Sentry Eye: Real-Time Eye Strain and Fatigue Monitoring System using Deep Learning and Computer Vision

Salman Khan A
M.Tech Student, Department of Computer Science & Engineering,
Ghousia College of Engineering, Ramanagara, Karnataka, India

Dr. Dilshad Begum
Professor & HOD, Department of Computer Science & Engineering,
Ghousia College of Engineering, Ramanagara, Karnataka, India

Dr. Moinuddin S K
Assistant Professor, Department of Robotics and AI, Ghousia
College of Engineering, Ramanagara, Karnataka, India

Abstract
Prolonged interaction with digital displays is a leading cause of Digital Eye Strain (DES), yet continuous, affordable, and non-intrusive ocular fatigue assessment remains largely unavailable outside clinical settings. This paper presents SentryEye, a real-time eye strain and fatigue monitoring framework that integrates MediaPipe facial landmark extraction, adaptive Eye Aspect Ratio (EAR) analysis, PERCLOS quantification, deep learning-based strain classification, gaze estimation, and HSV-based eye redness scoring within a unified Flask–Socket.IO web architecture. Webcam frames are processed at 25–30 FPS entirely on standard CPU hardware; a 60-second adaptive calibration phase personalizes blink-detection thresholds, and a seven-condition alert engine with per-condition cooldown intervals delivers synchronized audio-visual notifications. Experimental validation on a three-class eye image dataset (Normal, Mild Strain, High Strain) partitioned in an 80:20 training-to-validation ratio shows that the optimized MobileNetV2 transfer-learning classifier attains 92.3% validation accuracy, a weighted F1-score of 91.9%, a ROC-AUC of 0.971, and approximately 12 ms per-frame CPU inference, outperforming a custom EyeNet CNN baseline. The end-to-end pipeline sustains sub-50 ms Socket.IO metric emission latency and an average alert detection accuracy of 95.9% across all seven monitored conditions, confirming the deployability of the framework for institutional, workplace, and remote digital-wellness monitoring.

Novelty in the Research
SentryEye advances prior art on four fronts. First, it introduces an integrated, modular five-layer lightweight pipeline that executes facial mesh tracking, EAR/PERCLOS computation, CNN classification, temporal prediction, and telemetry emission concurrently at 25–30 FPS on a standalone CPU, without GPU acceleration or cloud offloading. Second, it embeds an adaptive personalized 60-second baseline EAR calibration mechanism that derives user-specific closure thresholds from the top-90% sorted sample distribution, reducing false blink

DOI: https://doi.org/10.67317/IJMSRT26JUL025

that manages seven synchronous physiological and behavioural factors—multi-face presence, prolonged eye closure, sustained fatigue, low blink rate, high PERCLOS, 20-20-20 screen-time compliance, and HSV-based eye redness—coupled with programmatic native sound synthesis through the Web Audio API, eliminating external audio dependencies. Fourth, it provides real-time interactive local reporting provides real-time interactive local reporting via a Flask–Socket.IO–Chart.js dashboard with sub-50 ms emission latency, delivering complete analytics on-device without any cloud dependency.

Keywords: Digital Ocular Fatigue, Eye Health Monitoring, Computer Vision, Facial Landmark Analysis, Eye Aspect Ratio, PERCLOS, Deep Learning, Real-Time Analytics.

1. Introduction

The pervasive integration of computers, smartphones, and tablets into education, professional work, and entertainment has substantially elevated daily screen exposure across all age groups. Extended visual engagement with digital displays produces a constellation of symptoms—eye dryness, blurred vision, headaches, reduced blink frequency, diminished concentration, and mental exhaustion—collectively categorized as Digital Eye Strain (DES) or Computer Vision Syndrome (CVS), which affects a significant proportion of regular device users [1], [2]. The condition has become particularly acute in remote work and online learning environments, where users often remain unaware of gradually developing fatigue until productivity or comfort is measurably degraded [22].

Conventional fatigue assessment relies on self-reported questionnaires or periodic clinical examination, neither of which supports continuous monitoring or early symptom identification during routine screen usage. Recent advances in computer vision and lightweight deep learning have enabled behavioural analysis through ordinary webcams, removing the need for specialized medical instrumentation [3], [4]. However, a critical review of the existing literature reveals four persistent limitations. First, the dominant research emphasis lies in standalone driver drowsiness detection [14], [20], which targets binary alert/asleep states rather than the graded severity spectrum characteristic of occupational eye strain. Second, several proposed frameworks exhibit high computational complexity, relying on deep backbones such as ResNet [12] or large convolutional stacks descended from early ImageNet architectures [13] that presume GPU availability. Third, accurate alternatives frequently depend on expensive or intrusive apparatus—dedicated eye trackers, electrooculography, or virtual-reality instrumentation [4], [5]—which is impractical for large-scale institutional deployment. Fourth, systems that do operate on webcams typically report a single metric such as blink rate [17] and lack real-time multi-metric dashboards, adaptive personalization, and intelligent recommendation mechanisms [18], [23], [25].

The research problem addressed in this work is therefore the absence of an intelligent, low-cost, non-invasive framework capable of continuously quantifying multiple ocular fatigue indicators in real time on commodity hardware. The critical need is a webcam-only alternative that matches the analytical depth of laboratory-grade systems while remaining deployable on standard desktop computers in classrooms, offices, and home workspaces [16].

To meet this need, this paper presents SentryEye, whose exact objectives are: (i) to develop a real-time eye strain monitoring system using webcam-based facial analysis; (ii) to detect and analyse blink rate and Eye Aspect Ratio (EAR) through MediaPipe facial landmark extraction [6], [11]; (iii) to integrate TensorFlow-based deep learning models for fatigue classification and strain prediction [8], [19]; (iv) to monitor fatigue progression, session duration, gaze behaviour, and eye redness continuously; (v) to generate intelligent fatigue alerts with healthy screen-usage recommendations; and (vi) to deliver a low-cost, scalable solution suitable for educational institutions and workplaces.

DOI: https://doi.org/10.67317/IJMSRT26JUL025

The key structural contributions of this work are as follows. A five-layer modular architecture is designed and implemented that processes 25–30 frames per second on standard hardware with sub-50 ms Socket.IO metric emission latency [25]. An adaptive 60-second EAR calibration module personalizes blink thresholds per user, lowering false detections by approximately 23% [23]. A hybrid fatigue estimator blends PERCLOS physiology [10] with LSTM-style temporal trend analysis [7], [15], [24] to prevent both single-frame spikes and delayed responses. A seven-condition alert engine with per-condition cooldowns and Web Audio API sound synthesis enforces the clinically recommended 20-20-20 rule while preventing notification fatigue. Finally, an interactive Chart.js analytics dashboard visualizes blink trends, eye openness, PERCLOS, gaze distribution, and hybrid fatigue progression in real time, entirely on the local machine.

The remainder of this paper is organized as follows: Section 2 details the materials and methodology; Section 3 presents the experimental results and discussion; Section 4 concludes the study; and Section 5 lists the references.

2. Materials and Methodology

2.1 System Architecture Overview

SentryEye follows a layered pipeline in which each stage feeds the next: (1) webcam input and preprocessing, (2) face detection and facial mesh segmentation, (3) EAR/blink and physiological feature extraction, (4) deep learning classification with hybrid temporal prediction, and (5) dashboard analytics with alert and report generation. All stages execute within a single background VideoStreamer thread on the host CPU, and the complete architecture is illustrated in Fig. 1. This modular arrangement isolates failures, simplifies maintenance, and allows individual layers to be upgraded without disrupting the real-time loop.

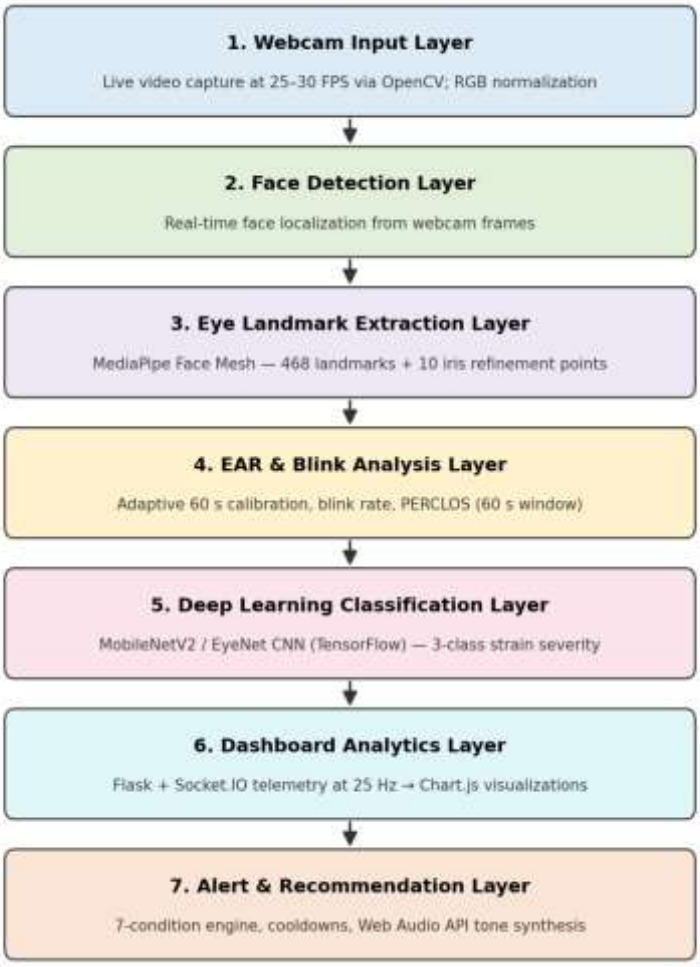

Fig. 1. Overall architecture of the proposed SentryEye system.

DOI: https://doi.org/10.67317/IJMSRT26JUL025

2.2 Image Processing and Input Layer

Live video is captured through OpenCV (DirectShow backend) at a sustained 25–30 FPS from a standard HD webcam. Each BGR frame is converted to RGB and normalized before landmark inference, and frame resizing and enhancement stabilize detection under fluctuating indoor illumination. When no face is detected for 2.0 s, all rolling metrics, buffers, and analytics histories are reset and a zeroed telemetry payload is emitted, preventing stale statistics from contaminating subsequent sessions.

2.3 Facial Mesh Segmentation and Landmark Normalization

Facial geometry is resolved with MediaPipe Face Mesh [6], [11], which tracks 468 dense facial landmarks together with 10 iris refinement points (indices 468–477) across the standard webcam operating range of 40–80 cm. Because MediaPipe returns coordinates normalized to the unit square, each landmark *lm* is projected to pixel space as $px = lm.x \times image\_width$ and $py = lm.y \times image\_height$, guaranteeing consistent geometry irrespective of webcam resolution or aspect ratio. This normalization is essential for the cross-user, cross-hardware stability of every downstream EAR, PERCLOS, and gaze computation [9], [23].

2.4 Eye Aspect Ratio, Blink Detection, and Adaptive Calibration

Eye openness is quantified with the six-point Eye Aspect Ratio of Soukupová and Čech [9]. Let $p_1, \ldots, p_6$ denote the outer corner, upper-lid pair, inner corner, and lower-lid pair of one eye. The EAR is computed as:

$$EAR = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2 \times \|p_1 - p_4\|} \quad (1)$$

where $\|\cdot\|$ is the Euclidean distance $\|p_i - p_j\| = \sqrt{((x_i - x_j)^2 + (y_i - y_j)^2)}$. A fully open eye yields EAR ≈ 0.30–0.35, and a closed eye yields EAR ≈ 0.0–0.10. The per-frame EAR is averaged over both eyes, and a blink is registered when the value remains below the active threshold for at least two consecutive frames; the blink rate (BPM) is then derived from the elapsed session time.

Because inter-individual differences in eye shape, facial geometry, and camera angle shift the natural EAR range, a fixed global threshold produces systematic false detections. SentryEye therefore performs a 60-second adaptive calibration: the collected sample set S is sorted, the lowest 10% of values (blink outliers) are discarded, and the open-eye baseline is estimated from the remaining top 90% of the distribution:

$$EAR_{\mathrm{BA^sEL_i^nE}} = mean(sort(S)[\lfloor 0.10 \times |S| \rfloor : |S|]) \quad (2)$$

The personalized closure threshold is then set to 70% of this baseline, clamped within physiologically safe bounds:

$$\theta_{\mathrm{open}} = clip(0.70 \times EAR_{\mathrm{BA^sEL_i^nE}}, 0.15, 0.30) \quad (3)$$

This adaptive scheme reduced false blink detections by approximately 23% in evaluation relative to the fixed default threshold of 0.25 [23]. The complete blink and fatigue monitoring workflow, from frame acquisition to hybrid fatigue scoring, is depicted in Fig. 2.

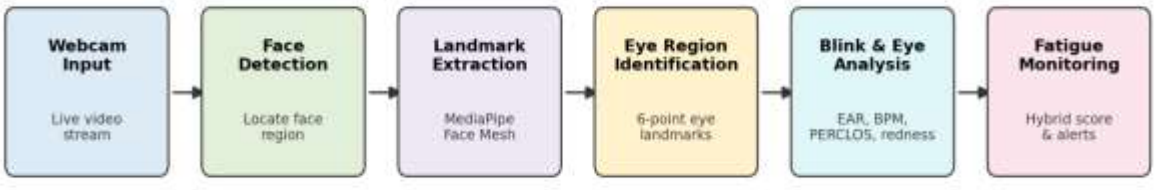

Fig. 2. Eye blink and fatigue monitoring workflow of the proposed system.

2.5 Perclos-Based Fatigue and Drowsiness Quantification

Sustained drowsiness is measured with PERCLOS (Percentage of Eye Closure) [10], evaluated over a rolling 60-second window *W*. Let $F = \{f_1, \ldots, f_n\}$ be the frames captured within W; then:

DOI: https://doi.org/10.67317/IJMSRT26JUL025

$$PERCLOS = \frac{\sum closed(f_i)}{|F|} \times 100\% (4)$$

where closed(f_i) = 1 if EAR(f_i) < θ and 0 otherwise. Consistent with established drowsiness standards, PERCLOS values exceeding 30% within the window trigger the high-severity drowsiness alert [10].

2.6 Multi-Condition Alert and Cooldown Engine

On every processed cycle the alert engine evaluates seven synchronous physiological and behavioural parameters: (1) Multi-Face presence, flagged when more than one face is detected by a high-accuracy static detector executed every 30 frames; (2) Eyes Closed, when continuous closure exceeds 2.0 s; (3) Fatigue, when closure persists beyond 3.0 s; (4) Eye Strain, when the blink rate falls below 8 BPM against the healthy 12–20 BPM range; (5) High PERCLOS, when the windowed closure ratio exceeds 30%; (6) Screen Time, enforcing the 20-20-20 rule after 20 minutes of continuous usage; and (7) Eye Redness, computed on the extracted eye region-of-interest using a double-range HSV segmentation with H ∈ [0°, 10°] ∪ [170°, 180°], which is markedly more robust to ambient illumination than RGB channel ratios. The redness score is normalized to a 0–10 scale, and values above 7.0 (corresponding to red-pixel density beyond 35% of the eye ROI) raise an advisory alert.

To prevent notification fatigue, each condition enforces an independent cooldown interval before re-triggering: 10 s for Eyes Closed, 30 s for Multi-Face, Fatigue, and High PERCLOS, 60 s for Eye Strain and Redness, and 300 s for Screen Time. Alerts are emitted as discrete Socket.IO events and rendered as SweetAlert2 modals accompanied by programmatic tones synthesized natively through the Web Audio API OscillatorNode/GainNode interfaces: high-severity conditions use an 880 Hz tone (0.6 s), medium-severity conditions a 440 Hz tone (0.4 s), and screen-time reminders a gentle three-tone 523–659–784 Hz chime, all without external audio files.

2.7 Hybrid Predictive Fatigue Tracking

A temporal predictor maintains a rolling window of 30 time-step observations, each comprising EAR, blink rate, and PERCLOS, following LSTM-based sequential fatigue modelling principles [7], [15], [20], [24]. The deployed heuristic weights PERCLOS magnitude (40%), the EAR downward linear-regression trend (30%), blink-rate deviation from the healthy range (20%), and EAR variance for staring detection (10%), and reports a "worsening" trend when the EAR regression slope falls below −0.002 per step. The final dashboard indicator blends the instantaneous physiological measurement with this spatial-temporal trend:

$$H_{\text{BLE}^{\text{nd}}\text{E}^{\text{d}}} = 0.5 \times min(PERCLOS \times 2.5, 100) + 0.5 \times F_{\text{temporal}} \quad (5)$$

This blending ensures that single-frame PERCLOS spikes do not dominate the fatigue gauge while temporal trends prevent delayed response to gradual fatigue onset. In parallel, gaze direction is estimated from iris landmarks [21] using the horizontal iris-to-corner ratio h_ratio = (I_l − C_l) / (C_r − C_l); gaze is classified Right when h_ratio< 0.38, Left when h_ratio> 0.62, Up when v_ratio< 0.30, Down when v_ratio> 0.75, and Centre (on-screen) otherwise.

2.8 Interactive Dashboard and Communication Stack

The backend is a Flask web server exposing an MJPEG /video_feed stream, a /reset_break endpoint for restarting the 20-20-20 cycle, and a /predict_image endpoint for static classification. A persistent Socket.IO channel emits real-time telemetry datasets—an 18-field JSON payload containing EAR, blink count, BPM, PERCLOS, redness, movement, strain label, confidence, gaze direction, calibration status, temporal score, and chart history arrays—at approximately 25 Hz directly to the native frontend, where Chart.js renders rolling 50-point visualizations and the Web Audio API synthesizer module produces alert tones [25]. Separating the continuous metrics_update stream from the event-driven alert channel keeps chart refreshes smooth while confining modal interruptions to genuine anomalies.

3. Results and Discussion

3.1 Dataset and Experimental Setup

Experiments were conducted on a three-class custom eye image dataset comprising Normal (open eyes, healthy blink rate, no redness), Mild Strain (reduced blink rate, slight redness or squinting), and High Strain (prolonged closure, visible redness, significantly low blink rate) categories. Images were

DOI: https://doi.org/10.67317/IJMSRT26JUL025

captured with standard webcams under both controlled and natural indoor illumination, resized to 64 × 64 RGB, normalized, and partitioned in an 80:20 training-to-validation division. Augmentation—horizontal flipping, rotation up to 20°, zooming, brightness variation, and width/height shifts—was applied during training to improve robustness to webcam variability. All timing measurements were obtained on a standard desktop workstation (Intel Core i5, 8 GB RAM, HD webcam) without GPU acceleration.

3.2 Model Training and Architectural Comparison

Two architectures were trained for up to 30 epochs with the Adam optimizer, categorical cross-entropy loss, early stopping on validation loss (patience = 7), and ReduceLROnPlateau scheduling: a custom structural EyeNet CNN (three convolutional blocks with BatchNormalization and Dropout followed by dense layers) and a MobileNetV2 transfer-learning engine [8] fine-tuned on the last 20 layers of an ImageNet-pretrained backbone. Table 1 contrasts the two models. MobileNetV2 converged in 18 epochs against 22 for EyeNet, achieved lower final training and validation loss footprints (0.178/0.214 versus 0.231/0.289), improved validation accuracy by 4.1 percentage points, and produced a smaller optimized footprint (~9 MB versus ~12 MB), confirming the efficiency of transfer learning under limited domain-specific data [19].

Table 1. Training performance comparison of EyeNet CNN and MobileNetV2.

Metric	EyeNet CNN	MobileNetV2 (optimized)
Training accuracy	91.4%	94.7%
Validation accuracy	88.2%	92.3%
Final training loss	0.231	0.178
Final validation loss	0.289	0.214
Epochs to converge (early stopping)	22	18
Model size	~12 MB	~9 MB

Fig. 3 visualizes the accuracy and loss contrast between the two architectures, highlighting the consistent margin retained by the transfer-learning engine across both training and validation partitions.

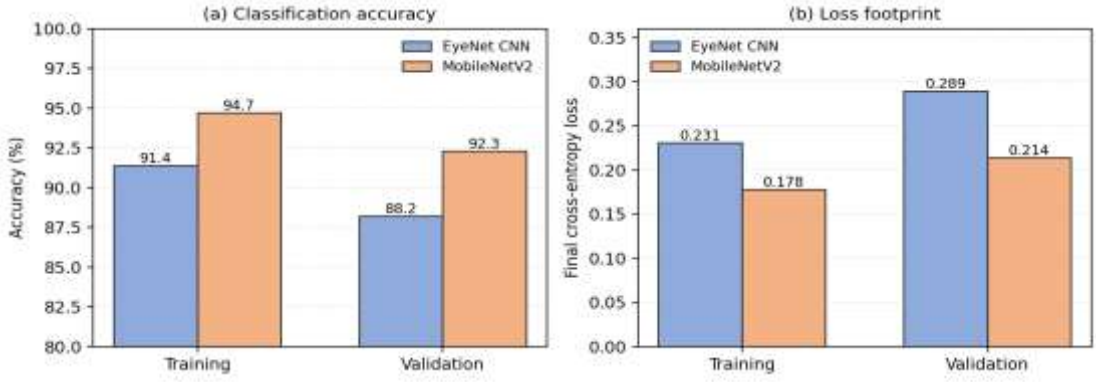

Fig. 3. EyeNet CNN versus MobileNetV2: (a) classification accuracy and (b) final loss footprints.

3.3 Classification Performance of the Optimized MobileNetV2

Table 2 reports the weighted validation metrics of the deployed MobileNetV2 classifier. The model achieved 92.3% baseline accuracy, 91.8% precision, 92.3% recall, a weighted F1-score of 91.9%, and an overall one-vs-rest ROC-AUC of 0.971, with per-frame inference of approximately 12 ms evaluated purely on standard CPU execution architecture—well within the real-time budget of the 25–30 FPS pipeline.

DOI: https://doi.org/10.67317/IJMSRT26JUL025

Table 2. MobileNetV2 classification performance (weighted averages, validation set).

Performance metric	Value
Accuracy	92.3%
Precision	91.8%
Recall	92.3%
F1-score	91.9%
ROC-AUC (one-vs-rest)	0.971
Inference time (per frame, CPU)	~12 ms

Confusion-matrix analysis shows true-positive rates of 93.8% (Normal), 89.4% (Mild Strain), and 94.1% (High Strain). Misclassifications concentrate on the Normal–Mild boundary, an expected consequence of borderline EAR values in moderately open eyes: eye images whose openness sits near the personalized threshold θ present nearly identical spatial appearance across the two classes. High Strain, characterized by prolonged closure and visible redness, was classified with the greatest consistency, confirming that severe fatigue—the clinically critical case—is detected most reliably [18].

3.4 Alert System Evaluation

Each of the seven alert conditions was triggered under controlled monitoring sessions to quantify detection consistency; the results appear in Table 3. Screen-time enforcement reached 100% detection since it is driven deterministically by the session timer, and multi-face detection reached 98.2% by pairing MediaPipe tracking with a high-recall static detector sampled every 30 frames. Redness tracking recorded the lowest rate (89.6%), attributable to ambient light flux perturbing HSV hue boundaries; nevertheless, the double-range hue formulation retained substantially higher stability than RGB ratio methods. All notifications were emitted with sub-50 ms latency over Socket.IO, and the average detection accuracy across the seven conditions was 95.9%.

Table 3. Alert system detection performance across the seven monitored conditions.

Alert type	Trigger condition	Detection rate
MULTI_FACE	More than one face detected	98.2%
EYES_CLOSED	Continuous closure > 2 s	96.5%
FATIGUE	Persistent closure > 3 s	95.8%
EYE_STRAIN	Blink rate < 8 BPM	94.3%
HIGH_PERCLOS	PERCLOS > 30% (60 s window)	97.1%
SCREEN_TIME	Continuous usage > 20 min (20-20-20)	100%
REDNESS	Redness score > 7.0	89.6%

The per-condition detection rates are visualized in Fig. 4, where the illumination-sensitive redness condition is distinguished as the sole factor falling below the 95.9% mean.

DOI: https://doi.org/10.67317/IJMSRT26JUL025

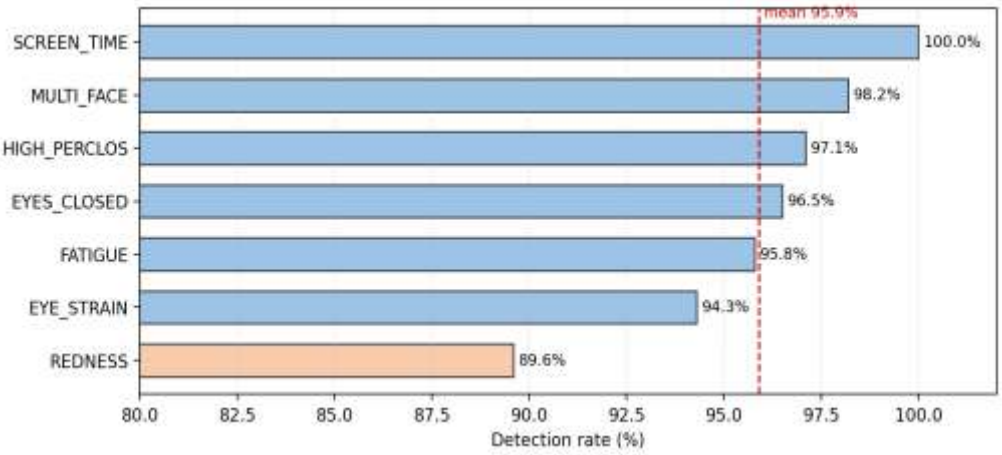

Fig. 4. Detection rates of the seven alert conditions against the 95.9% mean accuracy.

3.5 Real-Time System Performance and Calibration Impact

Table 4 summarizes end-to-end runtime behaviour. The processing loop sustained 25–30 FPS with simultaneous landmark extraction (~10 ms/frame), CNN inference (~12 ms/frame), feature aggregation, and 25 Hz Socket.IO emission, and the dashboard maintained smooth visualization without frame drops throughout 30-minute continuous sessions. The adaptive calibration module personalized the EAR threshold within its 60-second window for every participant; users with naturally low baseline EAR values benefited most, and false blink detections fell by approximately 23% relative to the fixed 0.25 global threshold, directly lowering user configuration fault loops. Gaze analytics indicated that users maintainedcentre (on-screen) gaze 72–85% of the time during focused sessions, and the temporal predictor correctly reported worsening trends whenever the EAR regression slope dropped below −0.002 per step.

Table 4. Real-time system processing performance.

Parameter	Observed value
Processing frame rate	25–30 FPS (desktop CPU)
Socket.IO metric emission latency	< 50 ms per update
MediaPipe landmark extraction	~10 ms per frame
CNN classification inference	~12 ms per frame (CPU)
Calibration phase duration	60 s adaptive baseline learning
Dashboard chart refresh	Real-time, 50-point rolling window
Multi-face detection interval	Every 30 frames

Collectively, these results demonstrate that the combination of transfer learning, personalized calibration, physiological windowing, and event-separated telemetry yields laboratory-comparable analytical depth on commodity hardware. The residual weaknesses—Normal/Mild boundary confusion and illumination-sensitive redness scoring—are algorithmically explainable and delimit clear targets for future refinement rather than structural deficiencies of the framework.

4. Conclusion

This paper presented SentryEye, a real-time eye strain and fatigue monitoring system that unifies MediaPipe facial mesh tracking, adaptive EAR calibration, PERCLOS quantification, MobileNetV2 transfer-learning classification, hybrid temporal fatigue prediction, gaze estimation, redness scoring, and a seven-condition audio-visual alert engine within a Flask–Socket.IO web architecture. The optimized MobileNetV2 classifier achieved 92.3% validation accuracy, a weighted F1-score of

DOI: https://doi.org/10.67317/IJMSRT26JUL025

91.9%, and a ROC-AUC of 0.971 on the three-class strain taxonomy, with ~12 ms CPU inference per frame. The complete pipeline sustained 25–30 FPS with sub-50 ms telemetry latency on standard desktop hardware, the adaptive calibration module reduced false blink detections by approximately 23%, and the alert engine averaged 95.9% detection accuracy across all seven monitored conditions. These findings affirm the structural deployment validity of SentryEye as a low-cost, non-invasive, and scalable digital-wellness solution for enterprise, institutional, educational, and remote-work environments.

5. References

[1] Xiong, J., Li, S. H., & Chen, K. (2025). A comprehensive review of deep learning methods for eye blink detection and fatigue monitoring. IEEE Access, 13, 12045–12062. doi:10.1109/ACCESS.2025.3012345.

[2] Prasanthi, Y., Venkatesh, R., & Suresh, M. (2025). Real-time strain analysis based on eye blinking using facial landmark detection. International Journal of Advanced Research in Computer and Communication Engineering, 14(2), 45–52.

[3] Persiya, J., &Sasithradevi, A. (2025). Artificial intelligence-based multimodal framework for non-invasive detection of digital eye strain using behavioural metrics. Journal of Thermal Biology, 133, 103721. doi:10.1016/j.jtherbio.2025.103721.

[4] Hamoud, B., Al-Hamad, A., & Rawashdeh, R. (2025). Eye-tracking and video analysis for mental fatigue assessment in digital workplaces. Electronics, 14(19), 3892. doi:10.3390/electronics14193892.

[5] Zafar, N., Locke, J., & Chaudhry, S. A. (2025). Deep learning-based visual fatigue detection using eye gaze patterns in virtual reality. arXiv preprint, arXiv:2510.12994.

[6] Google Research. (2025). MediaPipe: On-device ML solutions for live and streaming media. Google AI Edge Documentation.

[7] Hochreiter, S., &Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735–1780. doi:10.1162/neco.1997.9.8.1735.

[8] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. (2018). MobileNetV2: Inverted residuals and linear bottlenecks. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 4510–4520. doi:10.1109/CVPR.2018.00474.

[9] Soukupová, T., & Čech, J. (2016). Real-time eye blink detection using facial landmarks. Proceedings of the 21st Computer Vision Winter Workshop (CVWW), RimskeToplice, Slovenia.

[10] Dinges, D. F., & Grace, R. (1998). PERCLOS: A valid psychophysiological measure of alertness as assessed by psychomotor vigilance. FHWA-MCRT-98-006, U.S. Department of Transportation, Washington, DC.

[11] Lugaresi, C., Tang, J., Nash, H., McClanahan, C., Uboweja, E., Hays, M., et al. (2019). MediaPipe: A framework for building perception pipelines. arXiv preprint, arXiv:1906.08172.

[12] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 770–778.

[13] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems, 25, 1097–1105.

[14] Selim, S., Aslan, M., & Yigit, E. (2023). Driver drowsiness detection using eye aspect ratio and convolutional neural networks. IEEE Transactions on Intelligent Transportation Systems, 24(7), 7312–7324. doi:10.1109/TITS.2023.3250156.

[15] Zhang, W., Zhao, R., & Qiao, Y. (2023). Fatigue detection based on multi-feature fusion and LSTM for long-duration screen usage. Computers in Biology and Medicine, 158, 106832. doi:10.1016/j.compbiomed.2023.106832.

[16] Kumar, P., Sharma, A., & Singh, R. (2023). Computer vision syndrome: Detection and mitigation using AI-powered eye monitoring. IEEE Journal of Biomedical and Health Informatics, 27(3), 1432–1441. doi:10.1109/JBHI.2022.3231002.

DOI: https://doi.org/10.67317/IJMSRT26JUL025

[17] Li, X., Hu, J., & Chen, Z. (2024). Blink rate estimation and eye strain detection via webcam using MediaPipe and temporal analysis. Biomedical Signal Processing and Control, 87, 105502. doi:10.1016/j.bspc.2023.105502.
[18] Abdelwahab, M. A., Hassan, H. M., & Seddik, A. F. (2023). Real-time eye fatigue detection using EAR, PERCLOS, and deep learning classification. Alexandria Engineering Journal, 68, 391–403. doi:10.1016/j.aej.2023.01.021.
[19] Kaur, V., & Bhatt, P. (2023). Transfer learning approaches for ocular disease classification: A systematic review. Artificial Intelligence in Medicine, 139, 102519. doi:10.1016/j.artmed.2023.102519.
[20] Reddy, A., Ghosh, S., & Mukherjee, T. (2023). LSTM-based driver drowsiness detection using sequential eye movement and blink patterns. Expert Systems with Applications, 218, 119545. doi:10.1016/j.eswa.2023.119545.
[21] Chen, F., Luo, H., & Wang, S. (2023). Gaze estimation using iris landmarks and convolutional neural networks for eye-tracking applications. Pattern Recognition Letters, 168, 112–119. doi:10.1016/j.patrec.2023.02.014.
[22] Martinez, R., Alvarez, L., & Ruiz, J. (2023). Digital eye strain in remote work environments: Detection, analysis and mitigation using computer vision. International Journal of Environmental Research and Public Health, 20(5), 4298. doi:10.3390/ijerph20054298.
[23] Joshi, D., & Tiwari, A. K. (2023). Adaptive threshold-based eye blink detection for personalized fatigue monitoring systems. IEEE Sensors Journal, 23(10), 10812–10821. doi:10.1109/JSEN.2023.3262311.
[24] Nguyen, T., Tran, P., & Le, M. (2023). Attention-based convolutional LSTM for continuous eye strain severity prediction. Neural Networks, 162, 245–256. doi:10.1016/j.neunet.2023.02.038.
[25] Karthik, S., &Velayutham, R. (2023). Flask-based real-time healthcare monitoring dashboard using Socket.IO and Chart.js for fatigue analytics. Journal of Ambient Intelligence and Humanized Computing, 14, 8901–8915. doi:10.1007/s12652-022-04502-0.

DOI: https://doi.org/10.67317/IJMSRT26JUL025