Proceedings of the 2nd Indian International Conference on Industrial Engineering and Operations Management
Warangal, Telangana, India, August 16-18, 2022

Distracted Driver Detection: A Comparative Study Using CNN

Sujay H. Jagadale
M.Tech Student, Department of Production Engineering,
Veermata Jijabai Technological Institute (VJTI), Mumbai, 400019, India.
jagadalesujay@gmail.com, shjagadale_m20@pe.vjti.ac.in

Dr. P.R. Attar
Assistant Professor, Department of Production Engineering,
Veermata Jijabai Technological Institute (VJTI), Mumbai, 400019, India.
prattar@pe.vjti.ac.in

Abstract

Road accidents due to distracted driving have been on a rise in recent years. As per the Road Accident Report 2019, 11 people were killed each hour in India due to road accidents (Transport Research Wing, 2020). This makes it crucial to take measures to stop the number of road fatalities. It found that the major cause of these accidents was driver error. This paper proposed a solution to detect the distraction of drivers into different predefined classes. The use of different pre-trained Convolutional Neural Network (CNN) models viz. AlexNet, VGG16, and ResNet50, for the classification of distracted drivers according to the State Farm's Distracted Driver Detection challenge on Kaggle, are depicted in this paper, As well as MyModel is trained on Dataset consisting of different local driver's 2D dashboard camera images along with the State Farm's Dataset. After comparing the results of predefined models with MyModel, the best result has been found as a categorical cross-entropy loss of 0.2344 on the validation set and an accuracy score of 93.28%.

Keywords
Distracted Driver, Accident Prevention, CNN, Image Classification, and Deep Learning.

1. Introduction

Road Accidents are one of the leading causes of death, disability, and hospitalization of people worldwide in general and Indian particular. At least one out of 10 people killed on roads across the world is from India, according to the World Health Organization. As per the Report on 'Road accidents in India 2019', the accident-related deaths in India in 2019 were 1,51,113 in number and 451,361 injuries (Transport Research Wing, 2020). In this report, it is found that the major cause of accidents was 'Distracted Drivers Behavior', which includes major use of mobile phones while driving, eating, drinking, talking with nearby passengers, reaching behind, fixing hair, etc. To mitigate this problem, we classified driver's behavior into 10 different classes.

We input images of the driver into our model. Each image belongs to one of the 10 classes mentioned in the dataset section. The model then predicts the class of an image by giving an output as the probability for each class. If the software is integrated with the hardware, we can warn the driver if he's distracted or not, thus preventing accidents from happening. This paper compares the results of pre-trained models with MyModel made from scratch based on the categorical cross-entropy loss and Accuracy as a metric.

1.1 Related work

This challenge was hosted on the Kaggle platform by Statefarm Insurance company USA in 2016. Some recent works combine multiple models for distracted driver detection which include wearable sensors to measure physiological and biomedical signals such as brain activity, muscular activity, and heart rate. Due to the involvement of personnel and the cost of hardware this method failed (Fernández et al., 2016). Thereafter some solutions were based on the SVM model that detects the use of mobile phones while driving (Abouelnaga et al., 2018). The traditional methods use handcrafted features, specifically HOG and clustered SIFT descriptors using Bag of Words (BOW) passed into a tuned SVM classifier (Hssayeni et al., 2017). Some used hand and face

segmentation using RCNN (Mercanoglu et al., 2019). As fewer approaches based on deep CNN models pre-trained on AlexNet, ResNet-50, and VGG-16, we considered these algorithms for comparative study.

1.2 Objectives

This paper focuses on deep learning models using pre-trained algorithms like AlexNet, ResNet-50, and VGG-16, and one model from scratch namely MyModel. A comparison between my model and pre-trained algorithms has been depicted in this paper based on categorical cross-entropy or log loss and Accuracy as metrics.

1.3 Dataset

The dataset used for our research has been provided by the State Farms insurance company of the USA. They offered a dataset of 2D dashboard camera images for a Kaggle Public Challenge (Kaggle, 2016). The dataset had 22400 training images and 79727 testing images. The resolution of each image was 640 x 480 pixels. The sample input image is as in Figure 1.

Figure 1. Sample input image

The training images had corresponding labels attached. Labels belonged to one of the ten classes as mentioned below:
c0: normal driving
c1: texting – right
c2: talking on the phone – right
c3: texting – left
c4: talking on the phone – left
c5: operating the radio
c6: drinking
c7: reaching behind
c8: hair and makeup
c9: talking to the passenger

The goal was to predict the likelihood of what the driver is doing in each picture which is not the same as predicting the exact class of each driver. The training set consists of 22400 images which were split into train and validation sets. The training set was split in such a manner to ensure that the images in the validation set are not related to the images in the training set. This is because the images are highly correlated to each other.

2. Literature Review

Driver Distraction has been a major problem of concern in recent years which increases the risk of road accidents considerably. Various literature found solutions to the problem based on wearable sensors and computer vision. Most of the vision-based approaches used a two-step structure which includes extraction of features from raw data using hand-crafted methods and classifiers are fitted with hand-crafted features (Dunn et al., 2021). This two-step architecture cannot gain an optimal trade-off between the robustness of the trained classifier and the distinct hand-crafted features. In the last two decades, vision-based approaches to detect distracted drivers that are based on support vector machines (SVMs) and decision trees have dominated the research area. Recently, great research in deep learning especially convolutional neural networks (CNNs) has become the dominant approach to solving the distracted driving problem.

Satardekar et.al. (2018) studied several CNN models, their best ensemble was made after averaging the probabilities generated by VVG-16, VVG-19, and Inception V3. They got a log loss of 0.795.

Behera et al. (2019) used both AUC and State Farm's distracted driver detection dataset with sequence information of the images, i.e., they used the video version of the dataset. They proposed a novel deep neural network approach called Multi-stream LSTM (M-LSTM). They used appearance features and contextual information (e.g., pose and objects) obtained from another pre-trained CNN with distinct combinations as multi-stream inputs for their M-LSTM. They achieved 52.22% and 91.25% accuracy on AUC and State Farm's distracted driver datasets, respectively.

Alotaibi and Alotaibi (2020) prepared a model and applied on to State Farm distracted driver and AUC datasets. They got better performance than both ResNet and HRNN. Their method obtained 92.36% ResNet obtained 88.52%, and HRNN achieved 84.85%. So, their method is better than ResNet by around 3.80% and better than HRNN by around 7%.

Xing et al. (2019) have classified seven actions as normal driving, left mirror checking, right mirror checking, rear mirror checking, using the in-vehicle devices, answering the phone, and texting. They cropped the driver's body from the images and then applied the Gaussian Mixture Model (GMM) to extract the body features from the background. They used different types of CNN to classify the actions. The best result was achieved by using AlexNet as 91% in binary format predicting whether the driver was distracted.

Omerustaoglu et al. (2020) found that fusing sensor data with vision data increases the accuracy of distracted driver detection tasks successfully. Specifically, both hybrid and prediction level fusion increased the overall accuracy by 9%, from 76% to 85%, when compared to using only image data. It is found that using sensor data increased the normal driving detection accuracy from 74% to 85%. The statistical tests showed that both of these differences are significant.

3. Methodology

In this paper, we have used pre-trained deep learning models, viz. AlexNet, VGG-16, and ResNet50. Our model was developed by adding different layers with specific parameters to the sequential model. Pretrained models were used because they have been trained on a very large dataset (ImageNet), which has 1.2 million images with 1000 distinct classes. We have used the initial layers of pre-trained models, making their training off to gain model weights for our dataset. The reason is that the initial layers of the model include edge detection and shape detection modules, which are generalized for any image recognition application, and these become increasingly more abstract in the final layers, making them more specific to the application.

In the distracted driver case, the last layer gives an output of one of the 10 classes for a given image. Data has been split into training (80%) and validation (20%) sets. We performed all work on a notebook provided by Kaggle because it provides free access to NVIDIA K80 GPUs in kernels. That enabling a GPU to our Kernel results in a 12.5X speedup during the training of a deep learning model.

3.1 Convolutional Neural Network (CNN)

A neural network is a layered architecture containing neurons. A Convolutional Neural Network (CNN) is a deep learning algorithm that can take in an input image, assign learnable weights and biases to various aspects/objects in the image, and be able to differentiate one from the other. A convolutional neural network is the same as neural networks but for images. So, we provide images as input to the CNN model (Brownlee,2020). It consists of an input layer, an output layer, and a number of hidden layers. Hidden layers include the Convolution layer, Pooling layer, Rectified Linear Units layer, Dropout layer, and Fully Connected layer.

3.1.1 Input Layer

The input layer contains the raw pixel values of the images. In this case, images are colored with a resolution of 640*480 pixels which are scaled down to 224*224 to reduce training time.

3.1.2 Conv Layer

The Conv layer contains a set of learnable filters of small dimensions. These filters are mapped over the entire region of an input image and at each location, a dot product is taken with the weights of the filter and a small region beneath the filter. For this project, if 32 filters of size 3×3 are used then the output dimension would be 224*224*32.

3.1.3 Pooling Layer
The Pooling layers reduced the 2D dimensions of input volume to prevent overfitting or avoid computation difficulties. This is done by applying a small filter of size 2×2 to input data on each depth slice. There are various types of pooling filters like max-pooling which select a max value under the filter, average pooling, which takes average values of the elements in a predefined sized image section., etc.

3.1.4 ReLu Function
It applied an activation function to each element to increase the non-linearity of the model. The max(0, x) is example of an activation function.

3.1.5 Dropout Layer
The Dropout layer is added to prevent the model from overfitting. It is a regularization method that drops a few neurons from the neural network during the training process resulting in a reduced size of the model. In our models, we have added a dropout layer with a value of 0.5.

3.1.6 FC Layer
The Fully Connected layer consists of the weights and biases along with the neurons. These layers are usually placed before the output layer and form the last few layers of a CNN Architecture. This layer gives the final prediction for each class. In this project there are 10 classes, so the FC layer contains 10 neurons.

4. Proposed Models
4.1 AlexNet
AlexNet is a convolutional neural network that was designed by Alex Krizhevsky (Krizhevsky et al., 2012). In ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2010, this network was trained to classify 1.2 million high-resolution images into 1000 different classes. It achieved top-1 and top-5 error rates of 37.5% and 17%, which outperforms the best methods at that time.

AlexNet consists of eight layers in which five convolutional layers, are followed by max-pooling layers, two fully connected hidden layers, and one fully connected output layer (Le, 2020). For our dataset, we changed the input shape to 224*224*3 with 96 filters. After each convolutional block, we used batch normalization to avoid the un-stability of gradients within the neural network. After compiling the model for 10 epochs, we got a training accuracy score of 92.59% and a validation accuracy score of 91.72% as shown in Figure 2. The training loss is 0.3570 and the validation loss is 0.2888 as shown in Figure 3.

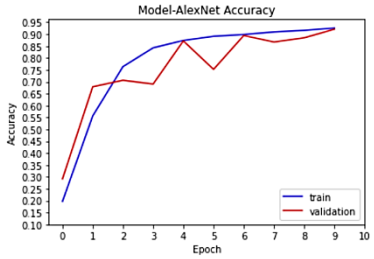

Figure 2. Training Accuracy and Validation Accuracy for AlexNet

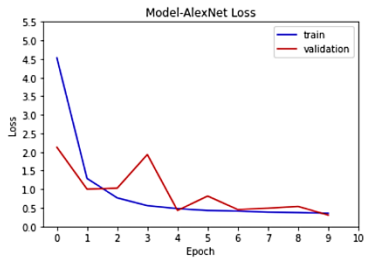

Figure 3. Training loss and Validation loss for AlexNet

4.2 VGG-16

VGG-16 is part of the VGG network architecture which was introduced in the paper Very Deep Convolutional Networks for Large Scale Image Recognition by Simonyan and Zisserman in 2014 (Simonyan and Zisserman, 2014). The number of weight layers in VGG-16 is 16. We made training off of the first 15 layers, did not include the top i.e., we removed the last softmax layer of VGG-16 and added a Fully Connected layer having 10 neurons to output the final predicted values. We trained the model for 10 epochs and got a training accuracy of 96.58% and validation accuracy of 94.12% as shown in Figure 4, and training loss of 0.1484 and validation loss of 0.2146 as shown in Figure 5.

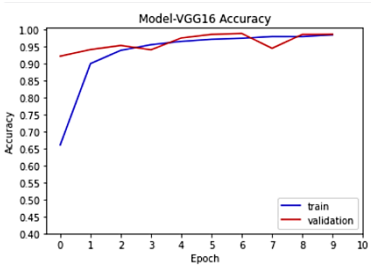

Figure 4. Training Accuracy and Validation Accuracy for VGG-16

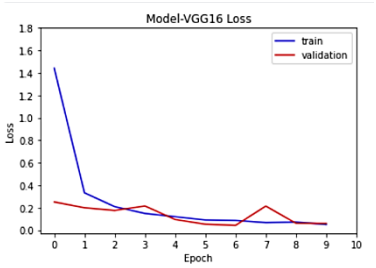

Figure 5. Training loss and Validation loss for VGG-16

4.3 ResNet-50

The ResNet model was invented by Microsoft researchers in 2016. The model has achieved a result of 96.4% in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) (Zhang et al., 2016). The network consists of 50 layers of deep neural network architecture. Also, the ResNet model introduced unique residual blocks in which the identity skip connections are used to address training a very deep architecture approach. Residual blocks copy and carry out the inputs of a specific layer to the next layer. The vanishing gradients issue was overcome by the identity skip connection step, which ensures that the next layer trains on something other than the input that the layer is familiar with.

In our project, we used pre-trained weights of ResNet-50 keeping training off for the first 49 layers. Instead of ResNet's last layer, we added a Fully Connected layer having 10 neurons to output the final predicted values. Also, used the Dropout layer with a value of 0.5. We trained the model for 10 epochs and got a training accuracy of 55.59% and validation accuracy of 75.27% as shown in Figure 6, and training loss of 2.3111 and validation loss of 0.8756 as shown in Figure 7.

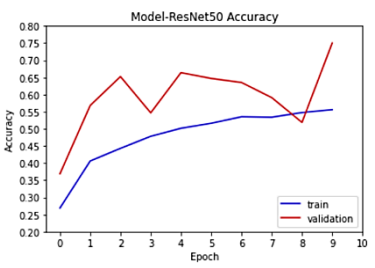

Figure 6. Training Accuracy and Validation Accuracy for ResNet-50

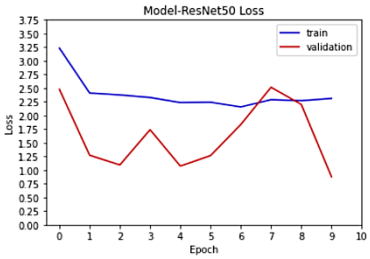

Figure 7. Training loss and Validation loss for VGG-16

4.4 MyModel

In this paper, we proposed a convolutional neural network architecture from scratch by adding layers to the sequential model. The initial input layer has a shape of 224×224×3, and the model consists of six CNN blocks with Conv2D and MaxPooling2D layers with a pool size of 2×2, each convolution layer consists of filters ranging from 32,64,128,256,512,1024, having a kernel size of 3×3 with the same padding. The activation function used was 'ReLu' with 'GlorotNormal' as a kernel initializer (Keras Documentation). After these convolution blocks, a dense layer with 500 neurons was added and followed by a dropout layer with a value of 0.5 to avoid overfitting. The last layer added was the dense layer having 10 neurons to output the final predicted values with Softmax as activation function (Jones, 2020).

The final model architecture has been shown in Figure 8. It describes the layer names with respective input and output shapes after performing the respective operations.

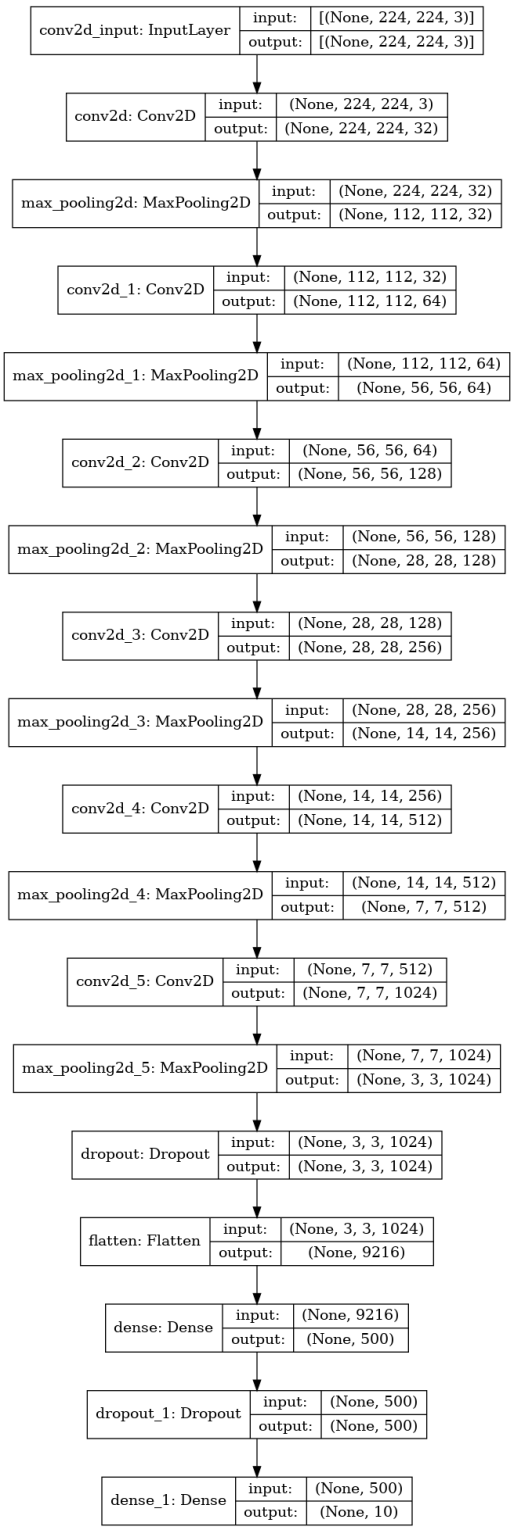

Figure 8. MyModel Architecture

For compiling our model, we used the loss function as categorical cross-entropy with the 'RMSprop' optimizer (Gandhi, 2018). We trained the model for 10 epochs and got a training accuracy of 93.26% and validation accuracy of 96.83% as shown in Figure 9, and training loss of 0.2538 and validation loss of 0.1295 as shown in Figure 10.

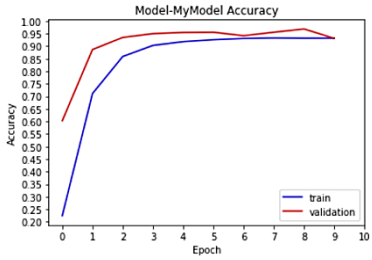

Figure 9. Training accuracy and Validation accuracy for MyModel

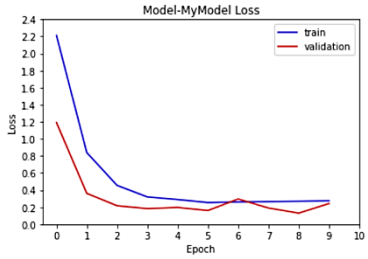

Figure 10. Training loss and Validation loss for MyModel

5. Results and Discussion

Classification Accuracy is what we usually mean, when we use the term accuracy. It is the ratio of number of correct predictions to the total number of input samples.

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions\ made}$$

It works well only if there are equal number of samples belonging to each class.

Logarithmic Loss or Log Loss, works by penalizing the false classifications. It works well for multi-class classification. When working with Log Loss, the classifier must assign probability to each class for all the samples (Bell,2020). Suppose, there are N samples belonging to M classes, then the Log Loss is calculated as below:

$$Log\ Loss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M=10} y_{ij}\log(p_{ij})$$

To evaluate the performance of models categorical cross-entropy or log loss was used as the metrics. Here N stands for a number of predictions and M is the number of classes i.e., 10 in our case. The value of y_{ij} is 1 if the image i belong to class j with the probability value of p_{ij}.
As per the objective we prepared models, based on the comparative study table 1. we compared results based on training and validation loss we found that,

1. AlexNet has more training loss than MyModel's training loss, thus MyModel performs better than AlexNet in training.
2. VGG-16 has low training loss than MyModel's training loss, hence VGG-16 performs better than MyModel in training.
3. ResNet-50 has more training loss than MyModel's training loss, hence MyModel is far better than ResNet-50 in training.
4. AlexNet has more validation loss than MyModel's validation loss, thus MyModel performs better than AlexNet in validation.
5. VGG-16 has more validation loss than MyModel's validation loss, hence MyModel performs better than VGG-16 in validation.
6. ResNet-50 has more validation loss than MyModel's validation loss, hence MyModel is far better than ResNet-50 in validation too.

Table 1. Comparative Study

Model Name	Loss		Accuracy	
	Training	Validation	Training	Validation
AlexNet	0.3570	0.2888	92.59%	91.72%
VGG-16	0.1484	0.2146	96.58%	94.12%
ResNet-50	2.3111	0.8756	55.59%	75.27%
MyModel	0.2538	0.1295	93.26%	96.83%

Likewise, from a comparative study in table 1. we compared results based on training and validation accuracy we found that,

1. AlexNet has less training accuracy than MyModel's training accuracy, thus MyModel performs better than AlexNet in training.
2. VGG-16 has more training accuracy than MyModel's training accuracy, hence VGG-16 performs better than MyModel in training.
3. ResNet-50 has less training accuracy than MyModel's training accuracy, hence MyModel is far better than ResNet-50 in training.
4. AlexNet has less validation accuracy than MyModel's validation accuracy, thus MyModel performs better than AlexNet in validation.
5. VGG-16 has less validation accuracy than MyModel's validation accuracy, hence MyModel performs better than VGG-16 in validation.
6. ResNet-50 has less validation accuracy than MyModel's validation accuracy, hence MyModel is far better than ResNet-50 in validation too.

6. Conclusion

We compared three well-known CNN models with our model for the distracted driver detection and classification problem. After trying out several CNN models, MyModel's Validation loss was 0.1295. We can conclude that MyModel performs far better in validation and training too. If we provide any random image from particular 10 classes, it will give better prediction results. Furthermore, current deep CNN models can be used to automatically detect distracted drivers and can be integrated with different hardware systems to warn the driver. However, the main constraint for distracted driver detection using a dashboard camera from the driver's viewpoint is the driver's privacy. In the future, if CNN models can recognize distracted behaviors using images from a dashboard camera that captures the drivers from different angles in real-time, and combining distracted driver detection with drowsy driver detection using one classification model.

References

Abouelnaga, Y., Eraqi, H., and Moustafa, M. *Real-time Distracted Driver Posture Classification*,2018.

Alotaibi, M., and Alotaibi, B. Distracted driver classification using deep learning. *Signal, Image and Video Processing*, *14*(3), 617–624. https://doi.org/10.1007/s11760-019-01589-z, 2020

Behera, A., Keidel, A., and Debnath, B. *Context-driven Multi-stream LSTM (M-LSTM) for Recognizing Fine-Grained Activity of Drivers: 40th German Conference, GCPR 2018, Stuttgart, Germany, October 9-12,*

2018, Proceedings (pp. 298–314). https://doi.org/10.1007/978-3-030-12939-2_21, 2019

Dunn, N. J., Dingus, T. A., Soccolich, S., and Horrey, W. J., Investigating the impact of driving automation systems on distracted driving behaviors. *Accident; Analysis and Prevention*, *156*, 106152., https://doi.org/10.1016/j.aap.2021.106152, 2020.

Fernández, A., Usamentiaga, R., Carús, J. L., and Casado, R. Driver distraction using visual-based sensors and algorithms. *Sensors (Switzerland)*, *16*(11), 1–44, 2016.

Hssayeni, M. D., Saxena, S., Ptucha, R., and Savakis, A. Distracted driver detection: Deep learning vs handcrafted features. *IS and T International Symposium on Electronic Imaging Science and Technology*, 20–26, 2017.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks. In F. Pereira, C. J. Burges, L. Bottou, and K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems* (Vol. 25). Curran Associates, Inc., 2012.

Mercanoglu, O., Gencoglu, S., Bacak, M., and Keles, H. *Hand and Face Segmentation with Deep Convolutional Networks using Limited Labelled Data*, 2019.

Omerustaoglu, F., Sakar, C. O., and Kar, G. Distracted driver detection by combining in-vehicle and image data using deep learning. *Applied Soft Computing Journal*, *96*, 106657, 2020.

Satardekar, S. *Distracted Driver Detection and Classification Distracted Driver Detection and Classification July*, 2–7, 2018.

Simonyan, K., and Zisserman, A.. Very Deep Convolutional Networks for Large-Scale Image Recognition. *ArXiv 1409.1556*, 2018.

Transport Research Wing, G. of I. Road Accidents in India, *Ministry of Road Transport and Highways, Transport Research Wing*. https://morth.nic.in/sites/default/files/RA_Uploading.pdf, 2020.

Xing, Y., Lv, C., Wang, H., Cao, D., Velenis, E., and Wang, F. Y. Driver activity recognition for intelligent vehicles: A deep learning approach. *IEEE Transactions on Vehicular Technology*, *68*(6), 5379–5390, 2019.

Zhang, X., Ren, S., and Sun, J. *Deep Residual Learning for Image Recognition*., 2016.

Biographies

Sujay H. Jagadale is pursuing a Master of Technology in Production Engineering from Veermata Jijabai Technological Institute (VJTI) Mumbai. He received Data Scientist and Machine Learning Engineer certification from Codespyder Technologies Pvt. Ltd. Pune.

Dr. P.R. Attar is currently working as Assistant Professor, Production Engineering Department, Veermata Jijabai Technological Institute (VJTI), Mumbai.