International Research Journal on Advanced Engineering and Management
https://goldncloudpublications.com
https://doi.org/10.47392/IRJAEM.2025.041

e ISSN: 2584-2854
Volume: 03
Issue:02 February 2025
Page No: 256-264

Driver Drowsiness Detection Using Deep Neural Networks

Charanya J[1], Kavitha Vasu[2], Shahidkhan S[3], NaveenKumar K[4], Naveen Ram C E[5], Vasikaran S[6]
[1]Assistant Professor, Artificial Intelligence, Kongu Engineering College, Erode, Tamil Nadu, India.
[2]Associative Professor, Computer Science, Velalar College of Engineering, Erode, Tamil Nadu, India.
[3,4,5,6]UG - Artificial Intelligence and Data Science, Kongu Engineering College, Erode, Tamil Nadu, India.
Email ID: *charanya.ece@kongu.ac.in[1], kavithamebit@gmail.com[2], shahidkhans.22aid@kongu.edu[3], naveenkumark.22aid@kongu.edu[4], naveenramce.22aid@kongu.edu[5], vasikarans.22aid@kongu.edu[6]*

Abstract
Driver drowsiness significantly impacts road safety, leading to numerous accidents. We developed a Driver Drowsiness Detection system using deep learning for binary classification, utilizing CNN, VGG16, GoogleNet, AlexNet, MobileNet_v2, and ResNet101 architectures. Our models were trained on an annotated dataset of driver images and evaluated on metrics like accuracy and F1-score. Results show that while deeper networks offer high accuracy, lightweight models like MobileNet_v2 provide a good balance of performance and computational efficiency. This work demonstrates the potential of these models for real-time drowsiness detection in advanced driver- assistance systems (ADAS) to enhance road safety.

Keywords: *Driver Drowsiness Detection, Deep Learning, Binary Classification, Road Safety, ADAS.*

1. Introduction
Driver drowsiness is a major contributor to road accidents, leading to severe injuries, fatalities, and significant economic losses globally. The ability to detect and respond to driver fatigue in real-time can drastically reduce the risk of accidents, making roads safer for everyone [1]. Traditional methods of detecting driver drowsiness, such as self-reporting and manual observation, are not only impractical for continuous monitoring but also often inaccurate due to human error and subjective judgment.With the advent of deep learning and advancements in computer vision, automated detection systems have become feasible. These systems can process visual data in real-time, identifying subtle signs of Drowsiness Detection system using deep learning for binary classification, identifying drivers as either "drowsy" or "alert". Several state-of-the-art was in convolutional neural network (CNN) architectures are employed in this study to evaluate their effectiveness in detecting driver drowsiness.The dataset used for training and evaluating these models comprises annotated images of drivers exhibiting various states of alertness. This dataset was meticulously preprocessed and augmented to ensure robustness and improve generalization. Key performance metrics such as accuracy, precision, recall, and F1-score were used to evaluate and compare the models [2]. The results of this study provide valuable insights into the strengths and weaknesses of different CNN architectures for the task of driver drowsiness detection. While deeper networks like ResNet101 and VGG16 typically offer higher accuracy, they come with increased computational demands. Conversely, lightweight models such as MobileNet_v2 provide a favorable trade-off between performance and efficiency, making them suitable for real-time applications. In conclusion, this research demonstrates the potential of deep learning models in enhancing driver safety through reliable drowsiness detection. The findings can inform the development of advanced driver-assistance systems (ADAS) that incorporate real-time drowsiness detection, ultimately contributing to safer road environments.

2. Literature Survey
Driver drowsiness detection has emerged as a critical area of research within the realm of road safety, owing to its pivotal role in averting accidents caused by driver fatigue. Traditional methodologies primarily relied on the observation of behavioral and physiological indicators, such as monitoring eye closure rates, head position changes, and

physiological signals like EEG patterns. While effective to some extent, these approaches were often limited by their intrusiveness and incapability to operate seamlessly in real-time scenarios The advent of machine learning, particularly deep learning, has revolutionized the landscape of driver drowsiness detection. Leveraging advanced computer vision techniques and sophisticated algorithms, deep learning models offer the potential for automatic feature extraction and high-accuracy analysis of visual data. Convolutional Neural Networks (CNNs), in particular, have garnered significant attention for their prowess in discerning intricate patterns and features from raw image data.Various CNN architectures have been explored for driver drowsiness detection, each with its unique characteristics and advantages. Models such as VGG16, GoogleNet, AlexNet, MobileNet_v2, and ResNet101 have been scrutinized for their effectiveness in accurately identifying signs of drowsiness in drivers. These architectures undergo extensive training on annotated datasets of driver images, which capture diverse states of alertness and fatigue. Through this training process, the models learn to recognize subtle cues and patterns indicative of drowsiness. Comparative studies have been conducted to evaluate the performance of these CNN architectures in detecting driver drowsiness. These studies often assess metrics such as classification accuracy, computational efficiency, and real-time capabilities. While deeper networks like ResNet101 and VGG16 typically exhibit higher accuracy rates, they also demand greater computational resources, rendering them less suitable for real-time deployment in resource-constrained environments. Conversely, lightweight models like MobileNet_v2 offer a compelling compromise between accuracy and computational efficiency, making them ideal candidates for real-time applications. The findings from these studies contribute to the ongoing development of robust and efficient drowsiness detection systems aimed at enhancing road safety. By leveraging the capabilities of deep learning, researchers endeavor to create solutions that can reliably identify and mitigate driver fatigue, thereby reducing the risk of accidents and preserving human lives on the road [3-7].

3. Materials

The dataset utilized for the driver drowsiness detection project, obtained from Kaggle, is structured to facilitate the development of machine learning models for identifying drowsiness in drivers. The dataset is partitioned into two primary classes, Table 1.

Table 1 File Segmentation

Total Files	41793
Number of Drowsy	22348
Number of Non-Drowsy	19445

Drowsy Class: This class comprises images capturing instances where drivers display signs of drowsiness. These signs may include closed eyes, drooping eyelids, or other facial expressions indicative of fatigue or reduced alertness [8-12].

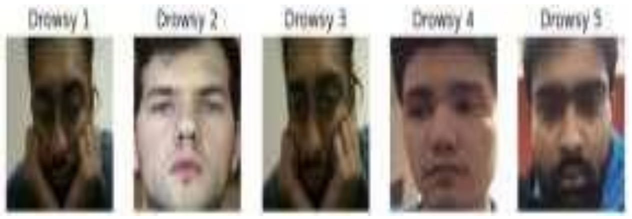


Figure 1 Drowsy Images

Non-Drowsy Class: In contrast, this class consists of images depicting drivers in an alert and attentive state. These images typically show open eyes, an a upright posture, and other facial features associated with wakefulness, Figure 1 [13, 14].

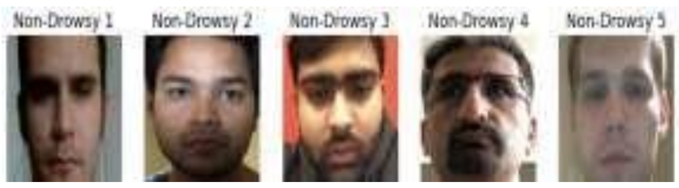


Figure 2 Non-Drowsy Images

Each class is segregated into separate folders within the dataset directory. By organizing the data in this manner, researchers and developers can effectively train and evaluate machine learning algorithms to recognize subtle visual cues indicative drowsiness, Figure 2.

4. Methodology

This section lacks a clear definition of the overall research methodology.

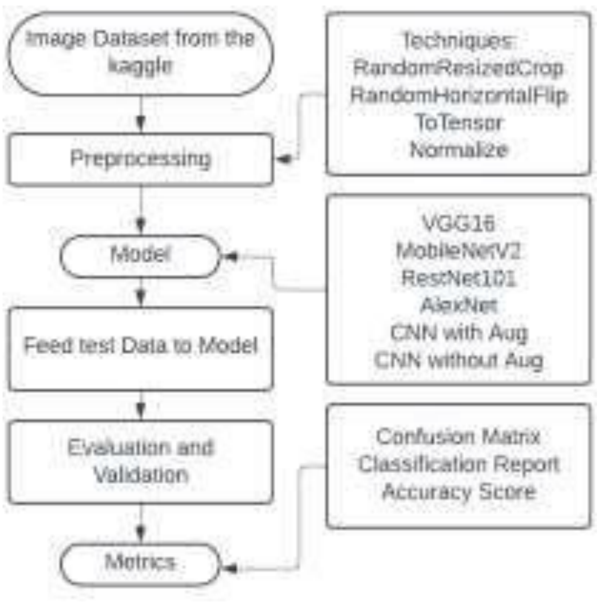


Figure 3 Workflow Chart

4.1. Algorithms

we developed a driver drowsiness detection system using multiple deep learning algorithms for binary classification, distinguishing between drowsy and non-drowsy states. The algorithms implemented include Convolutional Neural Networks (CNN), VGG16, GoogleNet, AlexNet, MobileNetV2, and ResNet101. Each model was pretrained on the ImageNet dataset and fine-tuned on a custom dataset of driver images. The architecture of each model was adjusted to include a Global Average Pooling layer followed by a dense layer with ReLU activation and afinal sigmoid output layer for binary classification. The models were trained using the Adam optimizer with binary cross-entropy loss. Performance metrics such as accuracy, precision, recall, and F1-score were used to evaluate and compare the efficacy of each model. Experimental results demonstrated that transfer learning with these architectures significantly improves the detection accuracy of driver drowsiness, Figure 3.

4.2. Convolutional Neural Network (CNN)

We have created two versions of the SimpleCNN model are defined, one without data augmentation and one with data augmentation. The SimpleCNN model is designed for image classification tasks, particularly for driver drowsiness detection in this context. It consists of three convolutional layers followed by max-pooling layers to extract and down sample image features. The convolutional layers, namely conv1, conv2, and conv3, are responsible for detecting various patterns and features in the input images. Each convolutional layer is followed by a Rectified Linear Unit (ReLU) activation function to introduce non-linearity and increase the model's expressive power. After the convolutional layers, max-pooling layers are applied to reduce the spatial dimensions of the feature maps while preserving the most important information. These pooling layers help in capturing the most salient features and reducing computational complexity.It consists of three convolutional layers followed by max-pooling layers to extract and down sample image features.

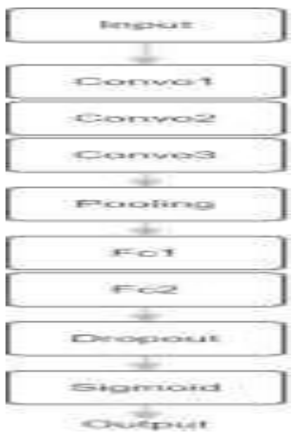


Figure 4 Architecture of CNN

Following the convolutional and pooling layers, the feature maps are flattened and fed into two fully connected layers: fc1 and fc2. These layers serve as the classifier, mapping the extracted features to the output classes (drowsy or non-drowsy). The first fully connected layer, fc1, has 512 neurons, followed by

e ISSN: 2584-2854
Volume: 03
Issue:02 February 2025
Page No: 256-264

another layer (fc2) with the number of neurons equal to the number of output classes. To prevent overfitting, a dropout layer with a dropoutrate of 0.5 is added after the first fully connected layer.Dropout randomly sets a fraction of input units to zeroduring training, which helps in regularizing the modeland reducing its reliance on specific features.Fig 5 shows the confusion matrix for Regarding data transformations, the version without augmentation applies simple preprocessing steps to the input images. It resizes each image to a fixed size of 224x224 pixelsand normalizes the pixel values using mean and standard deviation values computed from the ImageNet dataset. These transformations ensure uniformity in the input data and facilitate efficient training of the neural network. In contrast, the version with data augmentation incorporates additional random transformations to increase the diversity of the training data. These transformations include random resized cropping and horizontal flipping, whichintroduce variations in the training images. By augmenting the training data with these random transformations, the model becomes more robust to variations in input images and is less likely to overfit to the training set. Overall, both versions of the SimpleCNN model sharethe same architecture but differ in the data preprocessing steps applied to the input images. While the version without augmentation relies on fixed-size images and standard normalization, the version with augmentation leverages random transformations to enhance the model's generalization capability, Figure 4.

4.3. VGG16

VGG16, short for Visual Geometry Group 16, is a deep convolutional neural network architecture developed by the Visual Geometry Group at the University of Oxford. It is characterized by its simplicity, comprising a series of convolutional layers followed by max-pooling layers. The architecture consists of 13 convolutional layers and three fully connected layers.The model is employed as a feature extractor and classifier for driver drowsinessdetection. The model is loaded with pre-trained weights, which were learned on the ImageNet dataset, a large dataset with millions of labeled images across thousands of categories. By leveraging these pre- trained weights, the model has already learned to recognize a wide variety of visual features, making it well-suited for transfer learning.The last fully connected layer of the model isreplaced with a new fully connected layer. This new layer has the number of output neurons equal to the number of classes in the dataset (in this case, the classes are drowsy and non-drowsy). By replacing thelast layer, the model can be fine-tuned to classifyimages specific to the driver drowsiness detection task. During the training phase, the model is optimized using the stochastic gradient descent (SGD) optimizer with a learning rate of 0.001 and momentum of 0.9. The model's performance is evaluated using both training and validation data,with metrics such as loss and accuracy tracked over multiple epochs.

4.4. GoogleNet

We utilized GoogLeNet, a deep convolutional neural network, for the efficient detection of driver drowsiness. GoogLeNet's architecture, renowned for its inception modules, allows for efficient computation and deeper network designs. This makes it particularly suitable for complex tasks like image classification. We leveraged transfer learning by using a pre-trained GoogLeNet model, which was then fine-tuned for our specific task of binary classification: detecting drowsy versus non-drowsy drivers. It replacing its final fully connected layer with one that outputs two classes. This modification enabled the network to focus on our specific classification needs.To improve the model's generalization and robustness, we applied data augmentation techniques,such as random cropping and horizontal flipping, during the preprocessing stage. Test accuracy:0.9943

4.5. AlexNet

The AlexNet architecture was efficiently utilized for driver drowsiness detection by leveraging its powerful convolutional layers for feature extraction and its fullyconnected layers for classification. By fine-tuning a pre-trained AlexNet model, we adapted it to recognize drowsy versus non-drowsy states, which involved modifying the final layer to match the number ofclasses in our detection task.

e ISSN: 2584-2854
Volume: 03
Issue:02 February 2025
Page No: 256-264

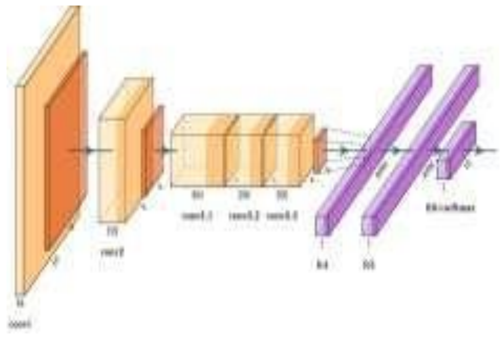


Figure 4 Architecture of AlexNet

We can observe the layers of Alexnet from fig.9. Transfer learning played a crucial role in thisprocess, allowing us to use the pre-trained weights of AlexNet, which had already learned rich features from a large-scale image dataset. This approach enabled us to achieve high accuracy even with a smaller,domain- specific dataset, as the model's extensive prior training helped in extracting meaningful patterns related to drowsiness.

4.6. MobileNet V2

MobileNetV2 was employed for detecting driver drowsiness, leveraging its efficient architecture designed for mobile and embedded applications . MobileNetV2, known for its balance between high accuracy and low computational cost, provided a robust solution for real-time drowsiness detection. We utilized the pre-trained MobileNetV2 model, which was initially trained on the ImageNet dataset. This pre-trained model served as a strong starting point due to its rich feature extraction capabilities.The final classification layer of MobileNetV2 was replaced with a new fully connected layer tailored to the specific number of classes in our drowsiness detection task. This adjustment allowed the model to output predictions aligned with our application. Instead of training the entire model from scratch, we fine-tuned it.

4.7. ResNet101

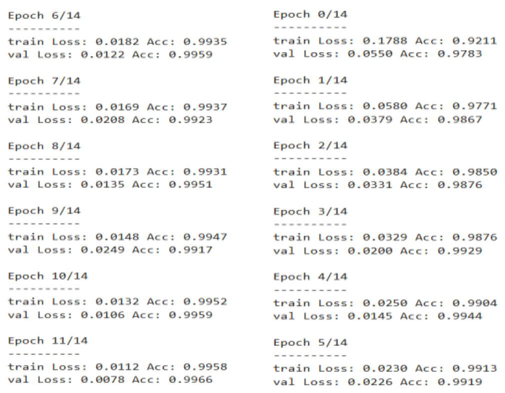
ResNet-101 was employed to detect driver drowsiness, utilizing its advanced deep learning structure to ensure high precision and reliability in this task. ResNet-101, with its deep architecture, is known for achieving high performance on complex tasks. Its residual connections help in training very deep networks efficiently, preventing issues like vanishing gradients. Despite its depth, ResNet-101 can be optimized for deployment in real-time systems with appropriate hardware, making it suitable for in-car embedded systems to monitor driver drowsiness in real-time. We used the ResNet-101 model pre-trained on the ImageNet dataset. This pre-trained model provided a strong foundation due to its extensive feature extraction capabilities. The final fully connected layer of ResNet-101 was replaced to match the number of classes in our drowsiness detection task. This allowed the model to output predictions relevantto our specific application.

4.8. Preprocessing

A standardized preprocessing pipeline was employed to optimize the input images for effective analysis. This pipeline comprised several essential transformations to ensure that the data was appropriately formatted and augmented for robust model training and evaluation. The first step involved applying a Random Resized Crop transformation, which randomly selected and resized regions of the images to a fixed size of 224x224 pixels. By introducing random cropping, the model was exposed to diverse sections of the images, facilitating the learning of invariant features and improving generalization performance. Following the crop operation, a Random Horizontal Flip was performed. This augmentation technique horizontally flipped the images with a certain probability, enriching the dataset with additional variations in orientation. This augmentationencouraged the model to learn features invariant to left-right orientation changes, thereby enhancing its ability to detect drowsiness from images captured under different conditions. Subsequently, the images were converted into PyTorch tensors using the ToTensor transformation. This conversion facilitated seamless integration with the PyTorch framework and enabled efficient computation on GPU devices duringboth training and inference stages. Finally, a Normalization step was applied to standardize thepixel values of the images. By adjusting the pixel intensities to have a mean of [0.485,0.456,0.406] and a standard deviation of [0.229,0.224,0.225], the input data was normalized to a common scale. This normalization ensured that the model received input data with consistent statistical properties, promoting stable and effective training across different architectures.

4.9. Training and Validation Phase

The training process involved multiple iterations (epochs) over the entire training dataset,during which the model learned to make accurate predictions and adjust its parameters based on the observed errors.



```
Epoch 6/14                              Epoch 0/14
----------                              ----------
train Loss: 0.0182 Acc: 0.9935          train Loss: 0.1788 Acc: 0.9211
val Loss: 0.0122 Acc: 0.9959            val Loss: 0.0550 Acc: 0.9783

Epoch 7/14                              Epoch 1/14
----------                              ----------
train Loss: 0.0169 Acc: 0.9937          train Loss: 0.0580 Acc: 0.9771
val Loss: 0.0208 Acc: 0.9923            val Loss: 0.0379 Acc: 0.9867

Epoch 8/14                              Epoch 2/14
----------                              ----------
train Loss: 0.0173 Acc: 0.9931          train Loss: 0.0384 Acc: 0.9850
val Loss: 0.0135 Acc: 0.9951            val Loss: 0.0331 Acc: 0.9876

Epoch 9/14                              Epoch 3/14
----------                              ----------
train Loss: 0.0148 Acc: 0.9947          train Loss: 0.0329 Acc: 0.9876
val Loss: 0.0249 Acc: 0.9917            val Loss: 0.0200 Acc: 0.9929

Epoch 10/14                             Epoch 4/14
----------                              ----------
train Loss: 0.0132 Acc: 0.9952          train Loss: 0.0250 Acc: 0.9904
val Loss: 0.0106 Acc: 0.9959            val Loss: 0.0145 Acc: 0.9944

Epoch 11/14                             Epoch 5/14
----------                              ----------
train Loss: 0.0112 Acc: 0.9958          train Loss: 0.0230 Acc: 0.9913
val Loss: 0.0078 Acc: 0.9966            val Loss: 0.0226 Acc: 0.9919
```

Figure 5 Training Epoch in Colab

Each model was initialized with its respective architecture, including pre-trained weights if applicable, Figure 5. The last fully connected layer of the model was modified to output predictions for the specific number of classes in the dataset. The training dataset was loaded into batches using PyTorch's DataLoader module. Batching allowed the model to process multiple samples simultaneously, leveraging parallelism for faster computation. Within each epoch,for both training and validation phases, the model performed a forward pass on the input data. The input images were fed into the model, and the model generated predictions for each sample in the batch.After obtaining the model predictions, the loss function was calculated to quantify the disparity between the predicted outputs and the ground truth labels. The cross-entropy loss, a common choice for classification tasks, was typically employed.With the loss computed, backpropagation was performed to calculate the gradients of the loss with respect to the model parameters. These gradients were then used to update the model's parameters (weights and biases) performance. At the end of each epoch, the model's performance was evaluated on both the training and Gradient Descent (SGD) or its variants. This process aimed to minimize the loss and improve the model's validation datasets. Metrics such as loss and accuracy were computed to assess the model's performance and monitor for overfitting. The training loop iterated over multiple epochs, with the model gradually improving its performance as it learned from the training data. Early stopping mechanisms might have been employed to prevent overfitting and determine the optimal number of epochs. loss computed, backpropagation was performed to calculate the gradients of the loss with respect to the model parameters. These gradients were then used to update the model's parameters (weights and biases) performance.At the end of each epoch, the model's performance was evaluated on both the training and

Gradient Descent (SGD) or its variants. This process aimed to minimize the loss and improve the model's validation datasets. Metrics such as loss and accuracy were computed to assess the model's performance and monitor for overfitting.The training loop iterated over multiple epochs, with the model gradually improving its performance as it learned from the training data. Early stopping mechanisms might have been employed to prevent overfitting and determine the optimal number of epochs, Table 2.

5. Result and Discussion

Table 2 Accuracy of All Models

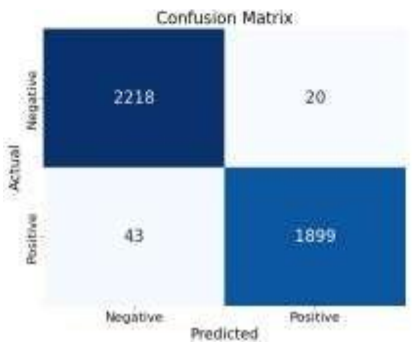
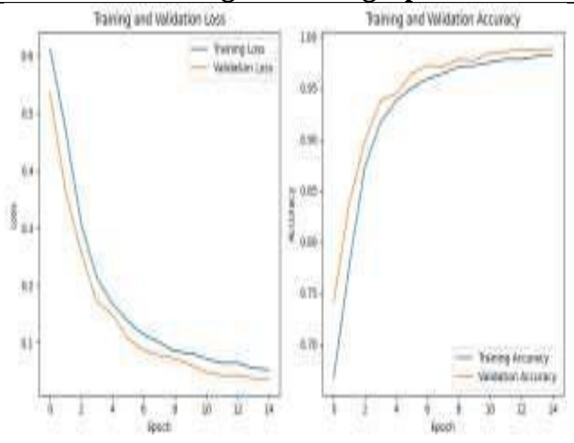
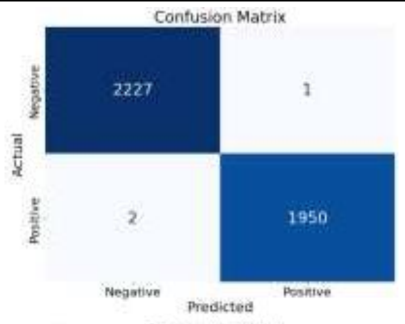
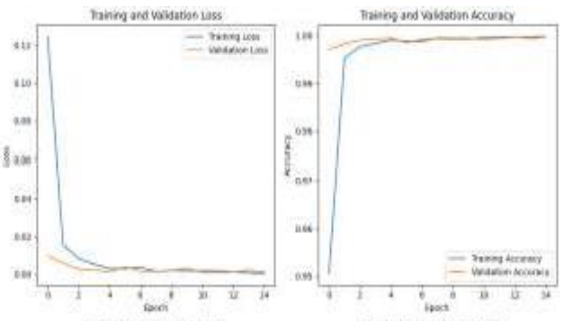
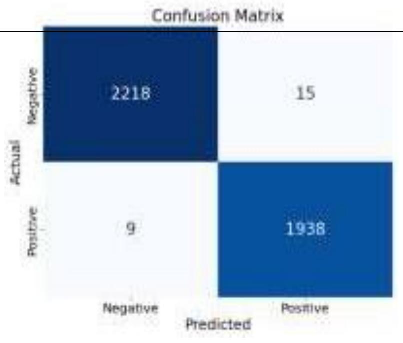
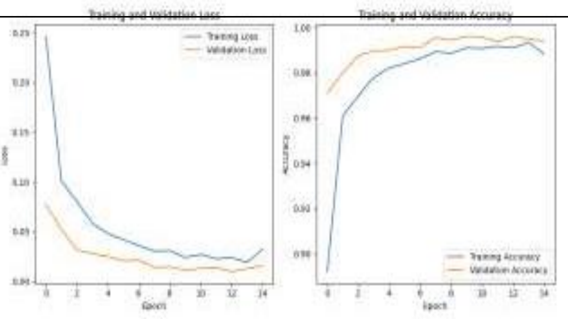
CNN without Augmentaion	0.9993
CNN with Augmentation	0.9849
AlexNet	0.9981
GoogleNet	0.9943
MobileNetV2	0.9988
VGG16	0.9981
ResNet101	0.9998

We have concluded the result with accuracy metrics Confusion matrix and Classification report. The evaluation of different deep learning models for driver drowsiness detection yielded notable results,

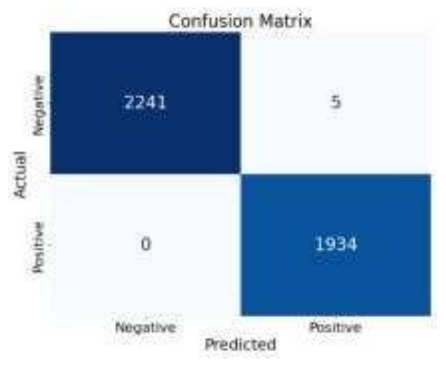
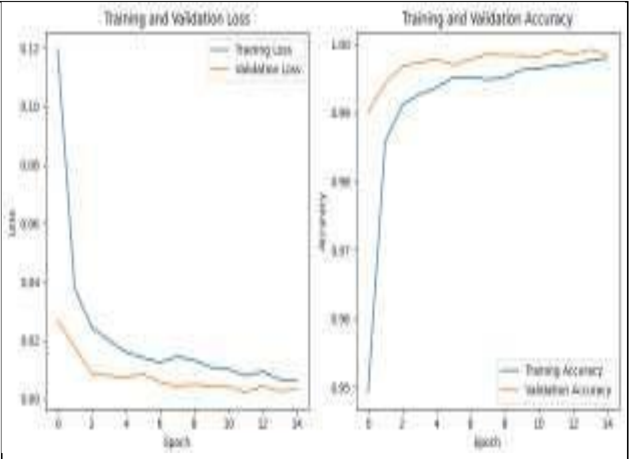
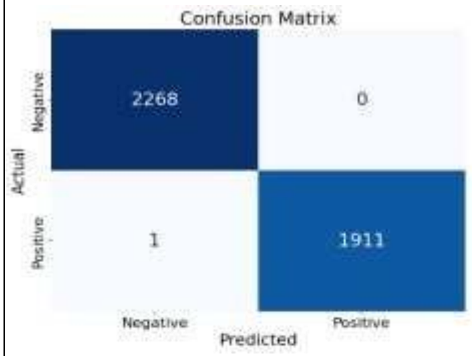
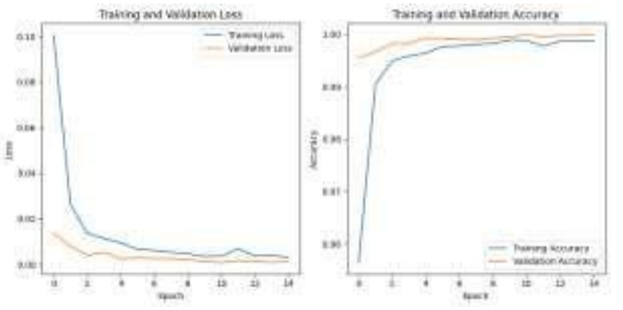
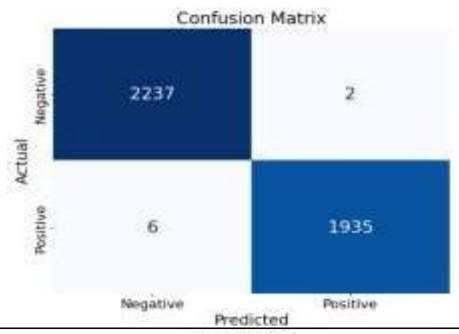
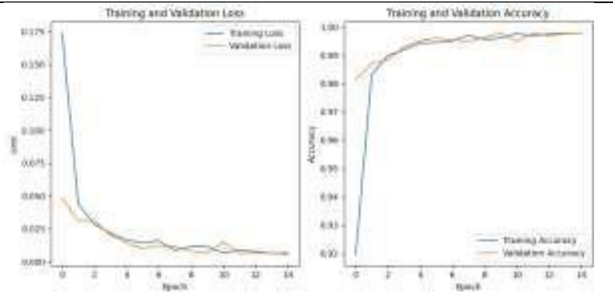
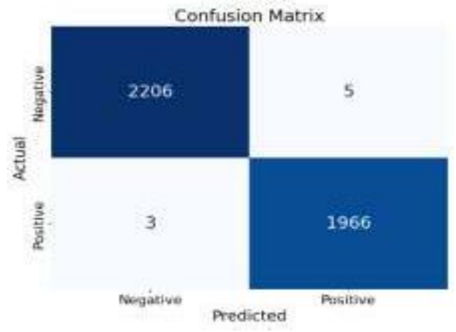
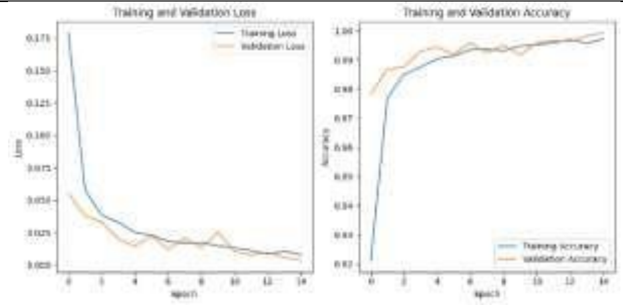
e ISSN: 2584-2854
Volume: 03
Issue:02 February 2025
Page No: 256-264

showcasing their efficacy in enhancing road safety. Among the models tested, ResNet-101 emerged as the top performer, achieving an impressive accuracy of 99.98%. Close behind, CNN without of 99.81%, indicating their reliability in drowsiness detection tasks. Despite slightly lower performance, GoogLeNet still achieved a commendable accuracy of 99.43%. The Simple CNN model, while exhibiting a lower accuracy of 98.49%, remains a viable option for drowsiness detection systems. These results underscore the effectiveness of deep learning architectures in accurately detecting driver drowsiness, thereby contributing to enhanced road safety measures. The high accuracy rates obtained by these models emphasize their potential to be integrated into real-time drowsiness detection systems, offering proactive measures to mitigate the risks associated with drowsy driving. Further research and development in this domain could lead to the implementation of more advanced and robust models, ultimately fostering safer transportation environments for all road users, shown in Table 3.

Table 3 Confusion Matrix and Graph

S.NO	Models	Confusion Matrix	Training and Loss graph
1	CNN		
2	CNN without Augmentation		
3	GoogleNet		

e ISSN: 2584-2854
Volume: 03
Issue:02 February 2025
Page No: 256-264

4	MobileNetV2		
5	ResNet101		
6	VGG16		
7	AlexNet		

Conclusion

The superior performance of models such as ResNet-101, CNN without augmentation, and MobileNetV2 highlights their promise for real-world implementation in driver drowsiness detection systems. Further research and development in this domain could lead to the implementation of more advanced and robust models, ultimately fostering safer transportation environments for all road users.

References

[1]. "Automatic detection of driver impairment based on pupillary light reflex," IEEE Transactions on Intelligent Transportation Systems, pp. 111.

[2]. J. H. Yang, Z. H. Mao, L. Tijerina, T. Pilutti, J. F. Coughlin, and E. Feron, "Detection of driver fatigue caused by sleep deprivation," IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, vol. 39, no. 4, pp. 694–705, 2009.

[3]. K. Fujiwara, E. Abe, K. Kamata, C. Nakayama, Y. Suzuki, T. Yamakawa, T. Hiraoka, M. Kano, Y. Sumi, F. Masuda, M. Matsuo, and H. Kadotani, "Heart rate variability-based driver drowsiness detection and its validation with EEG," IEEE Transactions on Biomedical Engineering, vol. 66, no. 6, pp. 1769–1778, Jun. 2019.

[4]. S. Hu and G. Zheng, "Driver drowsiness detection with eyelid-related parameters by support vector machine," Expert Systems with Applications, vol. 36, no. 4, pp. 7651–7658, 2009.

[5]. C. F. P. George, "Sleep apnea, alertness, and motor vehicle crashes," American Journal of Respiratory and Critical Care Medicine, vol. 176, no. 10, pp. 954–956, Nov. 2007.

[6]. S. Srivastava and P. K. Singh, "Proof of optimality based on greedy algorithm for offline cache replacement algorithm," International Journal of Next-Generation Computing, vol. 13, no. 3, 2022.

[7]. P. Smiti, S. Srivastava, and N. Rakesh, "Video and audio streaming issues in multimedia application," in 2018 8th International Conference on Cloud Computing, Data Science & Engineering (Confluence), IEEE, 2018.

[8]. P. Smriti, S. Srivastava, and S. Singh, "Keyboard invariant biometric authentication," in 2018 4th International Conference on Computational Intelligence & Communication Technology (CICT), IEEE, 2018.

[9]. M. Dua, S. Singla, R. Raj, and A. Jangra, "Deep CNN models-based ensemble approach to driver drowsiness detection," Neural Computing and Applications, 2020. doi:10.1007/s00521-020-052097

[10]. P. K. Mall and P. K. Singh, "Explainable deep learning approach for shoulder abnormality detection in X-ray datasets," International Journal of Next-Generation Computing, vol. 13, no. 3, 2022.

[11]. P. K. Mall and P. K. Singh, "BoostNet: A method to enhance the performance of deep learning models on musculoskeletal radiograph X-ray images," International Journal of System Assurance Engineering and Management, pp. 1–15, 2022.

[12]. P. K. Mall, P. K. Singh, and D. Yadav, "GLCM-based feature extraction and medical X-ray image classification using machine learning techniques," in 2019 IEEE Conference on Information and Communication Technology, IEEE, 2019, pp. 1–6.

[13]. P. K. Mall and P. K. Singh, "Credence-Net: A semi-supervised deep learning approach for medical images," International Journal of Nanotechnology, vol. 20, 2022.

[14]. D. Irfan, X. Tang, V. Narayan, P. K. Mall, S. Srivastava, and V. Saravanan, "Prediction of quality food sale in mart using the AI-based TOR method," Journal of Food Quality, vol. 2022, 2022.