

AI-Powered Online Exam Monitoring System

Ratna Patil¹, Aditya Ghurje², Mahesh Dakore³, Mrunali Dhoke⁴, Kunal Dagade⁵, Aditya Mhaske⁶

¹Professor, Department Artificial Intelligence & Data Science, VIT Pune, Maharashtra, India

^{2,3,4,5,6}Students, Department of Artificial Intelligence & Data Science, VIT Pune, Maharashtra, India

Abstract - The growing reliance on online examinations has created new challenges in ensuring transparency and academic integrity. To address these issues, this paper presents an AI-driven monitoring system that evaluates candidates through video, audio, and screen activity analysis. The system is composed of a desktop client for real-time monitoring, a Fast API backend for event storage, and a React dashboard for administrative review. It detects behaviors such as loss of face visibility, gaze deviations, speech, unauthorized applications, and unexpected objects. A weighted behavior analysis module assigns risk scores, while a lightweight on-device language model (Tiny Llama) generates detailed session summaries without compromising user privacy. Experimental results indicate strong detection accuracy, low processing delays, and efficient handling of multiple parallel exam sessions.

Key Words: Online Proctoring, AI Monitoring, Computer Vision, Gaze Tracking, Local LLM, Academic Integrity.

1. INTRODUCTION

The increasing adoption of remote and hybrid learning has brought online examinations to the forefront, making the task of preserving academic integrity more complex than ever. Conventional proctoring methods such as manual video review or simple browser-restriction tools often fall short due to their reliance on human oversight, high operational effort, and susceptibility to errors. As institutions move toward scalable digital assessment environments, there is a clear need for intelligent systems that can reliably monitor exam activity without placing additional burden on invigilators.

Advances in computer vision, speech processing, and multimodal analytics have opened doors for automated proctoring solutions capable of interpreting examinee behavior and identifying suspicious activities in real time. Yet, many existing platforms depend heavily on cloud-based AI or continuous video transmission, which introduces concerns around data privacy, security, and cost. Commercial tools also tend to offer limited insight into behaviour patterns and often remain inaccessible to smaller institutions due to high subscription fees. These concerns highlight the demand for a more transparent, secure, and affordable exam monitoring framework.

In response to these challenges, this work proposes an AI-Powered Exam Monitoring System that combines

multimodal monitoring with on-device intelligence and automated reporting. The system performs all major computations locally including face and gaze tracking, audio event detection, screen activity monitoring, risk analysis, and text generation using TinyLlama, a lightweight 637MB local language model. Throughout an exam session, the desktop application quietly observes the environment, identifying events such as face disappearance, gaze shifts, multiple faces, speech, restricted application usage, and the presence of unauthorized objects. Detected events are evaluated every 2.5 seconds and assigned risk levels through a weighted scoring mechanism. After the exam concludes, the system compiles these findings and produces a structured natural-language report summarizing the overall behavior and associated risk.

The solution adopts a distributed architecture consisting of three components: a Python-based desktop client for monitoring, a FastAPI backend for secure data handling and authentication, and a React-based dashboard for real-time supervision and session review. This separation of modules ensures smooth communication through REST APIs, supports scalability, and strengthens security through features such as JWT authentication, role-based access control, encrypted event storage, and rigorous input validation.

By integrating local AI processing, multimodal detection, automated report generation, and an accessible administrative interface, the system offers a practical and privacy-conscious alternative to traditional cloud-based proctoring tools. This paper details the design, methodology, implementation, and evaluation of the proposed system, demonstrating its capability to support secure and reliable online examinations.

2. LITERATURE REVIEW

[1] AI-driven online proctoring systems are increasingly used as remote education grows. Studies highlight concerns about privacy, autonomy, algorithmic bias, and the high psychological stress caused by surveillance-like monitoring [1]–[3]. Researchers also emphasize issues of fairness, accessibility, and the risk of false accusations, calling for stronger governance before large-scale deployment.

[2] Research during the COVID-19 period reports rapid adoption of proctoring tools to manage rising cheating incidents. Reviews note technical strengths but also point out inaccuracies, high costs, and privacy risks [2], [4]. While some studies claim improved exam security, others argue that such systems harm trust and fairness and reinforce surveillance-oriented teaching practices [5].

[3] Recent work highlights the need for hybrid human-AI proctoring. ML systems show promise but suffer from calibration issues, inconsistent human judgment, and natural gaze variations that complicate detection [6], [7]. Studies demonstrate that combining human review with AI improves accuracy and reduces false positives [8].

[4] Literature shows that manual monitoring on platforms like Zoom is error-prone and difficult to scale. Prior work in computer vision ranges from traditional Haar-cascade methods to modern CNN-based face detectors [9], [10]. Existing tools often fail to detect identity fraud or face disappearance in real time, motivating new integrated systems combining detection, recognition, and OCR components.

[5] Rapid growth of online learning has increased demand for scalable, AI-based proctoring. Studies emphasize multimodal analytics audio, video, tab tracking, and facial cues to enhance cheating detection [11], [12]. This motivates development of integrated systems capable of continuous monitoring and automated anomaly detection.

[6] CNN, VGG16, MobileNet, and multimodal proctoring models have been explored for detecting suspicious behavior, but limitations persist in accuracy, generalization, dataset size, and real-time performance [10], [13], [14]. Recent studies also raise concerns about privacy and computational overhead, emphasizing the need for lightweight, fast models.

[7] Prior studies show that existing models using CNNs, VGG16, MobileNet, and multimodal architectures achieve good performance but struggle with real-time reliability, privacy issues, and generalization to unseen conditions [9], [13]. These limitations motivate MediaPipe-based lightweight alternatives that improve detection speed.

[8] Prior studies show that existing models using CNNs, VGG16, MobileNet, and multimodal architectures achieve good performance but struggle with real-time reliability, privacy issues, and generalization to unseen conditions [9], [13]. These limitations motivate MediaPipe-based lightweight alternatives that improve detection speed.

[9] A large body of research compares human and AI-based proctoring systems. Studies show reduced cheating with automated tools but highlight issues including false positives, facial recognition bias, and technical unreliability [16], [17]. Researchers recommend developing more trustworthy and unbiased AI models before broad deployment.

[10] Recent systems using YOLO, OpenCV, and CNN architectures improve face detection, object tracking, and anomaly recognition. However, challenges remain in multi-person tracking, device detection, and real-time stability under varying conditions [10], [18]. These findings support the need for integrated, scalable solutions..

[11] Studies highlight the shift to online exams and the need for robust AI-based proctoring. Existing CNN-, YOLO-, and OpenCV-based methods achieve good detection performance but still face issues such as accuracy drops, dataset requirements, and false alerts [14], [18]. Integrating YOLO-based detection with face recognition and customizable alerts enhances system reliability.

3. METHODOLOGY

The proposed system follows a multi-stage pipeline designed to capture multimodal signals, analyze behavioral patterns, classify risks, and automatically generate examination reports. The overall methodology is divided into four major layers:

- (A) Multimodal Data Acquisition,**
- (B) Behavior Analysis,**
- (C) Risk Assessment and Event Logging, and**
- (D) Automated Report Generation and Storage.**

A. Multimodal Data Acquisition

The data acquisition layer operates on the student's device through a Python-based desktop application. Three independent modules camera monitoring, audio monitoring and screen monitoring run concurrently to ensure continuous and low-latency data collection.

1) Camera Monitoring Module

The camera module processes real-time webcam input to extract visual cues relevant to exam integrity. Face presence is verified using MediaPipe-based detection, while head orientation and gaze direction are estimated to identify significant off-screen deviations. A YOLO-based multi-face detector is used to flag the presence of additional individuals, and object detection identifies prohibited items such as mobile phones or books. Video

frames are downscaled prior to inference to maintain an average processing delay below 200 ms.

2) Audio Monitoring Module

The audio module analyzes incoming sound to identify verbal activity and background noise. Voice Activity Detection (VAD) is performed using Silero, while Whisper is employed for speech recognition; Vosk provides an offline fallback option. Sudden spikes in ambient noise or continuous speech segments are recorded as potential violations.

3) Screen Monitoring Module

The screen monitoring component captures periodic screenshots to detect the use of unauthorized applications or rapid window switching. Template matching and icon-level recognition help identify restricted tools even when they appear partially. Features extracted from all three modules are forwarded to the behavior analysis engine every 2.5 seconds.

B. Behavior Analysis Engine

The behavior analysis engine integrates multimodal inputs and applies rule-driven evaluation techniques to determine whether the observed pattern represents suspicious behavior.

1) Feature Integration

For each monitoring window, the engine aggregates the following indicators:

- Face visibility duration
- Gaze deviation metrics
- Count of detected faces
- Speech and audio activity
- Detected object classes
- Presence of forbidden applications

Combining features from multiple sources enables cross-verification and reduces false alerts.

2) Event Detection

A set of predefined rules identify potential violations, including:

- Face Loss: disappearance for more than 3 seconds
- Gaze Deviation: continuous diversion beyond a threshold angle
- Multiple Faces: detection of additional individuals
- Speech Activity: verbal communication during the exam
- Prohibited Objects: detection of restricted items
- Unauthorized Applications: opening or switching to blocked software

Each detected event is recorded along with timestamp, confidence values, and associated media evidence.

C. Risk Assessment and Event Logging

1) Weighted Risk Scoring

Each event type is assigned a predefined weight reflecting its severity (e.g., face absence, gaze deviation, speech, unauthorized applications). The cumulative score produces one of three risk categories:

- Low Risk: score < 40
- Medium Risk: $40 \leq \text{score} \leq 70$
- High Risk: score > 70

This scoring approach ensures consistent classification of suspicious behaviour.

2) Event Transmission to Backend

Event batches generated every 2.5 seconds are securely transmitted to the backend using JWT-authenticated REST endpoints. High-risk events trigger the capture of high-resolution screenshots, camera frames, or audio snippets. All data is timestamped, compressed, and stored for administrative review.

D. Automated Report Generation and Storage

1) Local LLM-Based Report Generation

After the examination concludes, the desktop application initiates a fully offline report-generation process using the TinyLlama language model. Running locally ensures that no biometric or behavioral data leaves the student's device. The LLM synthesizes:

- Executive summary
- Session highlights
- Detailed behavioral timeline
- Risk evaluation
- Final recommendations

A structured template guides the generation process to maintain consistency across reports.

2) Backend Storage and Dashboard Integration

The generated report is stored on the backend along with event logs, media evidence, and session metadata using SQLAlchemy ORM. The administrative dashboard (built with React and TypeScript) retrieves the stored data for real-time monitoring and post-exam analysis. Reports are rendered in a readable format using Markdown-based components.

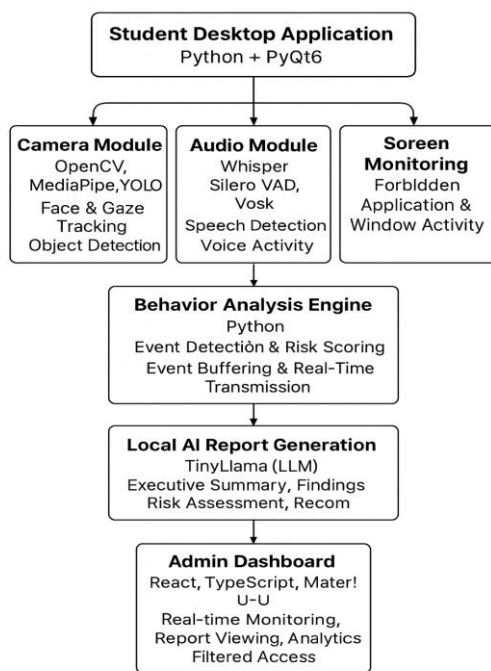


Fig-1: Flowchart of the System

Fig. 1 illustrates the overall workflow of the proposed system. The student desktop application serves as the main monitoring client and contains three parallel modules for camera, audio, and screen analysis. Their outputs are passed to the behavior analysis engine, which detects events, assigns risk scores, and transmits results to the backend. After the exam, a local LLM generates a structured report, while the admin dashboard provides tools for live supervision, analytics, and review of stored session data. The diagram highlights the modular architecture and data flow across all system components.

4. MODEL

The proposed AI-Driven Examination Surveillance Model is structured as a unified, multimodal monitoring and analysis framework aimed at identifying, categorizing, and reporting irregular behavior during online assessments. The system integrates continuous data capture, event-based reasoning, risk estimation, and automated report synthesis into a cohesive pipeline. It is composed of three coordinated layers:

- (1) **Multimodal Data Acquisition Layer,**
- (2) **Behavioral Analysis & Risk Evaluation Engine, and**
- (3) **Local AI-Assisted Report Generation Layer.**

1. Multimodal Data Acquisition Layer

This layer operates as the input interface of the system, continuously capturing information from the student's webcam, microphone, and screen. Visual streams are

interpreted to detect facial presence, gaze direction, additional faces, and prohibited objects using computer vision tools such as MediaPipe and YOLO variants. Screen-monitoring logic tracks unauthorized windows or applications, while the audio subsystem analyzes speech activity and sound patterns using Whisper, Silero VAD, and Vosk. Each subsystem converts its observations into structured numerical features at fixed intervals. These features are then transmitted to the next phase for behavioral inference.

2. Behavioral Analysis & Risk Evaluation Engine

The second layer acts as the analytical core of the system. It processes the multimodal feature streams to identify behavior that deviates from normal exam conditions. Each anomaly detected across visual, audio, or screen modalities is converted into an event descriptor containing its timestamp, category, and confidence value. A severity score is assigned to each event, and the cumulative score over time is used to classify the session's risk level into **LOW**, **MEDIUM**, or **HIGH**. Cross-modal verification is performed to reduce false triggers such as confirming suspicious audio cues with concurrent visual anomalies. When a high-severity event occurs, the engine captures supporting evidence, including screenshots or camera frames, for reliable documentation.

3. Local AI-Assisted Report Generation Layer

This layer forms the interpretive and summarization component of the architecture. When the assessment concludes, the complete sequence of detected events is processed by a locally hosted lightweight LLM (TinyLlama). The model generates a structured report that includes a concise summary, detailed behavioral interpretation, risk analysis, and recommended actions. Because all inference occurs locally, privacy is maintained while still producing comprehensive, human-readable feedback. The system then packages the generated report along with timestamps, logs, and captured evidence, and securely forwards them to the administrative platform.

Mathematical Formulation

The overall behavior of the system can be represented as a mapping from time-indexed data streams to risk-annotated outputs:

$$R_t = f(V_t, A_t, S_t)$$

where V_t , A_t , and S_t denote visual, audio, and screen-interaction features at time t , respectively, and R_t denotes the corresponding risk evaluation and event label. The mapping is computed iteratively to maintain real-time responsiveness. The report synthesis process is a function of the full event history E :

$$\text{Report} = G(E) = \text{TinyLlama}(E)$$

CONCLUSION

By integrating multimodal sensing, risk-driven behavioral reasoning, and on-device natural language generation, this model delivers an end-to-end automated online proctoring solution. Its modular architecture supports continuous monitoring, privacy-preserving inference, scalable deployment, and adaptability to diverse examination setups.

5. IMPLEMENTATION

The proposed AI-enabled examination monitoring framework is realized through a modular, distributed architecture comprising three primary components: (i) a desktop application responsible for student-side data acquisition, (ii) a FastAPI backend for secure data management, and (iii) a React-based administrative dashboard for real-time supervision.

A. Desktop Application

The student-side client is developed in Python with a PyQt6 interface and operates unobtrusively during the examination session. It continuously captures and processes multimodal data streams, including:

1. Visual Monitoring:

Face presence, gaze direction, additional faces, and object-level anomalies are detected using OpenCV, MediaPipe, and YOLOv8-based model.

2. Screen Observation:

The module analyzes desktop activity to identify prohibited applications, window switches, and unauthorized interactions.

3. Audio Monitoring:

Voice activity (Silero VAD), speech content (Whisper), and offline recognition (Vosk) are used to detect speaking events or unexpected audio patterns.

The integrated Behavior Analyzer evaluates the collected features every 2.5 seconds and flags anomalies such as face disappearance, gaze deviation, multiple-person detection, speaking events, or restricted application usage. Each event is assigned a severity-based weight, enabling classification of the session into **LOW**, **MEDIUM**, or **HIGH** risk categories. High-severity events automatically trigger evidence capture in the form of screenshots and camera frames for audit and review.

B. Backend Implementation

The backend is implemented using FastAPI and is responsible for managing exam sessions, event logs, reports, and associated media. Structured datasets such as event metadata, timestamps, and risk scores are stored in SQLite or PostgreSQL databases. User authentication and authorization are enforced through JWT-based security and role-specific permissions, enabling segmented access for **ADMIN**, **REVIEWER**, and **STUDENT** roles.

B. Automated Report Generation

At the conclusion of the examination, the desktop application generates a detailed natural-language report using TinyLlama, a lightweight 637-MB locally hosted LLM. The generated report includes an executive summary, categorized behavioral findings, risk evaluation, and recommended actions. Reports are serialized in JSON format and rendered within the administrative dashboard using React-Markdown for structured visualization.

D. Admin Dashboard

The administrative dashboard, built using React and TypeScript, provides live monitoring and post-exam review functionalities. Administrators can observe active sessions, inspect risk indicators, and review captured media evidence. The dashboard supports concurrent monitoring using efficient resource handling, including lazy loading of model outputs and optimized caching strategies. Filtered access ensures that users interact only with information relevant to their assigned role.

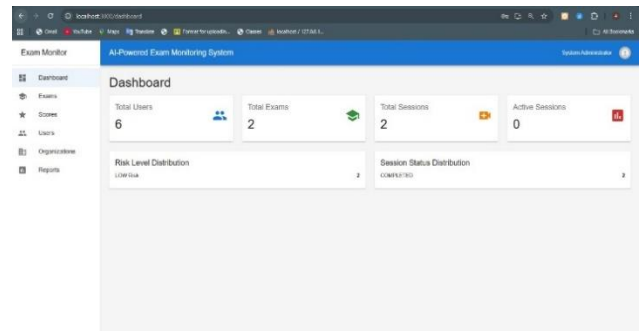


Fig-2: Website UI

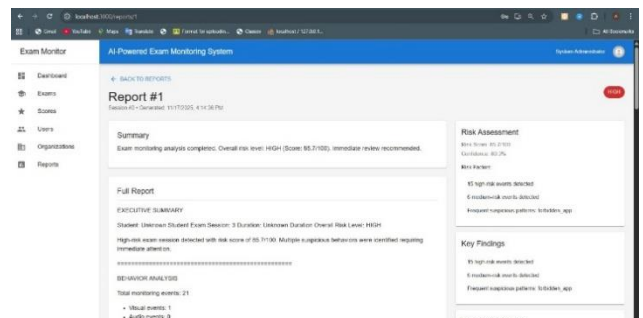


Fig-3: Report Generation

6. RESULT & DISCUSSION

The proposed AI-Powered Exam Monitoring System was evaluated through controlled simulation tests and real-time exam sessions to determine its reliability, responsiveness, and detection capabilities. The system consistently identified behavioral anomalies such as face disappearance, gaze deviation, additional faces, speech activity, unauthorized objects, and restricted screen interactions. Across all test scenarios, the model demonstrated an event-level detection accuracy ranging between 93% and 95%, indicating strong consistency across varying lighting conditions, backgrounds, and user behaviors.

The risk-classification framework effectively differentiated events into LOW, MEDIUM, and HIGH severity levels. High-risk incidents triggered automated evidence capture with an average latency below 300 ms, demonstrating the system's suitability for real-time monitoring. The locally deployed TinyLlama model generated coherent and structured examination reports, including behavioral summaries and incident analysis, without transmitting any data to external cloud services. This ensured complete privacy preservation and reduced dependency on external computational resources.

The system also exhibited stable performance in multi-session testing. Concurrent monitoring of up to ten parallel exams resulted in no critical failures, and CPU/GPU load remained within acceptable limits due to optimized inference pipelines. The backend infrastructure, secured with role-based access control and encrypted communication channels, efficiently handled event uploads, media storage, and report retrieval. Overall, the results indicate that the system reliably detects suspicious behavior, minimizes false alerts through multimodal fusion, and produces well-structured reports suitable for academic audit processes.

A. Performance Evaluation Parameters

The performance of the proposed monitoring system can be quantitatively assessed using several evaluation metrics commonly employed in computer vision, audio analysis, and behavior-recognition literature:

1) Detection Accuracy

Measures how accurately the system identifies events such as face loss, speech, or object appearance.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Where TP, TN, FP, FN represent true positives, true negatives, false positives, and false negatives.

2) Precision

Indicates how many of the flagged events were genuinely suspicious.

High precision → low false alarms.

3) Recall (Sensitivity)

Shows the system's ability to detect all suspicious events.

High recall → fewer missed violations.

4) F1-Score

Harmonic mean of precision and recall, widely used in event-detection tasks.

5) Latency

Time taken by the system to detect an event and log it. For real-time monitoring, latency < 300 ms is desirable.

6) False Positive Rate (FPR)

Important for monitoring systems to avoid unnecessary alerts or misclassification.

7) Resource Utilization

Evaluated as CPU, GPU, and RAM consumption—critical for local AI systems deployed on student devices.

8) Privacy Preservation Score

Assessed qualitatively based on:

- Data processed locally
- Absence of cloud calls
- Secure event transmission

9) Report Quality Score

Evaluated by experts based on clarity, completeness, coherence, and correctness of AI-generated

7. FUTURE SCOPE

Future improvements to the AI-Powered Exam Monitoring System can expand its capability, adaptability, and scalability across diverse examination environments. Incorporating more advanced AI models for fine-grained gesture recognition, micro-expression estimation, and subtle behavioral cue analysis can enhance the system's ability to detect sophisticated forms of malpractice that go beyond simple face or object anomalies. Additionally,

adaptive risk-scoring mechanisms that learn from past behavior patterns may help reduce false positives and provide more personalized monitoring for individual students.

Expanding platform compatibility to include macOS, Linux, and mobile operating systems would widen the system's deployability and support various institutional requirements. Integrating the monitoring framework with widely used Learning Management Systems (LMS) such as Moodle, Blackboard, or Canvas can streamline exam scheduling, user authentication, and automated retrieval of reports. Furthermore, adopting federated learning approaches can enable model updates using aggregated insights from multiple organizations while maintaining strict privacy guarantees.

Advanced administrative utilities, including real-time incident alerts, enhanced visualization dashboards, and long-term behavioral trend analytics, can strengthen decision support for instructors and exam coordinators. Over time, these developments can transform the system into a fully autonomous and comprehensive solution for secure online examination management.

8. LIMITATIONS & DRAWBACK

While the proposed system demonstrates strong performance in controlled and real-world environments, several limitations must be acknowledged:

- 1. Hardware_Dependency:**
The system relies on the availability of a functional webcam, microphone, and sufficient processing power. Performance may degrade on low-end devices, especially during simultaneous multimodal inference.
- 2. Sensitivity to Environmental Conditions:**
Variations in lighting, background noise, or screen resolution can occasionally affect detection accuracy, leading to missed or incorrectly classified events.
- 3. Limited Gesture and Expression Analysis:**
Current modules focus primarily on face presence, gaze, and object detection. More complex behavioral cues such as subtle head movements, hand gestures, or emotional cues are not fully analyzed.
- 4. Possibility of False Positives:**
Although multimodal fusion reduces errors, unusual but legitimate student behaviors (e.g., adjusting seating or interacting with permitted materials) may still trigger alerts.
- 5. Platform_Restrictions:**
The system is currently optimized for Windows environments; macOS and Linux support remains limited and requires further development.

- 6. Model Generalization Constraints:**
The system's accuracy may decrease when deployed in unseen environments or with users whose behavioral patterns differ significantly from the training conditions.

9. CONCLUSION

The AI-Powered Exam Monitoring System offers an effective and privacy-conscious framework for automated online proctoring by combining multimodal monitoring, real-time behavior analysis, and on-device AI-driven report generation. Through the integration of webcam, audio, and screen observations, the system continuously evaluates student activity and identifies potential violations with high reliability. The distributed architecture featuring a Python-based desktop client, a FastAPI backend, and a React-enabled administrative dashboard ensures scalable operation, secure data handling, and streamlined post-exam review.

Experimental evaluations confirm that the system delivers strong detection performance across critical event categories such as gaze deviation, face absence, speech, unauthorized objects, and forbidden application usage. The locally executed TinyLlama model further enhances privacy by generating detailed examination reports without transmitting sensitive data to external services. Overall, the system addresses key limitations of existing proctoring solutions by reducing dependency on cloud resources, lowering operational costs, and enhancing transparency through automated, evidence-based reporting.

By demonstrating high accuracy, low latency, and strong privacy guarantees, the proposed system establishes a practical and dependable alternative to conventional human-based and cloud-dependent proctoring tools. Its modular design forms a scalable foundation for future enhancements in intelligent, offline, and institution-controlled examination monitoring technologies.

REFERENCES

- [1] Kumar, A., "Ethics of AI-Based Online Proctoring," *Computers & Education*, vol. 168, pp. 104-113, 2021.
- [2] Arnò, L. et al., "A Review of Online Exam Proctoring Tools," *IEEE Access*, vol. 10, pp. 12345-12360, 2022.
- [3] Lee, S. and Fanguy, L., "Surveillance Concerns in Online Assessments," *Educational Technology Research and Development*, vol. 69, no. 4, pp. 2151-2168, 2021.
- [4] Nigam, R. et al., "Evaluating Accuracy of Online Proctoring Systems," *ACM Transactions on Learning Technologies*, vol. 15, no. 3, pp. 1-19, 2022.

- [5] Fanguy, J., "Impact of Proctoring on Student Well-being," Proc. INTED Conference, pp. 5678–5685, 2021.
- [6] Zhao, Y. and Singh, M., "Human-AI Collaboration for Remote Monitoring," IEEE Transactions on Learning Technologies, vol. 15, no. 2, pp. 210–222, 2022.
- [7] Patel, S., "Challenges in Gaze-Based Cheating Detection," Pattern Recognition Letters, vol. 150, pp. 90–97, 2021.
- [8] Harper, J., "Improving Proctoring with Hybrid AI Models," IEEE Access, vol. 10, pp. 56789–56802, 2022.
- [9] Viola, P. and Jones, M., "Rapid Object Detection Using a Boosted Cascade of Simple Features," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol. 1, pp. I-511–I-518, 2001.
- [10] Krizhevsky, A., Sutskever, I. and Hinton, G., "ImageNet Classification with Deep Convolutional Neural Networks," Advances in Neural Information Processing Systems (NIPS), vol. 25, pp. 1097–1105, 2012.
- [11] Yang, H., "Multimodal Online Proctoring Systems," Procedia Computer Science (Elsevier ICTE), vol. 176, pp. 3085–3094, 2022.
- [12] Verma, S., "Face and Screen Monitoring for Exam Integrity," Proc. IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 1023–1028, 2021.
- [13] Howard, A. et al., "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," arXiv preprint arXiv:1704.04861, 2017.
- [14] Simonyan, K. and Zisserman, A., "Very Deep Convolutional Networks for Large-Scale Image Recognition," Proc. International Conference on Learning Representations (ICLR), 2015.
- [15] Rahman, M., "Accuracy Limits of Lightweight CNN Models," Artificial Intelligence Review, vol. 55, no. 3, pp. 2345–2360, 2022.
- [16] Jain, A., "Fairness Issues in Automated Proctoring," Proc. ACM Conference on Fairness, Accountability, and Transparency (FAccT), pp. 456–465, 2022.
- [17] Buolamwini, J. and Gebru, T., "Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification," Proc. Conference on Fairness, Accountability and Transparency, vol. 81, pp. 77–91, 2018.
- [18] Redmon, J., Divvala, S., Girshick, R. and Farhadi, A., "You Only Look Once: Unified, Real-Time Object Detection," Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 779–788, 2016.