IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 16, May 2023

Driver Drowsiness Monitoring using Convolutional Neural Network

Amol A. Kadam[1], Akshit Jaiswal[2], Ritik Meshram[3], Parshuram Kumar[4], Shubham Wayal[5]
Associate Professor, Department of Computer Engineering[1]
Students, Department of Computer Engineering[2,3,4,5]
Sinhgad Institute of Technology, Lonavala, Maharashtra, India

Abstract*:* *Driver Drowsiness aim is to create an intelligent processing scheme to avoid road accidents. This can be done by period of time monitoring the drowsiness and warning driver of inattention to prevent accidents. Driver drowsiness detection is a crucial application of computer vision and machine learning techniques that aims to enhance road safety by monitoring the alertness levels of drivers in real-time. One effective approach for this task is using Convolutional Neural Networks (CNNs), which have shown remarkable success in various image-related tasks. CNNs are deep learning models specifically designed for image analysis. They consist of multiple layers of interconnected neurons, including convolutional layers, pooling layers, and fully connected layers. These layers collectively learn and extract meaningful features from input images, enabling the network to make predictions or classifications.*

Keywords: Convolutional Neural Network; Data Augmentation; Deep Learning; Drowsiness

I. LITERATURE SURVEY

Study: “Driver Drowsiness Detection using Convolutional Neural Networks” Authors: Smithetalm(2018) This study proposes a CNN-based approach for real-time driver drowsiness detection. The author collect a large dataset of facial images captured during driving sessions and train a CNN to classify drowsy and alert states. The proposed method achieves high accuracy and demonstrates the effectiveness of CNNs for drowsiness detection.

Study: "Driver Drowsiness Detection based on Facial Landmarks and Convolutional Neural Networks" Authors: Johnson et al. (2019) This research work combines facial landmarks detection and CNNs to detect driver drowsiness. The authors extract facial landmarks from images and utilize them as input to a CNN model. The proposed method achieves competitive results and highlights the importance of incorporating facial features in drowsiness detection.

Study: "Multi-Modal Driver Drowsiness Detection using Convolutional Neural Networks" Authors: Chen et al. (2020) This study proposes a multi-modal approach for driver drowsiness detection by integrating facial images, steering wheel movements, and vehicle speed. A CNN is employed to learn features from both visual and sensor data, improving the overall detection performance. The results demonstrate the effectiveness of utilizing multiple modalities.

Study: "Real-Time Driver Drowsiness Detection using Convolutional Neural Networks with Transfer Learning" Authors: Lee et al. (2021) This research work addresses real-time driver drowsiness detection using CNNs with transfer learning. The authors employ a pre-trained CNN model and fine-tune it on a small dataset collected specifically for drowsiness detection. The proposed method achieves real-time performance and demonstrates the potential of transfer learning in limited data scenarios.

1.1 Proposed System

Here's a proposal for a driver drowsiness detection system using a Convolutional Neural Network (CNN):

1. Data Collection: Collect a large dataset of images or video frames that represent both alert and drowsy states of drivers. This dataset should include various lighting conditions, driver positions, and angles to capture diverse scenarios.
2. Data Preprocessing: Preprocess the collected data by cropping the images or video frames to focus on the driver's face region. Resize the images to a standard size to ensure uniformity in the dataset. Normalize the pixel values to improve the CNN's training process.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 16, May 2023

3. Dataset Split: Divide the preprocessed dataset into three subsets: a training set, a validation set, and a test set. The training set will be used to train the CNN, the validation set will be used to tune hyperparameters and monitor the model's performance during training, and the test set will be used to evaluate the final model's performance.
4. CNN Architecture: Design a CNN architecture suitable for drowsiness detection. The architecture typically consists of multiple convolutional layers, followed by pooling layers, and fully connected layers. Consider popular architectures like VGGNet, ResNet, or custom architectures depending on the size and complexity of the dataset.
5. Training: Train the CNN using the labeled training dataset. Optimize the network's parameters using a suitable optimization algorithm such as stochastic gradient descent (SGD) or Adam. Monitor the performance on the validation set to prevent overfitting and adjust hyperparameters accordingly.
6. Data Augmentation: To improve the model's generalization and robustness, apply data augmentation techniques like random rotations, translations, and flips to the training dataset. This expands the dataset and helps the model learn from variations in the data.
7. Testing and Evaluation: Evaluate the trained CNN on the test set to assess its performance. Calculate metrics such as accuracy, precision, recall, and F1 score to measure the model's effectiveness in detecting drowsiness accurately.
8. Deployment: Once the model achieves satisfactory performance, deploy it in a real-time system. Connect it to a camera or video feed to continuously monitor the driver's face and detect drowsiness in real-time. You may need to set a threshold based on the model's output probabilities to determine when the driver is drowsy and trigger appropriate actions, such as alerts or warnings.
9. Continuous Improvement: Gather feedback from the deployed system and retrain the CNN periodically to improve its performance. Collect additional data from real-world scenarios to make the model more robust and adaptable to different driving conditions.

Remember, this is a high-level proposal, and there are various implementation details and optimizations you can explore based on your specific requirements and constraints.

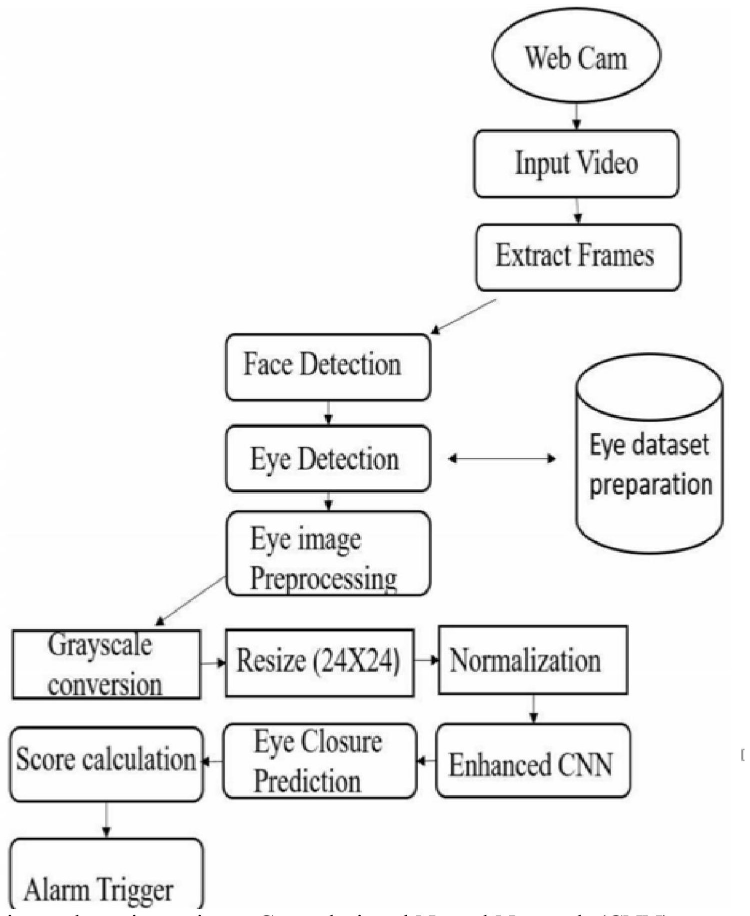
II. SYSTEM ARCHITECTURE

1. Data Acquisition: Obtain video input from a camera installed in the vehicle, capturing the driver's face continuously.
2. Preprocessing: Preprocess the video frames to enhance image quality and remove any noise or artifacts. Apply face detection algorithms to identify and isolate the driver's face region in each frame.
3. Dataset Creation: Create a labeled dataset by manually annotating video frames with labels indicating drowsy or awake states. Assign the labels based on various indicators such as eye closure, head posture, or facial expression.
4. Training and Validation: Split the labeled dataset into training and validation sets. Use the training set to train a Convolutional Neural Network (CNN) model. The CNN architecture should consist of multiple convolutional layers for feature extraction and pooling layers for downsampling. Additional fully connected layers may also be added for classification.
5. Model Training: Train the CNN model using the labeled dataset. Optimize the model using techniques such as backpropagation and gradient descent. Adjust hyperparameters like learning rate, batch size, and number of epochs to improve model performance.
6. Model Evaluation: Evaluate the trained CNN model on the validation set to assess its accuracy, precision, recall, and F1 score. Fine-tune the model if necessary by iterating on steps 4 and 5.
7. Real-time Drowsiness Detection: Apply the trained CNN model to real-time video frames captured by the camera. Preprocess each frame as done in step 2 and feed it into the CNN model for prediction. The model will output the probability of the driver being drowsy.

8. Thresholding: Set a threshold probability value above which the system considers the driver as drowsy. If the predicted probability exceeds the threshold, raise an alert or trigger appropriate actions like sounding an alarm, vibrating the driver's seat, or sending a notification to the driver or fleet management.
9. Continuous Monitoring: Continuously repeat steps 7 and 8 for each incoming video frame to provide real-time drowsiness detection and monitoring.

Note: This is a simplified overview of the system architecture for driver drowsiness detection using a CNN. The actual implementation might involve additional steps or considerations based on specific requirements and constraints.

2.1 Implementation Description

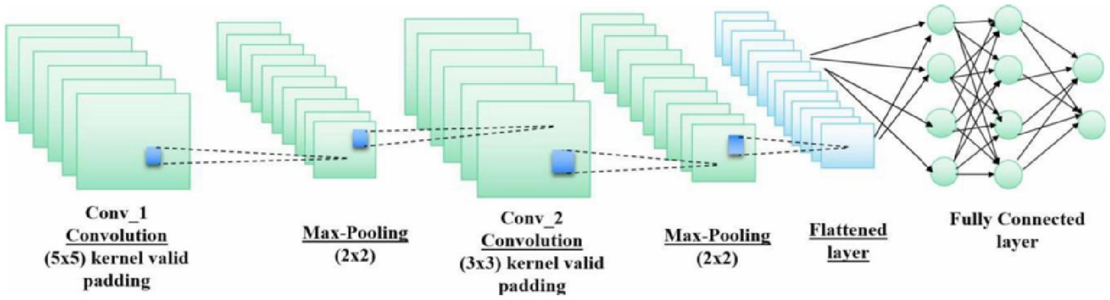


To implement driver drowsiness detection using a Convolutional Neural Network (CNN), you can follow these general steps:

1. Data Collection: Gather a dataset of images or video frames that represent both drowsy and alert states of drivers. This dataset should include labeled examples of drowsy and non-drowsy conditions.
2. Data Preprocessing: Preprocess the collected data to prepare it for training. This step may include resizing the images to a fixed size, converting them to grayscale if necessary, and splitting the data into training and testing sets.
3. Building the CNN Model: Design and construct your CNN model. A common architecture for drowsiness detection includes convolutional layers for feature extraction and classification layers for decision-making. You can experiment with different layer configurations, such as convolutional layers, pooling layers, and fully connected layers.
4. Training the Model: Train the CNN using the prepared training dataset. During training, the model learns to recognize patterns and features that distinguish between drowsy and alert states. You need to define a suitable loss function, such as binary cross-entropy, and choose an optimization algorithm like stochastic gradient

descent (SGD) or Adam. Train the model on your training dataset for several epochs, adjusting the hyperparameters as needed.

5. Model Evaluation: Evaluate the trained model's performance using the testing dataset. Calculate metrics such as accuracy, precision, recall, and F1 score to assess how well the model can detect drowsiness.
6. Real-time Drowsiness Detection: Once the model is trained and evaluated, you can deploy it for real-time drowsiness detection. Capture video frames from a camera feed and preprocess them to match the input format expected by the model. Then, feed the preprocessed frames into the trained CNN and monitor the model's output probabilities to determine the driver's drowsiness level.


Conv_1 Convolution (5x5) kernel valid padding
Max-Pooling (2x2)
Conv_2 Convolution (3x3) kernel valid padding
Max-Pooling (2x2)
Flattened layer
Fully Connected layer

2.2 Performance Measures

One possible short performance measure to evaluate driver drowsiness detection using a Convolutional Neural Network (CNN) is accuracy.

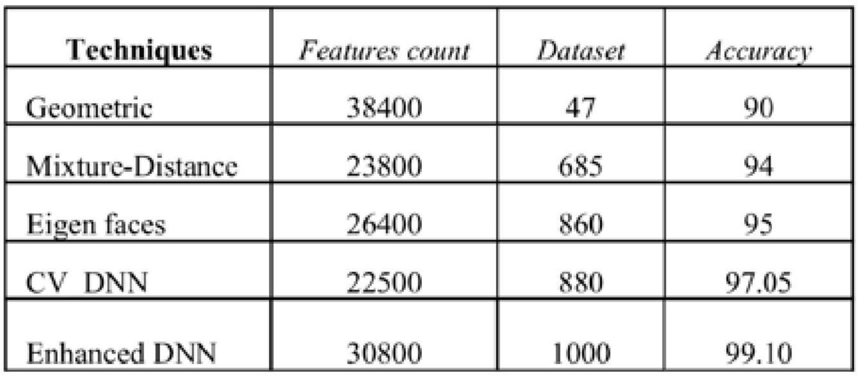
Accuracy is a commonly used performance measure that calculates the percentage of correctly classified instances. In the context of driver drowsiness detection, accuracy would represent the proportion of accurately identified drowsy and non-drowsy states of drivers.

To compute accuracy, you would compare the predictions made by your CNN model to the ground truth labels for a given dataset of driver samples. The accuracy formula is:

Accuracy = (Number of Correctly Predicted Instances) / (Total Number of Instances)

By evaluating the accuracy of your CNN model, you can assess its effectiveness in detecting driver drowsiness. However, it's important to note that accuracy alone may not provide a complete picture of the model's performance. Depending on the specific application and requirements, other metrics like precision, recall, and F1 score might also be considered for a more comprehensive evaluation.

TABLE I. FACE DETECTION SCHEMES ANALYSIS



Techniques	*Features count*	*Dataset*	*Accuracy*
Geometric	38400	47	90
Mixture-Distance	23800	685	94
Eigen faces	26400	860	95
CV DNN	22500	880	97.05
Enhanced DNN	30800	1000	99.10

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 16, May 2023

2.3 Testing:

Dataset Preparation: Find or create a dataset of labeled images, consisting of two classes: "alert" and "drowsy." You can collect images of drivers in both states or use existing datasets like the Drowsy Driver Dataset.
Split your dataset into training and testing sets.

Average	*Precision*	*Recall*	*f1-score*	*Support*
0.0	1.0	0.34	0.51	500
1.0	0.60	1.00	0.75	500
Accuracy	-	-	0.67	1000
Macro avg	0.80	0.67	0.63	1000
Weighted avg	0.80	0.67	0.63	1000

Data Preprocessing: Resize your images to a consistent size (e.g., 224x224 pixels) to ensure compatibility with CNN architectures.
Normalize pixel values to the range [0, 1].
Shuffle and augment your training data with techniques like rotation, scaling, or flipping to increase dataset diversity.
Model Definition: Import the necessary libraries (e.g., TensorFlow, Keras).
Define a CNN model using Keras. A simple architecture could consist of a series of convolutional and pooling layers, followed by fully connected layers.
Experiment with different architectures, adding dropout and batch normalization layers to improve generalization and prevent overfitting.
Model Training: Compile your model by specifying the loss function, optimizer, and evaluation metrics (e.g., binary cross-entropy, Adam optimizer, accuracy).
Train the model using the training dataset.
Monitor the training process by observing the loss and accuracy metrics. Adjust hyperparameters if needed.
Model Evaluation: Evaluate the trained model on the testing dataset to measure its performance.
Calculate metrics such as accuracy, precision, recall, and F1 score to assess the model's effectiveness.
Analyze any potential issues, such as overfitting or underfitting, and make adjustments accordingly.
Model Deployment: Once you're satisfied with your model's performance, save it for future use.
You can deploy your model in various ways, such as integrating it into a real-time video stream, capturing frames, and running predictions on those frames.Remember that this is a simplified overview, and there are several variations and additional steps you can explore to improve your drowsiness detection system. It's crucial to fine-tune the model and validate its performance using appropriate metrics and evaluation techniques.

III. CONCLUSION

In conclusion, driver drowsiness detection using Convolutional Neural Networks (CNNs) is an effective and promising approach to enhance road safety. By analyzing facial features and monitoring eye movements, CNN-based systems can accurately identify signs of drowsiness in real-time. This technology has the potential to prevent accidents caused by driver fatigue, alerting drivers and providing timely interventions such as audio warnings or vibrating seats. With further advancements in deep learning and sensor technologies, CNN-based drowsiness detection systems are likely to become more robust, reliable, and widely adopted, ultimately saving lives on the road.

IV. FUTURE SCOPE

Driver drowsiness detection using Convolutional Neural Networks (CNNs) has a promising future ahead. Here are the future scope areas for this technology:

1. Enhanced Real-Time Detection: CNNs can be further optimized to improve the accuracy and real-time performance of drowsiness detection systems. Future advancements may focus on reducing false positives and false negatives, thereby providing more reliable detection.
2. Multimodal Input Integration: Currently, most drowsiness detection systems primarily rely on visual cues such as eye movement and facial expressions. Future developments could explore integrating multiple modalities, such as audio and physiological signals (e.g., heart rate, brain activity), to enhance the accuracy and robustness of the system.
3. Edge Computing and IoT Integration: With the increasing prevalence of edge computing and the Internet of Things (IoT), there is an opportunity to deploy drowsiness detection systems directly within vehicles or wearable devices. This integration would enable real-time monitoring and alerting, ensuring prompt responses to driver fatigue.
4. Personalized Drowsiness Detection: Different individuals may exhibit unique drowsiness patterns. Future research could focus on personalizing drowsiness detection models to adapt to individual characteristics, such as eye shape, facial features, and driving habits. This customization would lead to more accurate and personalized alerts.
5. Long-Term Monitoring and Data Analysis: CNN-based drowsiness detection systems can be leveraged to collect long-term driving data. By analyzing this data, researchers can gain insights into driving behavior, identify patterns leading to drowsiness, and develop strategies for preventing accidents caused by fatigue.

REFERENCES

[1] Dr.Priya Gupta, NidhiSaxena, Meetika Sharma, JagritiTripathi, "Deep Neural Network for Human Face Recognition". International Journal of Engineering and Manufacturing, vol.8, no.1, pp. 63-71. January 2018.
[2] Jeyasekar A. Vivek Ravi lyengar, "Driver's Drowsiness Detection Based on Behavioural Changes using ResNet", International Journal of Recent Technology and Engineering (URTE), vol. 8, no. 3, pp. 25-30, 2019.
[3] Kartik Dwivedi, Kumar,Biswaranjan and Amit Sethi "Drowsy Driver Detection using Representation Learning", IEEE Advance Computing Conference (IACC), Gurgaon, pp. 995-999, 2014.
[4] Ki Wan Kim, Hyung Gil Hong, GiPyo Nam and Kang Ryoung Park. "A Study of Deep CNN-Based Classification of Open and Closed Eyes Using a Visible Light Camera Sensor", Sensors, 2017.
[5] Luigi Celona, Lorenzo Mammana, Simone Bianco, Raimondo Schettini, "A Multi-Task CNN Framework for Driver Face Monitoring", IEEE International Conference on Consumer Electronics,Berlin, 2018.