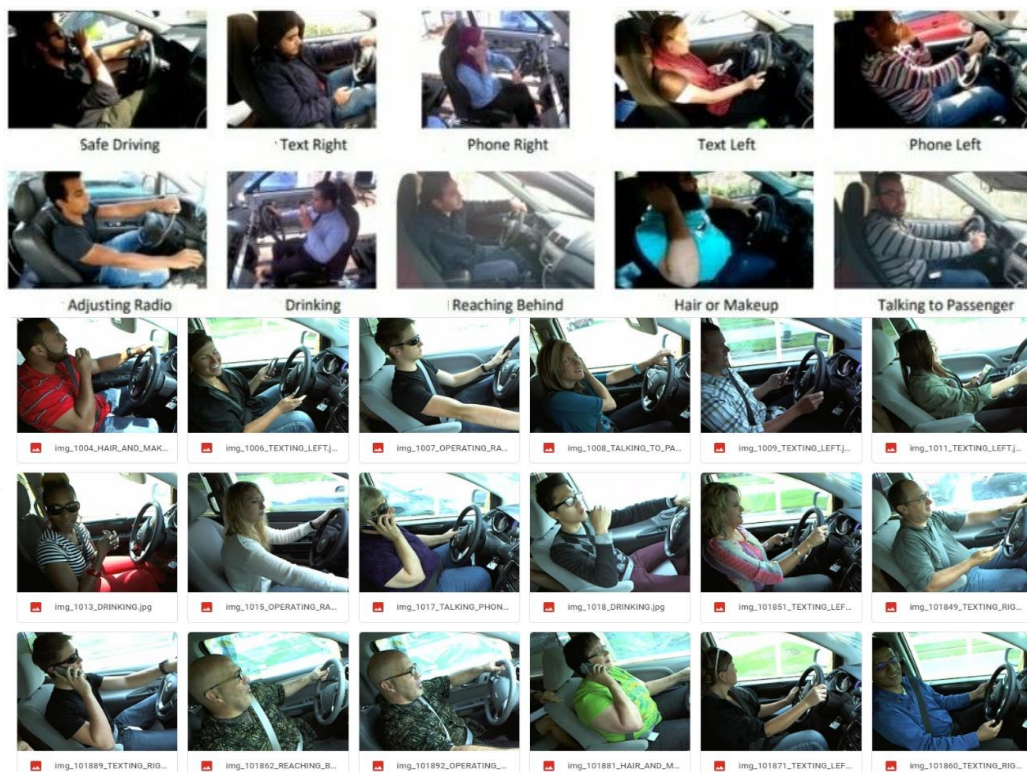


Figure 2. Different driver postures

2. PROBLEM STATEMENT

Over the past 10-15 years, car usage has increased rapidly and simultaneously, and road accidents are also increased. According to the National Highway Traffic Safety Administrator (NHTSA) survey, nearly 2 of 10 accidents are caused by distracted drivers. Distracted driver detection uses convolutional neural network (CNN) in computer vision, and we present a CNN-based system that detects the distracted driver and classifies the cause of distraction. This project aims to apply advanced techniques and evaluation metrics to evaluate whether the provided image classifies the best detection, and based on that, we can warn the distracted driver. Thereby accidents may reduce, and we can save humankind.

An approach to classify distraction driver detection system by using ... (Reddy Shiva Shankar)

3. RELATED WORK

Omerustaoğlu *et al.* [1] proposed five image classification strategies. The dataset contains 22,424 images categorized into ten classes from State Farm. Initially, they started by processing the photos if required. Then they used the multi-class linear support vector machine (SVM) method. The input for each image is a flattened vector representing one image obtained after pre-processing. Later, they use different models like soft max, naive Bayes (NB), decision tree (DT) and two layered neural network (NN) to classify the distraction. To distinguish those models, they have used accuracy as a performance measure. Among all the explored algorithms, Two Layer Net has the highest accuracy score at 90%, but it performs poorly in "Talking to the Phone-Left" and "Hair and Makeup" classes. Costa *et al.* [2] used a non-intrusive AI-based monitoring system to identify driver weariness, distraction, and activity. This experiment used 20 volunteers to observe the different activities with different scenarios. Here the data is divided into three distinct parts for different phases. They applied SVM and DT models to the dataset. From the experiment, DT proved more efficient in recognizing fatigue situations with an Accuracy of 93%, which is 2% more than SVM.

Chawan *et al.* [3] proposed different models of CNN for domain-driven design (DDD) and classification. Data is collected from Kaggle, which state Farm Company releases. The used models are Small CNN, VGG16, VGG19 and InceptionV3. The library in the DL used for this purpose is Keras, which runs on top of the tensor flow. Liang and Lee *et al.* [4] approached a hybrid Bayesian Network to detect driver cognitive distraction. The data set used in this experiment was collected by observing nine people (19 years of experience in driving) doing various activities during the driving session. Then three detection algorithms, such as the layered algorithm, original dynamic Bayesian network algorithm, and SVM Algorithm, are applied to the dataset. The results of the Layered experiment algorithm have shown better computational efficiency than the other two algorithms. From the experiment, it is included that data mining methods will have more efficiency. Jin *et al.* [5] created models for identifying the driver's cognitive distraction condition based on driving performance metrics. The data was taken using 12 people doing different activities in different phases. Low cognitive distraction (LCD), no cognitive distraction (NCD), and high cognitive distraction (HCD) were used to build distinct SVM models such as the NL Model, NH Model, LH Model, and NLH Model (HCD). Each SVM model in this experiment has a mean accuracy of around 74%, nearly similar to driving performance activity in the trial, allowing it to monitor the driver's cognitive state. For classification of distracted driver Abouelnaga *et al.* [6] suggested a genetically-weighted ensemble of CNN. The experiment is based on the State Farm DDD using the Kaggle data set. Face and hands detection, CNN, and a Weighted ensemble of classifiers employing genetic algorithm (GA) were employed in this experiment. The best result came from a genetically weighted ensemble of CNN, which had 95.98 % accuracy. Atiquzzaman *et al.* [7] focused on developing techniques for detecting distractions in driving by using two activities such as vehicle control and driving performance measures. The data was collected from 35 participants using simulated driving scenarios in 9 phases. From that, 3 stages were randomly picked and used for the experiment. He used the DM techniques such as latent dirichlet allocation (LDA), logistic regression (LR), and SVM to achieve the results. The investigation found that SVM has produced an Accuracy of 84.33% and 79.53% for texting and eating distractions.

Tran *et al.* [8] developed a driver distraction detection system and conversational warning system based on deep learning that identifies drivers' activities and alerts them. They used a dataset which consists of different images of drivers with standard and distracted driving postures. The development and implementation authors included four DCNNs such as VGG-16, Alex Net, Google Net and residual network. Google Net produced a good accuracy of 89% of these four networks and is considered the best model for driver distraction detection systems. Shankar *et al.* [9] proposed an approach for detecting the rider who was not wearing a helmet by reading the bike number plate number from the recovery time objective (RTO) database. By transmitting the detecting the position details, it keeps a database for the number of detections of that particular vehicle. Using CNN, Yasaswini *et al.* [10] proposed work to identify the factors contributing to fatal accidents. To enhance the accident occurrences could be determined by considering various criteria. They have considered the FARS dataset by generating risk factors which cause fatal accidents. Dhakate and Dash [11] proposed DDD System from distracting activities using ensemble techniques. They used the Stacking process to DDD posture in a real-time environment and got 9% accuracy. Sajid *et al.* [12] used the proposed efficient model's DDD dataset containing eight classes. They have used five variants in the efficient model for DDD. It achieves Map Of 99.16% with parameter setting. It will help potentially te drivers to maintain safe driving habits.

Alzubi *et al.* [13] proposed a unique technique for detecting driver distraction caused by in-hand electronic devices. This work has been submitted as accurate, energy, and memory efficient. Because they impair a driver's ability to make quick decisions and reduce their reaction time in emergencies, portable device distractions have been a critical factor in several fatal traffic incidents. Many incidents may have been averted if drivers were more alert instead of trying to multitask while behind the wheel. Compared to the

heavyweight network's accuracy, which was 0.86, our compressed neural network had an accuracy of 0.83. Leekha *et al.* [14] presented an architecture using CNN. They have used the ConvNet model on two datasets, SFD3 and AUCD2, with an accuracy of 98.48% and 95.64%. Their model can identify distractions in real-time without the need for parallel processing.

Oliveira *et al.* [15] conducted a comparative study of three learning techniques on concentration or distraction driver moments. VGG19, Inception v3, Resnet152 and Densenet161 were evaluated on deep CNN. Eraqi *et al.* [16] presented DDD with existing postures. They proposed a reliable deep learning-based model to achieve 90% accuracy. They investigated numerous visual aspects in DDD using face and hand localizations and skin segmentation, with an accuracy of 84.64 % in a real-time scenario. Social group optimization (SGO) and APSO are employed in the threshold function to coordinate an adaptive thresholding-based wavelet denoising approach [17]. The method of colouring an image has a stronger influence in various fields, such as astronomical photography, electronic microscope visualizations, and CCTV surveillance, to name a few of these fields' applications. The deep learning (DL) approach is used to develop an automated system for colour grayscale photographs. This study Shankar *et al.* [18] focuses on analyzing CNN's programming and content.

Devareddi and Srikrishna [19] conducted a poll on CBIR to determine the extent to which it provides a holistic view. The input is then converted into a feature map, which generates latent sharp images at each layer and delivers the final sharp image at full resolution by converting back into the original format at the decoder. The input is then down-sampled at the encoder, which results in a series of fuzzy images at various resolutions. At the decoder, the original format is converted back into. As a result, the findings are more stable and lower time complexity [20].

Mase *et al.* [21] proposed a new driver DDD that captures spectral-spatial characteristics of images utilizing CNN and stacked bidirectional long short-term memory (LSTM) networks. They presented a two-stage methodology: first, they used pre-trained CNNs to extract spatial posture information, and then they used BiLSTMs to remove spectral features from the pre-trained CNNs. They could attain an average classification accuracy of 92.7% using the model. Shankar *et al.* [22] used YOLOv4 to detect the object performance, and they did the work on facial expressions using Bezier curves.

Torres *et al.* [23] proposed a CNN that can detect and classify a motorist using a mobile phone video surveillance. They could get a DDD accuracy of 99% by employing this approach. Arun *et al.* [24] used vehicle-based, physiological, and behavioural markers to determine the level of driver inattention. They created a mechanism that activates the alert system only when the driver's distraction level exceeds the allowed limit. Karel explored DDD by Hurts *et al.* [25] as a road safety problem. Their findings were based on primary driving tasks and additional tasks unrelated to driving. They've also looked towards determining what caused the distraction. They looked at current research and measurement models in distraction research recommendations, standards, anti-distraction gadgets, and legislation. Finally, they talked about driver distraction related to integrated safety vision.

Mofid *et al.* [26] used the sensor fault detection and diagnosis (SFDD) dataset to solve the problem of distracted driving by creating a multi-class classifier that could detect and classify different types of driver inattention. Skin segmentation, facial blurring, and classic augmentation techniques are all used in this model. These techniques are used to boost NN's performance on DDD tasks.

4. RESEARCH METHOD

The work process follows in the Figure 3 flow chart analysis, and the DDD flow of structure is shown in Figure 4. It includes the collection of distracted driver data used for detecting the inactiveness of the driver and classifying the data into 10 different classes using deep learning algorithms. Then the proposed VGG16 Algorithm is applied to the dataset to classify the driver's behaviour.

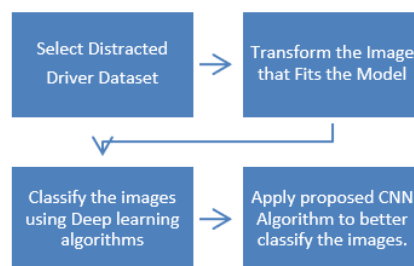


Figure 3. Flow chart analysis of work

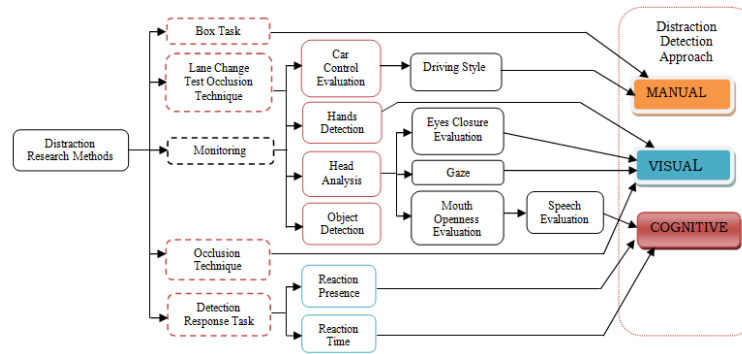


Figure 4. Structure of DDD

4.1. Objectives

Manually collect the data from various sources, merge them with related features, clean the dataset, pre-process it, or download it from different resources. To divide the dataset into different classes based on their features. Compare various machine learning (ML) models based on different metrics and select the best model for detecting drivers' distractions. Using evaluation measures, assess the model's performance. The proposed CNN algorithm is applied to the driver images to classify them into the corresponding classes.

4.2. Data exploration

Distraacted Driver images were taken from a Kaggle (state farm distracted driver detection competition) of 102,150 images with 10-class labels. The dataset comprises colour photos that vary in height and width between the low hundreds to low thousands. Among the 102,150 images, 17,939 training images, 4485 validation images, and 79,726 training images. All images used for training and validation fall into one of the ten categories listed in Table 1. The photos are 640×480 pixels in size and colour.

Table 1. Dataset description

Class	Description	Class	Description
C0	Safe Driving	C5	Operating the Radio
C1	Texting-Right	C6	Drinking
C2	Talking on the Phone-Right	C7	Reaching behind
C3	Texting-Left	C8	Hair and Makeup
C4	Talking on the Phone-Left	C9	Talking to Passenger

First, we must define the testing, training and model directories. It will be done with the help of the os module in python. If paths are not present, we will create the directories for train, test, and model training. For easy tracking of files, training images with their associated classes and testing images presented for. We create a CSV file. Here, the train data has the filename for the corresponding image, and then we have its corresponding class name. We identify the samples for each corresponding class shown in Figure 5. To get the count number of samples per class, we use each class's value_counts() method. Then we will find the presence of duplicate data. Now the data is almost balanced. Then we convert CLASS LABELS into numerical values. These values are future used under the meaning of their corresponding class. Training samples are further divided into training and validation samples in the 80 and 20. Then what we are doing is we are translating the square unit size image into the 64*64 image vector, and then we will flatten it.

```

labels_list = list(set(data_train['ClassName'].values.tolist()))
labels_id = {label_name:id for id,label_name in enumerate(labels_list)}
print(labels_id)
data_train['ClassName'].replace(labels_id,inplace=True)

{'c6': 0, 'c5': 1, 'c1': 2, 'c8': 3, 'c9': 4, 'c3': 5, 'c7': 6, 'c2': 7, 'c4': 8, 'c0': 9}
    
```

Figure 5. Feature used under the meaning of their corresponding class

Here stack collects a complete matrix where each row has a feature vector. So, before we build our model and start the training process, we should pre-process. The photos are separated into two groups during pre-processing: training and validation. The photos are then downsized to 64*64 pixel square images. Because all the images are coloured, we employ all three channels in the training process. By dividing each pixel in each image by 255, the images are normalized. A value of 0.5 is deducted from the mean to guarantee it is zero.

4.3. Algorithms used (convolutional neural networks)

A neural network has three layers. It consists of input layer, hidden layer and output layer. Each layer was explained in the following section. The architecture of the neural network is shown in Figure 6.

- Input layer: this layer contains the model's input. The total number of features in our dataset equals the number of neurons in this layer.
- Hidden layer: the hidden layer's input comes from the input layer's output. Many hidden layers can exist depending on our model and data set. The number of neurons in each hidden layer outnumbers the number of features in each hidden layer. We employ a matrix multiplication method with some biases, followed by an activation function, which makes the network nonlinear, to determine the output.
- Output layer: the output layer takes the input from the output of the hidden layer. Now, the input of each class is transformed by using the logistic function like sigmoid or SoftMax based on each class probability. After passing the data to the feed-forward model, each layer's output is obtained. The error is then calculated using error functions like square loss and cross-entropy. After that, backpropagation is done to minimize the loss. Neural networks which share their parameters are convolution neural networks, simply CNN. With the help of CNN, we can represent an image with its length, width and height.

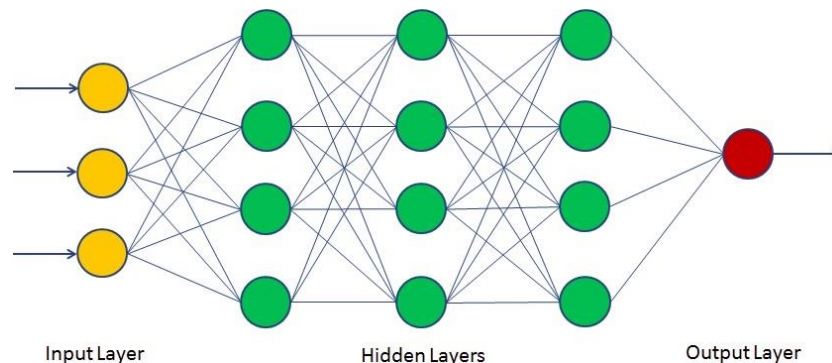


Figure 6. Neural network

Applying a neural network on a small patch of an image, as shown in Figure 6, represents the output vertically. Sliding is done on the whole image; we will get another image with different dimensions. We have more channels with lesser height and width, known as convolution, along with red, green, and blue (RGB) channels. It becomes a regular neural network if the patch size equals the image. We got fewer weights due to this small patch. Convolution layers contain a set of filters. Each filter has a small height and width. The depth is the same as the input volume.

- For example, we have to run convolution on an image having dimensions $12 \times 12 \times 3$. A $X \times X \times 3$ can be the filter possible, where 'a' is small compared to the dimension of the image.
- Sliding is done on every filter on the total input volume step by step during the forward pass. Here each step is called a stride. After that, we can calculate the dot product between the patch from the input volume and the weights of filters.
- As the sliding is done on every filter, we will get a 2-D output. The number of filters equals the depth of this output volume. Then the filters will be learned by the network.

Following are the layers used to build convolution neural networks:

A layer sequence in which each layer uses a differentiable function to transition from one volume to another. For example, CNN runs on a $16 \times 16 \times 3$ pixel image.

- Input layer: the input layer retains image input with width 16, height 16, and depth 3.
- ConvolutionLayer: the dot product between the filters and the image patch gives us the output volume by utilizing the convolution layer. If we apply eight filters to this layer, the output volume will be $16 \times 16 \times 8$.

- Activation function layer: this layer applies an activation function to the convolution layer's output. The activation functions include RELU, Tanh, Sigmoid, and others. The output volume has the exact dimensions as the input volume: $16 \times 16 \times 8$.
- Pooling layer: it is used in CNN. The function of the pooling layer is to reduce volume size, making the calculation fast. It also reduces memory, and also there may be no overfitting. Max pooling and average pooling layers are present in the pooling layer, as shown in Figure 7.

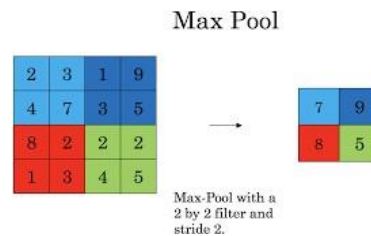


Figure 7. Dimensionality reduction in max-pooling layer

- Fully-ConnectedLayer: it takes the input from the pooling layer and calculates the output volume.
- Epoch: The whole iteration on all the training data, or simply the forward and backward pass on all the training data. The epoch must be determined based on the batch size and data count.
- Steps per Epoch: it's only the count of the image dataset divided by the batch size. It can be stated as follows: If the dataset comprises 10,000 images and the batch size is 100, the steps per epoch will be $10000/100=100$.
- Batch-Size simply refers to the number of training samples taken for a single forward and backward pass. They can be 10, 32, 64, 100, or any other number. However, increasing the batch size causes the RAM to fill up, making the process longer to complete.

Here we created a standard CNN architecture, as shown in Figure 8. Initially, it was created and trained on input data (i.e., training and validation data). Here we created four convolutional layers, and four Max pooling layers in between filters are increased from 64 to 512 in every convolutional layer. The dropout layer is used alongside the Flattening layer before the Fully Connected layer. Altogether the CNN model has 2 FCL. Several nodes in the last follicle center lymphoma (FCL) were set up as 10 (because we have 10 different classes). The softmax and rectified linear unit (ReLU) activation functions are used for all the other layers. Xavier initialization was used in each of the layers.

Image Net is one of the Image classification models. It categorizes and labels the images into nearly 22,000 different object categories for computer vision (CV) research. Majorly we use this Image classification model in DL and CNN. The main goal is to train the model that classifies the image correctly into 1,000 different object categories. These categories are classes of our daily lifestyle. It includes species like cats, vehicle types, dogs, and various objects. The pre-trained models in CNN on ImageNet are: i) VGG16, ii) VGG19, iii) ResNet50, and vi) inception.

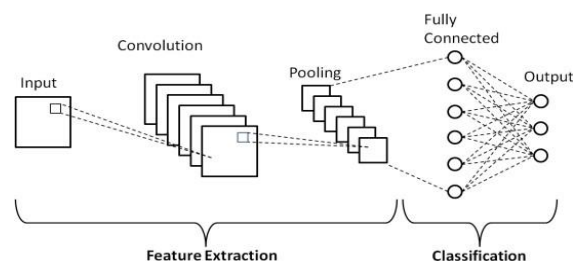


Figure 8. CNN architecture

4.3.1. Model architecture

The top layer of the VGG16 model was removed in this instance. As a result, the flattened-image vector predicts the last layer using the n-1 layers of VGG16. The final layer is a dense layer, which represents

the features of a specific image. This layer feature is being routed through a global average pooling 2D Layer. Each of the ten classes has its dense softmax layer.

After this, for improvement in the loss, transfer learning was applied to VGG16 model architecture showed promising results and was improved further by using the below techniques, and architecture was shown in Figure 9.

- A dropout layer was added to account for overfitting.
- We use Xavier initialization instead of random initialization of weights.
- Zero mean was ensured by subtracting 0.5 during pre-processing.
- The training was carried out with 450 epochs and with a batch size of 16.
- VGG16 and the Model Architecture are selected for further improvement of the loss metric, and fine-tuning is applied.
- The SGD optimizer was utilized with a 1e-4 learning rate. With SGD, a momentum of 0.9 was applied. The SGD optimizer was used with a 1e-4 learning rate. With SGD, a speed of 0.9 was involved.

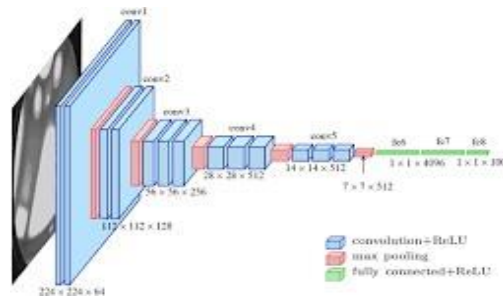


Figure 9. Architecture for VGG16

5. RESULTS ANALYSIS

The results of the models created are analyzed to assess the performance of each model on the dataset. Classification metrics like accuracy, precision, recall and f1-score were used to evaluate the performance of the models trained and tested on the dataset. We can see that almost training loss and validation loss are decreased, and in the same way, we are improving training and validation accuracy. From Table 2, we can compare models with accuracy and loss. The accuracy score for the model is 89.48%, and the loss score is 0.377. The VGG16 Algorithm surpassed CNN with a loss of 0.298 and a 91.7% accuracy.

We can see that almost training loss and validation loss are decreased. In the same way, we are getting improvements over training and validation accuracies shown in Figures 10 and 11. Table 2, Comparing VGG16 with the CNN graph of accuracy in Figure 12 and various evaluation metrics were shown in Figure 13.

Table 2. Evaluated evaluation metrics

Algorithm	Accuracy	Precision	Recall	F1-score	Loss
CNN	89.48	89.56	89.8	89.02	0.377
VGG16	91.7	91.85	91.76	91.08	0.298

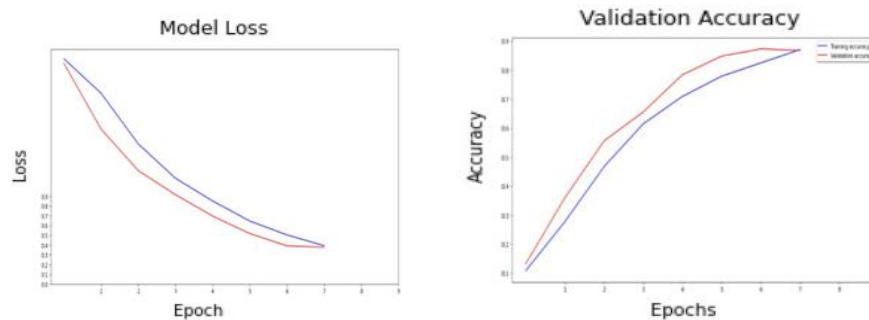


Figure 10. Plotting of training loss, validation loss for CNN

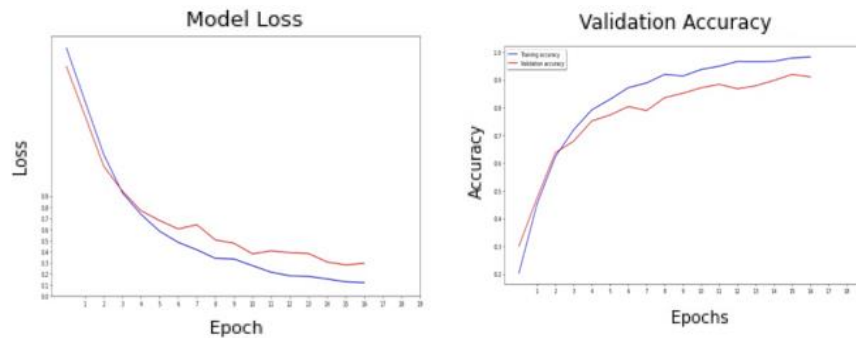


Figure 11. Plotting of training loss, validation loss for VGG16

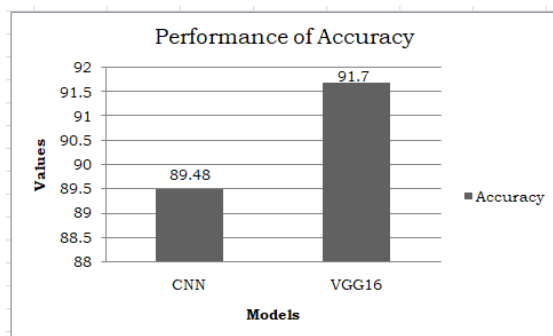


Figure 12. Comparison graph for accuracy

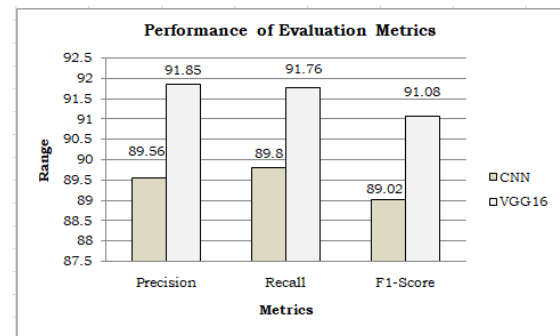


Figure 13. Comparison graph for accuracy

6. CONCLUSION

The scientific literature on distracted driving was examined in this research. Several previously studied methods have been incorporated into a single framework to identify different forms of driver distraction. The framework's description includes data collection, processing, inference of behaviour, and differentiation between types of distractions. Therefore, this article serves as an overview for scholars interested in driver distraction research and driver distraction detection systems, making the study necessary for practitioners of the transportation system. It can be done by predicting the likelihood of the driver's actions in each picture. Each classifier has its Accuracy and time requirements. Consider that if the CPU is replaced with a GPU, the classifier can execute with greater accuracy and less time, resulting in superior results. VGG16 gives more accuracy because it is the best ML Algorithm for image processing compared to other algorithms. The VGG16 Algorithm surpassed CNN with a loss of 0.298 and a 91.7% accuracy. The holistic driver distraction framework that we present summarises all contemporary ways of driver distraction and considers the most recent advanced computer vision techniques.





REFERENCES

- [1] F. Omerustaoglu, C. O. Sakar, and G. Kar, "Distracted driver detection by combining in-vehicle and image data using deep learning," *Applied Soft Computing Journal*, vol. 96, p. 106657, Nov. 2020, doi: 10.1016/j.asoc.2020.106657.
- [2] M. Costa, D. Oliveira, S. Pinto, and A. Tavares, "Detecting Driver's fatigue, distraction and activity using a non-intrusive Ai-based monitoring system," *Journal of Artificial Intelligence and Soft Computing Research*, vol. 9, no. 4, pp. 247–266, Oct. 2019, doi: 10.2478/jaiscr-2019-0007.
- [3] P. M. Chawan, S. Satardekar, D. Shah, R. Badugu, and A. Pawar, "Distracted driver detection and classification," *Journal of Engineering Research and Application*, vol. 8, no. 4, pp. 60–64.
- [4] Y. Liang and J. D. Lee, "A hybrid Bayesian Network approach to detect driver cognitive distraction," *Transportation Research Part C: Emerging Technologies*, vol. 38, pp. 146–155, Jan. 2014, doi: 10.1016/j.trc.2013.10.004.
- [5] L. Jin, Q. Niu, H. Hou, H. Xian, Y. Wang, and D. Shi, "Driver cognitive distraction detection using driving performance measures," *Discrete Dynamics in Nature and Society*, vol. 2012, pp. 1–12, 2012, doi: 10.1155/2012/432634.
- [6] Y. Abouelnaga, H. M. Eraqi, and M. N. Moustafa, "Real-time distracted driver posture classification," *arXiv preprint*, 2017, [Online]. Available: <http://arxiv.org/abs/1706.09498>.
- [7] M. Atiquzzaman, Y. Qi, R. Fries, and T. R. Board, "Exploring Distracted driver detection algorithms using a driving simulator study," *Journal of Transportation Research Board*, p. 17p, 2017, [Online]. Available: <https://trid.trb.org/view/1439226>.




- [8] D. Tran, H. Manh Do, W. Sheng, H. Bai, and G. Chowdhary, "Real-time detection of distracted driving based on deep learning," *IET Intelligent Transport Systems*, vol. 12, no. 10, pp. 1210–1219, Dec. 2018, doi: 10.1049/iet-its.2018.5172.
- [9] R. S. Shankar, C. H. Raminaidu, D. Ravibabu, and V. Gupta, "A survey to raise the awareness of road accidents due to not-wearing helmet," *International Journal of Industrial Engineering and Production Research*, vol. 31, no. 3, pp. 367–377, 2020, doi: 10.22068/ijiepr.31.3.367.
- [10] L. Yasaswini, G. Mahesh, R. S. Shankar, and L. V. Srinivas, "Identifying road accidents severity using convolutional neural networks," *International Journal of Computer Sciences and Engineering*, vol. 6, no. 7, pp. 354–360, Jul. 2018, doi: 10.26438/ijcse/v6i7.354360.
- [11] K. R. Dhakate and R. Dash, "Distracted driver detection using stacking ensemble," in *2020 IEEE International Students' Conference on Electrical, Electronics and Computer Science, SCEECS 2020*, Feb. 2020, pp. 1–5, doi: 10.1109/SCEECS48394.2020.184.
- [12] F. Sajid, A. R. Javed, A. Basharat, N. Kryvinska, A. Afzal, and M. Rizwan, "An efficient deep learning framework for distracted driver detection," *IEEE Access*, vol. 9, pp. 169270–169280, 2021, doi: 10.1109/ACCESS.2021.3138137.
- [13] J. A. Alzubi, R. Jain, O. Alzubi, A. Thareja, and Y. Upadhyay, "Distracted driver detection using compressed energy efficient convolutional neural network," *Journal of Intelligent & Fuzzy Systems*, vol. 42, no. 2, pp. 1253–1265, Jan. 2022, doi: 10.3233/JIFS-189786.
- [14] M. Leekha, M. Goswami, R. R. Shah, Y. Yin, and R. Zimmermann, "Are you paying attention? detecting distracted driving in real-time," in *2019 IEEE Fifth International Conference on Multimedia Big Data (BigMM)*, Sep. 2019, pp. 171–180, doi: 10.1109/BigMM.2019.00-28.
- [15] F. R. da Silva Oliveira and F. C. Farias, "Comparing transfer learning approaches applied to distracted driver detection," in *2018 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*, Nov. 2018, pp. 1–6, doi: 10.1109/LA-CCI.2018.8625214.
- [16] H. M. Eraqi, Y. Abouelnaga, M. H. Saad, and M. N. Moustafa, "Driver distraction identification with an ensemble of convolutional neural networks," *Journal of Advanced Transportation*, vol. 2019, pp. 1–12, Feb. 2019, doi: 10.1155/2019/4125865.
- [17] V. M. Gupta, K. Murthy, and R. S. Shankar, "A novel approach for image denoising and performance analysis using SGO and APSO," *Journal of Physics: Conference Series*, vol. 2070, no. 1, p. 012139, Nov. 2021, doi: 10.1088/1742-6596/2070/1/012139.
- [18] R. S. Shankar, G. Mahesh, K. V. S. S. Murthy, and D. Ravibabu, "A novel approach for gray scale image colorization using convolutional neural networks," in *2020 International Conference on System, Computation, Automation and Networking (ICSCAN)*, Jul. 2020, pp. 1–8, doi: 10.1109/ICSCAN49426.2020.9262377.
- [19] R. B. Devareddi and A. Srikrishna, "Review on content-based image retrieval models for efficient feature extraction for data analysis," in *2022 International Conference on Electronics and Renewable Systems (ICEARS)*, Mar. 2022, pp. 969–980, doi: 10.1109/ICEARS53579.2022.9752281.
- [20] R. S. Shankar, G. Mahesh, K. V. S. S. Murthy, and J. Rajanikanth, "A novel approach for sharpening blur image using convolutional neural networks," *Journal of Critical Reviews*, vol. 7, no. 7, pp. 139–148, Apr. 2020, doi: 10.31838/jcr.07.07.22.
- [21] J. M. Mase, P. Chapman, G. P. Figueredo, and M. T. Torres, "A hybrid deep learning approach for driver distraction detection," in *International Conference on ICT Convergence*, Oct. 2020, vol. 2020–October, pp. 1–6, doi: 10.1109/ICTC49870.2020.9289588.
- [22] R. S. Shankar, L. V. Srinivas, P. Neelima, and G. Mahesh, "A framework to enhance object detection performance by using YOLO algorithm," in *International Conference on Sustainable Computing and Data Communication Systems, ICSCDS 2022 - Proceedings*, Apr. 2022, pp. 1591–1600, doi: 10.1109/ICSCDS53736.2022.9760859.
- [23] R. Torres, O. Ohashi, E. Carvalho, and G. Pessin, "A deep learning approach to detect distracted drivers using a mobile phone," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10614 LNCS, 2017, pp. 72–79.
- [24] S. Arun, K. Sundaraj, and M. Murugappan, "Driver inattention detection methods: A review," in *2012 IEEE Conference on Sustainable Utilization and Development in Engineering and Technology (STUDENT)*, Oct. 2012, pp. 1–6, doi: 10.1109/STUDENT.2012.6408351.
- [25] K. Hurts, L. S. Angell, and M. A. Perez, "The distracted driver," *Reviews of Human Factors and Ergonomics*, vol. 7, no. 1, pp. 3–57, Sep. 2011, doi: 10.1177/1557234X11410387.
- [26] N. Mofid, J. Bayrooti, and S. Ravi, "Keep your AI-es on the road: tackling distracted driver detection with convolutional neural networks and targeted data augmentation," *arXiv preprint*, 2020, [Online]. Available: <http://arxiv.org/abs/2006.10955>.

BIOGRAPHIES OF AUTHORS






Reddy Shiva Shankar     is an Assistant Professor at the Department of Computer Science and Engineering in Sagi RamaKrishnam Raju Engineering College, Bhimavaram, Andhrapradesh, India. He is pursuing PhD degree in Computer Science and Engineering with a specialization in Medical Mining and Machine Learning. His research areas are image processing, medical mining, machine learning, deep learning and pattern recognition. He published 30+ papers in International Journals and Conferences. S.S. Reddy has filed 04 patents. His research interests include image processing, medical mining, machine learning, deep learning and pattern recognition. He can be contacted at email: shiva.shankar591@gmail.com.






Pilli Neelima    is Assistant Professor at Sagi Ramakrishnam Raju Engineering College, Department of Computer Science and Engineering, India. She Received a B.Tech. degree in Sagi Ramakrishnam Raju Engineering College, Department of Computer Science and Engineering, in 2006. She holds an M.Tech. degree in Sagi Ramakrishnam Raju Engineering College, Department of Computer Science and Engineering, in 2010. Her research areas are cloud computing, fog computing, edge computing, machine learning and IoT. She can be contacted at email: neelima.p47@gmail.com.



Voosala Priyadarshini    is Assistant Professor at Sagi Ramakrishnam Raju Engineering College, Department of Computer Science and Engineering, India. She Received a B.Tech. degree from Sri Vishnu Engineering College for Women, Department of Computer Science and Engineering, in 2005. She holds an M.Tech. degree in Sagi Ramakrishnam Raju Engineering College, Department of Computer Science and Engineering, in 2010. Her research areas are cloud computing, fog computing, edge computing, machine learning and image processing. She can be contacted at email: priyavoosala@gmail.com.



Swaroop Ravi Chigurupati    is Assistant Professor at Sagi Ramakrishnam Raju Engineering College, Department of Computer Science and Engineering, India. He received B.Tech. degree in Sagi Ramakrishnam Raju Engineering College, Department of Information Technology in 2012. He holds an M.Tech. degree in Sagi Ramakrishnam Raju Engineering College, Department of Information Technology, in 2018. His research areas are Image Processing, Bioinformatics, Machine Learning, Deep Learning, and Data Mining. He can be contacted at email: raviswaroop.chigurupati@gmail.com.