



Vision Transformer-Based Driver Fatigue Detection in Real-Time Embedded Systems

Zahra Elmi¹ · Firdevs Eyidoğan² · Zeynep Sena Kaya² · Ali Buğuş² · Soheila Elmi³

Received: 10 November 2025 / Revised: 1 May 2026 / Accepted: 6 May 2026
© The Author(s), under exclusive licence to ITS Japan 2026

Abstract

Driver fatigue is a major contributor to traffic accidents, yet reliable early warning remains challenging under real-world and embedded constraints. This study presents a camera-only driver-fatigue detection pipeline that couples lightweight face tracking (MediaPipe) with a single Vision Transformer (ViT) classifier ($\approx 86\text{M}$ parameters) to infer alert vs. drowsy states from facial cues. The model is trained on a multi-source dataset of $\sim 4,000$ images aggregated from seven public fatigue datasets and evaluated on a held-out test split, achieving 95.0% accuracy, 93.8% F1-score, and ROC-AUC of 0.986. On Raspberry Pi 5, end-to-end inference runs at 55.6 ms/frame (18.0 FPS). Compared with a detection-driven YOLO baseline using the same labeling protocol and test split, the proposed ViT improves accuracy (95.0% vs. 93.2%) and ROC-AUC (0.986 vs. 0.979) while trading throughput (18.0 vs. 28.0 FPS), clarifying the accuracy–latency balance for deployment. Real-world validation with 12 participants (22–47 years) across day and night sessions (~ 9 hours total) shows high agreement with manual annotations (92.8%), supporting operational feasibility. By combining a single-backbone ViT inference pipeline with explicit embedded benchmarking, matched baseline comparison, and in-vehicle validation, the proposed system provides a deployment-oriented evaluation package for camera-only fatigue monitoring under embedded ITS constraints.

Keywords Driver fatigue · Image processing · Vision transformer · Real-time systems · Traffic safety · Deep learning

1 Introduction

Driver fatigue is a major traffic-safety risk: sleep deprivation and mental fatigue reduce attention, slow perception, and prolong reaction time, thereby impairing hazard recognition and decision-making. These effects become particularly critical in monotonous settings such as long-distance and nighttime driving. NHTSA's *Traffic Safety Facts* research note reports 633 fatalities in U.S. crashes involving drowsy drivers in 2023 [1], underscoring the need for reliable early-warning monitoring under real-world conditions.

Among available sensing modalities, camera-based fatigue monitoring is especially attractive for in-cabin deployment because it is non-contact while directly capturing facial indicators such as eyelid closure, blink dynamics, mouth opening, and head pose variation [2–4]. However, deployment-ready visual systems remain difficult to realize in practice. Many high-performing studies rely on temporal or multi-stage hybrids such as CNN–LSTM [5] and 3DCNN–LSTM [6], which increase integration overhead and complicate embedded real-time execution. In addition,

✉ Zahra Elmi
zehraelmi@beykoz.edu.tr
Firdevs Eyidoğan
030721057@std.izu.edu.tr
Zeynep Sena Kaya
kaya.s@std.izu.edu.tr
Ali Buğuş
bugus.ali@std.izu.edu.tr
Soheila Elmi
selmi24@ku.edu.tr

¹ Department of Software Engineering, Beykoz University, Muhtar Sokak No: 3, Istanbul 34010, Kavacak, Turkey

² Department of Computer Engineering, Istanbul Sabahattin Zaim University, Halkalı Cad., Istanbul 34303, Küçükçekmece, Turkey

³ Department of Electrical and Electronics Engineering, Koç University, Rumelifeneri Yolu, Istanbul 34450, Turkey

maintaining an effective accuracy–latency balance under in-vehicle compute, memory, and power constraints remains challenging [7], and models validated only under laboratory conditions may not transfer cleanly to real driving. More concretely, many prior fatigue-detection studies do not jointly report measurable deployment indicators such as model size, end-to-end on-device latency/FPS on a stated embedded platform, and controlled baseline comparisons under identical splits, making embedded feasibility difficult to assess objectively.

Vision Transformers (ViT) provide a promising alternative because self-attention enables a single backbone to capture both local facial cues and longer-range dependencies without relying on multi-stage temporal fusion [8]. Motivated by this, we develop a ViT-based fatigue-detection pipeline operating on monocular facial images, designed for real-time embedded deployment, with a single-backbone classifier and lightweight face tracking. On a $\sim 4,000$ -image multi-source dataset, the proposed model achieves 95.0% accuracy, 93.8% F1-score, and 0.986 ROC-AUC, while running end-to-end at 55.6 ms/frame (18.0 FPS) on Raspberry Pi 5. Under the same split and binary labeling protocol, the proposed ViT pipeline also outperforms a detection-driven YOLO baseline in accuracy (95.0% vs. 93.2%) and ROC-AUC (0.986 vs. 0.979), although at lower throughput (18.0 vs. 28.0 FPS). Accordingly, the study emphasizes explicit embedded benchmarking through model size reporting (≈ 86 M parameters), end-to-end Raspberry Pi 5 runtime, and direct comparison with a YOLO-based baseline under a matched protocol.

The main contributions of this work are threefold: (1) a camera-only, single-backbone ViT fatigue-detection pipeline with lightweight MediaPipe-based face tracking for real-time embedded deployment; (2) a harmonized multi-source dataset of approximately 4,000 images from seven public datasets under a unified binary label space (alert vs. drowsy); and (3) explicit embedded benchmarking via Raspberry Pi 5 deployment, a matched YOLO-based baseline comparison, and real-world in-vehicle validation with manual-annotation agreement analysis. Importantly, the contribution should be understood as a deployment-oriented system and evaluation contribution rather than as the proposal of a new transformer architecture or learning algorithm. By combining embedded ViT-based inference, justified binary label-space harmonization, matched baseline evaluation, and in-vehicle validation within a single reproducible protocol, the study supports objective assessment of accuracy, latency, and operational feasibility for embedded intelligent transportation systems.

2 Literature Review

Existing driver-fatigue detection studies can be broadly grouped into three categories: physiological signal-based, vehicle behavior-based, and vision-based approaches.

Because prior studies use different datasets, label definitions, evaluation metrics, and hardware settings, the following review is intended as a qualitative positioning of sensing modalities, model families, and deployment characteristics rather than a direct numerical ranking across papers.

Physiological methods, particularly EEG-based systems, have shown strong correlations between fatigue and spectral changes. Lal and Craig [9] linked increased theta/delta activity with fatigue transitions, and strong performance has been reported within study-specific EEG protocols, for example, 94.3% in a multistage EEG analysis setting [10]. Continuous monitoring has also been explored in wearable settings [11]. Despite their effectiveness, electrode-based sensing and specialized hardware reduce user comfort and limit practical automotive deployment, motivating non-invasive alternatives.

Vehicle behavior-based methods infer fatigue indirectly from driving dynamics such as lane deviation, steering variability, and pedal usage. Doudou et al. [12] reviewed vehicle-, video-, and physiology-based fatigue detection approaches and highlighted their trade-offs. Vehicle-based systems can operate in real time, for example, through wavelet features and ANN-based lane-deviation analysis [13], but they may remain sensitive to road geometry, traffic conditions, and other external factors. Vision-based approaches, by contrast, directly analyze facial cues such as blink behavior, eyelid closure, yawning, and head movement, making them especially suitable for in-cabin monitoring.

2.1 CNN-Based Visual Fatigue Detection

CNN-based pipelines and CNN–RNN extensions have dominated vision-based fatigue detection. Guo and Markoni [14], for example, reported 84.85% accuracy on NTHU using a CNN–LSTM model. More broadly, many prior visual systems rely on CNN backbones, CNN–RNN combinations, or multi-sensor fusion [15–17]. These approaches have produced competitive results, but they often depend on temporal fusion, auxiliary sensing, or multi-component processing pipelines.

2.2 Hybrid Architectures (CNN–LSTM / Multi-Stage / CNN–Transformer)

Hybrid designs typically combine multiple signals, processing stages, or architectural blocks to improve performance. In physiological settings, for example, Sahayadhas et al. [18] reported up to 96.7% accuracy within a frequency-based EEG/ECG/EMG setting, while Khushaba et al. [19] reported 95–97% using wavelet-derived EEG features and fuzzy-logic classification under their own evaluation protocol. In vision settings, hybrid CNN–Transformer or

CNN–temporal pipelines combine convolutional locality with attention-based global context. Although effective, such designs generally increase sensing, preprocessing, and integration complexity. Importantly, many studies still do not report end-to-end deployment indicators such as parameters, FLOPs, latency, or FPS on the target device, which complicates fair embedded benchmarking. Since this work targets stable real-time execution on Raspberry Pi 5, we intentionally adopt a single-backbone ViT pipeline with a simpler inference path; hybrid variants remain relevant future directions when paired with embedded optimization or hardware acceleration.

2.3 Transformer-Based Approaches (ViT/attention)

Recent studies have explored transformer backbones for driver drowsiness detection to leverage global facial context through self-attention. Hassan et al. [6] reported a real-time transformer-based approach, Jarndal et al. [20] proposed a real-time ViT-based system with deployment-oriented considerations, and Azmi and Zaman [17] further examined ViT-based driver drowsiness detection. These studies suggest that transformer architectures can be competitive for visual fatigue analysis.

Taken together, the literature suggests five derived requirements for an embedded fatigue-detection system: (R1) non-contact, camera-only sensing for practical in-cabin use; (R2) a low-complexity inference path that avoids physiological instrumentation and multi-stage or temporal-fusion overhead; (R3) explicit deployment reporting through measurable indicators such as model size and device-level latency/FPS; (R4) evaluation under a unified label space with a matched baseline protocol; and (R5) evidence beyond laboratory-only settings. These requirements directly inform our design: R1 is addressed through a monocular image pipeline, R2 through a single-backbone ViT with lightweight MediaPipe face tracking, R3 through reporting both model size ($\approx 86\text{M}$ parameters) and Raspberry Pi 5 runtime, R4 through harmonizing seven public datasets into a binary alert/drowsy label space and comparing against a YOLO baseline under the same split and evaluation protocol, and R5 through real-world in-vehicle testing with manual-annotation agreement analysis. Overall, unlike hybrid or transfer-heavy fatigue-detection pipelines, the proposed method prioritizes architectural simplicity, deployment-oriented embedded feasibility, and a reduced inference path.

3 Materials and Methods

This section describes the datasets, preprocessing and labeling strategy, ViT-based model design, training protocol, and real-time embedded integration used for driver-fatigue detection.

3.1 Datasets

We analyzed seven open-access driver-fatigue datasets: YawDD, NTHU Drowsy Driver, UTA-RLDD, Kaggle Drowsiness Detection, DROZY, DFAT, and EyeClose-Videos [21–27]. These sources include both image data and video data converted through frame extraction and were inspected for data size, resolution, label quality, and facial visibility. Because the proposed model operates on single-frame inputs, we selected approximately 4,000 suitable samples from the seven datasets and mitigated class imbalance through balanced sampling to obtain an approximately balanced two-class set (drowsy vs. alert) across sources. Samples were obtained either directly from labeled images or from videos at fixed intervals. A compact dataset-level overview is provided in Supplementary Table 6.

The seven public datasets cover heterogeneous viewpoints, illumination conditions, and subject appearance; however, most do not consistently report demographic metadata (e.g., age, ethnicity) or detailed behavioral context, which prevents fine-grained demographic stratification using public annotations alone. To partially address this limitation, we report the characteristics of our real-world cohort (12 participants, 22–47 years; day/night driving) and explicitly discuss low-light operation as a key limitation for camera-only systems. Accordingly, the aggregated dataset includes both daylight/indoor and lower-light scenes, although low-light labels are not uniformly available across sources.

Low-resolution facial images collected from university student volunteers were used only for embedded real-time testing; no personal data or biometric records were collected, and ethical committee approval was not required.

3.2 Data Preprocessing and Labeling

To ensure consistency across sources, all images were converted to RGB, resized to the ViT input resolution (224×224), and normalized using the standard `ViTImageProcessor` associated with the selected `google/vit-base-patch16-224` model. This step applies the model-specific pixel-value scaling and channel-wise normalization expected by the pre-trained ViT backbone. Separate brightness/contrast jitter was used only as data augmentation during training and was not treated as the normalization procedure itself.

We initially defined fine-grained labels such as “eyes closed/open,” “head down/up,” and “mouth open/closed.” However, this granularity did not yield stable gains because: (i) several fine-grained states were severely underrepresented, and (ii) single-frame ambiguity between adjacent states (e.g., brief blinks vs. sustained closure, transient head

movements vs. sustained drowsiness) increased label noise and inter-class overlap. In addition, fine-grained definitions were not fully consistent across the merged public datasets, reducing inter-source coherence. Therefore, we adopted a binary labeling scheme (drowsy vs. alert) aligned with the practical early-warning objective and more stable training under heterogeneous sources. The effect of this label-granularity decision is quantified in Section 4 (Experiment 3) by comparing behavior before and after label refinement.

3.3 Model Architecture—Vision Transformer (ViT)

The core model is a Vision Transformer (ViT) classifier (Fig. 1). In our implementation, we used the Hugging Face `google/vit-base-patch16-224` configuration (224×224 RGB) with 12 encoder layers. Input images are divided into fixed-size patches, embedded with positional information, and processed by the Transformer encoder; the final [CLS] representation is mapped to a binary prediction (alert vs. drowsy) by an MLP head. The model was optimized with AdamW, trained with binary cross-entropy, and exported to ONNX for embedded inference.

We initialized the network with ImageNet pre-trained weights and fine-tuned it on the prepared fatigue dataset to improve generalization under limited task-specific data. Although smaller ViT variants may offer lower memory use and latency, ViT-Base was selected to maintain sufficient representation capacity for the heterogeneous multi-source corpus; combined with MediaPipe-based face cropping and ONNX deployment, this choice remained compatible with real-time Raspberry Pi 5 operation. The model configuration corresponds to ViT-Base (ViT-B/16) and is summarized in Table 1: 196 patches (14×14), embedding dimension 768, 12 encoder layers with 12 attention heads, MLP dimension 3072, and ~ 86 M parameters.

Table 1 ViT hyperparameter configuration

Feature	Value
Patch size	16×16
Input size	224×224 RGB
# patches	$14 \times 14 = 196$
Embed dim.	768
Layers	12
Heads	12
Hidden size	768
MLP dim.	3072
Params	~ 86 M
Pre-training	IN-21k $\rightarrow$ IN-1k

3.4 Feature Extraction and MediaPipe Integration

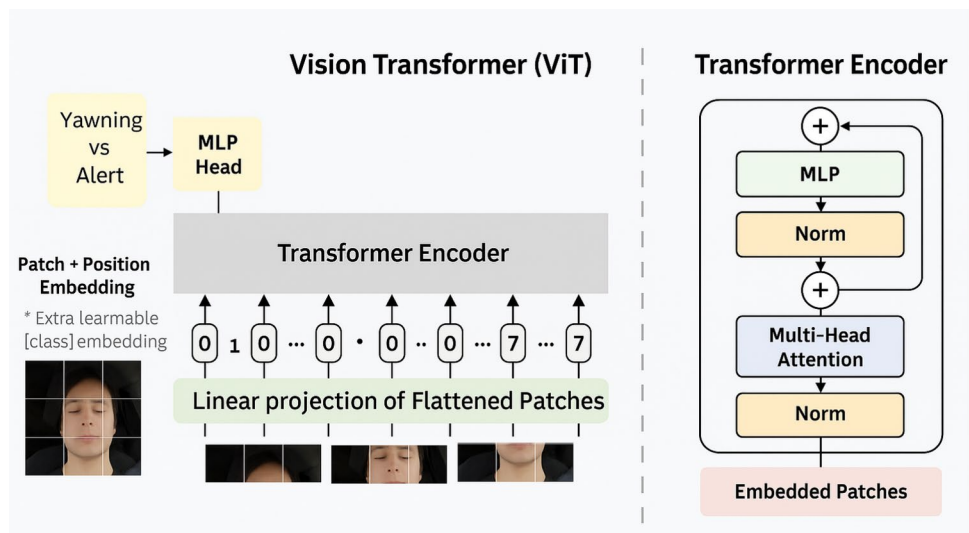
For efficient real-time operation, we use a two-stage pipeline. First, MediaPipe Face Detection (BlazeFace; `model_selection=0`) localizes the facial ROI, reducing background clutter and unnecessary computation. The detected face is then cropped using the relative bounding box, converted to RGB, resized to 224×224 , and normalized with the same `ViTImageProcessor`-based preprocessing used during training. The processed crop is passed to the exported ONNX model for inference via ONNX Runtime on Raspberry Pi 5.

This design focuses inference on the facial ROI rather than the full frame, improving efficiency under embedded constraints. MediaPipe landmarks were retained only for optional diagnostics or simple rule-based checks, while the final alert/drowsy decision was produced by the ViT classifier.

3.5 Model Training

The compiled dataset was partitioned into training, validation, and test subsets. The Vision Transformer model

Fig. 1 Architectural structure of the Vision Transformer model



was trained using AdamW with a batch size of 32, an initial learning rate of 1×10^{-4} , dropout of 0.1, and weight decay of 1×10^{-4} under a fixed random seed of 42. To improve optimization stability, the learning rate was scheduled using ReduceLROnPlateau with a reduction factor of 0.1, patience of 3 epochs, and a minimum learning rate of 1×10^{-6} . Training was run for up to 20 epochs with early stopping based on validation loss (patience = 5), and the final reported test result for each experiment was obtained from the checkpoint selected by the lowest validation loss.

Final model training was implemented in TensorFlow with NVIDIA GPU support, whereas PyTorch was used during the prototyping stage. Detailed optimizer, scheduler, and reproducibility settings are reported in Supplementary Table 8. A compact summary of the hardware platform, device operating system, and software-library versions used for both training and embedded inference is provided in Supplementary Table 9.

3.6 Real-Time System and Hardware Integration

We integrated the proposed model into an in-vehicle prototype built on Raspberry Pi 5, using a dashboard-mounted wide-angle camera to preserve facial visibility across typical head poses. For portability, the system was powered by a 5 V/3 A power bank. A Python-based application captures video frames via OpenCV, performs face detection using MediaPipe, and executes the exported ONNX model through ONNX Runtime on Raspberry Pi 5. The application is launched automatically at boot via a startup script.

During testing, system status was monitored remotely via SSH, and the Raspberry Pi was connected to the vehicle audio system over Bluetooth to deliver warning sounds through the speakers. When fatigue is detected, the system triggers an audible alarm and maintains the warning while drowsy predictions persist, using simple temporal thresholding to reduce nuisance alarms. Figure 2 shows the external casing and internal hardware layout of the prototype.

All components were optimized for real-time operation under Raspberry Pi resource constraints through resolution control and runtime-level optimization where applicable. The resulting prototype functions as a standalone embedded device using only camera input and does not require an external server. For clarity, TensorFlow/PyTorch was used during model development, whereas the embedded deployment pipeline on Raspberry Pi 5 used OpenCV, MediaPipe, Python, and ONNX Runtime.

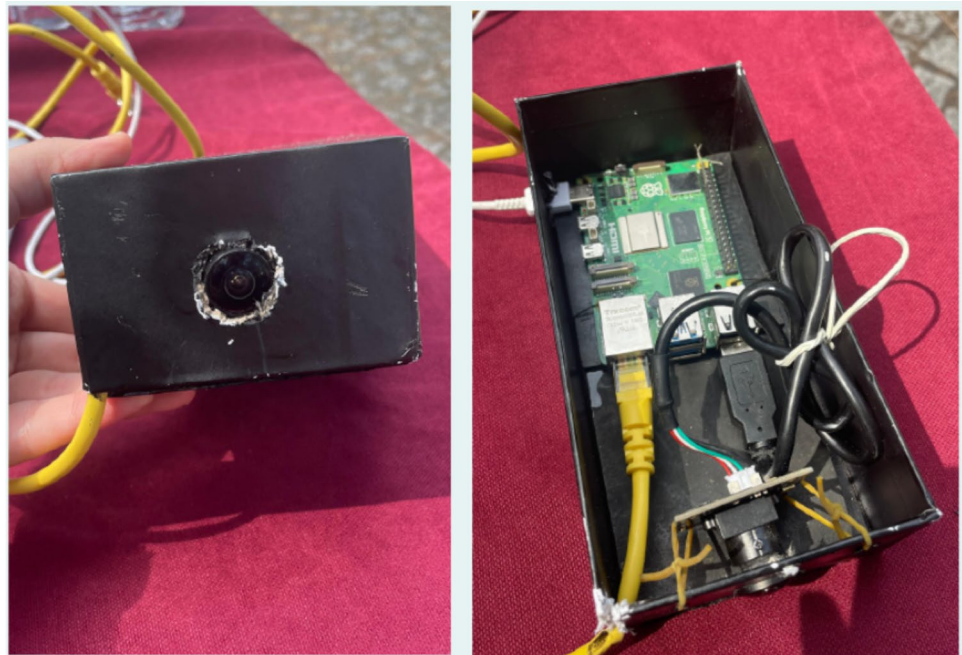
4 Results and Discussion

This section reports (i) model experiments conducted during development, (ii) prototype integration and real-world tests, and (iii) an overall discussion of strengths and limitations.

4.1 Model Experiments and Ablation Validation

We conducted three controlled experiments as an ablation-style validation to identify a model that generalizes reliably while remaining feasible for embedded deployment.

Fig. 2 External casing and internal hardware layout of the embedded in-vehicle prototype



Specifically, the study isolates the effect of: (i) adding an auxiliary graph-attention branch, (ii) removing this auxiliary branch while retaining fine-grained labels, and (iii) preserving the same ViT backbone while refining the label space to a balanced binary alert/drowsy scheme.

In Experiment 1, we evaluated a hybrid ViT+Graph Attention Network (GAT) configuration, in which Mediapipe facial landmarks were represented as a graph to capture structural relationships among facial regions. Although the model converged rapidly on the training set, reaching training accuracy above 95%, validation accuracy fluctuated and remained consistently below training performance, indicating overfitting and unstable generalization (Fig. 3). The auxiliary branch also increased integration complexity and runtime overhead, which is undesirable for real-time embedded deployment.

In Experiment 2, we simplified the pipeline to a pure ViT classifier without any auxiliary network. This reduced architectural complexity and improved overall consistency compared with Experiment 1; however, a non-negligible train-validation gap and occasional validation instability were still observed (Fig. 4). These results suggested that architecture simplification alone was insufficient and that the learning target itself required refinement.

In Experiment 3, we retained the same ViT architecture but revised the labeling strategy from fine-grained states to a balanced binary scheme (alert vs. drowsy). This refinement better matches the practical early-warning objective and improves label consistency across data sources. After label refinement, the training and validation curves became substantially more stable (Fig. 5), and the model achieved the strongest and most consistent performance among the three experiments. Table 2 reports validation accuracy and

test accuracy at the checkpoint selected by the lowest validation loss for each experiment.

Taken together, these ablation results show that removing the auxiliary GAT branch and adopting a balanced binary label space improve both training stability and test performance, supporting the selection of the final single-backbone ViT design for embedded deployment.

To provide an explicit embedded benchmark, we report the model size of the final ViT configuration ($\approx 86\text{M}$ parameters), its end-to-end Raspberry Pi 5 latency/FPS, and direct comparisons with a YOLO-based baseline and a lightweight MobileNetV2 baseline under the same evaluation settings. The overfitting observed in the ViT+GAT setting is attributed mainly to the limited information content of the facial-landmark graph relative to appearance cues and to the added optimization difficulty of the auxiliary branch. For these reasons, we prioritized the single-backbone ViT pipeline as the most stable and practically deployable solution. To further assess embedded accuracy-efficiency trade-offs, we additionally evaluated post-training INT8 quantization on the same held-out test split and Raspberry Pi 5 setup (Supplementary Table 7) and repeated the experiments over five random seeds, reporting accuracy, F1-score, and ROC-AUC as mean $\pm$ standard deviation (Supplementary Table 10).

4.2 Prototype Integration and Real-Time Tests

We integrated the refined ViT model into the embedded prototype and evaluated it in a real vehicle across daytime/nighttime sessions, multiple drivers, and diverse road types (urban/highway). The system ran continuously on Raspberry Pi 5 and monitored the driver's face via the on-board camera. Figure 6 shows the in-vehicle setup and a snapshot of system operation.

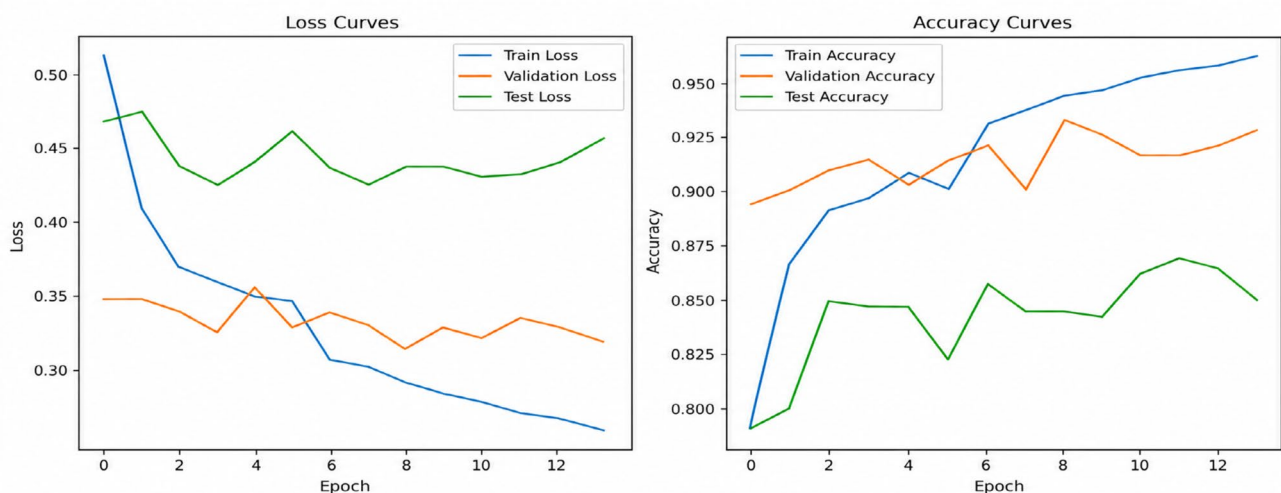


Fig. 3 Training and validation loss/accuracy curves for the ViT+GAT model. Training stopped early at epoch 14, while the checkpoint selected for evaluation corresponded to the lowest validation loss at epoch 9

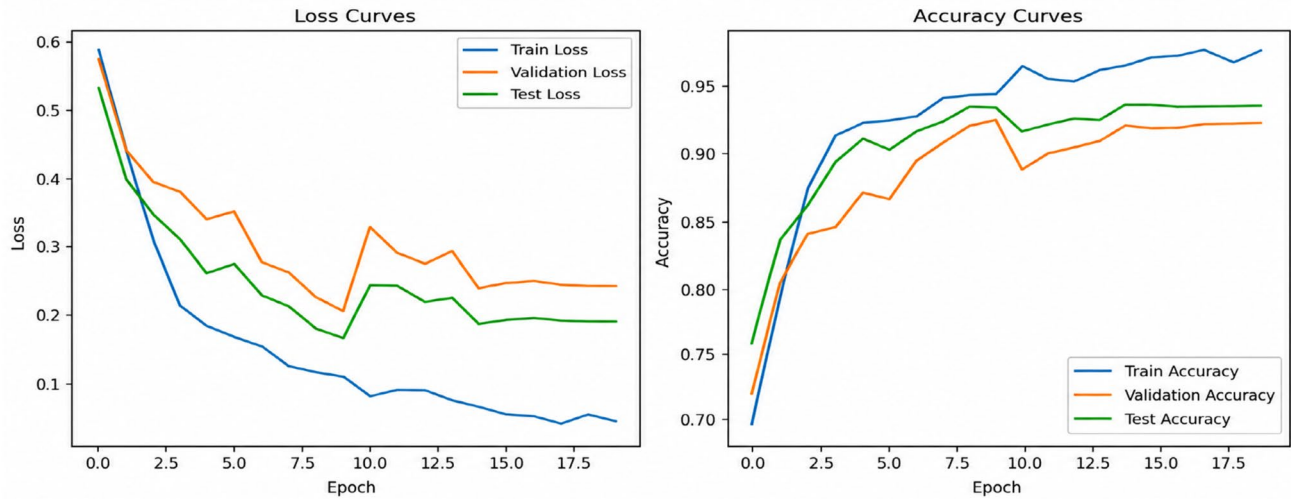


Fig. 4 Training and validation loss/accuracy curves for the pure ViT model with fine-grained labels. Removing the auxiliary GAT branch improves stability, although a residual train–validation gap remains

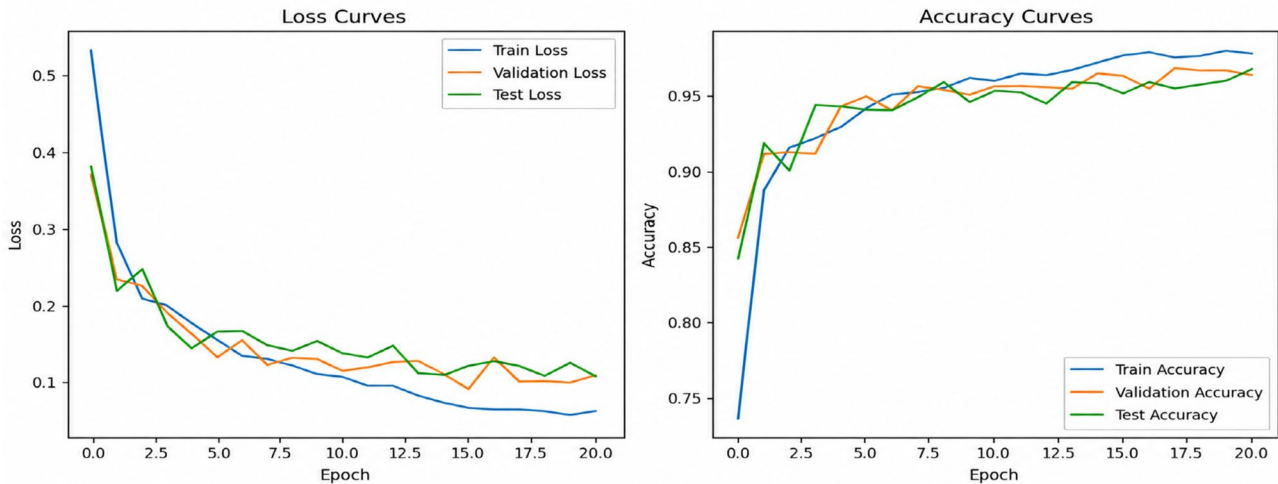


Fig. 5 Training and validation loss/accuracy curves for the final pure ViT model with balanced binary labels. Label refinement yields the most stable convergence among the three evaluated settings

Because missed-fatigue cases are safety-critical, we report accuracy, precision, recall, F1-score, and ROC-AUC. Table 3 compares the proposed ViT pipeline with a lightweight MobileNetV2 classifier and a YOLO-based

Table 2 Ablation-style validation experiments used to select the final embedded-feasible model design. Each experiment was trained for up to 20 epochs, with early stopping based on validation loss. Validation and test accuracy are reported at a single checkpoint selected by the lowest validation loss

Exp.	Setup	Selected epoch	Val. (%)	Test (%)
Exp1	ViT+GAT hybrid	Ep9	93.33	91.20
Exp2	Pure ViT, fine-grained labels	Ep10	93.17	92.85
Exp3	Pure ViT, balanced binary labels	Ep18	96.75	95.00

detection-driven baseline under the same face-crop input, binary label space, and evaluation split. MobileNetV2 provides the highest throughput (31.5 FPS) and YOLO also remains faster than ViT (28.0 FPS), whereas the proposed ViT achieves the strongest classification performance overall, including the highest ROC-AUC (0.986) and drowsy-class sensitivity of 94.6%.

Figure 7 presents a representative confusion matrix of the ViT classifier on the held-out test split. Because Table 3 reports classification performance as mean $\pm$ standard deviation over repeated runs, the confusion matrix is presented for qualitative error analysis rather than as an exact numerical reconstruction of the averaged metrics. We further analyze ROC and PR curves to assess threshold-dependent

Fig. 6 The prototype device facing the driver in the vehicle and a snapshot of the system's operation while the vehicle is in motion

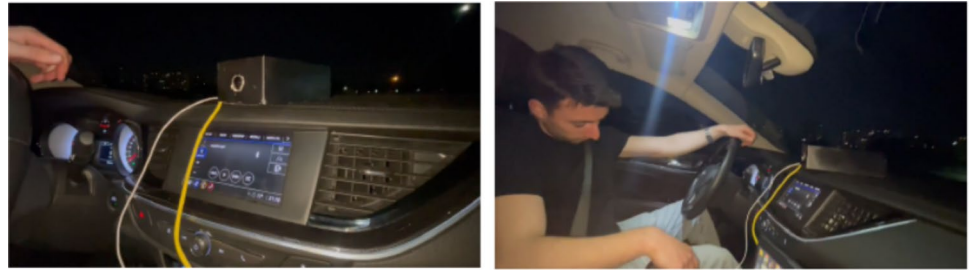


Table 3 Head-to-head comparison of the proposed ViT, MobileNetV2, and YOLO baselines on Raspberry Pi 5. Classification metrics, latency, and FPS are reported as mean $\pm$ standard deviation across repeated runs, whereas Params (M) is reported as a fixed model property

Metric	ViT	MNetV2	YOLO
Accuracy (%)	95.0 $\pm$ 0.2	92.2 $\pm$ 0.3	93.2 $\pm$ 0.3
Precision (%)	93.2 $\pm$ 0.3	90.5 $\pm$ 0.3	91.6 $\pm$ 0.3
Sensitivity/Recall (%)	94.6 $\pm$ 0.3	91.8 $\pm$ 0.3	92.2 $\pm$ 0.3
F1-score (%)	93.8 $\pm$ 0.3	91.0 $\pm$ 0.3	91.9 $\pm$ 0.3
ROC-AUC	0.986 $\pm$ 0.002	0.951 $\pm$ 0.003	0.979 $\pm$ 0.002
Params (M)	86.0	3.5	7.2
FPS	18.0 $\pm$ 0.2	31.5 $\pm$ 0.3	28.0 $\pm$ 0.3
Latency (ms/frame)	55.6 $\pm$ 0.7	31.7 $\pm$ 0.4	35.7 $\pm$ 0.5

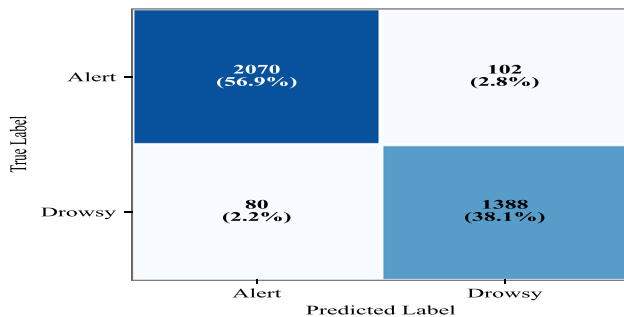
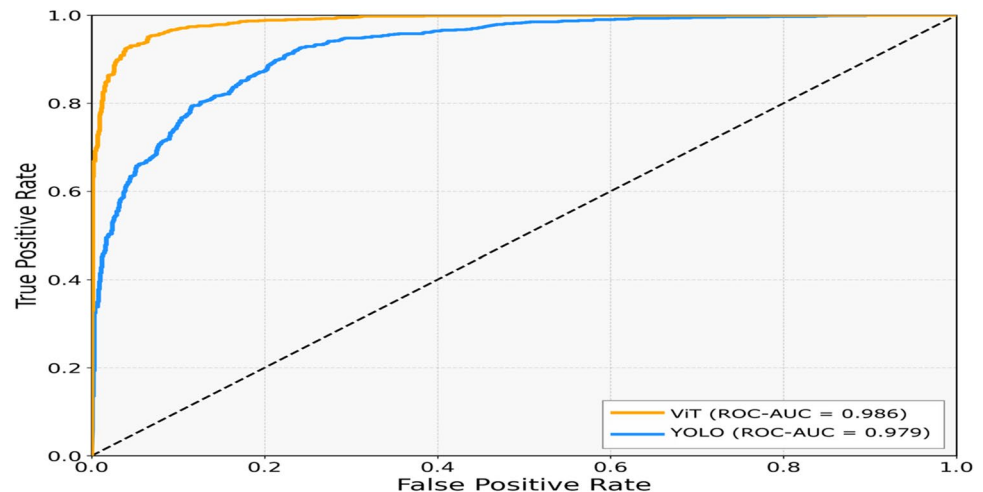


Fig. 7 Confusion matrix of the proposed ViT classifier on the held-out test split (drowsy vs. alert)

discrimination. Figure 8 reports ROC-AUC values of 0.986 (ViT) and 0.979 (YOLO), and Fig. 9 presents the corresponding precision–recall curves.

Fig. 8 ROC curves on the held-out test split comparing the proposed ViT pipeline and the YOLO-based baseline (ROC-AUC: 0.986 vs. 0.979)



For real-world evaluation, 12 participants (22–47 years) drove in both daytime and nighttime sessions, totaling approximately 9 hours of video. In prototype operation, the alert logic was triggered when eye closure persisted for more than 2 s. Yawning-related cues were observed during manual review, but they were not used as a separate hard threshold in the prototype alert logic. System outputs were verified by manual annotation, and the protocol summary is given in Table 4. To quantify illumination effects, Table 5 further reports manual-annotation agreement stratified by daytime and nighttime, together with 95% confidence intervals. As expected, nighttime agreement is lower due to reduced cue visibility, while the confidence intervals make the uncertainty explicit and the overall agreement remains 92.8%.

Fig. 9 Precision–recall curves on the held-out test split for the proposed ViT pipeline and the YOLO-based baseline

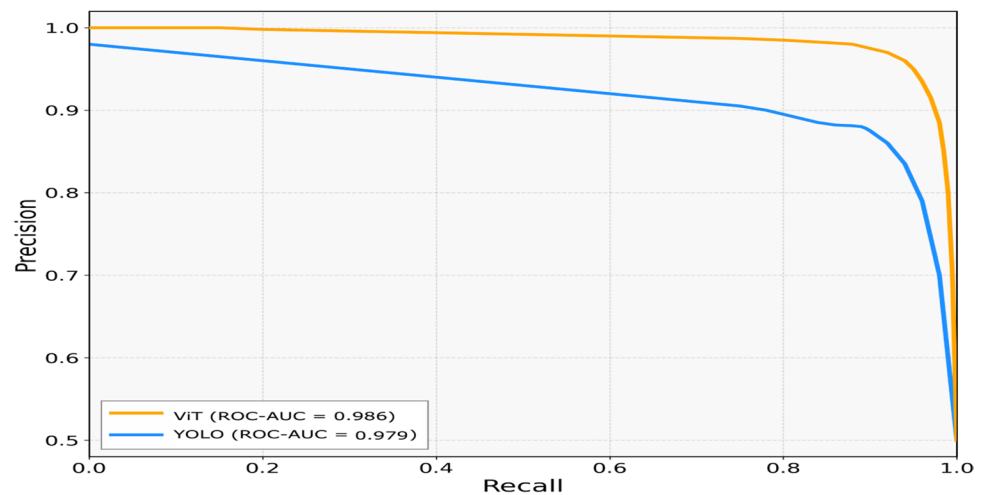


Table 4 Test scenario details

Test Parameter	Value
Number of participants	12
Age range	22–47
Total recording duration	~9 h
Day/night sessions	Both
Operational alert criteria	Eye closure > 2 s
Manual-annotation agreement	92.8%

Table 5 Real-world evaluation protocol and manual-annotation agreement, stratified by illumination condition, with 95% confidence intervals

Condition	Agreement (%)	95% CI	N
Daytime	93.5	[90.6–95.5]	390
Nighttime	91.0	[85.3–94.6]	150
Overall	92.8	[90.3–94.7]	540

Overall, the proposed ViT pipeline remains embedded-feasible (Table 3) and performs robustly under real driving, with low missed-fatigue and high manual agreement. Most false alarms were associated with benign behaviors such as frequent blinking or brief mirror-check head movements; mitigation strategies are discussed in Section 4.4. Low-light/night conditions remain a key limitation for camera-only operation, and practical mitigation directions include IR/NIR imaging and illumination-aware training. While we did not run a formal user study, frequent false alarms may still cause alarm fatigue; production systems should therefore incorporate conservative thresholds, temporal confirmation, alert-rate limiting (cooldowns), and optional personalization.

4.3 System Strengths

The prototype demonstrates practical feasibility for real-time, in-vehicle fatigue monitoring. The end-to-end pipeline runs on Raspberry Pi 5 with timely alert delivery through the vehicle audio system. Robust facial ROI acquisition is maintained under natural head-pose variation using the

wide-angle camera and MediaPipe tracking, while automated startup and remote monitoring support integration-oriented testing across drivers and scenarios.

For deployment, privacy-preserving operation is essential. The proposed system supports on-device inference without routine raw-video storage or cloud transmission. Where diagnostic recordings are required, anonymization, access control, and explicit consent should be applied. Because commercial driver-monitoring systems typically rely on proprietary datasets and undisclosed protocols, direct product-level benchmarking is not feasible; instead, we provide reproducible evidence through embedded runtime, matched baseline comparison, and real-world validation.

4.4 System Limitations

The present results also make the prototype’s main limitations more concrete. First, the real-world evaluation remains limited in scale, comprising 12 participants and approximately 9 hours of driving data. Second, illumination sensitivity is measurable: manual-annotation agreement decreases from 93.5% in daytime driving to 91.0% at night, although the overall agreement remains 92.8% (Table 5). Third, nuisance alerts are still possible. Across five evaluation runs on the held-out test split, the proposed ViT achieved $93.2 \pm 0.3\%$ precision and $94.6 \pm 0.3\%$ sensitivity/recall for the drowsy class (Table 3), indicating relatively few missed-fatigue cases overall, although false alarms can still arise during benign behaviors such as frequent blinking or brief mirror-check head movements. Finally, although the embedded runtime of 55.6 ms/frame (18.0 FPS) on Raspberry Pi 5 is real-time, it leaves less throughput margin than lighter baselines such as MobileNetV2 (31.5 FPS) and YOLO (28.0 FPS). Taken together, these results indicate that low-light robustness, false-alarm mitigation, broader real-world validation, and further deployment optimization remain important next-step priorities before production-level integration.

5 Conclusion

This work presented a camera-only, real-time driver-fatigue detection pipeline for embedded deployment using a single Vision Transformer (ViT) classifier with lightweight face tracking. On a multi-source dataset of $\sim 4,000$ images from seven public datasets, the model achieved 95.0% accuracy, 93.8% F1-score, and 0.986 ROC-AUC. On Raspberry Pi 5, the end-to-end pipeline operated at 55.6 ms/frame (18.0 FPS). Under the same split and binary labeling protocol, the proposed method outperformed a YOLO-based baseline in both accuracy (95.0% vs. 93.2%) and ROC-AUC (0.986 vs. 0.979), while operating at lower throughput (18.0 vs. 28.0 FPS). Real-world evaluation with 12 participants across day/night driving scenarios (~ 9 hours) yielded 92.8% agreement with manual annotations.

These findings position the work as a deployment-oriented system and evaluation contribution rather than as a new transformer architecture or learning algorithm. Its main value lies in jointly integrating embedded ViT-based inference, justified binary label-space harmonization, matched baseline benchmarking, and in-vehicle validation within a reproducible evaluation setting for embedded intelligent transportation systems.

The results suggest that a simpler inference path and a harmonized binary label space can improve generalization while remaining feasible for embedded deployment. The prototype also supports privacy-preserving in-cabin monitoring through on-device inference without routine cloud processing or raw-video transmission. The main limitations are reduced robustness under low-light conditions, occasional false positives during benign driver behaviors, and the limited scale of real-world validation. Future work will focus on illumination-aware training, IR/NIR support, lightweight temporal stabilization, and broader real-world testing before production-level use.

Appendix A Supplementary Material

We performed post-training INT8 quantization on the final ViT-B/16 model after training convergence. The quantized model was evaluated on the same held-out test split, preprocessing pipeline, binary label space distinguishing alert and drowsy states, and Raspberry Pi 5 runtime setup used in the main manuscript.

Table 6 Compact overview of the seven public datasets used in this study

Dataset	Type	Key characteristics
YawDD	Vid.	In-vehicle fatigue/yawning cues; frame extraction; facial visibility screening.
NTHU	Vid.	Heterogeneous fatigue behaviors; viewpoint/illumination diversity.
UTA-RLDD	Vid.	Realistic RGB driving data; screened for single-frame suitability.
Kaggle DD	Img.	Close-up fatigue cues; mixed acquisition conditions.
DROZY	Multi.	Controlled drowsiness recordings; visual component only.
DFAT	Img./Frm.	Heterogeneous fatigue samples; binary relabeling-compatible subset used.
EyeClose-Videos	Vid.	Eye-closure-centered cues; usable frames retained.

Table 7 Post-training INT8 quantization results for ViT-B/16 on Raspberry Pi 5

Model	Format	Acc.	F1	AUC	MB	ms	FPS
ViT-B/16	FP32	95.0	93.9	0.986	330–345	55.6	18.0
ViT-B/16	INT8	94.5	93.3	0.983	85–95	42.0	23.8

Note: The FP32 row reports the selected final checkpoint used for the FP32-versus-INT8 comparison, whereas Tables 3 and A5 report mean $\pm$ standard deviation over five fixed random seeds

Table 8 Training hyperparameters and reproducibility settings

Setting	Value
Random seed	42
Optimizer	AdamW
Initial learning rate	1×10^{-4}
β_1	0.9
β_2	0.999
ϵ	1×10^{-8}
Weight decay	1×10^{-4}
Batch size	32
Dropout	0.1
Scheduler type	ReduceLROnPlateau
Scheduler parameters	factor = 0.1, patience = 3, min_lr = 1×10^{-6}
Early stopping patience	5
Maximum epochs	20
Checkpoint selection	Best validation loss
Final training framework	TensorFlow
Prototyping framework	PyTorch

Table 9 Hardware, OS, and software environment used for training and embedded inference

Stage	Hardware / OS	Software versions
Training	Intel Core i9-12900K, NVIDIA RTX 3090, 64 GB RAM, Ubuntu 22.04 LTS	TensorFlow 2.13, CUDA 11.8, cuDNN 8.6, Python 3.10
Prototyping	Intel Core i7-12700H laptop, 32 GB RAM, Ubuntu 22.04 LTS	PyTorch 2.1, Python 3.10
Embedded inference	Raspberry Pi 5 (8 GB RAM), Raspberry Pi OS 64-bit (Bookworm)	ONNX Runtime 1.17.1, OpenCV 4.8.0, MediaPipe 0.10.14, Python 3.11.2
Input device	Raspberry Pi Camera Module 3 Wide	RGB face crop, 224×224 input

Table 10 Repeated evaluation under five fixed random seeds for the ViT, MobileNetV2, and YOLO models on the same held-out test split. Seed rows report raw results; the Overall row reports mean $\pm$ standard deviation across the five runs

Model	Seed	Acc. (%)	F1 (%)	AUC
ViT	42	94.8	93.6	0.985
ViT	123	95.1	94.0	0.987
ViT	256	94.9	93.8	0.986
ViT	512	95.3	94.2	0.988
ViT	1024	94.7	93.5	0.984
ViT	Overall	95.0 ± 0.2	93.8 ± 0.3	0.986 ± 0.002
MobileNetV2	42	92.1	90.9	0.949
MobileNetV2	123	92.4	91.2	0.953
MobileNetV2	256	92.0	90.8	0.948
MobileNetV2	512	92.5	91.3	0.954
MobileNetV2	1024	91.9	90.7	0.951
MobileNetV2	Overall	92.2 ± 0.3	91.0 ± 0.3	0.951 ± 0.003
YOLO	42	93.1	91.8	0.978
YOLO	123	93.4	92.1	0.980
YOLO	256	93.0	91.7	0.977
YOLO	512	93.5	92.2	0.981
YOLO	1024	92.9	91.6	0.977
YOLO	Overall	93.2 ± 0.3	91.9 ± 0.3	0.979 ± 0.002

Author Contributions Z. E. Conceptualization, validation, visualization, writing—original draft. F. E., Z. K., A. B. Data curation, formal analysis, investigation, methodology, software, and validation. S. E. Validation, visualization, writing, review, and editing.

Funding The authors received no specific funding for this work.

Data Availability The public datasets used are cited in the manuscript; derived/curated data are available from the corresponding author on reasonable request.

Code Availability Code for preprocessing and embedded deployment is available from the corresponding author on reasonable request.

Declarations

Ethics approval Ethical approval was waived according to institutional policy for non-invasive in-vehicle recordings; all procedures complied with applicable guidelines.

Consent to participate Informed consent was obtained from all participants.

Consent for publication All authors consent to the publication of this manuscript.

Competing interests The authors have no relevant financial or non-financial interests to disclose.

References

1. National Highway Traffic Safety Administration: Drowsy Driving. <https://www.nhtsa.gov/risky-driving/drowsy-driving>. Accessed 19 July 2025
2. Zhao, Z., Zhou, N., Zhang, L., Yan, H., Xu, Y., Zhang, Z.: Driver fatigue detection based on convolutional neural networks using em-cnn. *Comput. Intell. Neurosci.* **2020**, 7251280 (2020). <https://doi.org/10.1155/2020/7251280>
3. Phan, A.C., Nguyen, N.H.Q., Trieu, T.N., Phan, T.C.: An efficient approach for detecting driver drowsiness based on deep learning. *Appl. Sci.* **11**(18), 8441 (2021). <https://doi.org/10.3390/app11188441>
4. Florez, R., Palomino-Quispe, F., Coaquira-Castillo, R.J., Herrera-Levano, J.C., Paixão, T., Alvarez, A.B.: A cnn-based approach for driver drowsiness detection by real-time eye state identification. *Appl. Sci.* **13**(13), 7849 (2023). <https://doi.org/10.3390/app13137849>
5. Gomaa, M.W., Mahmoud, R.O., Sarhan, A.M.: A cnn-lstm-based deep learning approach for driver drowsiness prediction. *J. Eng. Res.* **6**(4), 59–70 (2022)
6. Alameen, S.A., Alhothali, A.M.: A lightweight driver drowsiness detection system using 3dcnn with lstm. *Comput. Syst. Sci. Eng.* **44**(2), 895–912 (2023). <https://doi.org/10.32604/CSSE.2023.027665>
7. Hu, S., Gao, Q., Xie, K., Wen, C., Zhang, W., He, J.: Efficient detection of driver fatigue state based on all-weather illumination scenarios. *Sci. Rep.* **14**, 17075 (2024). <https://doi.org/10.1038/s41598-024-67131-5>
8. Hassan, O.F., Ibrahim, A.F., Gomaa, A., Makhlof, M.A., Hafiz, B.: Real-time driver drowsiness detection using transformer architectures: a novel deep learning approach. *Sci. Rep.* **15**, 17493 (2025). <https://doi.org/10.1038/s41598-025-02111-x>
9. Lal, S.K.L., Craig, A.: A critical review of the psychophysiology of driver fatigue. *Biol. Psychol.* **55**(3), 173–194 (2001)
10. Budak, U., Bajaj, V., Akbulut, Y., Atila, O., Sengur, A.: An effective hybrid model for eeg-based drowsiness detection. *IEEE Sens. J.* **19**(17), 7624–7631 (2019)
11. Li, G., Lee, B.-L., Chung, W.-Y.: Smartwatch-based wearable eeg system for driver drowsiness detection. *IEEE Sens. J.* **15**(12), 7169–7180 (2015)
12. Doudou, M., Bouabdallah, A., Berge-Cherfaoui, V.: Driver drowsiness measurement technologies: Current research, market solutions, and challenges. *Int. J. Intell. Transp. Syst. Res.* **18**, 297–319 (2019)
13. Ma, J., Murphey, Y.L., Zhao, H.: Real time drowsiness detection based on lateral distance using wavelet transform and neural network. In: *Proc. IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 411–416 (2015)
14. Guo, J., Markoni, H.: Driver drowsiness detection using hybrid convolutional neural network and long short-term memory. *Multimedia Tools and Applications.* **78**, 29059–29087 (2019)
15. Hashemi, M., Mirrashid, A., Beheshti Shirazi, A.: Driver safety development: Real-time driver drowsiness detection system based on convolutional neural network. *SN Comput. Sci.* **1**, 289 (2020). <https://doi.org/10.1007/s42979-020-00300-7>
16. Balandong, R.P., Ahmad, R.F., Saad, M.N.M., Malik, A.S.: A review on eeg-based drowsiness detection systems. *IEEE Access.* **6**, 22908–22919 (2018)
17. Ed-Doughmi, Y., Idrissi, N., Hbali, Y.: Real-time system for driver fatigue detection based on a recurrent neuronal network. *J. Imaging.* **6**(8), 1–14 (2020)
18. Sahayadhas, A.S., Sundaraj, K., M., M.: Detecting driver drowsiness based on sensors: A review. *Sensors.* **12**(12), 16937–16953 (2012)
19. Khushaba, R.N., Kodagoda, S., Lal, S., Dissanayake, G.: Driver drowsiness classification using fuzzy wavelet-packet-based feature-extraction algorithm. *IEEE Trans. Biomed. Eng.* **58**(1), 121–131 (2010)
20. Jarndal, A., Tawfik, H., Siam, A.I., Alsyoud, I., Cheaitou, A.: A real-time vision transformers-based system for enhanced driver drowsiness detection and vehicle safety. *IEEE Access.* **13**, 1790–1803 (2025). <https://doi.org/10.1109/ACCESS.2024.3522111>
21. Roboflow Community: Drowsy bybng Dataset (2023). <https://universe.roboflow.com/company-m1rh5/drowsy-bybng/dataset/1/images>. Accessed 20 July 2025
22. Shankar, S.: Drowsiness SGVF2 Dataset (2023). <https://universe.roboflow.com/saujanya-shankar/drowsiness-sgvf2/dataset/11>. Accessed 20 July 2025
23. Nasri, I.: Driver Drowsiness Dataset (DDD) (2022). <https://www.kaggle.com/datasets/ismailnasri20/driver-drowsiness-dataset-ddd?resource=download>. Accessed 20 July 2025
24. Siamdip: Driver Fatigue Detection Dataset (2022). <https://www.kaggle.com/datasets/siamdip/driver-fatigue-detection>. Accessed 20 July 2025
25. Muc, M.: Frame-level Driver Drowsiness Detection (2022). <https://www.kaggle.com/datasets/matjazmuc/frame-level-driver-drowsiness-detection-fl3d>. Accessed 20 July 2025
26. Fahmy, H.: Opened/Closed Eyes Dataset (2021). <https://www.kaggle.com/datasets/hazemfahmy/opened-closed-eyes>. Accessed 20 July 2025
27. Dataport, I.: YAWDD: Yawning Detection Dataset (2020). <https://iee-dataport.org/open-access/yawdd-yawning-detection-dataset>. Accessed 20 July 2025

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.