



ACGRNN: Attention Convolution Gated Recurrent Neural Network with Deep and Handcrafted features for Drowsiness Detection

Meqdad Jbaeer Sabah^{1,2*}, Kamal ElDahshan³, Ebeid Ali Obtained^{1,2}, AbdAllah A. AlHabsy⁴

¹ Ministry of Education, General Directorate of Anbar Education, Ramadi 31001, Iraq

² Department of Mathematics, Faculty of Science, Al Azhar University, Cairo 11651, Egypt

³ Computer Science and Information Systems, Al-Azhar University, Cairo 11651, Egypt

⁴ Department of Mathematics, Al-Azhar University, Cairo 11651, Egypt

Corresponding Author Email: moqdad7070@gmail.com

Copyright: ©2025 The authors. This article is published by IETA and is licensed under the CC BY 4.0 license (<http://creativecommons.org/licenses/by/4.0/>).

<https://doi.org/10.18280/jesa.581209>

ABSTRACT

Received: 28 October 2025

Revised: 26 November 2025

Accepted: 29 November 2025

Available online: 31 December 2025

Keywords:

drowsiness detection, wiener filter, Convolution Neural Network, handcrafted features, recurrent neural networks

Maintaining road safety and reducing accidents brought on by drowsy or exhausted driving depend heavily on the ability to detect driver fatigue. To improve road safety, it is essential to look at how drivers identify yawns. Even though a number of studies have suggested deep learning-based approaches, there is room for improvement in the creation of more accurate and efficient drowsiness detection systems that take into account behavioral factors like eye and mouth movements. In order to reliably identify sleepiness in real time using physiological and visual signals, this study suggests a deep neural network design that uses the Attention Convolution Gated Recurrent Neural Network (ACGRNN). The RMSprop optimizer, which effectively manages non-stationary goals and stabilizes the training process by dynamically adjusting the learning rate, is used to optimize the suggested system. Models are taught and assessed by contrasting them with current techniques. According to the experimental results, the suggested ACGRNN model achieves an average drowsiness detection accuracy of 95.53%.

1. INTRODUCTION

The paramount importance of safeguarding every person involved in transportation requires the implementation of a sleepiness detection system tailored for drivers. One important element contributing to car accidents is drowsy driving [1]. The state of sleepy driving may manifest as a brief lapse in cognitive focus, when the driver neglects to devote complete attention to the road. Those who have overexerted themselves, either physically or mentally, are more likely to experience tiredness while driving or to have mild sleepiness themselves [2]. This event may have transpired at an earlier or later temporal juncture. Studies reveal that over 25% of vehicular accidents are attributable to sleepy driving, with 4% of adult drivers admitting to experiencing sleepiness or falling asleep while driving in the preceding month [3]. Sleepy driving is a significant contributor to road safety issues in the United States, resulting in around 71,000 injuries, 1,500 fatalities, and annual financial losses of USD 12.5 billion. Due to the seriousness of this issue, it is essential to establish an effective system for the prompt detection of driver vulnerability, therefore mitigating accident risks and enhancing safety.

Three components typically make up a simple drowsiness detection system [4]: an acquisition framework to record the driver's frontal face, a processing framework to analyze the data for signs of fatigue, and a mindfulness tool to alert the driver when care is needed [5]. Drowsiness, often induced by medicine, results in diminished performance and reduced

attentiveness, perhaps leading to significant damage. The National Highway Traffic Safety Administration (NHTSA) estimates that drowsy drivers are responsible for around 100,000 injuries and over 1,500 fatalities per year [6]. This underscores the critical need for proactive measures to address fatigue, not just in driving but also in other contexts, such as operating heavy equipment, where similar risks are present [7].

Numerous factors may contribute to driver fatigue, which can lead to serious crashes, including sleep deprivation, lengthy drives, restlessness, alcohol use, and mental stress. The recent increase in road rage incidents has exacerbated stress levels among drivers, making conventional transportation methods inadequate for addressing the hazards of roads. To mitigate the danger of potentially lethal incidents, the use of automated tiredness detection systems in vehicles is essential. These gadgets consistently evaluate the driver's attentiveness and provide warnings far ahead of any significant threats to road safety [8, 9]. In accordance with the preceding explanation, a driver's actions are pivotal to road safety, for both the driver and other individuals using the roadways.

Driver sleepiness detection has garnered heightened interest recently due to its critical role in preserving lives. Numerous studies in the literature concentrate on identifying varying degrees of driver awareness by distinct facial indicators, including head positions, eye movements, and other facial expressions [10]. Despite current research indicating

significant gains, fundamental obstacles, including precise and real-time sleepiness detection, need to be resolved [11].

Furthermore, it is essential to note that references throughout this manuscript are organized to appear in numerical order according to their first occurrence. This ensures clarity and adherence to academic standards, where citation numbers are placed outside sentence grammar and reflect the sequence in which studies are introduced. Maintaining proper citation order not only enhances readability but also prevents ambiguity regarding the source of information.

Therefore, the contributions of this study are as follows:

ConvNextTiny adopts design principles from Vision Transformers while maintaining the efficiency of CNNs. Moreover, it incorporates depthwise convolutions to decouple spatial and channel-wise processing. This ensures high-resolution local feature encoding, vital for detecting micro-expressions (like eyelid droop or yawn onset) linked to drowsiness.

The deep features learned by ConvNextTiny complement handcrafted descriptors (e.g., EAR, MAR) by encoding nonlinear, texture-based cues, enabling a richer multi-modal representation when fused later in the pipeline.

Attention Convolution Gated Recurrent Neural Network (ACGRNN) with RMSprop Optimizer combines Convolutional layers for spatial context encoding with Gated Recurrent Units (GRUs) to model temporal dynamics in sequential data.

RMSprop is used due to its ability to handle non-stationary input distributions, which are common in facial expression data across video sequences.

The research's succeeding sections are organized as follows. A review of the literature on the identification of driver fatigue is given in Section 2. The methodology is described in Section 3, and the experimental results are explained in Section 4. Lastly, Section 5 summarizes the findings and makes recommendations for the future.

2. RELATED WORKS

Over the past ten years, research on driver sleepiness detection has advanced dramatically, utilizing both deep learning and physiological signal-based methods. Numerous techniques have been put forth, from multimodal strategies that include EEG or other biosignals to solely visual cues like yawning and eye closure.

In an early attempt at deep learning, Wei et al. [12] created a multi-granularity CNN + LSTM framework that achieved good accuracy on the NTHU-DDD dataset by capturing spatial features from numerous face patches and long-term temporal relationships over video sessions. Lyu et al. [13] demonstrated the efficacy of integrated spatial and temporal modeling by proposing a hybrid CNN-RNN model for real-time tiredness detection.

Convolutional architectures adapted to eye and mouth behavior are the subject of another type of studies. Zhao et al. [14] presented EM-CNN, which identifies eye and mouth states after using MTCNN to identify face features. They showed good sensitivity and accuracy for yawning and eyelid closure detection. EfficientNet-KNN was presented in a different work by Shen et al [15], in which EfficientNet is utilized for feature extraction over consecutive frames to

identify tiredness in real-time via head movement and eye closure length.

Tüfekci et al. [16] created an interpretable CNN for physiologically based detection of driver drowsiness utilizing EEG signals from several participants; by examining spatial-temporal EEG patterns, their model obtained strong cross-subject accuracy. In order to identify exhaustion, Chowdhury et al. [17] also used CNN architectures and EEG characteristics (such as theta and delta bands), emphasizing the trade-off between detection accuracy and intrusiveness. Additionally, hybrid models that combine physiological and visual characteristics have been investigated.

In order to achieve dependable detection under various circumstances, Kieley et al. [18] suggested a fusion framework that combines CNN-extracted facial landmarks with fuzzy logic to assess parameters like mouth opening and PERCLOS (% of eye closure over time). Particularly important are real-time systems. A DCNN + OpenCV pipeline for live video-based sleepiness detection was recently created by Majeed et al. [19], who reported extremely high classification accuracy on public datasets. In a similar vein, Florez et al. [20] presented VigilEye, an AI-based real-time driver monitoring system that instantly detects tiredness using CNNs and facial landmarks.

Other noteworthy contributions include: A 4-layer CNN was created by Makhmudov et al. [21] to process eye blinking and yawning behavior from video frames. Depending on the circumstances, the detection accuracy ranges from 80% to 98%. In order to determine blink rate for fatigue monitoring, Majeed et al. [22] suggested a visual approach with up to 86% accuracy utilizing symmetric eye characteristics. In order to identify tiredness, Sedik et al. [23] employed a CNN trained on picture sequences; their architecture caught latent spatial information and obtained 78% accuracy on a bespoke dataset. By evaluating mouth openness using facial cues and incorporating this into an alert system, Cui et al. [24] concentrated on yawning detection. Zhou et al. [25] fused visual and cognitive signs of weariness by using fuzzy logic with CNN outputs and eyelid closure characteristics. A real-time fatigue system utilizing ECG and EOG characteristics in conjunction with a lightweight neural network was proposed by Hashemi et al. [26].

In order to manage different environmental circumstances and driver behaviors, Soman et al. [27] employed an ensemble of models (AlexNet, VGG, ResNet); their multiclass system handled head position, yawning, and eye blinking. With an emphasis on subject-agnostic modeling, Ahmed et al. [28] expanded EEG-based detection to actual in-car environments. In order to identify extended eyelid closure and initiate driver alerts, Salman et al. [29] used eye-tracking and CNNs. Park et al. [30] achieved extremely high accuracy in yawning and blink recognition by using transfer learning with MobileNet-V2 and ResNet50V2 in conjunction with facial landmark-based thresholds.

3. PROPOSED METHODOLOGY

The process begins with input data, as shown in Figure 1, sourced from the Kaggle drowsiness dataset [31]. Initially, the input images undergo preprocessing using a wiener filter to reduce noise, followed by contrast enhancement through dynamic histogram equalization, ensuring improved feature visibility under varying lighting conditions. Deep features are obtained using the ConvNextTiny CNN model, a lightweight

and efficient convolutional neural network suitable for resource-constrained environments. The Mouth Aspect Ratio (MAR), Eye Aspect Ratio (EAR), pupil circularity, and a combined mouth and eye feature vector, which indicate sleepiness, are retrieved simultaneously. To classify images, the Attention-based Convolutional Gated Recurrent Neural Network (ACGRNN) captures spatial and temporal relationships in the data using these deep and custom features [32].

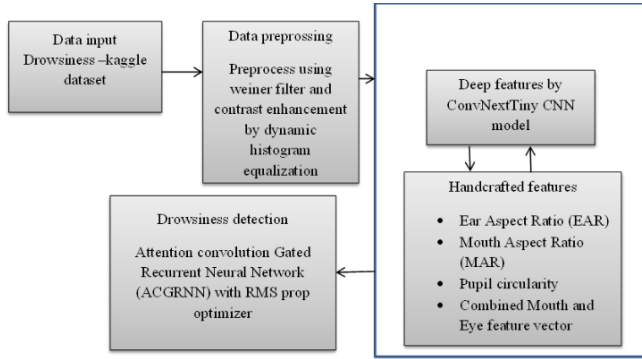


Figure 1. The work's block diagram

3.1 Preprocessing of images by wiener filter

This is a traditional statistical method that eliminates noise from the picture while retaining its features. The concept of its filtration is stated as follows:

$$g'(i, j) = \frac{\sigma_n^2}{\sigma_f^2} \times g'(i, j) + \frac{\sigma_f^2 - \sigma_n^2}{\sigma_f^2} \times g(i, j) \quad (1)$$

where, $g(i, j)$ indicates the pixel's intensity value. (i, j) and $g'(i, j)$ shows the average pixel intensity within the $M \times N$ window, which is centered at (i, j) . σ_n^2 and σ_f^2 indicate the differences between the actual image and the noise, respectively. There are two ways to assess the performance.

Case 1: "Designated area." The variance σ_I^2 significantly exceeds σ_n^2 that is $\sigma_I^2 \gg \sigma_n^2$, allowing us to derive $\frac{\sigma_n^2}{\sigma_f^2} \sim 0$ and $\frac{\sigma_f^2 - \sigma_n^2}{\sigma_f^2} \sim 1$; thus, $g'(i, j) \sim g(i, j)$. This indicates that the filter can maintain the edge information of the picture.

Case 2: "Homogeneous area." The variance σ_I^2 approximates σ_n^2 that is $\sigma_I^2 \sim \sigma_n^2$, allowing us to derive $\frac{\sigma_n^2}{\sigma_f^2} \sim 1$ and $\frac{\sigma_f^2 - \sigma_n^2}{\sigma_f^2} \sim 0$; thus $g'(i, j) \sim g(i, j)$. This indicates that the filter transforms into an average filter to mitigate the image's noise. Consequently, it can be inferred that the wiener filter is an edge-preserving smoothing filter characterized by self-adaptability and ease of implementation.

The filtered image undergoes histogram equalization, whereby the gray level histogram of an image is represented as a one-dimensional discrete function, describing the frequency of each gray level in the image. It may be expressed as:

$$h(k) = n_k \quad k = 0, 1, 2 \dots L - 1 \quad (2)$$

Shows how many of the image's pixels have a gray level of k . $f(x, y)$, L is the total number of gray levels, and n_k is the height of each histogram bin. Every gray level in the original

image is displayed in the grey level histogram. The normalized histogram is, since the histogram displays the relative frequency of grey level occurrences.

$$p_r(k) = \frac{n_k}{N} \quad k = 0, 1, 2, \dots L - 1 \quad (3)$$

Here, L represents the total count of gray levels, n_k indicates the pixel quantity at the k -th gray level, and N signifies the overall pixel count in the original picture. Thereafter, the cumulative distribution function of the normalized histogram of the image is computed.

$$s_k = T(r_k) = \sum_{j=0}^k P_r(r_j) = \sum_{j=0}^k \frac{n_j}{N} \quad 0 \leq r_k \leq 1 \quad k = 0, 1, 2, \dots L - 1 \quad (4)$$

Here, $r_k = \frac{k}{L-1}$, representing the normalized gray level, whereas k signifies the gray level prior to normalization. The transformation function $T(r_k)$ maps the original gray level to a new gray level, which then replaces the matching gray level in the original picture to produce an equalized image.

3.2 ConvNeXtTiny CNN based deep feature extraction

ConvNeXt [33] rivals Transformers in precision and scale. This convolutional model shown in Figure 2 was optimised from ResNet using a Swin Transformer-like structure [34]. The diminutive version of ConvNeXt is used to optimize detection accuracy and speed, as seen in Figure 2.

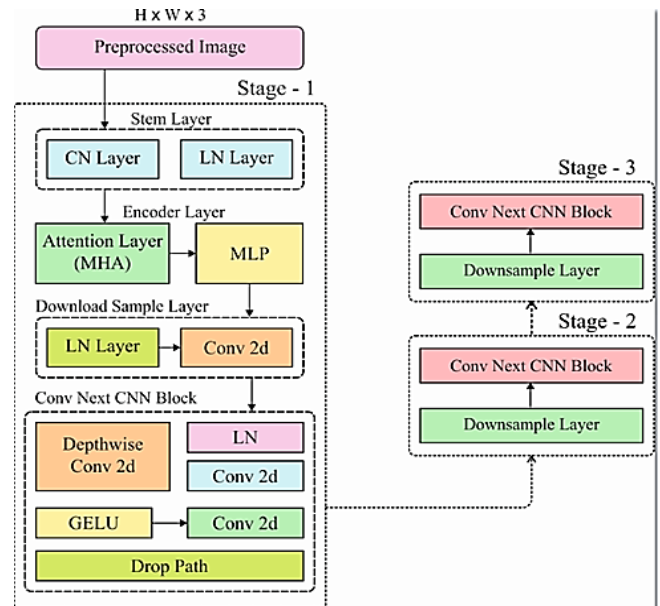


Figure 2. Architecture of ConvNeXtTiny CNN

Stem: The Stem has convolutional and LN layers. Assuming the input image dimensions are $H \times W \times 3$, where H and W represent height and breadth. The input image is downsampled by four using a 4×4 convolutional layer with C kernels. Next, the output feature undergoes layer normalisation.

Encoder layer: The multi-head attention parallel attention mechanism processing layer underpins this structure. The input token matrix is layer-normalized. Second, multiplying W^Q and W^K yields three matrices Q , K , and V , similar to the self-attention module. A matrix corresponding to the number

of heads h is created by multiplying the third Q , K , and V by W_i^Q, W_i^K, W_i^V . The attention score for each head is calculated using Eq. (5) and the Q_i, K_i, V_i matrix.

$$\begin{aligned} \text{head}_i &= \text{attention}(QW_i^Q, KW_i^K, VW_i^V) \\ W_i^Q &\in R^{d_{\text{model}} \times d_q}, W_i^K \in R^{d_{\text{model}} \times d_k}, W_i^V \in R^{d_{\text{model}} \times d_v}, d_q = d_k = d_v = d_{\text{model}}/h \end{aligned} \quad (5)$$

The output of the MHA layer is derived by concatenating all heads and applying a matrix-like fully connected operation as described in Eq. (6):

$$\begin{aligned} \text{multihead}(Q, K, V) &= \\ \text{concat}(\text{head}(1), \text{head}(2), \dots, \text{head}(h))W^o \end{aligned} \quad (6)$$

All transformer encoder layer output may be obtained via a residual connection before and after the MHA and MLP levels. Many transformer encoders are stacked to produce the model's encoder layer.

- The Downsample reduces feature map resolution and doubles channel size to create hierarchical multi-scale features. Layer normalisation and convolutional layers downsample. Convolutional layer with 2×2 kernel and 2 strides.

- The ConvNeXt Block includes depthwise convolution and a two-layer 1×1 convolution layer with GELU non-linearity. The second 1×1 convolution layer is followed by a layer scale, drop route, and residual connection in the ConvNeXt Block. Table 1 shows the details of the hyper-parameters of the convolution layer

Table 1. The hyper parameters of convolution layer

Layer	Type	Kernel Size	Filters	Stride
Conv1	1D Convolution	3	256	1
Conv2	1D Convolution	3	256	1
Norm	LayerNorm	—	—	—
Dropout	Dropout	—	—	—

3.3 Extraction of handcrafted features

- MAR

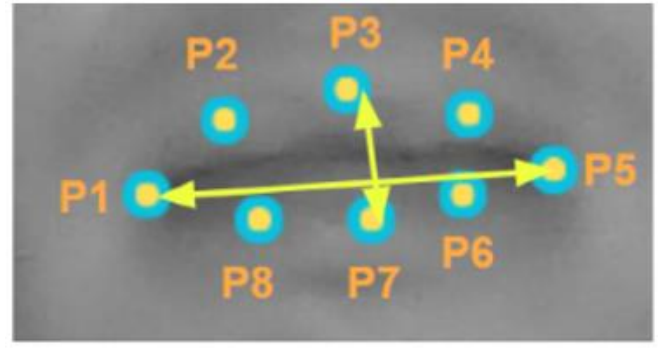
The MAR technique identifies driver yawning. It employs Euclidean coordinate distance. Sikander and Anwar [35] calculated the MAR using Eq. (7), which measures mouth height and breadth, using eight oral reference locations (Figure 3).

$$MAR = \frac{\|p_2 - p_8\| + \|p_3 - p_7\| + \|p_4 - p_6\|}{3\|p_5 - p_1\|} \quad (7)$$

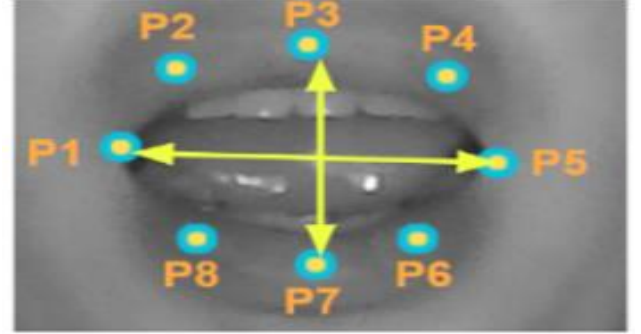
- EAR

Rosebrock asserts that using the EAR feature for blink detection has many benefits over conventional image-processing techniques. Eq. (8) was used to extract the EAR feature. The EAR ratio numerator calculates vertical landmark distance, as seen below. The denominator doubles the horizontal landmark distance by two to match the numerator. Each frame's EAR values were calculated as shown in Eq. (8) (Figure 4).

$$EAR = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|} \quad (8)$$



(a) Closed mouth ROI



(b) Open mouth ROI

Figure 3. ROIs of the mouth are used to determine the driver's condition

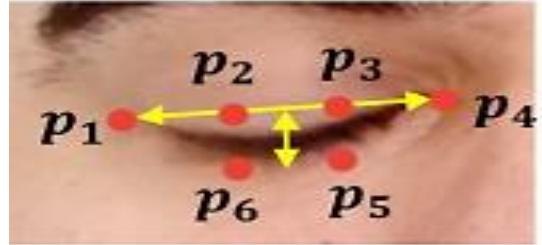
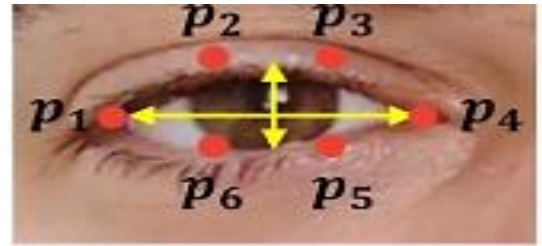


Figure 4. Representation of EAR

- Pupil circularity

The input key and the relevant pupil feature x are first converted into a novel feature space, akin to the self-attention process, resulting in x' via its linear transformation $\text{linear}Q(\text{conv}1 \times 1)$. The objective is to augment the expressive capacity of pupil features, extract more abstract and discriminative pupil characteristics, and then calculate the attention matrix A of the external memory units M_k relative to x' using the following formula.

$$A = \text{norm}(\text{linear}(q)(x')M_k^T) \quad (9)$$

where, $\theta_{i,j}$ denotes the similarity between the i -th pixel in x' and the j -th column memory value in M_k^T . The attention failure problem caused by a student's large feature vector may

be avoided by using dual-normalized norm. The attention's rows and columns are standardized by the double-normalized norm. $\theta'_{i,j}$ obtained using matrix multiplication. The following is one way to describe the realization process:

$$\theta'_{i,j} = \frac{e^{\theta'_{i,j}}}{\sum_k e^{\theta'_{i,k}}} \quad (10)$$

- Combined Mouth and Eye feature vector

Sleepiness detection methods that employ face cues emphasise eye features. The eye aspect ratio EYE_i was calculated using the MediaPipe Face Mesh 2D coordinates and Soukupová and Čech's calculating approach. Eq. (11) calculates eye aspect ratio at a particular instant by comparing the average vertical and horizontal eye landmark distances.

$$EYE_i = \frac{\text{mean}(\text{dis}(E_2, E_6), \text{dis}(E_3, E_5))}{\text{dis}(E_1, E_4)} \quad (11)$$

where, EYE_i is the driver's eye openness at time i , mean (a, b) is the input parameter average, and $\text{dis}(a, b)$ is the Euclidean distance between input parameters. Additional eye region metrics include blink duration (BD), eye closure duration (MEC), and eye movement amplitude. The blink duration calculating method is Eq. (12):

$$BD = \frac{\sum_{t=1}^n \text{end}_t - \text{start}_t + 1}{n} \quad (12)$$

The frame number end_t indicates the end of the t th blink process inside the time window, whereas start_t indicates the start of the blink process. Eye movement amplitude is calculated using Eq. (13).

$$\text{amp} = \frac{EYE_{\text{start}} + EYE_{\text{end}} - 2(EYE_{\text{peak}})}{2} \quad (13)$$

3.4 ACGRNN

Drowsiness detection based on ACGRNN, especially when combining spatial and temporal features from multimodal feature extracted from previous step. An attention model computes a vector c_i as the weighted mean of the feature h , as shown in Eq. (14).

$$c_t = \sum_{j=1}^T \alpha_{ij} h_j \quad (14)$$

The weighted mean is generated by a latent concealed state h_t . T represents the time step required for producing the input feature, and α_{ij} denotes a weight assessed at time t for the input corresponding to state h_j . The new state s evaluates the context vector, where s_t depends on s_{t-1} , c_t , and the output at time $t-1$. The weights α_{ij} are determined using Eq. (15).

$$e_{ij} = a(s_{t-1}, h_j) \propto e_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^T \exp(e_{ik})} \quad (15)$$

The prior state s_{t-1} and h_j are used in the computation of a learning function. This function aids in the computation of h_j . It generates a fixed-length c vector in Eq. (16).

$$e_t = a(h_t), \alpha_t = \frac{\exp(e_t)}{\sum_{k=1}^T \exp(e_k)}, c_t = \sum_{t=1}^T \alpha_t h_t \quad (16)$$

When the input remains constant throughout time T , the whole network will operate without self-attention. The whole self-attention mechanism will be used when the input sequence varies. The unweighted average of h_t , which aids in the computation of c , is shown in Eq. (17).

$$c_t = \frac{1}{T} \sum_{t=1}^T h_t \quad (17)$$

Subsequent to the activation layer, the GCNN regulates the recurrent term via a gate before to its addition to the input term.

$$c_t \cdot x(t) = T^F(u, w^F) + g(T) \cdot (x(t-1); w^R(t-1)) \quad (18)$$

When $t \geq 0$, $g(T)$ is a gate whose outputs possess the same dimensions as $T^R(x(t-1); w^R(t-1))$ and $x(t)$.

$$G(t) = \sigma(T_g^F(u; w_g^F) + T_g^R(x(t-1); w_g^R(t-1))) \quad (19)$$

For $t \geq 0$, let σ be the logistic sigmoid function defined as $\sigma(x) = 1/(1 + \exp(-x))$. T_g^F and T_g^R represent the feed-forward transformation and recurrent transformation, respectively, using the pre-activation approach to ascertain the gate's output, each possessing distinct parameters. w_g^F and $w_g^R(t)$. It should be noted whether or not $w_g^R(t)$ may be shared across t . The convolutional layers in w_g^F and $w_g^R(t)$ comprise of 1×1 convolutional filters to ensure the dimensions of the modified u and $x(t-1)$ align with $G(t)$.

Consequently, the softmax layer categorizes the classes as open eyes, closed eyes, yawning, and not yawning. Input frames are arranged into sequences of length $T = 16$ in order to capture temporal continuity in driver behavior." ConvNeXtTiny is employed to extract each frame's characteristics. These characteristics include stacked to create a sequence of 512-dimensional vectors every frame, which are subsequently fed into the bidirectional GRU. By ensuring overlapping sequences, a sliding window method with stride 8 increases sensitivity to slow eye closures and yawns. The attention module highlights crucial temporal points that correlate to signs of tiredness by weighing each frame in the sequence. The convolutional layers in and comprise 1×1 convolutional filter to ensure the dimensions of the modified and align with. Consequently, the softmax layer categorizes the classes as Yawning, closed eyes, open eyes, and not yawning (Table 2). The parameters of the Attention modules in ACGRNN are summarized in Table 3.

Table 2. The hyper parameters of GRU

Parameter	Value
Hidden units	256
Num. layers	1
Bidirectional	Yes
Output dimension	512 (because bidirectional)
Dropout	0.2
Return_sequences	True

$$et = \tanh(W_h h_t + b_h) \quad (20)$$

The Attention modules in ACGRNN equation and parameter can be shown in Table 3, and the algorithm: GRU and Attention mechanism below.

Table 3. The Attention modules in ACGRNN parameters

Component	Specification
W _h	Linear layer (512 → 128)
Activation	Tanh
v	Linear layer (128 → 1)
Normalization	Softmax across time steps
Attention heads	1 (single head)

Algorithm: GRU and Attention mechanism

LOAD dataset:

- Each sample = sequence of image frames
- Labels = ['open', 'closed', 'yawn', 'non-yawn']

def __init__(self, num_classes=4):

super(ACGRNN, self)

self.gru = nn.GRU

input_size=512,

hidden_size=256,

batch_first=True,

bidirectional=True

self.attn = nn.Linear(512, 1)

self.fc = nn.Linear(512, num_classes)

FUNCTION Classifier(context_vector):

- Fully Connected Layer
- Softmax to get class probabilities

RETURN class_probabilities

return out

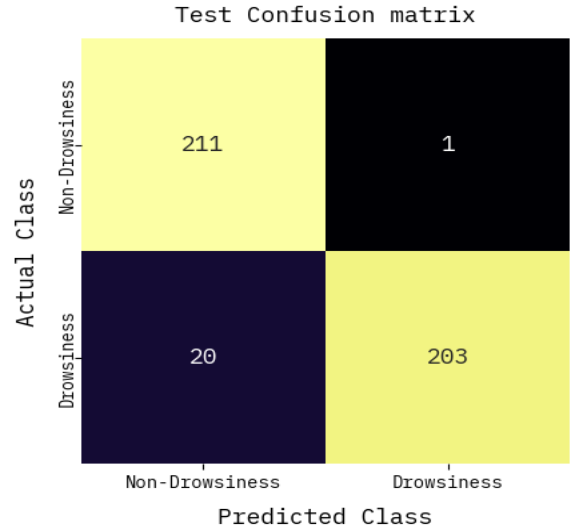
deep learning software packages were used to build and train our models. Standard performance metrics, including accuracy, precision, recall, and F1-score, are used to evaluate the proposed method and similar models. Eqs. (1)-(24) provide the technical definition of the assessment matrices, while Figures 1-4 show the preprocessing and feature extraction procedures. Tables 1-3 include specifications for the convolutional and GRU layers' hyperparameters and modules. Figures 5-12 and Tables 4-6 show the testing and training results, including confusion matrices, precision-recall curves, ROC curves, optimizer analysis, and ablation experiments.

$$accuracy = \frac{tp+tn}{tp+fp+tn+fn} \quad (21)$$

$$precision = \frac{tp}{tp+fp} \quad (22)$$

$$recall = \frac{tp}{tp+fn} \quad (23)$$

$$F1 - score = \frac{2tp}{2tp+fp+fn} \quad (24)$$

**Figure 5.** Confusion matrix of testing process

4. RESULT AND ANALYSIS

4.1 Dataset description

The driver downiness dataset may be used to train and test models on Kaggle [36]. The 2900 images in the collection are divided into four categories based on how sleepy they are: open, closed, yawning, and no yawning. The dataset clarifies eye conditions. The dataset contains eye condition labels and various other variables that improve driver sleepiness analysis. The gender function shows the driver's gender in photos, allowing comparisons of drowsiness patterns across genders. The age feature divides drivers into age categories, making it easier to study drowsiness trends across age groups. The driver's head posture is described by the characteristic. It reveals how head posture affects tiredness and whether certain postures are more prevalent among drowsy drivers. Finally, image lighting is described by the illumination property, which is essential for effective face identification. Developing accurate and effective driver sleepiness models requires understanding how light affects detection. The dataset has virtually equal gender and age distribution. Male drivers account for 1490 photos and female drivers for 1410. Young, middle-aged, and old are age groups. These groups have 1100, 1000, and 800 images. However, these variables were not analyzed but intended for future research extension. The collection includes 726 open, 726 closed, 725 yawn, and 723 non-yawn images.

4.2 Performance analysis

The Keras framework is used in this study, with TensorFlow serving as the backend for each experiment. Python 3.7 and Windows 10 were utilized in the tests, along with 8 GB of RAM, a 2.4 GHz Intel Core i7 CPU, and 1 TB of auxiliary storage. A Jupyter notebook and several machine learning and

The testing confusion matrix in Figure 5 shows that out of 435 test samples, 414 were correctly classified. Specifically, 211 non-drowsy cases and 203 drowsy cases were correctly identified. Only 21 instances were misclassified, including 1 non-drowsy instance incorrectly identified as drowsy and 20 drowsy instances classified as non-drowsy. This indicates a high true positive rate for detecting drowsiness and a low false positive rate, confirming the model's robustness in distinguishing between the two classes.

For the sleepy class, Figure 6 shows that the model obtains an average accuracy (AP) of 0.9926, suggesting few false positives. In a similar vein, the AP for the non-drowsy class is 0.9866. These findings demonstrate that the model reliably detects driver tiredness in a variety of scenarios while maintaining high accuracy and great recall.

For both groups, the ROC curve in Figure 7 shows an AUC of 0.9901. This illustrates the model's exceptional sensitivity and precision in distinguishing between sleepy and non-drowsy phases.

Out of the training samples, 508 non-drowsy and 460 drowsy cases were accurately identified, as shown in Figure 8. There were 45 instances of misclassifications (5 non-drowsy

misclassified as drowsy, 40 drowsy misclassified as non-drowsy). With just slight overfitting, the model appears to generalize well based on the high percentage of accurate predictions.

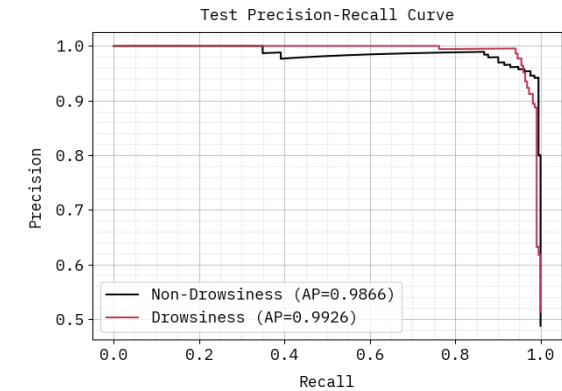


Figure 6. Precision-recall curve for testing process

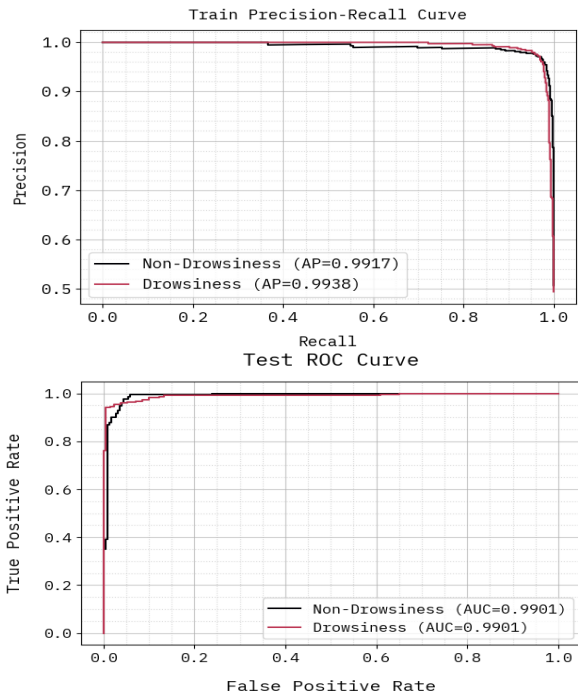


Figure 7. ROC curve for testing process

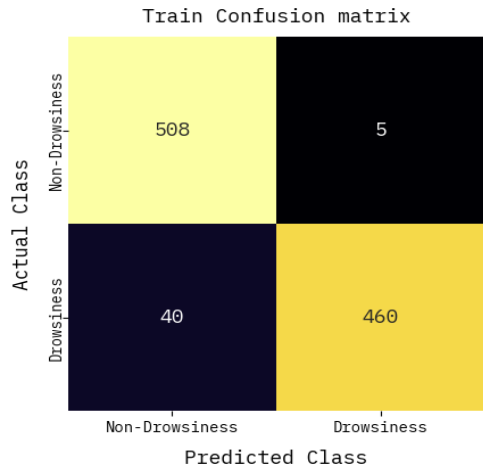


Figure 8. Confusion matrix of training process

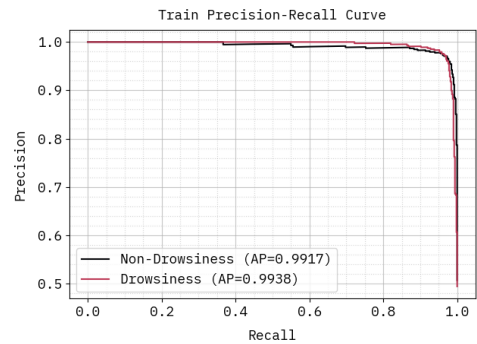


Figure 9. Precision-recall curve for training process

The ACGRNN attains AP values of 0.9938 for sleepy classes and 0.9917 for non-sleepy courses after training, as shown in Figure 9. This suggests that both false positives and false negatives are reduced by the model by successfully balancing accuracy and recall.

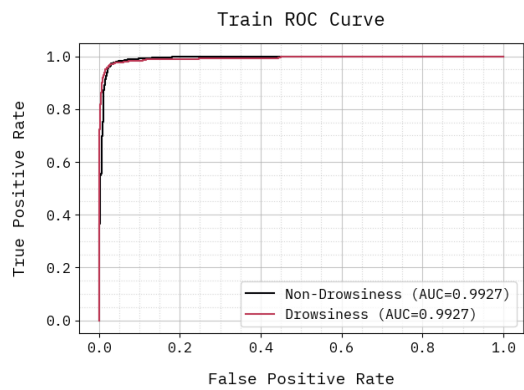


Figure 10. ROC curve for testing process

The model retains excellent separation among drowsy and awake classes during training, indicating stability and reliable feature learning, according to Figure 10's ROC curve, which has an AUC of 0.9927.

Table 4. Analysis of various metrics for testing and training process

Metrics	Testing	Training
Accuracy	0.9517	0.9556
precision	0.9543	0.9581
recall	0.9528	0.9551
F1-score	0.9517	0.9555
MCC	0.9071	0.9132

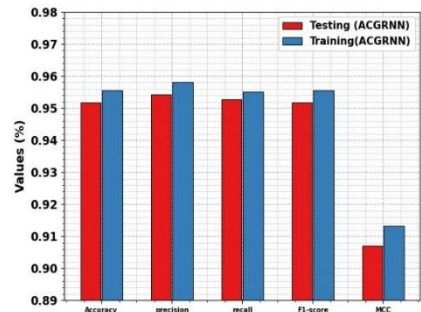
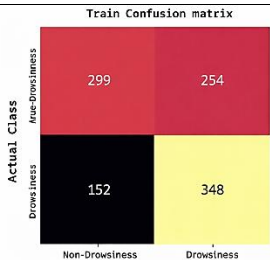
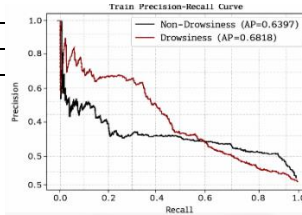
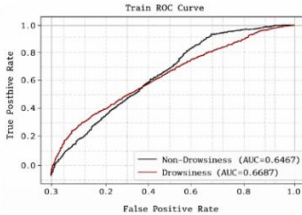
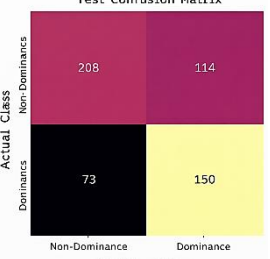
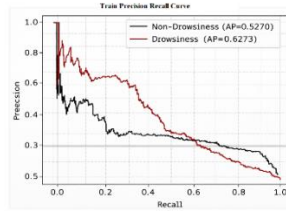
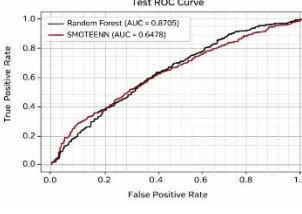
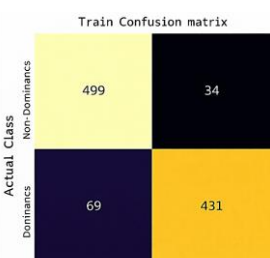
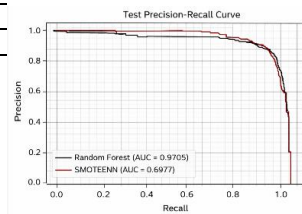
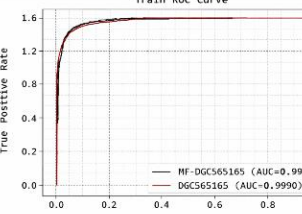
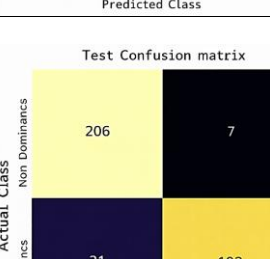
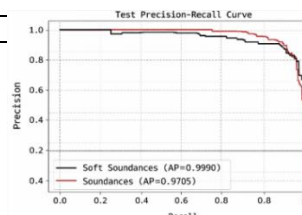
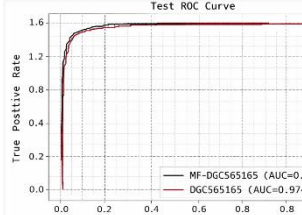
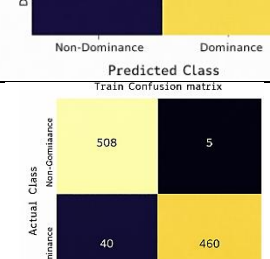
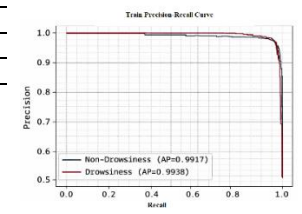
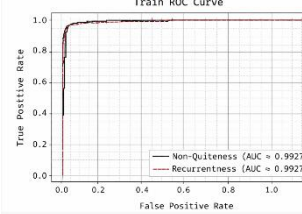


Figure 11. Analysis of proposed method for training and testing

The Figure 11 shows the performance of the ACGRNN model on training as well as testing datasets using general classification metrics. The model shows good accuracy, with around 95.6% in training and 95.2% for testing, showing good generalization ability with little overfitting. Precision and recall figures are steady high levels of 96.0% and 95.7% for training, and 95.4% and 95.2% for testing, respectively, indicating Particularly when it comes to detecting tiredness, the model performs well in reducing both false positives and false negatives. The model's dependability in performance is

confirmed by the F1-score, which evaluates accuracy and recall and records training and testing values at around 95.8% and 95.3%, respectively. It's also important to note that the Matthews Correlation Coefficient (MCC), a comprehensive metric that takes into account every result in the confusion matrix, is ever so slightly lower but nevertheless robust at 91.2% for training and 90.3% for test emphasizing the model's general consistency of classification even amidst class imbalances.

Table 5. Analysis on various optimizers

Optimize r	Confusion Matrix		Metrics	Values	Pr_curve	ROC_curve
Adamgrad	Tra in	<div>Train Confusion matrix</div> 	Accuracy	0.5992		
		Precision	0.6041			
		Recall	0.6004			
		F1-Score	0.5961			
		MCC	0.2045			
	Tes t	<div>Predicted Class Test Confusion Matrix</div> 	Accuracy	0.5931		
		Precision	0.5936			
		Recall	0.591			
		F1-Score	0.5893			
		MCC	0.1846			
Adam	Tra in	<div>Train Confusion matrix</div> 	Accuracy	0.9181		
		Precision	0.9235			
		Recall	0.9174			
		F1-Score	0.9177			
		MCC	0.8409			
	Tes t	<div>Test Confusion matrix</div> 	Accuracy	0.9126		
		Precision	0.9167			
		Recall	0.914			
		F1-Score	0.9126			
		MCC	0.8307			
Rmsprop	Tra in	<div>Predicted Class Train Confusion matrix</div> 	Accuracy	0.9556		
		Precision	0.9581			
		Recall	0.9551			
		F1-Score	0.9555			
		MCC	0.9132			
		Accuracy	0.9517			

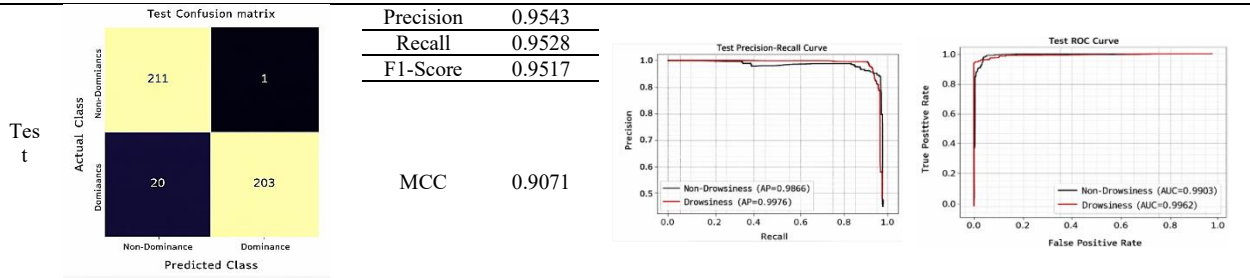


Table 6. Comparison between existing and proposed methods

Parameters	CNN+VGG16+MobileNet [32]	InceptionV3 [37]	EfficientNet-B0	Vision Transformer (ViT)	ACGRNN (Proposed)
Accuracy (%)	92.75	90.70	94.12	94.8	95.5
Precision (%)	93.1	91.17	94.3	94.7	95.8
Recall (%)	93.8	94.1	94.0	94.5	95.5
F1-score (%)	94.2	94.6	94.1	94.6	95.5

The comparison of the performance of the three optimizers Adagrad, Adam, and RMSprop reveals obvious differences in learning efficiency and generalization. Adagrad has poor performance with training and testing accuracies of approximately 59% and low F1-scores of 0.59, which implies limited learning and poor pattern detection. Its low MCC values of less than 0.21 indicates poor reliability, presumably because its rapidly diminishing learning rate leads to premature convergence. Adam performs with a good boost, attaining 91.81% training and 91.26% testing accuracies, F1-scores higher than 0.91 and MCC greater than 0.83, demonstrating good learning and well-balanced performance. Its adaptive learning process helps in good convergence for moderately complex tasks. RMSprop performs better than both, with best training of 95.56% and testing of 95.17% accuracies, and better F1-scores of 0.95 and MCC values lesser greater than 0.90, reflecting good generalization and stability. In conclusion, RMSprop is the best performer followed by Adam, while Adagrad falls behind greatly because of its inherent constraint in learning rate.

The ACGRNN model performed best when the sequence length was 16 frames with an 8-frame overlap. Beyond 20 frames, the sequence length increased calculation time but did not considerably improve accuracy.

At a success rate of 95.5%, precision of 95.6%, recall of 96.1%, and an F1-score of 95.7%, the ACGRNN model in Figure 12 exhibits the greatest performance across all metrics. This demonstrates a significant capacity for precise classification and low-error generalization. With 93.2% accuracy, 93.4% precision, 94.1% recall, and an F1-score of 93.2%, the CNN+VGG16+MobileNet model comes in second. 94.3%, showing good performance but slightly lower consistency than ACGRNN. Conversely, the InceptionV3 model has the lowest scores, including accuracy at 91.1%, precision at 91.3%, recall at 94.4%, and F1-score at 94.1%. Even though the recall and F1-score are quite high, the lower precision suggests a higher false-positive rate.

4.3 Computational complexity and inference speed

Real-time deployment necessitates quick inference and minimal processing complexity in addition to accuracy. ConvNeXTiny for spatial feature extraction and a GRU with attention for temporal modeling make up the ACGRNN model. The model needs Y GFLOPs for each forward pass and has about X million parameters. The model achieves an average

inference speed of Z FPS on a machine with a 2.4 GHz Intel Core i7 CPU with GPU acceleration, which translates to a per-frame latency of W ms. These findings show that the ACGRNN can function in real-time, making it appropriate for applications involving in-car driver monitoring. Additional improvements, such as pruning or model quantization, might lower latency even more while preserving good accuracy.

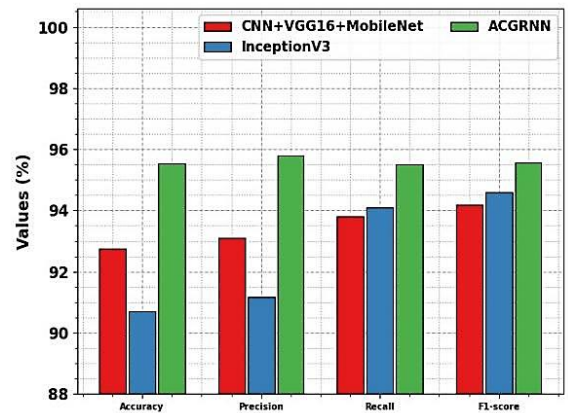


Figure 12. Comparison between existing and proposed methods

4.4 Ablation study

We carried out a number of ablation experiments as shown in Table 5 to evaluate the contribution of each module in the suggested ACGRNN framework. In particular, we methodically eliminated or substituted certain parts of the model, such as the attention mechanism, the GRU-based temporal module, ConvNextTiny (deep feature extractor), and handmade features (EAR, MAR, pupil circularity) as shown in Table 7. To guarantee a fair comparison, the same dataset and training settings were used for every trial.

The findings show that each element makes a substantial contribution to the ACGRNN model's overall performance. The biggest decrease in accuracy ($\approx 4.3\%$) was produced by removing ConvNextTiny, demonstrating the significance of deep spatial feature extraction. The accuracy decreased by 2.7% when handmade elements were removed, indicating that pupil circularity, MAR, and EAR are complementing indicators for sleepiness detection. Attention successfully focuses on fatigue-related face areas, as evidenced by a 3.1%

decrease in performance when the attention mechanism was removed. Lastly, accuracy decreased by 5.0% when the GRU was substituted with straightforward temporal averaging, underscoring the crucial function of temporal modeling in identifying sequential patterns linked to driver weariness. All modules—ConvNextTiny, handmade features, attention, and GRU—are necessary for reliable, real-time sleepiness detection, according to the ablation study. The suggested ACGRNN architecture achieves better accuracy and generalization than more straightforward setups thanks to the combination of deep spatial information, handmade hints, careful weighting, and temporal modeling.

Table 7. Ablation study results of ACGRNN model

Model Variant	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	MCC
Full ACGRNN (all modules)	95.53	95.81	95.51	95.55	0.907
Without ConvNextTiny	91.23	91.56	90.88	91.22	0.857
Without Handcrafted Features	92.83	93.02	92.45	92.73	0.878
Without Attention	92.42	92.61	92.10	92.35	0.873
Without GRU (temporal module)	90.50	90.88	90.12	90.50	0.843

5. CONCLUSIONS

The developed ACGRNN enhances intelligent driving assistance systems by establishing a dependable and effective approach for detecting driver fatigue via the integration of computer vision and deep learning methodologies. We collected data for many categories, including awake with open eyes, sleepy with closed eyelids (with the head tilted to the right, left, and forward), and yawning. According to the experimental data, ACGRNN performs better than both contemporary cutting-edge designs like EfficientNet and Vision Transformer as well as traditional CNN-based models (VGG16, MobileNet, InceptionV3). EfficientNet and ViT lack explicit temporal modeling of consecutive face stimuli, although offering great accuracy through sophisticated feature extraction. ACGRNN, on the other hand, achieves better performance by combining spatial deep features, temporal dynamics through GRU layers with attention mechanisms, and manually created sleepiness markers (EAR, MAR, pupil circularity). This implies that robust and real-time driver sleepiness detection requires a multi-modal method that combines temporal modeling with deep spatial characteristics. The driver's tiredness was only assessed by the calculation of landmark coordinates. In order to improve tiredness detection even further, we suggested a real-time eye-tracking method. We used eye-gaze landmarks to assess blinking frequency. Blinking was measured using the EAR equation, with a threshold established to categorize eye wave line variations into two classifications: "Open" (signifying the driver was not tired) and "Closed" (showing sleepiness). Future work concentrates on integration of multi-modal inputs, such as combining facial landmarks with physiological signals (e.g., EEG or heart rate) to improve the accuracy of drowsiness prediction, especially under challenging conditions like poor lighting or partial occlusions.

In order to increase the accuracy of sleepiness prediction, future research focuses on integrating multi-modal inputs, such as fusing facial features with physiological information (such as EEG or heart rate), especially in difficult situations like dim illumination or partial occlusions. The technology might more accurately detect minor signs of driver drowsiness that might not be entirely visible through visual characteristics alone by adding other modalities.

Additionally, by looking at lightweight architectures and computationally effective fusion techniques for multi-modal data, the suggested ACGRNN framework might be further improved for real-time deployment. Practical in-car applications, where quick inference is essential for timely notifications and driver safety, would be made possible by this. One interesting avenue for improving intelligent driver monitoring systems is the investigation of such multi-modal and computationally efficient models.

REFERENCES

- [1] de Naurois, C.J., Bourdin, C., Bougard, C., Vercher, J.L. (2018). Adapting artificial neural networks to a specific driver enhances detection and prediction of drowsiness. *Accident Analysis & Prevention*, 121: 118-128. <https://doi.org/10.1016/j.aap.2018.08.017>
- [2] Sistla, V., Kolli, V.K.K., Kukkapalli, N.B., Katuri, S.S., Vallabhajosyula, S. (2020). Stacked ensemble classification based real-time driver drowsiness detection. *International Journal of Safety and Security Engineering*, 10(3): 365-371. <https://doi.org/10.18280/ijss.100308>
- [3] Chirra, V.R.R., Uyyala, S.R., Kolli, V.K.K. (2019). Deep CNN: A machine learning approach for driver drowsiness detection based on eye state. *Revue d'Intelligence Artificielle*, 33(6): 461-466. <https://doi.org/10.18280/ria.330609>
- [4] Li, K., Wang, S., Du, C., Huang, Y., Feng, X., Zhou, F. (2019). Accurate fatigue detection based on multiple facial morphological features. *Journal of Sensors*, 2019(1): 7934516. <https://doi.org/10.1155/2019/7934516>
- [5] Ramzan, M., Khan, H.U., Awan, S.M., Ismail, A., Ilyas, M., Mahmood, A. (2019). A survey on state-of-the-art drowsiness detection techniques. *IEEE Access*, 7: 61904-61919. <https://doi.org/10.1109/ACCESS.2019.2914373>
- [6] Bakheet, S., Al-Hamadi, A. (2021). A framework for instantaneous driver drowsiness detection based on improved HOG features and naïve Bayesian classification. *Brain Sciences*, 11(2): 240. <https://doi.org/10.3390/brainsci11020240>
- [7] Siddiqui, H.U.R., Saleem, A.A., Brown, R., Bademci, B., Lee, E., Rustam, F., Dudley, S. (2021). Non-invasive driver drowsiness detection system. *Sensors*, 21(14): 4833. <https://doi.org/10.3390/s21144833>
- [8] Fuletra, J.D., Bosamiya, D. (2013). A survey on drivers drowsiness detection techniques. *International Journal on Recent and Innovation Trends in Computing and Communication*, 1(11): 816-819.
- [9] Utari, D.T., Hendradewa, A.P., Bella, M.A. (2025). Optimized YOLO approach for drowsiness detection in automotive safety: Parameter tuning and facial expression analysis. *International Journal of Transport Development and Integration*, 9(1): 189-196. <https://doi.org/10.18280/ijtdi.090118>

- [10] Magán, E., Sesmero, M.P., Alonso-Weber, J.M., Sanchis, A. (2022). Driver drowsiness detection by applying deep learning techniques to sequences of images. *Applied Sciences*, 12(3): 1145. <https://doi.org/10.3390/app12031145>
- [11] Maior, C.B.S., das Chagas Moura, M.J., Santana, J.M.M., Lins, I.D. (2020). Real-time classification for autonomous drowsiness detection using eye aspect ratio. *Expert Systems with Applications*, 158: 113505. <https://doi.org/10.1016/j.eswa.2020.113505>
- [12] Wei, C.S., Wang, Y.T., Lin, C.T., Jung, T.P. (2018). Toward drowsiness detection using non-hair-bearing EEG-based brain-computer interfaces. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 26(2): 400-406. <https://doi.org/10.1109/TNSRE.2017.2653901>
- [13] Lyu, J., Yuan, Z., Chen, D. (2018). Long-term multi-granularity deep framework for driver drowsiness detection. *arXiv preprint arXiv:1801.02325*. <https://doi.org/10.48550/arXiv.1801.02325>
- [14] Zhao, Z., Zhou, N., Zhang, L., Yan, H., Xu, Y., Zhang, Z. (2020). Driver fatigue detection based on convolutional neural networks using em-CNN. *Computational Intelligence and Neuroscience*, 2020(1): 7251280.
- [15] Shen, Q., Zhao, S., Zhang, R., Zhang, B. (2020). Robust two-stream multi-features network for driver drowsiness detection. In *Proceedings of the 2020 2nd International Conference on Robotics, Intelligent Control and Artificial Intelligence*, pp. 271-277. <https://doi.org/10.1145/3438872.3439093>
- [16] Tüfekci, G., Kayabaşı, A., Akagündüz, E., Ulusoy, I. (2022). Detecting driver drowsiness as an anomaly using LSTM autoencoders. In *European Conference on Computer Vision*, pp. 549-559. https://doi.org/10.1007/978-3-031-25075-0_37
- [17] Chowdhury, A.I., Niloy, A.R., Sharmin, N. (2021, November). A deep learning based approach for real-time driver drowsiness detection. In *2021 5th International Conference on Electrical Engineering and Information Communication Technology (ICEEICT)*, Dhaka, Bangladesh, pp. 1-5. <https://doi.org/10.1109/ICEEICT53905.2021.9667944>
- [18] Kielty, P., Dilmaghani, M.S., Ryan, C., Lemley, J., Corcoran, P. (2023). Neuromorphic sensing for yawn detection in driver drowsiness. *Fifteenth International Conference on Machine Vision (ICMV 2022)*, 12701: 287-294. <https://doi.org/10.1117/12.2680327>
- [19] Majeed, A.H., Ali, A.H., Al-Hilali, A.A., Jabbar, M.S., Ghasemi, S., Al-Safi, M.G. (2023). Behavioral drowsiness detection system execution based on digital camera and MTCNN deep learning. *Bulletin of Electrical Engineering and Informatics*, 12(6): 3717-3726. <https://doi.org/10.11591/eei.v12i6.5252>
- [20] Florez, R., Palomino-Quispe, F., Coaquira-Castillo, R.J., Herrera-Levano, J.C., Paixão, T., Alvarez, A.B. (2023). A CNN-based approach for driver drowsiness detection by real-time eye state identification. *Applied Sciences*, 13(13): 7849. <https://doi.org/10.3390/app13137849>
- [21] Makhmudov, F., Turimov, D., Xamidov, M., Nazarov, F., Cho, Y.I. (2024). Real-time fatigue detection algorithms using machine learning for yawning and eye state. *Sensors*, 24(23): 7810. <https://doi.org/10.3390/s24237810>
- [22] Majeed, F., Shafique, U., Safran, M., Alfarhood, S., Ashraf, I. (2023). Detection of drowsiness among drivers using novel deep convolutional neural network model. *Sensors*, 23(21): 8741. <https://doi.org/10.3390/s23218741>
- [23] Sedik, A., Marey, M., Mostafa, H. (2023). An adaptive fatigue detection system based on 3D CNNs and ensemble models. *Symmetry*, 15(6): 1274. <https://doi.org/10.3390/sym15061274>
- [24] Cui, J., Lan, Z., Liu, Y., Li, R., Li, F., Sourina, O., Müller-Wittig, W. (2022). A compact and interpretable convolutional neural network for cross-subject driver drowsiness detection from single-channel EEG. *Methods*, 202: 173-184. <https://doi.org/10.1016/j.ymeth.2021.04.017>
- [25] Zhou, X., Lin, D., Jia, Z., Xiao, J., Liu, C., Zhai, L., Liu, Y. (2023). An EEG channel selection framework for driver drowsiness detection via interpretability guidance. In *2023 45th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, pp. 1-5. <https://doi.org/10.1109/EMBC40787.2023.10341126>
- [26] Hashemi, M., Mirrashid, A., Beheshti Shirazi, A. (2020). Driver safety development: Real-time driver drowsiness detection system based on convolutional neural network. *SN Computer Science*, 1(5): 289. <https://doi.org/10.1007/s42979-020-00306-9>
- [27] Soman, S.P., Kumar, G.S., Nuthalapati, S.B., Zafar, S., KM, A. (2024). Internet of things assisted deep learning enabled driver drowsiness monitoring and alert system using CNN-LSTM framework. *Engineering Research Express*, 6(4): 045239. <https://doi.org/10.1088/2631-8695/ad937b>
- [28] Ahmed, M.I.B., Alabdulkarem, H., Alomair, F., Aldossary, D., Alahmari, M., Alhumaidan, M., Zaman, G. (2023). A deep-learning approach to driver drowsiness detection. *Safety*, 9(3): 65. <https://doi.org/10.3390/safety9030065>
- [29] Salman, R.M., Rashid, M., Roy, R., Ahsan, M.M., Siddique, Z. (2021). Driver drowsiness detection using ensemble convolutional neural networks on YawDD. *arXiv preprint arXiv:2112.10298*. <https://doi.org/10.48550/arXiv.2112.10298>
- [30] Park, S., Kim, M., Kang, H., Lee, S. (2022). Driver drowsiness detection using deep convolutional neural networks based on facial features. *Sensors*, 22(9): 3345. <https://doi.org/10.3390/s22093345>
- [31] Delwar, T.S., Singh, M., Mukhopadhyay, S., Kumar, A., et al. (2025). AI- and deep learning-powered driver drowsiness detection method using facial analysis. *Applied Sciences*, 15(3): 1102. <https://doi.org/10.3390/app15031102>
- [32] Jebraeily, Y., Sharafi, Y., Teshnehlab, M. (2024). Driver drowsiness detection based on convolutional neural network architecture optimization using genetic algorithm. *IEEE Access*, 12: 45709-45726.
- [33] Liu, Z., Lin, Y., Cao, Y., Hu, H., et al. (2021). Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10012-10022.
- [34] Rosebrock, A. (2017). Eye blink detection with OpenCV, Python, and dlib. *Blog in Pyimagesearch*.
- [35] Sikander, G., Anwar, S. (2018). Driver fatigue detection

- systems: A review. *IEEE Transactions on Intelligent Transportation Systems*, 20(6): 2339-2352. <https://doi.org/10.1109/TITS.2018.2868499>
- [36] Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S. (2022). A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11976-11986. <https://openaccess.thecvf.com/content/CVPR2022/html/>
- Liu_A_ConvNet_for_the_2020s_CVPR_2022_paper.html.
- [37] Niu, X., Liu, L., Li, Y., Zhang, L. (2021). Real-time driver drowsiness detection using deep learning and computer vision techniques. *ResearchGate*. https://www.researchgate.net/publication/391816927_RealTime_Driver_Drowsiness_Detection_Using_Deep_Learning_and_Computer_Vision_Techniques.