



Edge-Computing Deployment and Evaluation of YOLOv10 for Driver Drowsiness Detection

Andika Setiawan¹, Radhinka Bagaskara², Muhammad Fida Raditya³, Mugi Praseptiawan⁴

^{1,2,3,4} Department of Informatics Engineering, Sumatera Institute of Technology, Indonesia

⁴ Human Centered Computing and Information Systems Lab (HCC-ISL), Universiti Teknikal Malaysia Melaka, Malaysia

¹andika.setiawan@if.itera.ac.id, ²radhinka.bagaskara@if.itera.ac.id, ³muhammad.120140037@student.itera.ac.id, ⁴mugi.praseptiawan@if.itera.ac.id

ARTICLE INFORMATION

Article History:

Received: August 20, 2026

Last Revision: September 16, 2026

Published Online: September 30, 2026

KEYWORDS

YOLOv10,
Driver Drowsiness Detection,
Edge Computing,
NVIDIA Jetson Orin Nano,
Driver Monitoring

CORRESPONDENCE

Phone: +6285728117270

E-mail: andika.setiawan@if.itera.ac.id

ABSTRACT

Driver drowsiness is a major safety risk requiring early and computationally efficient on-board monitoring. This Research evaluates the implementation of a previously trained YOLOv10n object detection model on an NVIDIA Jetson Orin Nano edge device for driver drowsiness detection. The model detects two visual indicators, eye closure and yawning, which are combined with timing rules to classify the driver as alert or drowsy and to trigger local audible warnings. Unlike the previous model comparison study, this research focuses on post-training behaviour, scenario robustness and hardware resource use on the embedded platform. The system was evaluated using thirteen pre-recorded driving simulation scenarios that varied lighting, use of glasses and head orientation, comprising one false-positive control and twelve drowsiness scenarios with 79 annotated drowsiness events. Across the drowsiness scenarios, 19 of the 79 events were detected with a single false alarm, giving an event-level recall of 24.05%, a precision of 95.00% and an F1-score of 38.38%, while the control scenario raised no false alarm. The mean model inference time was 46.37 ms per frame, mean CPU and GPU usage were 69.02% and 66.57%, and the average RAM usage ratio was 66% of the total system RAM. Although the raw model latency was compatible with near-real-time processing, reliability deteriorated markedly under low light, eyeglass reflections, head movements and temporary loss of detection, which reset the temporal drowsiness counters. The results show that YOLOv10n can run locally on the Jetson Orin Nano, but its reliability must be improved before practical in-vehicle deployment.

1. INTRODUCTION

Road safety is heavily influenced by human factors, in particular fatigue and sleepiness, which impair attention, reaction time and decision-making capacity. Recent crash patterns analyses have further identified drowsy driving as a significant contributor to road safety risks [1],[2]. As signs of sleepiness often develop gradually, the on-board monitoring system should detect early visual signs before the driver reaches a critical loss of consciousness.

Computer vision-based monitoring of drivers offers a non-contact approach to observe facial and behavioural indicators such as closing of eyes, yawning and head tilt.

Within Advanced Driver Assistance Systems (ADAS), these visual signals can be processed locally to create alerts without sending continuous video to a remote server [3]. Recent object detection architectures have further improved the trade-off between recognition performance and computational latency, for example YOLOv10 introduced an end-to-end detection architecture that aims to reduce post-processing overhead while maintaining competitive performance in the detection domain [4].

A previous study by Raditya et al. compared YOLOv10n and EfficientDet using the same image set and training settings. YOLOv10n achieved 91.76% accuracy

for closed-eye detection and 93.57% for yawning, with an average inference time of 27.59 ms on the development environment, outperforming EfficientDet in both detection performance and latency [5]. These findings have motivated the selection of YOLOv10n for the next phase of the research: deployment on an embedded edge computing platform.

Deployment introduces limitations that are not visible during the evaluation of the model at the image level. Embedded devices have limited CPU, GPU, memory, and thermal headroom, and driver monitoring performance may degrade if the lighting changes, glasses reflect, or the driver turns away from the camera. The practical question is therefore not only whether the detector will perform well on a fixed set of images, but also whether the full temporal decision pipeline can be performed continuously on edge equipment and remain robust in the event of scenario changes.

This Research addresses this deployment-oriented issue by running the selected YOLOv10n model on the NVIDIA Jetson Orin Nano and evaluating both the detection behaviour and the use of computational resources in 13 simulated driving scenarios. The contribution is a system-wide assessment linking image-based detection, temporal decision rules, embedded inference, local warning activation and scenario-specific fault analysis. This Research does not repeat the comparison of YOLOv10 and EfficientDet models. It extends the previous work to edge deployment and reliability assessment.

2. RELATED WORK

Recent studies of driver drowsiness have increasingly combined visual facial recognition with real-time inference. Soe et al. used YOLO for abnormal driver behaviour detection in ADAS settings and reported high accuracy of recognition in multiple driving modes and lighting conditions [6]. Ayachi et al. implemented EfficientDet for driver fatigue detection and demonstrated the utility of single-stage object detection in ADAS applications in real time [7]. These studies show that eye and mouth-related visual indicators can support the monitoring of non-contact fatigue, but their main focus is on detection performance rather than on the embedded resource profile.

A second line of work focuses explicitly on embedded deployment. Civik and Yuzgec implemented CNN-based eye and mouth classification on an NVIDIA Jetson Nano and reported real-time operation at approximately 6 FPS [8]. Kim et al. combined facial landmarks, eye-closure analysis, head-pose recognition, and infrared imaging, showing that low-light robustness can be improved when the sensing pipeline is designed for challenging illumination [9]. Rahman et al. also demonstrated an embedded Jetson Nano system that handled variations in lighting, eyeglasses, viewing distance, and oblique face projection while combining computer vision with physiological monitoring [10].

Other recent studies have explored different efficiency and robustness strategies. Sahin evaluated several machine-learning classifiers on Jetson Nano and Jetson TX2 platforms [11]. Florez et al. proposed a real-time

embedded eye-and-mouth system using a Jetson Nano and near-infrared camera, obtaining 14 FPS in real-environment tests [12]. Debsi et al. introduced an attention-enhanced YOLOv8 model for driver distraction and fatigue detection [13], while Fang et al. used an improved YOLOv5-based multi-feature approach to increase robustness to variations in facial appearance and driver state [14]. Jia et al. combined multiple lightweight modules to reduce sensitivity to external factors such as illumination and occlusion [15].

The literature also shows that sensing and environmental conditions strongly affect deployment reliability. Beles et al. combined multiple drowsiness indicators for autonomous-driving contexts [16]. Liu et al. specifically targeted complex lighting conditions with YOLO-FDCL, emphasising that illumination variation remains a major source of performance degradation [17]. Dhasarathan et al. demonstrated that edge optimisation with TensorRT can substantially increase inference throughput on Jetson-class hardware [18]. Phan et al. integrated deep-learning drowsiness detection with an IoT warning system on a Jetson Nano and evaluated practical variations including glasses, head tilt, and other facial cues [19]. Zhou et al. also proposed a deep-learning fatigue-detection and early-warning system, reporting real-time visual monitoring across multiple fatigue levels [20]. Together, these studies show that embedded feasibility and high benchmark accuracy do not automatically imply robustness across real cabin conditions.

Table 1 contrasts representative studies with the present work in terms of the detector and platform, the evaluation focus, and the main methodological or experimental difference. Studies that used the earlier-generation NVIDIA Jetson Nano are identified as such, the present work uses the newer Jetson Orin Nano.

TABLE 1. COMPARISON OF RELATED DRIVER DROWSINESS STUDIES WITH THE PRESENT WORK

Study	Detector / platform	Evaluation focus	Difference from this research
Raditya et al. [5]	YOLOv10n vs. EfficientDet; development PC	Image-level accuracy (91.76% closed eye, 93.57% yawn); 27.59 ms	No embedded deployment, temporal rules, warning, or scenario test
Soe et al. [6]	YOLO; ADAS setting	Recognition accuracy across driving modes and lighting	Detection-level focus; edge resource profile not emphasised
Ayachi et al. [7]	EfficientDet; ADAS setting	Real-time single-stage fatigue detection	Detection-level focus; no event-level scenario analysis
Civik & Yuzgec [8]	CNN eye/mouth classifier; Jetson Nano	Real-time feasibility (~6 FPS)	Earlier Jetson generation; classifier rather than one-stage detector
Rahman et al. [10]	Vision + physiological sensing; Jetson Nano	Lighting, eyeglasses, distance, oblique face	Multimodal sensing; this research is vision-only with resource logging
Florez et al. [12]	CNN + mouth aspect ratio;	14 FPS in real-environment tests	NIR sensing; this research uses a standard USB

Study	Detector / platform	Evaluation focus	Difference from this research
	Jetson Nano + NIR camera		webcam and factorial scenarios
Liu et al. [17]	YOLO-FDCL (improved YOLOv8)	Robustness to complex lighting	Architecture modification; this research evaluates an unmodified detector after deployment
Dhasarathan et al. [18]	TensorRT-optimised model; Jetson-class edge device	Inference throughput	Speed optimisation; this research reports scenario-level reliability
Phan et al. [19]	Deep learning + IoT alerting; Jetson Nano	Glasses, head tilt, facial cues; smart alerting	Earlier Jetson and detector; no 3×2×2 scenario design or CPU/GPU/RAM logging
This Research	Pre-trained YOLOv10n + temporal rules + audible warning; Jetson Orin Nano	13 scenarios (3×2×2 + control); event-level recall, precision, F1 and false-positive control; latency; CPU/GPU/RAM; warning-path analysis	—

The remaining gap relevant to this work is the joint evaluation of a selected YOLOv10n drowsiness detector after deployment on a Jetson Orin Nano, including event-level drowsiness decisions, scenario-dependent behaviour, and simultaneous CPU, GPU, memory, and latency measurements. Existing work often reports either model accuracy, embedded speed, or robustness to a subset of environmental factors. The present study combines these perspectives using a controlled set of thirteen scenarios that vary lighting, eyeglass use, and head orientation, while retaining the same YOLOv10n model selected by the preceding comparative study [5].

Accordingly, the novelty of this paper lies in the deployment-level evaluation rather than in proposing a new YOLO architecture. Compared with the studies in Table 1, this work differs in two ways. First, the YOLOv10n detector benchmarked in [5] is evaluated without modification, together with its temporal decision rules and audible warning, as one system on the Jetson Orin Nano, so the observed behaviour reflects deployment and not a change of model. Second, the system is tested in a $3 \times 2 \times 2$ factorial scenario set (illumination $\times$ eyeglasses $\times$ head orientation) plus a false-positive control, performance is reported with event-level recall, precision and F1-score, and latency, CPU, GPU and RAM usage are logged together with the detection outcomes. The analysis shows where a detector that performs well on still images loses reliability once temporal rules and embedded hardware are added.

Specific objectives are: (1) to deploy a pre-trained YOLOv10n detector on the NVIDIA Jetson Orin Nano, (2) to integrate temporal eye-closure and yawning rules with a local warning actuator, (3) to quantify the inference latency and the use of CPU, GPU, and RAM, and (4) to analyse

event-level detection performance and failure modes in 13 driving simulation scenarios.

3. METHODOLOGY

This Research used an applied experimental approach. In this research, the detector itself was not retrained. Instead, the previously selected YOLOv10n weights have been treated as an input for the Deployment Phase. This separation preserves the difference in scope between the previous publication on model comparison and the current evaluation of embedded systems [5].

3.1 Research Workflow

The workflow is shown in Figure 1. It consists of six phases: preparation of the YOLOv10n pre-trained model, development of the inference application, deployment of the application and weights on the NVIDIA Jetson Orin Nano, running a scenario-based video test, collecting measurements of the detection and resource usage, and validation of live-alert implementation. Figure 1 therefore represents the full post-training evaluation pipeline and not the training process of the model.

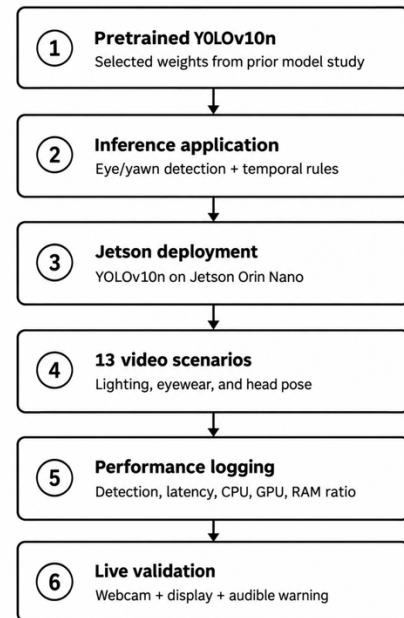


FIGURE 1. POST-TRAINING DEPLOYMENT AND EVALUATION WORKFLOW

In the previous study, the YOLOv10n model was trained with a batch size of 16, a 512×512 pixel input resolution, and 25 epochs. In the original test setting, it achieved 91.76% accuracy for closed eyes detection and 93.57% for yawning, with an average inference time of 27.59 ms [5]. These values serve as an offline baseline against which to discuss embedded deployment behaviour. They are not recalculated in this experiment.

A Python inference application was then developed to read frames from a pre-recorded video or webcam, run YOLOv10n inference, keep time counters for eye closing and yawning, display the predicted status, and record hardware statistics. The same decision-making logic was applied during the pre-recorded video evaluation and the final live demonstration to allow for the attribution of

behavioural changes to the input scenario rather than to a different application logic.

The model and application have been deployed on the Jetson Orin Nano and tested in 13 pre-recorded scenarios. Resource monitoring was done during the inference using statistics from the Jetson platform [21]. After the offline evaluation of the scenario, the same application was connected to the webcam and external speaker for live demonstration.

3.2 Hardware Configuration

The drowsiness detection system was implemented as a self-contained edge prototype following an input–processing–output structure, with a webcam and local power supply as inputs, a single embedded device handling inference and decision logic, and a display and speaker as outputs.

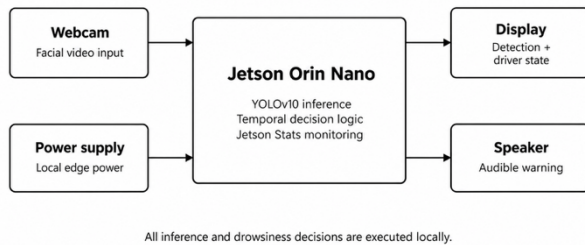


FIGURE 2. HARDWARE CONFIGURATION AND LOCAL PROCESSING PATH

Figure 2 shows hardware configuration and path to local processing. The NVIDIA Jetson Orin Nano Developer Kit (8 GB unified memory, flashed on a 256 GB NVMe M.2 SSD and running Linux for Tegra) served as the YOLOv10n edge computing platform. The camera provided images of faces, the external display displayed the detection output, and the external speaker functioned as a warning actuator. Processing was done locally on the Jetson device without cloud inference. The inference script was written in Python 3.10 with OpenCV, used the YOLOv10 implementation of Wang et al. [4] (THU-MIG repository, based on Ultralytics 8.1.34) on PyTorch 2.3.0 for aarch64, with the trained weights executed directly (no TensorRT conversion), and read hardware statistics through the jtop interface of Jetson Stats [21]. The webcam and speaker were connected through USB. Here, “Jetson Orin Nano” denotes the platform used in this research, and “Jetson Nano” denotes the earlier-generation device used in [8], [10]–[12], [19].

This configuration has been chosen to assess whether a compact embedded platform can maintain a full detection and warning pipeline and not only perform isolated model inference. The hardware settings were kept constant in all pre-recorded scenarios to ensure that changes in the measured behaviour were related to the video conditions and not to hardware changes.

3.3 Driver Drowsiness Decision Rules

Object detector outputs eye and mouth states, but these are not sufficient for classification as drowsiness. Therefore, temporal rules were used to distinguish the

normal blinking or talking from the persistent drowsiness indicators. This design follows the research logic used in the source experiment and is summarised in Table 2.

Eye closing was considered an indication of drowsiness if the closed eye state lasted longer than 0.5 s. Previous drowsiness studies associate blinks longer than 0.4 s with drowsiness [22]. The evaluated implementation applied a slightly stricter threshold of 0.5 s, which is the value used for all results in this paper. Yawning was considered an indicator of drowsiness if the mouth-open state lasted for more than 5 s, which is consistent with the duration of yawning standardised in the original study [23].

Formally, let $T_{\text{eye}}(t)$ be the continuous duration for which the eye-closed state has been detected up to frame t , and $T_{\text{mouth}}(t)$ the continuous duration of the mouth-open (yawning) state. The driver state is $D(t) = 1$ (Drowsy) if $T_{\text{eye}}(t) > 0.5 \text{ s}$ OR $T_{\text{mouth}}(t) > 5 \text{ s}$, and $D(t) = 0$ (Alert) otherwise. Each counter increases while its state is detected in consecutive frames and is reset to zero when the opposite state (eye open or mouth closed) is detected or when the corresponding state is not detected in a frame. Consequently, normal blinks ($\leq 0.5 \text{ s}$) and mouth openings during speech ($\leq 5 \text{ s}$) leave the state Alert, and the two conditions are evaluated independently, so either one alone is sufficient to classify the driver as Drowsy. Because the counters depend on successive detections, the accumulated time may be reset by the temporary loss of the eye or mouth detection. This dependence becomes relevant in the analysis of failures reported in Section 4.4.

TABLE 2. RULES FOR DECISION ON DRIVER DROWSINESS (DROWSY IF RULE 1 OR RULE 2 IS SATISFIED)

No.	Eye State	Mouth State	Class (0/1)
1	Closed $> 0.5 \text{ s}$	Any	Drowsy (1)
2	Any	Open (yawning) $> 5 \text{ s}$	Drowsy (1)
3	Open, or closed $\leq 0.5 \text{ s}$ (blink)	Closed, or open $\leq 5 \text{ s}$ (e.g., speaking)	Alert (0)

The drowsy state is connected to the audible warning mechanism, so that a detected drowsiness event triggers an immediate local warning as soon as $D(t) = 1$. This actuator integration is part of the deployment contribution as it shows the full sensor–decision–warning chain of the edge device and not just the detector output.

Evaluation videos were recorded with the vehicle stationary while a single demonstrator seated in the driver’s seat simulated normal and drowsy driving behaviour. The thirteen scenarios systematically varied three practical factors in a $3 \times 2 \times 2$ factorial design: illumination (daytime, afternoon, and night, where the night condition reproduced normal street lighting by directing a dim lamp towards the face), eyeglass use, and head orientation (forward versus looking towards the right and left A-pillars). Scenario 1 was a special false positive control containing normal speech and rapid blinking, while scenarios 2 to 13 contained scripted drowsiness patterns used to detect events. Figure 3 visualises the behaviour timeline and Table 3 shows the full set of scenarios.

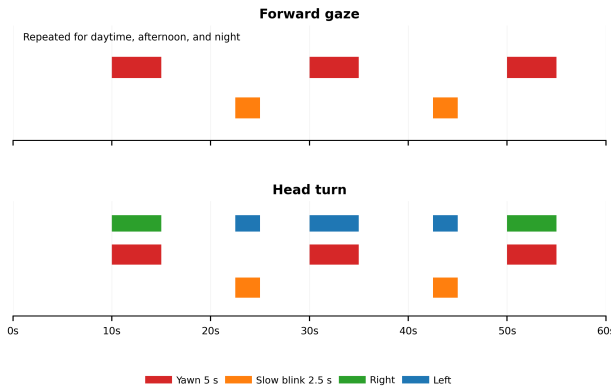


FIGURE 3. BEHAVIORAL TIMELINE USED IN PRERECORDED DRIVING SCENARIOS

TABLE 3. TESTING SCENARIOS

Video No.	Testing Scenario
1	False-positive control
2	Glasses, afternoon, looking left/right
3	Glasses, afternoon, looking ahead
4	Glasses, night, looking left/right
5	Glasses, night, looking ahead
6	Glasses, daytime, looking left/right
7	Glasses, daytime, looking ahead
8	No glasses, afternoon, looking left/right
9	No glasses, afternoon, looking ahead
10	No glasses, night, looking left/right
11	No glasses, night, looking ahead
12	No glasses, daytime, looking left/right
13	No glasses, daytime, looking ahead

Each scenario was recorded as one video of approximately one minute, giving a total duration of 13 min 51 s (831 s). The videos were captured with the rear camera of a Samsung Galaxy Tab S7 tablet (SM-T875) at 1920×1080 pixels and approximately 30 frames per second (29.8 fps), and the frame rate was read from each file. Before inference every frame was resized directly to 512×384 pixels (4:3, without preserving the 16:9 aspect ratio), which the detector letterboxes to its 512-pixel training resolution [5], and detections were accepted at a confidence threshold of 0.6. The temporal counters convert the number of consecutive frames containing a state into seconds using the frame rate of the file, so the decision thresholds refer to video time rather than to processing time.

Ground truth was defined at event level from the recording script: the time (in whole seconds) of every scripted drowsiness event was stored as metadata for each video. Scenarios 2-13 contain 79 drowsiness events in total (five to seven per video, Table 6), whereas Scenario 1 contains only negative behaviour (normal blinking and speech) and no drowsiness event. Each scenario was

recorded once and each video was processed once on the device in a single continuous run. The detector was trained on the Driver Eyes and Mouth State dataset (version 6; 10,218 training and 2,903 validation images derived from DMD and YawDD footage) [5], so none of the evaluation videos were used for training. Table 4 summarises the dataset and acquisition settings.

TABLE 4. EVALUATION DATASET AND ACQUISITION SETTINGS

Item	Description
Participant	One demonstrator simulating driving
Recording setting	Stationary vehicle, driver's seat, front-facing view
Camera / native resolution	Samsung Galaxy Tab S7 (SM-T875) / 1920×1080 px, ≈ 30 fps (29.8 fps)
Processing resolution / confidence	Frames resized to 512×384 px; confidence threshold 0.6
Model input size	512 px (letterboxed), as in training [5]
Videos	13 (1 false-positive control + 12 drowsiness scenarios), ≈ 1 min each
Scenario factors	Illumination (day, afternoon, night with dim lamp) $\times$ eyeglasses (yes/no) $\times$ head orientation (forward, towards right/left A-pillar)
Total duration	13 min 51 s (831 s)
Annotated drowsiness events	79 event times in Scenarios 2-13 (5-7 per video, 1-s resolution); none in Scenario 1
Repetitions	One recording per scenario; one processing run per video
Training data of the detector	Driver Eyes and Mouth State v6: 10,218 train / 2,903 validation images (DMD + YawDD) [5]; no overlap with test videos
Edge platform / software	Jetson Orin Nano Developer Kit (8 GB, 256 GB NVMe, Linux for Tegra); Python 3.10, PyTorch 2.3.0 (aarch64), YOLOv10 (THU-MIG, Ultralytics 8.1.34 base), OpenCV, jtop

3.4 Performance Evaluation

Each scenario has been individually processed using the same software and hardware. The experiment recorded the number of expected and observed drowsiness events, along with the average latency of the model inference, CPU usage, GPU usage, and RAM statistics returned by the monitoring program. The thirteen videos contain 79 annotated drowsiness events, all of which belong to Scenarios 2-13.

Detection performance was evaluated at event level. A predicted drowsiness event is a transition of $D(t)$ from 0 to 1, whose video time was written to the result log of each run. Predicted events were matched one-to-one with the annotated event times of the same video, pairing the closest times first, and a pair was accepted when the two times differed by no more than 3 s. Matched predictions are true positives (TP), unmatched predictions are false positives (FP) and unmatched annotated events are false negatives (FN). Recall, precision and F1-score are then:

$$\text{Recall} = TP / (TP + FN) \quad (1)$$

$$\text{Precision} = TP / (TP + FP) \quad (2)$$

$$F1 = 2 \times \text{Precision} \times \text{Recall} / (\text{Precision} + \text{Recall}) \quad (3)$$

The ± 3 s window reflects the 1-s resolution of the annotations. The results are identical for windows of 3-5 s and decrease for tighter windows (e.g., ± 2 s: recall 17.72%, precision 70.00%), so the reported values are not sensitive to the exact choice above 3 s. The quantity reported as “inference accuracy” in the source experiment was obtained by comparing the numbers of predicted and annotated events per video. For all scenarios except Scenario 7 it equals the time-matched recall (see Section 4.1). Scenario 1 contains only non-drowsy behaviour and therefore serves as the false-positive control, with:

$$FPR = FP / N_{neg} \quad (4)$$

where N_{neg} is the number of negative (speech and rapid-blink) episodes in Scenario 1. Scenario 1 measures a different outcome (correct rejection of normal behaviour) from Scenarios 2-13 (detection of drowsiness events). It is therefore reported separately and excluded from the recall average.

The average inference time represents the latency per frame, while the total wall clock processing time includes frame acquisition, visualisation, timing logic, logging, and file I/O. The CPU and GPU values represent the average usage during the execution. The monitoring script read `jetson.stats['RAM']`, Jetson Stats defines this simplified RAM statistic as the ratio of RAM used to total RAM [21]. Therefore, recorded values of 0.63-0.69 are treated here as a ratio of the use of RAM at system level, not as megabytes.

4. RESULT AND DISCUSSION

4.1 Performance Evaluation of the Proposed Driver Drowsiness Monitoring System

Table 5 summarises the results of the 13 scenarios. The scenarios were more varied than the image-based test suite used in the previous publication, as they introduced time continuity, lighting changes, reflections from glasses, and head motion. This difference explains why the end-to-end event detection rate should not be compared directly with the previous accuracies for closed eyes and yawning [5].

The total video duration was 13 min 51 s and the final processing on the Jetson Orin Nano took 28 min 58 s according to the result log, i.e., the complete offline pipeline ran at approximately 0.48 times real-time speed (831 s of video in 1,738 s). The longer wall-clock time reflects the full Python pipeline, including video decoding, object detection, visualisation, time-state evaluation, logging of resources, and output processing. On the contrary, the latency values in Table 5 represent the observed component of the inference per frame.

TABLE 5. PERFORMANCE EVALUATION RESULTS OF THE YOLOV10N INFERENCE MODEL

Scenario	Event Recall (%)	Inference time (ms)	CPU Usage (%)	GPU Usage (%)	RAM Utilisation (ratio)
1	n/a*	45.48	64.32	69.59	0.63
2	0.00	44.72	64.42	69.81	0.63
3	28.57	45.86	67.26	67.57	0.64
4	0.00	45.32	66.79	71.80	0.64

Scenario	Event Recall (%)	Inference time (ms)	CPU Usage (%)	GPU Usage (%)	RAM Utilisation (ratio)
5	0.00	46.22	69.80	68.06	0.65
6	0.00	46.26	69.23	67.67	0.66
7	100.00†	46.99	71.43	67.45	0.66
8	14.29	47.45	73.72	65.03	0.67
9	14.29	47.23	72.65	67.97	0.67
10	0.00	46.50	72.07	62.32	0.68
11	42.86	47.20	71.61	61.85	0.68
12	42.86	46.93	71.41	61.73	0.69
13	57.14	46.64	62.61	64.54	0.69
Mean**	25.00	46.37	69.02	66.57	0.66

* Scenario 1 is the false-positive control (0 false alarms, i.e., 100% correct rejection) and is excluded from the recall mean. ** Recall averaged over Scenarios 2-13; latency and resource values averaged over all 13 scenarios. † Time-matched recall (5 of 5 events); the value of 83.33% in the source experiment corresponds to 5 correct out of 6 predicted events.

Scenario 1, the false-positive control, raised no false alarm: the normal speech and rapid blinking it contains did not trigger the drowsiness logic (false-positive rate 0%, 100% correct rejection). Because this result measures specificity rather than the detection of drowsiness events, it is reported separately and is no longer included in the recall average. The twelve drowsiness scenarios showed much greater variability: the highest recall was 100% for Scenario 7 (glasses, daytime, looking ahead), while several combinations of low light, glasses and head motion (Scenarios 2, 4, 5, 6, and 10) produced 0%. The macro-averaged recall over Scenarios 2-13 is 25.00% (23.61% if the count-based value of the source experiment is used for Scenario 7), and the pooled recall is 24.05% (19 of 79 events). This separation gives a clearer picture of the low sensitivity of the existing temporal pipeline in demanding drowsiness scenarios.

Table 6 reports the time-matched event-level results. Of the 79 annotated drowsiness events, 19 were detected and 60 were missed, while only one of the 20 predicted events was a false positive. The pooled recall is therefore 24.05%, the precision 95.00% and the F1-score 38.38%, and the control scenario produced no detection (FPR = 0%). The single false positive occurred in Scenario 7, where the warning was triggered twice within 1.5 s around the same annotated event. The error profile is thus dominated by missed events rather than by false alarms, which is consistent with the counter-reset failure mechanism discussed in Section 4.4. The source experiment observed that most misses occurred for yawns, which must be tracked continuously for 5 s, whereas prolonged eye closures (> 0.5 s) were still detected in several scenarios.

TABLE 6. EVENT-LEVEL DETECTION RESULTS PER SCENARIO

Sc.	E	Pred	TP	FP	FN	Rec. (%)	Prec. (%)	F1 (%)
1*	0	0	0	0	-	n/a	n/a	n/a
2	7	0	0	0	7	0.00	n/a	0.00
3	7	2	2	0	5	28.57	100.00	44.44
4	6	0	0	0	6	0.00	n/a	0.00
5	7	0	0	0	7	0.00	n/a	0.00
6	5	0	0	0	5	0.00	n/a	0.00
7	5	6	5	1	0	100.00	83.33	90.91
8	7	1	1	0	6	14.29	100.00	25.00
9	7	1	1	0	6	14.29	100.00	25.00
10	7	0	0	0	7	0.00	n/a	0.00
11	7	3	3	0	4	42.86	100.00	60.00
12	7	3	3	0	4	42.86	100.00	60.00
13	7	4	4	0	3	57.14	100.00	72.73
2-13	79	20	19	1	60	24.05	95.00	38.38

E = annotated drowsiness events. Pred. = predicted events (0-to-1 transitions of D(t)); matching window ± 3 s. *Scenario 1 (control) contains only non-drowsy behaviour and produced no prediction (FPR = 0%). Precision is undefined (n/a) when no event was predicted.

4.2 Computational Performance Analysis

Computational performance was significantly more stable than detection performance. The latency of the inference was only slightly different in the 13 scenarios, suggesting that illumination, use of glasses, and head orientation had a greater influence on the robustness of the recognition than the computational costs of running the detector.

The average recorded time for model inference was 46.37 ms per frame, corresponding to approximately 21.6 model inferences per second when considered separately. This value supports near-real time raw inference on the Jetson Orin Nano. However, the total time of the wall clock video processing was longer than the time of the input video, so that the full experimental pipeline cannot be described as a strict real-time video playback.

The average CPU and GPU usage was 69.02% and 66.57%, respectively. The RAM utilisation ratio for Jetson Stats was an average of 0.66, which corresponds to about 66% of the total system RAM used during the trials. This statistic represents memory usage at system level, and not memory that is exclusively attributable to YOLOv10n. These results suggest that the detector was able to perform without saturating the measured CPU and GPU resources, but also indicate that the computational bandwidth alone was not sufficient to reliably detect drowsiness events. In this experiment, the limiting factor was reliability rather than raw computational availability.

This observation is in line with the embedded driver monitoring studies which report feasible performance on Jetson-class hardware, while also highlighting the importance of lighting control, optimised sensing and model adaptation [8]-[12]. Recent work has shown that TensorRT and model specific optimisation can

significantly increase throughput [18]. This optimisation is relevant for future releases of the system, but does not directly address the detection loss mechanisms identified in this document.

4.3 Live Detection Implementation

After the pre-recorded video evaluation, the application was run with live video input to verify the integrated detection and warning path. Figure 4 shows the final implementation of the Jetson Orin Nano, including the monitoring interface and the driver-side display for the detection.

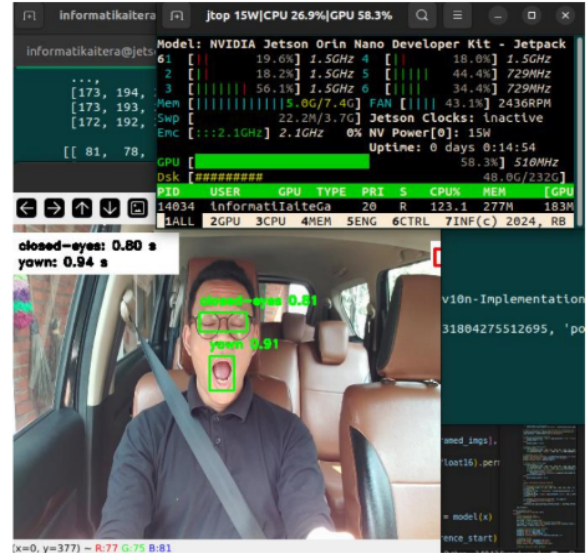


FIGURE 4. REAL-TIME IMPLEMENTATION OF THE PROPOSED DRIVER DROWSINESS DETECTION SYSTEM

During live operations, each frame was processed by YOLOv10n, the detected eye and mouth states were aggregated by the temporal logic, and the predicted driving status was displayed. If either the eye-closure or the yawning rule was met, the external speaker played an audible warning. This confirms that the full application can run locally without cloud inference.

Because the live application uses exactly the same detector, temporal rules and warning trigger as the scenario evaluation, with only the input source changed from a video file to the webcam, the warning path can be quantified from the measured pipeline, as summarised in Table 7. Every drowsiness event detected in Table 6 corresponds to a warning activation (the live script starts the audio playback in a separate thread at the 0-to-1 transition of D(t), so playback does not block the processing loop), so the warning activation accuracy on the 79 annotated events is 24.05% (19 events), with one false warning in the drowsiness scenarios and none in the control scenario. The warning is issued in the same processing cycle in which a counter exceeds its threshold, so the delay after threshold crossing is bounded by one processing cycle, whose model-inference component averaged 46.37 ms (range 44.72-47.45 ms across scenarios). The delay from the onset of a behaviour to the warning is therefore dominated by the deliberate thresholds of 0.5 s for eye closure and 5 s for yawning.

TABLE 7. QUANTITATIVE PERFORMANCE OF THE WARNING PATH

Metric	Value
Correct warning activations / annotated drowsiness events	19 / 79
Warning activation accuracy (%)	24.05
False warnings in drowsiness scenarios	1 (precision 95.00%)
False warnings in control scenario	0
Model inference per processing cycle (ms)	46.37 (44.72-47.45)
Warning delay after threshold crossing	≤ 1 processing cycle
Minimum delay from behaviour onset, eye closure (s)	> 0.5
Minimum delay from behaviour onset, yawning (s)	> 5
End-to-end offline throughput ($\times$ real time)	0.48

These values characterise the warning path under the recorded scenarios. A timed live trial with an independent reference for behaviour onset and audio onset was not part of the source experiment. Such a trial is required to measure the acoustic latency of the speaker and the live end-to-end frame rate, and is listed as future work in Section 5.

The live results show the feasibility of deployment, but should be interpreted in conjunction with the results of the scenario evaluation: the hardware can execute the pipeline, whereas the current detector and timing logic still require robustness improvements prior to deployment in uncontrolled driving environments.

4.4 Discussion

The main finding is the large gap between the performance of the detector at the offline image level and the event-level drowsiness detection performance after deployment. A previous model comparison study reported over 91% accuracy for both visual classes [5], yet the deployed temporal pipeline achieved a pooled event-level recall of 24.05% (F1-score 38.38%) across the twelve drowsiness scenarios. This difference is not contradictory, as both experiments assess different levels of the system: isolated object recognition versus continuous event detection with time and environmental variability.

Four recurrent failure mechanisms were observed in the source experiment. First, temporary loss of detection reset the eye-closure or yawning counters before they reached the required duration. Second, reflections from glasses reduce the visibility of the eye area. Third, rotating the head when blinking or yawning resulted in missed detection when facial features were moved out of the frontal view. Fourth, low lighting reduced the image quality and reliability of the detectors. These mechanisms are in line with recent literature which identifies illumination, occlusion, head position and sensor configuration as the main factors for robustness [9], [10], [15], [17].

Failure modes also explain why improvements to an object detector alone may not be sufficient. A practical driver monitoring system should preserve time evidence

during short detector breaks, e.g., by tracking, smoothing, hysteresis, or a state estimation system that can tolerate a limited number of missed images. Moreover, infrared or low-light cameras may reduce sensitivity to low illumination, while more extensive training data may improve generalisation to eyeglasses and non-frontal head positions [9], [12], [17].

This Research has several limitations. Simulated videos were recorded with stationary vehicles and a single demonstrator, so results should not be generalised to naturalistic driving populations. Each scenario was recorded once, which precludes statistical testing of differences between scenarios. The annotations have a resolution of 1 s, so events were matched within a ± 3 s window, and the live warning latency was characterised from the measured pipeline rather than from a timed live trial. Future experiments should include more drivers, moving vehicle data, repeated recordings per condition, annotation of non-drowsy behaviour in all scenarios, timed live trials, FPS from end to end, power and temperature measurements, and repeat testing under controlled lighting.

5. CONCLUSIONS

This Research deployed and evaluated a previously selected YOLOv10n drowsiness detector on an NVIDIA Jetson Orin Nano edge device, and its findings address the four objectives stated at the end of Section 2.

For objectives (1) and (2), the pre-trained detector was integrated with temporal eye-closure (> 0.5 s) and yawning (> 5 s) rules and an audible warning actuator, and the complete sensing-decision-warning chain ran locally without cloud inference. The warning was activated for 19 of the 79 annotated drowsiness events (24.05%) with one false warning, which did not occur in the control scenario, and a delay after threshold crossing bounded by one processing cycle. For objective (3), the mean model inference time was 46.37 ms per frame (about 21.6 inferences per second), with mean CPU and GPU usage of 69.02 and 66.57% and a system RAM utilisation ratio of 0.66, although the complete offline pipeline ran at only about 0.48 times real-time speed. For objective (4), the twelve drowsiness scenarios yielded a pooled event-level recall of 24.05%, a precision of 95.00% and an F1-score of 38.38%, with per-scenario recall ranging from 0 to 100%, while the false-positive control raised no false alarm.

These results indicate that the main limitation was detection reliability, while the computational resources of the device were sufficient. The dominant failure mechanisms were temporary loss of detection, eyeglass reflections, head motion outside the frontal area and insufficient illumination. Because these failures interrupt the temporal counters, they directly reduce the probability that a true drowsiness event reaches the required duration threshold. These conclusions are bounded by the study's limitations: stationary-vehicle recordings with a single demonstrator, one recording per scenario, event annotations at 1-s resolution, and a warning-path latency derived from the measured pipeline rather than from a timed live trial.

Therefore, future work should prioritise temporal smoothing or object tracking to preserve evidence across short detection gaps, low-light or infrared imaging, more

extensive and varied training data covering eyeglasses and non-frontal poses, TensorRT optimisation with explicit end-to-end throughput and power profiles, and testing with multiple drivers under moving conditions, including timed live trials of the warning latency. These improvements should help to close the gap between strong offline object detection metrics and reliable detection of drowsiness events in the real world.

REFERENCES

- [1] World Health Organization, Global Status Report on Road Safety 2023. Geneva, Switzerland: WHO, 2023.
- [2] M. A. Rahman, S. Das, and X. Sun, "Understanding the drowsy driving crash patterns from correspondence regression analysis," *Journal of Safety Research*, vol. 84, pp. 167-181, 2023, doi: 10.1016/j.jsr.2022.10.017.
- [3] A. V. Postolit, "Prospects for the Use of Artificial Intelligence and Computer Vision in Transport Systems and Connected Cars," *World of Transport and Transportation*, vol. 19, no. 1, pp. 74-90, 2021, doi: 10.30932/1992-3252-2021-19-1-74-90.
- [4] A. Wang et al., "YOLOv10: Real-Time End-to-End Object Detection," *arXiv:2405.14458*, 2024.
- [5] M. F. Raditya, A. Setiawan, and I. F. Ashari, "Perbandingan Kinerja YOLOv10 dan EfficientDet dalam Deteksi Mata Tertutup dan Mulut Terbuka sebagai Parameter Deteksi Pengemudi Mengantuk berbasis Citra Digital," *Seminar Nasional CORISINDO*, pp. 160-165, 2024.
- [6] M. T. Soe, A. Zaw Min, H. T. Kyaw, M. Min Paing, S. M. Htet, and B. Aye, "Abnormal Behavior Detection in Real-time for Advanced Driver Assistance System (ADAS) using YOLO," in *2022 IEEE Symposium on Industrial Electronics & Applications (ISIEA)*, 2022, pp. 1-6, doi: 10.1109/ISIEA54517.2022.9873672.
- [7] R. Ayachi, M. Afif, Y. Said, and A. Ben Abdelali, "Drivers Fatigue Detection Using EfficientDet in Advanced Driver Assistance Systems," in *2021 18th International Multi-Conference on Systems, Signals & Devices (SSD)*, 2021, pp. 738-742, doi: 10.1109/SSD52085.2021.9429294.
- [8] E. Civik and U. Yuzgec, "Real-time driver fatigue detection system with deep learning on a low-cost embedded system," *Microprocessors and Microsystems*, vol. 99, Art. no. 104851, 2023, doi: 10.1016/j.micpro.2023.104851.
- [9] D. Kim, H. Park, T. Kim, W. Kim, and J. Paik, "Real-time driver monitoring system with facial landmark-based eye closure detection and head pose recognition," *Scientific Reports*, vol. 13, Art. no. 18264, 2023, doi: 10.1038/s41598-023-44955-1.
- [10] A. Rahman, M. B. H. Hriday, and R. Khan, "Computer vision-based approach to detect fatigue driving and face mask for edge computing device," *Heliyon*, vol. 8, no. 10, Art. no. e11204, 2022, doi: 10.1016/j.heliyon.2022.e11204.
- [11] M. E. Sahin, "Real-Time Driver Drowsiness Detection and Classification on Embedded Systems Using Machine Learning Algorithms," *Traitement du Signal*, vol. 40, no. 3, pp. 847-856, 2023, doi: 10.18280/ts.400302.
- [12] R. Florez, F. Palomino-Quispe, A. B. Alvarez, R. J. Coaquira-Castillo, and J. C. Herrera-Levano, "A Real-Time Embedded System for Driver Drowsiness Detection Based on Visual Analysis of the Eyes and Mouth Using Convolutional Neural Network and Mouth Aspect Ratio," *Sensors*, vol. 24, no. 19, Art. no. 6261, 2024, doi: 10.3390/s24196261.
- [13] A. Debsi, G. Ling, M. Al-Mahbashi, M. Al-Soswa, and A. Abdullah, "Driver distraction and fatigue detection in images using ME-YOLOv8 algorithm," *IET Intelligent Transport Systems*, vol. 18, no. 10, pp. 1910-1930, 2024, doi: 10.1049/itr2.12560.
- [14] H.-J. Fang, H.-Z. Dong, S.-X. Lin, J.-Y. Luo, and Y. Fang, "Driver fatigue state detection method based on multi-feature fusion," *Journal of Zhejiang University (Engineering Science)*, vol. 57, no. 7, pp. 1287-1296, 2023, doi: 10.3785/j.issn.1008-973X.2023.07.003.
- [15] H. Jia, Z. Xiao, and P. Ji, "Real-time fatigue driving detection system based on multi-module fusion," *Computers & Graphics*, vol. 108, pp. 22-33, 2022, doi: 10.1016/j.cag.2022.09.001.
- [16] H. Beles, T. Vesselenyi, A. Rus, T. Mitran, F. B. Scurt, and B. A. Tolea, "Driver Drowsiness Multi-Method Detection for Vehicles with Autonomous Driving Functions," *Sensors*, vol. 24, no. 5, Art. no. 1541, 2024, doi: 10.3390/s24051541.
- [17] G. Liu, K. Wu, W. Lan, and Y. Wu, "YOLO-FDCL: Improved YOLOv8 for Driver Fatigue Detection in Complex Lighting Conditions," *Sensors*, vol. 25, no. 15, Art. no. 4832, 2025, doi: 10.3390/s25154832.
- [18] C. Dhasarathan et al., "Tensor RT optimized driver drowsiness detection system using edge device," *Ain Shams Engineering Journal*, vol. 16, no. 10, Art. no. 103620, 2025, doi: 10.1016/j.asej.2025.103620.
- [19] A.-C. Phan, T.-N. Trieu, and T.-C. Phan, "Driver drowsiness detection and smart alerting using deep

- learning and IoT," Internet of Things, vol. 22, Art. no. 100705, 2023, doi: 10.1016/j.iot.2023.100705.
- [20] L. Zhou, S. Li, and Y. Wang, "Fatigue Detectic Early Warning System for Drivers Based on Learning," in 2023 IEEE 3rd International Confere: Data Science and Computer Application (ICD 2023), pp. 1348-1351, 10.1109/ICDSCA59871.2023.10392792.
- [21] R. Bonghi, "Jetson Stats," 2024. [Online]. Available: https://rnext.it/jetson_stats.
- [22] T. Danisman, I. M. Bilasco, C. Djeraba, and N. Ihaddadene, "Drowsy driver detection system using eye blink patterns," in 2010 International Conference on Machine and Web Intelligence, 2010, pp. 230-233, doi: 10.1109/ICMWI.2010.5648121.
- [23] H. A. G. Teive, R. P. Munhoz, C. H. F. Camargo, and O. Walusinski, "Yawning in neurology: a review," Arquivos de Neuro-Psiquiatria, vol. 76, no. 7, pp. 473-480, 2018, doi: 10.1590/0004-282X20180057.

AUTHORS



Andika Setiawan

is a lecturer in the Department of Informatics Engineering at Institut Teknologi Sumatera (ITERA), Lampung, Indonesia. He received the bachelor's degree from Universitas Sebelas Maret and the master's degree in Computer Science from Universitas Gadjah Mada, Indonesia. His research interests include artificial intelligence, computer vision, image processing, and machine learning.



Radhinka Bagaskara

is a lecturer in the Department of Informatics Engineering at Institut Teknologi Sumatera (ITERA), Lampung, Indonesia. He received the bachelor's degree from Telkom University and master's degrees from Institut Teknologi Bandung, Indonesia, and Kanazawa University, Japan. His research interests include artificial intelligence, computational science, machine learning, and high-performance computing.



M. Fida Raditya

received the bachelor's degree in Informatics Engineering from Institut Teknologi Sumatera (ITERA), Lampung, Indonesia. His research interests include computer vision, deep learning, object detection, edge computing, and intelligent transportation systems.



Mugi Praseptiawan

is a Lecturer at the Institut Teknologi Sumatera (ITERA), Indonesia, and a PhD Student at Universiti Teknikal Malaysia Melaka (UTeM), Malaysia, affiliated with the Human-Centered Computing and Information Systems Lab (HCC-ISL). His research interests are software quality, online learning, and decision support systems, in particular the quality assessment of digital learning systems and multi-criteria decision making approaches.