Decision Making for Driver Assistant Systems using Computer Vision

Maddipati Rakesh Kumar[1[0000-0001-8466-8013]], Rashmi Agarwal[3[0000-0003-1778-7519]]

[1,2] RACE, REVA University, Bangalore, 560064, India
maddipatirakeshk.ai01@race.reva.edu.in[1]
rashmi.agarwal@reva.edu.in[2]

Abstract Advance Driver Assistant Systems (ADAS) are a group of electronic technologies that assist drivers in driving and parking systems, bringing safer, easier transportation. ADAS systems use advanced technologies to assist the driver by using a combination of sensor technologies, Artificial Intelligence (AI), radars, and cameras to perceive the world around the vehicle. Computer vision is the core of AI to decode and understand the visual data acquired from cameras. The decoded data from the camera helps in activating driver assistant systems such as Adaptive Cruise Control (ACC), Collision Warning Systems (CWS), Intelligent Speed Adaption (ISA), and automatic windshield wiper activation. These driver assistant systems are activated based on the weather condition predicted by computer vision models. In adverse weather conditions, the camera output becomes noisy and becomes challenging for computer vision algorithms to decode the visual data.

Multiple computer vision algorithms are proposed by many research institutes and leading automakers for a faster and more accurate classification of images. Decision-making for activating the driver assistant systems using various image classification algorithms are dependent mainly on Frames processed Per Second (FPS) and the accuracy of image classification models on streaming data. A comparative study of these algorithms on the benchmark DAWN dataset was conducted in this research.

Keywords: Autonomous cars, Safety systems, ADAS, Image classification, DAWN dataset

1 Introduction

ADAS is a group of electronic technologies that assist drivers during driving and parking functions. ADAS are developed to automate, adapt and enhance vehicle technology for safer and better driving comfort. ADAS functions can be taken as two main groups mainly, custom functions and safety functions. Custom functions mainly include giving a warning trigger to the driver such as a flashing light, or vibration alert. Safety functions take care of the vehicle control in case the driver is not responding to the potential warning alerts. Some of the custom and safety functions

are Forward Collision Warning (FCW), Lane Departure Warning (LDW), Automatic high beam assist, pedestrian detection, Automatic Emergency Braking (AEB).

Computer vision algorithms play a crucial role in the development of ADAS. These algorithms are responsible to deliver accurate and clear image data by processing the raw data from edge devices so that ADAS systems can better understand the environment around the car. The raw data from the camera may not be clear always due to blur, dust on the camera, and also due to harsh weather conditions. Some of the applications of computer vision in ADAS are:

I. Emergency brake systems
II. Automatic windshield wiper activation
III. Pedestrian detection
IV. Rear camera parking assistance

2 Literature Review

Computer vision-based driver assistant systems are blooming with a wide range of application and being a promising technology for solving real-world problems such as driver monitoring [1,2], speed adaptation [3,4], parking [5,6], lane departure warning [7], intelligent headlamp control [8]. For driver monitoring, the AdaBoost algorithm is used to extract the potential features from a large set of available features in [1]. For detection and recognition of road signs, and alerting the driver an integrated approach is designed using neural networks in [2]. For 360-degree parking assistant systems, a unique brightness balance algorithm is proposed in [6] that can adjust the brightness between input data streaming from 4 fisheye cameras around the vehicle. Various approaches for utilizing visual sensors in ADAS are discussed in [7,27-30] explaining the overall architecture for computer vision applications.

These assistant systems need to operate on any occurring environmental conditions such as heavy rain, bad lighting conditions, tunnels, highway, intercity, etc. Some of the studies are conducted for analyzing the effect of adverse weather conditions on primary sensors such as LiDAR, cameras, and radar by using simulation. In various studies, it is demonstrated that in adverse weather conditions the ADAS sensors fail regardless of vehicle speed [9]. In some of these studies, it is proved that LiDAR visibility and intensity is severely affected by bad weather condition [13- 15]. For example, in Lane Detection Warning System (LDWS) image detection, it is analyzed that environmental factors affect the performance of sensors [10,12] and it is proved there is a need for complex sensors and regional factors considered for better performance. A systematic view of faults and corresponding measures to avoid them is discussed in detail in [16]. Faults related to adverse weather conditions are discussed majorly in this research and these faults are highlighted as the primary issues in extracting the features. In addition to this, in some of the studies, it is proved rain is the main source of radar signal abrasion [16]. Radars are also affected by snow or mist in the form of abrasion and backscatter [17].

In this aspect, there is a need to build an accurate and faster image classification technique for weather detection. FPS of 4.4 is achieved by ResNet15 on a normal CPU setup in [18]. On the weather data considered in this study, the number of layers is reduced from 50 to 15 for ResNet architecture. This resulted in faster FPS without reducing the model accuracy. The present study is focused on applying computer vision algorithms for weather detection and comparing the speed and accuracy of the results. VGG16 [20], ResNet50 [21], EfficientNet [22], and InceptionNet [23] architectures are proposed by various researchers for image classification problems.

3 Proposed methodology

As an alternate approach to the conventional way of using sensing technologies such as LiDAR and radar, a new workflow is designed by using cameras for some of the decision-making requirements in ADAS. This can benefit in cost, time, and computational power. With advancements in machine learning and deep learning techniques, this study helps in understanding the applicability of various image classification algorithms. This helps in predicting the weather for each frame in the streaming video data on a real-time basis.

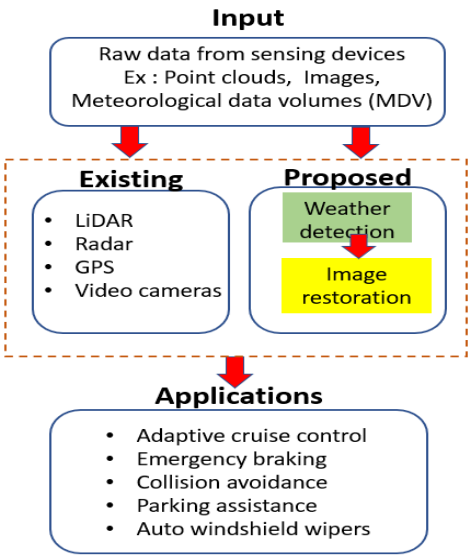

Fig. 1. Traditional approach and proposed methodology

The present study is limited to some of the decision-making requirements such as emergency braking, traffic sign recognition, and obstruction detection. The proposed methodology as shown in Fig. 1 is a novel approach for weather detection to most of the decision-making requirements, where the prediction result can be helpful in applying the appropriate denoising algorithms on the input data and also helps in faster decision-making for autonomous driving conditions.

The new approach can be stated as a pre-processing step before applying any denoising algorithm. This approach of using vision cameras is planned to eliminate high-cost sensing technologies in some of the applications where performance degrades in adverse weather conditions. This approach is also targeted for faster decision-making skills with minimal time for weather detection.

In the proposed methodology, computer vision models are trained on the benchmark dataset DAWN for detecting four types of weather conditions where the vision becomes noisy. The benchmark dataset is used in training various image classification algorithms such as VGG16, ResNet, InceptionNet, and EfficientNet. The trained models are fine-tuned for better accuracy and FPS using various hyperparameters such as learning rate, number of epochs, and dropout ratio.

4 Data Generation and Understanding

For deep learning applications to real-time in-stream video data, a higher system configuration is preferred for faster execution. In this study, Google Colaboratory (GC) is chosen for running different image classification models. GPU runtime with specifications mentioned in the table is used for faster runtime execution and FPS comparison. The runtime configurations used in this study are shown in Table 2.

Table 2. Runtime configuration

Runtime source	Specification
GPU	Up to Tesla K80 with 12 GB of GDDR5 VRAM, Intel(R) Xeon(R) CPU with 2 cores @ 2.20 GHz, and 13 GB RAM

Table 3. Classes in DAWN dataset

Class	Rain	Fog	Snow	Sandstorm
Samples	250	250	250	250

For image classification algorithm training DAWN dataset is used. This contains 4 different classes of weather such as rain, snow, fog, and sandstorm with 250 images for each class as shown in Table 3. This dataset also contains images captured during dawn, day, dusk, and night. Images in each class of this dataset are re-sized to 224* 224 pixels before passing to the image classification models. The DAWN dataset chosen for this study covers all the lighting conditions and weather conditions on a high level from the dashboard camera [19]. The dataset also contains object bounding boxes for autonomous driving and video surveillance scenarios.

5 Implementation

Tensorflow framework and Keras library are used in the present study. As discussed in the earlier section, original dataset contains 4 folders containing images captured in different weather conditions and bounding boxes for object detection. In the present study, bounding boxes are ignored as they are not relevant for the image classification tasks. Implementation steps for this study are shown in Fig. 2.

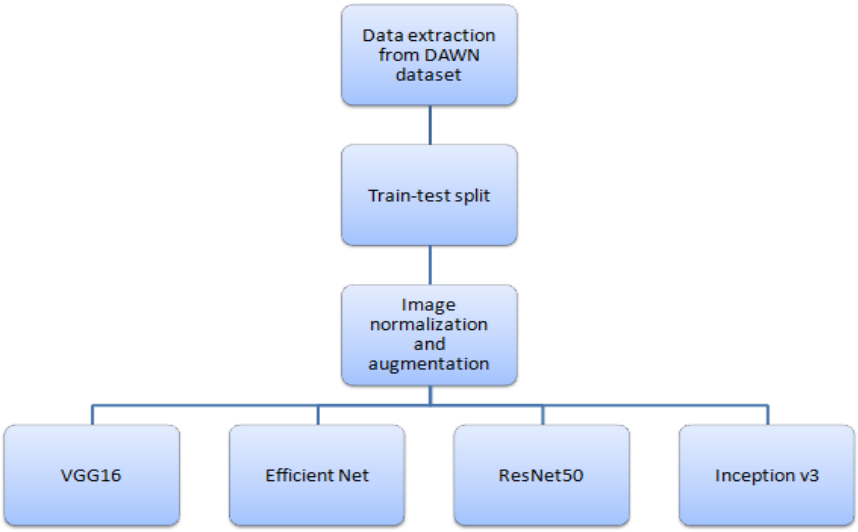

Fig. 2. Implementation flowchart

5.1 VGG16

This algorithm is one of the popular algorithms for object detection and image classification tasks. Transfer learning technique makes this algorithm adopted in an easy and faster way for computer vision tasks. This architecture contains 21 layers including pooling. The number 16 in the algorithm name represents the number of layers with weights. The architecture consists of 13 convolution layers, 5 pooling layers, and 3 dense layers at the end. This algorithm takes input tensor size as 224*224 pixel with RGB data. The base network is trained on the 'ImageNet' dataset containing more than 14 million high-quality images with 1000 target classes. Using the transfer learning techniques, the weights for the hidden layers are adopted making the learning process faster and simpler. For the last dense layer, the target classes are set to 4 based on the DAWN dataset.

5.2 Inception-v3

Inception-v3 is a CNN architecture for image classification and it is a superior version of Inception-v1. Compared to the VGG-16, inception architectures like GoogleNet, and InceptionNet are computationally efficient both in terms of

economical cost and the number of parameters generated. The major changes done to Inception-v3 compared to predecessors are the utility of auxiliary classifiers, factorization into smaller convolutions, spatial factorization into asymmetric convolutions, and efficient grid size reduction. The architecture contains a total of 42 layers with 24 M parameters. The default input image size is 299*299, and that can be changed to 224*224 for the present study. The ImageNet pre-trained weights are used for the transfer learning technique.

5.3 ResNet50

ResNet50 is convolution neural network architecture with 50 layers and it is a variant of residual networks. In previous CNN models, deep networks are adopted for increasing the accuracy scores. This approach leads to training huge neural networks and vanishing gradients. In ResNet architecture, skip-connections are used to solve the vanishing gradient problem. This network can be easily scalable for a different number of layers such as ResNet18, and ResNet34. ResNet101, ResNet110, ResNet154. The network has an input image size of 224*224 pixels as shown in Fig. 3.

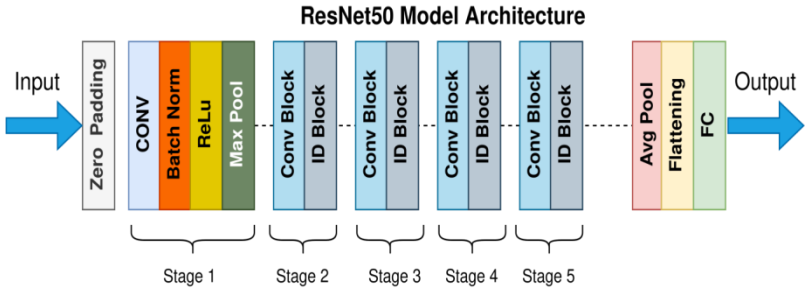

Fig. 3. ResNet50 Architecture

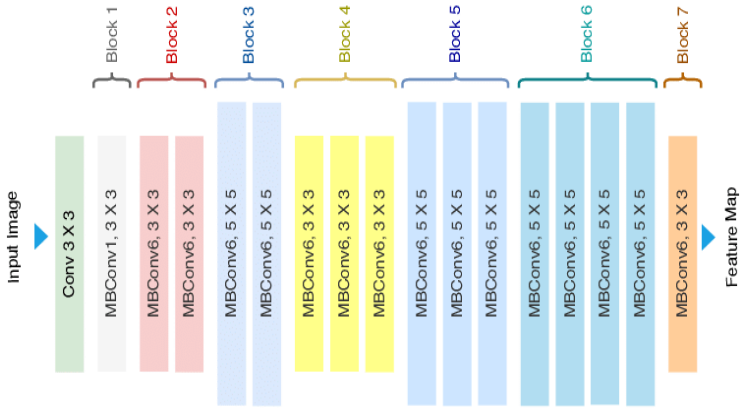

Fig. 4. EfficientNet Architecture

5.4 EfficientNet

Efficientnet is a CNN architecture and scaling method that uniformly scales all the dimensions of width, depth, and resolution using a compound coefficient. Before EfficientNet architecture is proposed, the most common way of improving the convolutional networks is scaling in one of the 3 dimensions i.e., the depth of the network, the width of the network, and the resolution of the image. But in EfficientNet, compound scaling is performed. This compound scaling takes care of balancing all the 3 dimensions of the network. Different scaling architectures are shown in Fig. 4. There are several architectures ranging from EfficientNet-B0 to EfficientNet-B7.

Table 3. Number of parameters for each architecture

Architecture	VGG16	ResNet50	InceptionV3	EfficientNet
Parameters (Millions)	14.7	23.5	21.8	64

The number of parameters increases with each variant of the architecture. The base architecture EfficientNet-B0 contains 5.3 million parameters and EfficientNet-B7 contains 66M parameters. EfficientNet-B7 achieved state-of-the-art accuracy scores compared to other CNNs on ImageNet data as shown in fig. Compared to other CNN models, EfficientNet-B7 is lightweight and the accuracy score is high. In the present study, EfficientNet-B7 is used. Total number of parameters in each architecture is shown in the Table 3.

6 Testing and Validation

In this study for comparing the different architectures, “accuracy score” and FPS are considered performance metrics for evaluation. From the original data, 20% of the data is used for model testing. Test data is randomly selected from the original data ensuring the proper class distribution. Test data is augmented using the “ImageDataGenerator” utility from the Keras library. FPS is calculated using 10 unseen images. These unseen images are resized to 224*224 pixel size, preprocessed similar to the training dataset, and then fed to the trained CNN architecture. FPS for a single image is calculated by taking the average time taken for 10 unseen images.

In the **first iteration** (i1), a dropout ratio of 0.2 and a learning rate of 0.00005 with 20 epochs are chosen as parameters. The accuracy scores and FPS values are shown in Table 4. In the **second iteration** (i2), a dropout ratio of 0.2 and a learning rate of 0.0001 with 20 epochs are chosen as parameters. For this iteration, the accuracy scores and FPS are shown in Table 4.

Table 4. Accuracy and FPS for Iteration 1 and 2

Architecture	Train (%)		Test (%)		Time (in sec for 10 Frames)		FPS (f/s)	
	i1	i2	i1	i2	i1	i2	i1	i2
VGG16	76	70	73	74	614	608	16	16
ResNet50	87	85	79	75	883	623	11	16
Inception	83	83	80	77	681	661	14	15
EfficientNetB7	86	88	74	66	1000	1004	10	9

Table 5. Accuracy and FPS for Iteration 3 and 4

Architecture	Train (%)		Test (%)		Time (in sec for 10 Frames)		FPS (f/s)	
	i3	i4	i3	i4	i3	i4	i3	i4
VGG16	74	68	75	70	70	618	16	16
ResNet50	86	85	78	78	78	655	15	15
Inception	81	82	78	78	78	691	14	14
EfficientNetB7	86	86	37	64	64	1009	9	9

In the **third iteration** (i3), a dropout ratio of 0.1 and a learning rate of 0.00005 with 20 epochs are chosen as parameters. The accuracy scores shown in Table 5.
In the **fourth iteration** (i4), a dropout ratio of 0.25 and a learning rate of 0.00005 with 20 epochs are chosen as parameters. The accuracy scores are shown in Table 5.

7 Analysis and Results

Various experiments were conducted by changing the hyper parameters such as learning rate, dropout ratio, and the number of epochs. Results from these experiments are noted and compared for all 4 architectures chosen.

In first iteration, VGG16 network proved to give higher FPS due to less number of parameters, but the accuracy scores are less. Based on accuracy scores and FPS, Inception v3 architecture is giving better results in comparison to other architectures. With EfficientNet architecture, accuracy can be tuned and overfitting can be avoided, but the FPS value is lower than the industrial standards.

In second iteration, VGG16, ResNet50, and Inception v3 proved to deliver higher FPS scores. In terms of model accuracy, model overfitting and underfitting Inception v3 proved to deliver the best scores. EfficientNet model delivered a very low FPS score and the model is overfitting for the DAWN dataset. Model training and testing accuracy score comparison is shown in Fig. 5 and Fig. 6. FPS comparison for all 4 iterations is shown in Fig. 7.

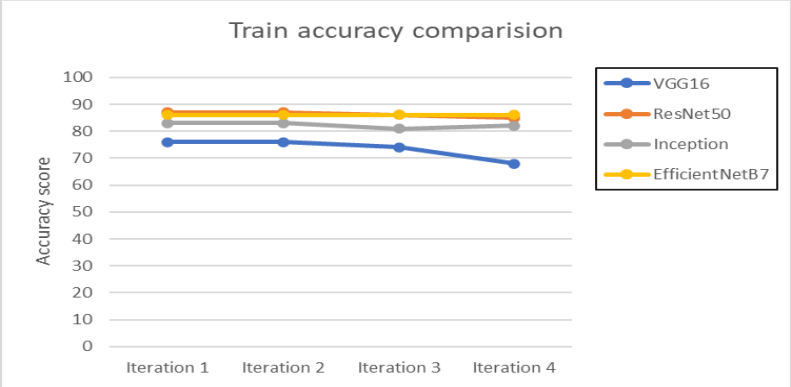

Fig. 5. Training set accuracy scores comparison

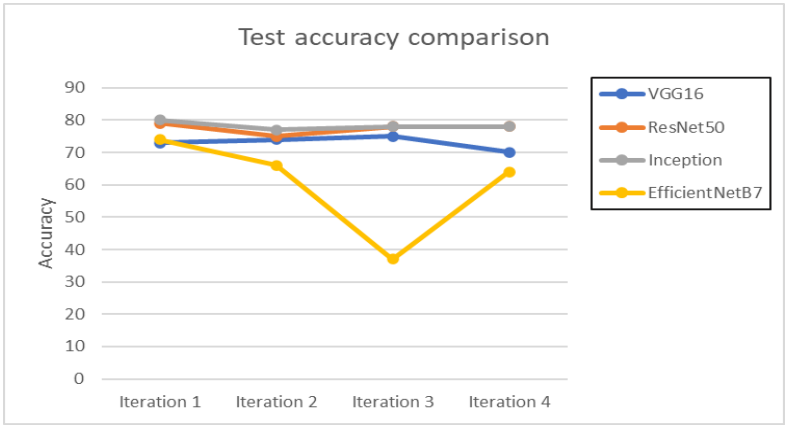

Fig. 6. Test set accuracy scores comparison

In third iteration, VGG16, ResNet50, and Inception v3 proved to deliver higher and similar FPS scores. In terms of model accuracy, model overfitting and underfitting Inception v3 are proved to deliver the best scores.

In fourth iteration, similar to the previous iterations, Inception v3 architecture delivered higher FPS and accuracy scores.

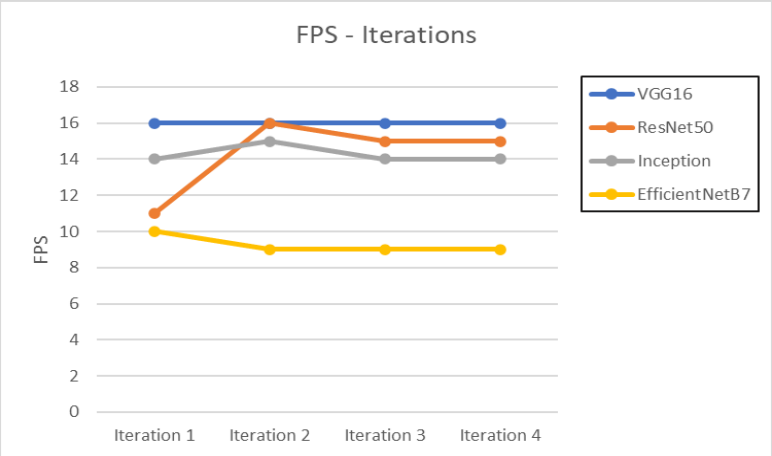

Fig. 7. FPS comparison

8 Conclusion and Future scope

The above-discussed results are compared in terms of accuracy scores and also FPS values. In all 4 architectures, EfficientNet B7 with 64 million parameters is overfitting and delivered lower FPS for the DAWN dataset. VGG16 and ResNet50 architectures delivered higher FPS, but the model accuracy is low. Inception v3 architecture with 21.8 million parameters is proved to deliver higher accuracy scores and FPS. With the different experiments in this study, it can be concluded that inception v3 architecture with transfer learning technique is the best fit for deep learning architecture in comparison to VGG16, ResNet50, and EfficientNet B7.

The study can be extended by developing image restoration architectures under adverse weather conditions. As the DAWN dataset contains only 250 images per class, a new dataset with more images per class taken from the car dashboard cameras will help to achieve higher accuracy scores. Also, deep learning architectures in this study are limited to 4, and this study can be taken forward with more lightweight architectures for real-time edge devices.

References

1. Viola, P., Jones, M.J. Robust Real-Time Face Detection. International Journal of Computer Vision 57, 137–154 (2004). https://doi.org/10.1023/B:VISI.0000013087.49260.fb

2. E. Murphy-Chutorian and M. M. Trivedi, "Head Pose Estimation in Computer Vision: A Survey," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 31, no. 4, pp. 607-626, April 2009, doi: 10.1109/TPAMI.2008.106.

3. M. L. Eichner and T. P. Breckon, "Integrated speed limit detection and recognition from real-time video," 2008 IEEE Intelligent Vehicles Symposium, 2008, pp. 626-631, doi: 10.1109/IVS.2008.4621285.

4. Kheyrollahi, A., Breckon, T.P. Automatic real-time road marking recognition using a feature driven approach. Machine Vision and Applications 23, 123–133 (2012). https://doi.org/10.1007/s00138-010-0289-5

5. Yu, M. and Ma, G., "360° Surround View System with Parking Guidance," SAE Int. J. Commer. Veh. 7(1):19-24, 2014, https://doi.org/10.4271/2014-01-0157.

6. C. Wang, H. Zhang, M. Yang, X. Wang, L. Ye, and C. Guo. Automatic parking based on a bird's eye view vision system. Advances in Mechanical Engineering, Article ID 847406, 13 pages, 2014.

7. R. Preiß, C. Gruner, T. Schilling, and H. Winter. Mobile vision - Developing and testing of visual sensors for driver assistance systems. In Advanced Microsystems for Automotive Applications (J. Valldorf and W. Gessner, editors), VDI-Buch, pp. 95–107, 2004.

8. J. C. Rubio, J. Serrat, A. M. Lopez, and D. Ponsa. Multiple target tracking for intelligent headlights control. IEEE Trans. Intelligent Transportation Systems, 13:594–609, 2012.

9. Roh, Chang-Gyun, Jisoo Kim, and I-JeongIm. 2020. "Analysis of Impact of Rain Conditions on ADAS" Sensors 20, no. 23: 6720. https://doi.org/10.3390/s20236720
10. Hadi M., Sinha P., Easterling J.R., Iv J.E. Effect of Environmental Conditions on Performance of Image Recognition-Based Lane Departure Warning System. Transp. Res. Rec. J. Transp. Res. Board.

11. Jeong K.M., Song B.C. Fog Detection and Fog Synthesis for Effective Quantitative Evaluation of Fog–detection-and-removal Algorithms. IEIE Trans. Smart Process. Comput. 2018;7:350–360. doi: 10.5573/IEIESPC.2018.7.5.350.

12. Tumas P., Nowosielski A., Serackis A. Pedestrian Detection in Severe Weather Conditions. IEEE Access. 2020;8:62775–62784.

13. Hasirlioglu S., Riener A. Introduction to rain and fog attenuation on automotive surround sensors; Proceedings of the 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC); Yokohama, Japan. 16–19 October

14. Hadj-Bachir M., De Souza P. LIDAR Sensor Simulation in Adverse Weather Condition for Driving Assistance Development. [(accessed on 22 September 2020)];2019 – 01998668.

15. G. P. Kulemin, "Influence of propagation effects on millimeter wave radar operation", Proc. SPIE Conf. Radar Sensor Technol., vol. 3704, pp. 170-178, Apr. 1999.

16. R. Lhermitte, "Attenuation and scattering of millimeter wavelength radiation by clouds and precipitation", J. Atmos. Oceanic Technol., vol. 7, no. 3, pp. 464-479, 1990.

17. The Impact of Adverse Weather Conditions on Autonomous Vehicles: How Rain, Snow, Fog, and Hail Affect the Performance of a Self-Driving Car

18. Jingming Xia, Dawei Xuan, Ling Tan, Luping Xing, "ResNet15: Weather Recognition on Traffic Road with Deep Convolutional Neural Network", Advances in Meteorology, vol. 2020, Article ID 6972826, 11 pages, 2020. https://doi.org/10.1155/2020/6972826

19. KENK, Mourad (2020), “DAWN”, Mendeley Data, V3, doi: 10.17632/766ygrbt8y.3

20. Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings. International Conference on Learning Representations, ICLR.

21. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Vol. 2016-December, pp. 770–778). IEEE Computer Society. https://doi.org/10.1109/CVPR.2016.90

22. Tan, M., & Le, Q. V. (2019). EfficientNet: Rethinking model scaling for convolutional neural networks. In 36th International Conference on Machine Learning, ICML 2019 (Vol. 2019-June, pp. 10691–10700). International Machine Learning Society (IMLS).

23. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., … Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Vol. 07-12-June-2015, pp. 1–9). IEEE Computer Society. https://doi.org/10.1109/CVPR.2015.7298594

24. https://www.synopsys.com/content/dam/synopsys/designware-ip/diagrams/adas-fig1.jpg.imgw.850.x.jpg, Retrieved August 15, 2022

25. https://upload.wikimedia.org/wikipedia/commons/9/98/ResNet50.png, Retrieved August 15, 2022,

26. https://www.researchgate.net/profile/Tashin-Ahmed/publication/344410350/figure/fig4/AS:1022373302128641@1620764198841/Architecture-of-EfficientNet-B0-with-MBConv-as-Basic-building-blocks.png, Retrieved August 15, 2022

27. Janai, J., Güney, F., Behl, A., & Geiger, A. (2017). Computer Vision for Autonomous Vehicles: Problems, Datasets and State of the Art. ArXiv. https://doi.org/10.48550/arXiv.1704.05519

28. B. Kanchana, R. Peiris, D. Perera, D. Jayasinghe and D. Kasthurirathna, "Computer Vision for Autonomous Driving," 2021 3rd International Conference on Advancements in Computing (ICAC), Colombo, Sri Lanka, 2021, pp. 175-180, doi: 10.1109/ICAC54203.2021.9671099.

29. Agarwal, Nakul & Chiang, Cheng-Wei & Sharma, Abhishek. (2019). A Study on Computer Vision Techniques for Self-driving Cars. 10.1007/978-981-13-3648-5_76.

30. Lubna, N. Mufti, and S. A. A. Shah, “Automatic Number Plate Recognition:A Detailed Survey of Relevant Algorithms,” Sensors, vol. 21, no. 9, p. 3028, Apr. 2021, doi: 10.3390/s21093028.