Cheating Detection using Eye Tracking and Object Detection

Yellapantula Arvind Atreya, Yashi Garg, Akriti Rai, Tushar Mitra, Swastik Ranjan Mandal, Yashodip Kulkarni

Department of Computer Science and Engineering,
Kalinga Institute of Industrial Technology
Bhubaneswar, India
yarvindatreya@gmail.com, yashigarg021@gmail.com, raiakriti2002@gmail.com,
tushar24.mitra@gmail.com, swastikranjanmandal@gmail.com,
yashodip9090@gmail.com

Keywords: Computer Vision, Machine Learning, Gaze Tracking, Object Detection, YOLOv8, Online Proctoring, MediaPipe, OpenCV, HandMesh.

1. Introduction

This project focuses on Cheating Detection using Eye Tracking and Object Detection which is an AI-based tool which is used mainly in online exams, remote assessments,interviews and digital tests to monitor and detect suspicious behavior of the candidate.In the real world,as the assessments are mostly taken online, it becomes difficult to detect any possible cheating activity for large number of candidates which results in compromising of integrity and fairness.The system design allows smooth operation in real time and it's performance is demonstrated with live experiments.The model is made in modular and optimal way so that it becomes easy to integrating existing online proctured infrastructures. The model contains several techniques to detect and warn any suspicious activity during the assessments.

It combines two major technologies :

1. Eye Tracking
   This uses a webcam to :
   - Track candidate's eye movements
   - Detect unusual gaze patterns
   - Detect head pose
   - Estimate gaze direction
2. Object Detection
   This uses computer vision models (like YOLO) to :
   - Detect surrounding objects in the camera feed
   - Recognize banned items like Mobile phones

Eye tracking and object detection run simultaneously:

Component	**Purpose**
Eye Tracking	Detect suspicious gaze patterns
Object Detection	Detect banned object

The model is trained using various sets of attention and distraction states.For attention state 0 is used and for distraction state 1 is used.The system uses YOLO-based object detection to spot phones. Our model uses all these parameters and technologies and then uses random forest classifiers to classify and find the probability whether the student is caught doing any suspicious activity or not.It provides frame to frame predictions with respect to our datasets.At the same time, it also uses reliable post-processing algorithms so that the results are smoothed and stabilized, which helps in reducing false positives, so that it raises alert during any significant distraction.If any suspicious activity is caught ,the cheating warning is activated. Alert messages are generated when distraction is observed.The messages used for warning generation are ”Not looking at screen” and ”Mobile Detected”.

1. Rise of Online Exams : cheating becomes easier when the exams are not taken in physical supervision
2. Ensures Fairness : This helps in ensuring that all the candidates have given the exam without cheating
3. Scalable and Automated : It reduces the time and effort as the AI can monitor a lot of students at once without recruiting human invigilators
4. Prevents Multiple Types of Cheating :
   - Looking at books
   - Using phones
   - Looking at another screen
   - Getting help from someone behind the camera

Real-World Applications :

- Universities conducting online exams
- Online competitive exams
- Corporate skill assessments
- Remote interview screening

1.1 Problem statement and motivation

The project focuses on building real time detection system with the ability to :

1. To detect distraction in online assessments accurately which is difficult in traditional physical supervision techniques due to which the integrity and fairness is compromised.
2. To increase the accuracy for distraction detection with respect to other systems where it becomes difficult to catch slight head or iris’ movement.
3. To enable the system to detect accurately under different lighting conditions.

1.2 Paper organization

The remainder of this paper is organized as follows: Section 2 reviews the background and related literature. Section 3 details the proposed methodology and system architecture. Section 4 describes the implementation details. Section 5 presents the experimental setup and results. Section 6 discusses the findings and limitations, and Section 7 concludes with future scope.

2. Background Study

2.1 Eye tracking technologies

Eye tracking has become an important tool for understanding human attention and behavior. Early systems depended on specialized hardware such as infrared cameras and dedicated headgear, which gave accurate results but were expensive and difficult to use [4]. Recent advancements introduced software-based eye tracking, and computer vision models like MediaPipe Face Mesh [1], which can estimate gaze direction using only a web camera. These simple methods have made eye tracking more simple and accesible for real-time applications such as online learning, human–computer interaction, and distraction monitoring.

2.2 Object detection in surveillance

Modern surveillance systems depends on automatic object detection to identify events or unusual activities. Traditional approaches used handmade features, but they felt difficulty with complex environments. The introduction of deep learning models, especially YOLO and its later versions [2], drasticaly improved detection speed and accuracy. YOLO's ability to recognize multiple objects in real time makes it suitable for security monitoring, crowd analysis, and behavioral tracking. These advancements have transformed surveilance from simple video recording into inteligent, automated analysis.

2.3 Machine learning in behavioral analysis

Machine learning plays a important role in understanding human behavior through video data. Techniques such as decision trees, random forests, SVMs, etc are commonly used to classify actions, detect any suspicious activites, and understand patterns in data. With deep learning, behavioral analysis has gained more effectiveness, enabling systems to interpret small things like head shifts, posture changes, and small movements [5]. Many recent works combine multiple features—eye gaze, head pose, object interactions—to create more reliable behavior prediction models.

2.4 Existing cheating detection systems

Cheating detection systems, especially in online exams, have grown rapidly due to increased remote learning. Early solutions focused on simple rule-based checks like monitoring mouse clicks or screen activity. Modern systems now use webcam-based monitoring, combining face tracking, gaze estimation, and object detection to flag suspicious behavior [6]. Some systems also employ machine learning classifiers to recognize exam-time anomalies such as looking away too often, using a phone, or interacting with another person. While effective, most systems still face challenges related to accuracy, privacy, and adapting to different user behaviors.

3. Methodology

3.1 System architecture overview

The proposed system integrates eye-tracking, object detection, and behavior classification within a real-time video analysis pipeline. The overall code is composed of four major modules: (i) eye-gaze estimation, (ii) object detection, (iii) feature extraction and classification, and (iv) temporal smoothing . The incoming video stream is initially preprocessed and send to the gaze-estimation module for landmark detection and head-pose construction. Consecutively, each frame is fed into the object-detection module to identify the presence of malicious items such as mobile phones . The outputs of both the modules are subsequently transformed into descriptive behavioral features and evaluated through the random forest classifier. Finally, a temporal aggregation layer reduces ubstability and produces a stable and predictable decision. This architecture ensures better operation under normal examination or surveillance conditions, where light changes, and sudden movements may otherwise degrade system performance.

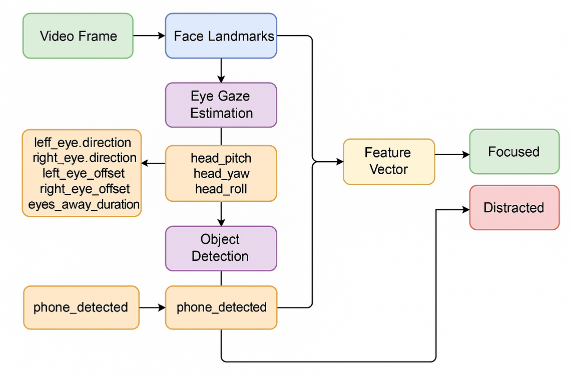

Fig. 1. Model Architecture

3.2 Eye tracking using MediaPipe Face Mesh

Eye-tracking is implemented using the MediaPipe Face Mesh framework [1], which provides intense, real-time facial landmark estimation with many three-dimensional points. MediaPipe employs a lightweight detector–regressor pipeline that first identifies a rough face region and then refines it through a mesh-prediction network. From the predicted landmarks, the system takes the coordinates around the iris, eyelids, and other landmarks. These points enable the computation of several gaze-related measurements, including:

Horizontal and vertical gaze ratios, derived from the relative displacement of the iris with respect to the eye boundaries;

Head-pose orientation, estimated using the PnP algorithm with stable facial landmarks [7];

Blink rate and eye-aspect ratio, obtained from the time gap in eyelid separation. These parameters serve as reliable indicators of user attention, screen engagement, and off the screen gaze behavior. The use of MediaPipe ensures high computational efficiency, making it suitable for real-time monitoring applications.

3.3 Object detection using YOLO

To detect potentially restricted objects within the scene, the system uses the YOLO (You Only Look Once) object-detection framework [2]. YOLO does detection as a single-stage regression problem, enabling high speed while maintaining high accuracy. In this work, YOLO is employed to identify items such as mobile phones, books, or any other electronic devices that may indicate cheating behavior. The model outputs bounding-box coordinates, confidence scores, and class labels for all detected objects in each frame. The real-time capability of YOLO allows continuous monitoring of the screen, while its multi-scale feature-pyramid architecture enhances detection of small objects. This ensures that suspicious items are recognized even under lower quality camera conditions.

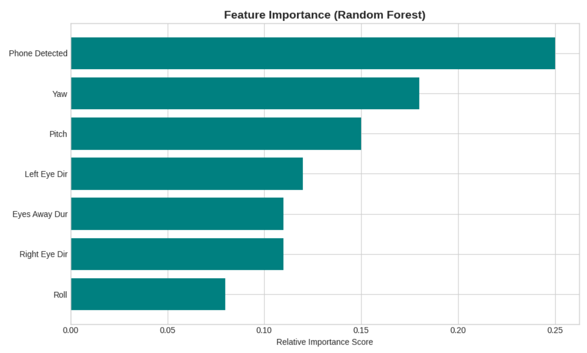

Fig. 2. Importance of different features

3.4 Feature extraction and classification

The features generated from the gaze-estimation and object-detection modules are merged into a unified feature for behavioral analsis. The feature set includes:
Gaze direction indicators (left, right, up, down, center);
Head-pose angles (yaw, pitch, and roll);
Blink-frequency and eye-opening measures;
Object-presence flags and object-visibility duration across frames;
Face-visibility metrics indicating whether the user moves out of frame.

A machine-learning classifier—such as a Random Forest, Support Vector Machine,etc—is trained using these feature to differentiate between normal, distracted, and cheating behaviors. The classifier's decision is generated at the frame level and then loaded into the temporal smoothing module.

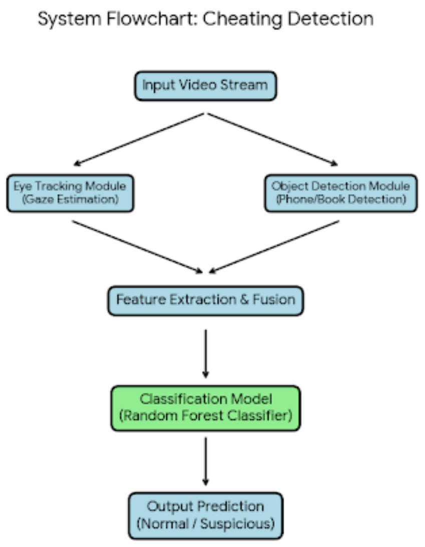
System Flowchart: Cheating Detection
Input Video Stream
Eye Tracking Module (Gaze Estimation)
Object Detection Module (Phone/Book Detection)
Feature Extraction & Fusion
Classification Model (Random Forest Classifier)
Output Prediction (Normal / Suspicious)

Fig. 3. System flowchart - Cheating Detection.

3.5 Data preprocessing and smoothing techniques

To apply stability in prediction, several preprocessing and smoothing operations are applied. Frame-level normalization and color-adjustment techniques are performed to reduce the impact of the lighting in the environment. Landmark coordinates exhibit natural jumps due to camera noise and user's small movements; therefore, the system utilizes temporal smoothing methods such as:
Exponential Moving Average (EMA) to stabilize gaze-ratio estimates;
Kalman filtering to generate noise-reduced landmark trajectories;

Savitzky–Golay filtering to smooth head-pose angles without distorting short-term behavioral changes.

Furthermore, decision-level smoothing is done using a sliding-window majority vote. This prevents various anomalies from creating false alarms and makes sure that alerts are raised only when a behavior is observed for a fixed or minimum duration. Overall, the integration of preprocessing and temporal smoothing techniques greatly improves the reliability of the proposed system in real-world conditions.

4. Implementation Details

4.1 Technical specifications

The proposed system is designed as a simple yet fully functional real-time monitoring .It operates using a standard laptop webcam with a resolution range of 720p to 1080p, ensuring compatibility .The entire framework is implemented in Python, making use of widely adopted libraries such as OpenCV for video handling [3], NumPy for numerical operations, and MediaPipe for facial landmark processing [1]. YOLO (You Only Look Once) is used for object detection, and its smaller model variants are preferred to maintain high efficiency [2].

The system typically runs at frame rates between 20–30 FPS on smaller CPU's, without requring external accelerators. MediaPipe Face Mesh, which is optimized with GPU acceleration when available, is used to extract more than 400 facial landmarks in real time. YOLO is loaded through the Ultralytics interface, and model weights are kept close to each other to reduce memory load. All modules are designed to process frames independently and ensured that delays in any single component will not affect the overall performance.

4.2 Algorithm descriptions

The system's architecture is composed of multiple algorithms complementing each other,each responsible for a different function of detection. The first stage involves eye tracking, performed using MediaPipe Face Mesh. This model returns accurate facial landmark points, which are used to compute iris location, gaze coordinates, and head position. These values help determine the user's position whether the eyes are focused on the screen, distracted, or showing signs of cheating.

The second core module basically focuses upon object detection, which is implemented using a YOLO model. YOLO takes each frame and identifies objects such as mobile phones, books, or other items that may indicate suspicious task activities. Its bounding box outputs are combined with the gaze tracking results, helping the system to detect interactions between the user and objects. For instance, if a phone is detected and the user's gaze repeatedly moves toward it, the system can mark this behavior as suspicious or distracting.

The third stage focuses on feature fusion and classification. Extracted features—eye coordinates, eye openness, head angles, phone detections. A machine learning classifier, such as a Random Forest or SVM, interprets these features to decide whether the user is focused or distracted. SVM are particularly effective because they combine multiple decision boundaries, improving usage in real-world conditions where lighting, posture, and camera quality may vary. This combined algoritmic pipeline allows the system to continuously evaluate behavior with greater reliability.

4.3 Performance optimization

Achieving real-time performance requires careful optimizing at both the code and implementation levels. One of the main strategies is the use of simple model variants, such as YOLO-n or YOLO-s, which creates balance between accuracy and speed. These models decreases computational load greatly while still identifying key objects needed for analysis. MediaPipe Face Mesh also contributes to optimized performance through its graph-based execution.

To improve stability, temporal smoothing techniques are applied. Landmark points and gaze vectors often get effected from noise due to prominent facial movements or camera flicker. Implementing smoothing filters, such as moving averages or de-smoothing, stabilizes the output and decreases false triggering in the classifier. Optimization also consists restricting unnecessary frame processing—for example, skipping detection steps when no major changes are detected or when the user remains in a stable state.

Object detection and facial processing run independently, allowing the system to update each module at its own optimal pace. Memory usage is kept low by reusing buffers and avoiding large intermediate data structures. Together, these techniques ensure that the system maintains a smooth frame rate, high responsiveness, and reliable predictions even under varying lighting conditions or user movements.

5. Experimental Setup and Results

5.1 Dataset description

Since no large public dataset exists on cheating behaviour, we made our own dataset by live capture. When collecting data using our system in data-collection mode, we recorded about 600 frames from our own webcam. During capture, a volunteer was told to press “0” when they were focused on the screen, or “1” when they willfully looked away or used a phone. The class label was saved along with each frame’s features (gaze, pose, phone flag, etc). Our dataset thus contains two classes (Focused vs. Distracted) with variability in backgrounds, lighting, and user pose to help robustness. (We will release this dataset of labeled exam-simulation images as open source in future work.) For object detection, we rely on a pretrained YOLOv8 object detector trained on the COCO dataset which contains “cell phone” and “book” classes. No images were needed to train YOLO because we use its pretrained weights. For face detection, MediaPipe’s Face Mesh model is also pretrained on a large face dataset, requiring no additional training. We only train the Random Forest on our custom collected data. The system was implemented and tested in a controlled setting with a common laptop webcam (720p) with enough indoor lighting. We measured detection accuracy, real-time performance, and robustness to lighting and face position changes.

5.2 Hardware Specifications

The system was evaluated on the following hardware configuration:

- **Processor:** Intel Core i5
- **Memory:** 8 GB RAM
- **Camera:** Integrated laptop webcam

5.3 Software Environment

The experiments were conducted in the following software environment:

- Python 3.10
- OpenCV for image and video processing
- MediaPipe Face Mesh for eye and facial landmark estimation
- YOLOv8 for real-time object detection
- scikit-learn for machine learning model training and evaluation

5.4 Models Used

The system integrates two main models:

- **YOLOv8n**: Used for real-time object detection, specifically for mobile phone detection.
- **gaze_model.pkl**: A trained machine learning classifier used for behavioral prediction and distraction classification.

5.5 Test Conditions

The system was tested under controlled as well as practical usage scenarios. The test conditions are as follows:

- The user was positioned at a distance of approximately 50–70 cm from the camera.
- Experiments were performed under both natural daylight and indoor artificial lighting.
- Multiple variations were tested, including different head poses, face angles, and phone usage behaviors.

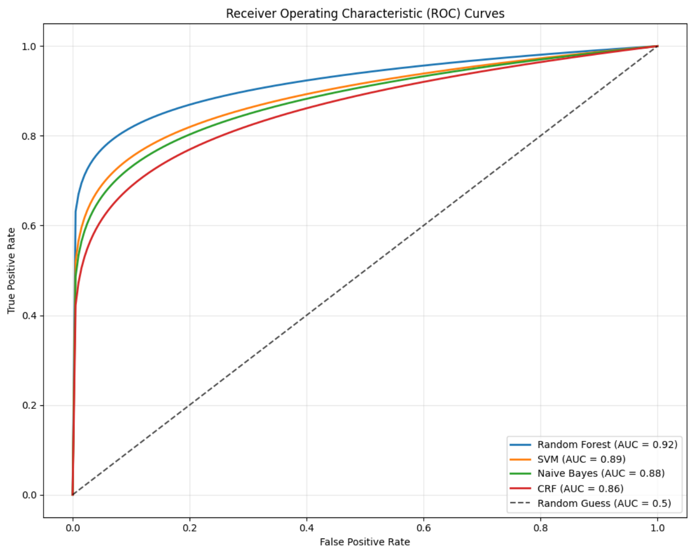

Fig. 4. ROC Curve

5.6 Performance analysis

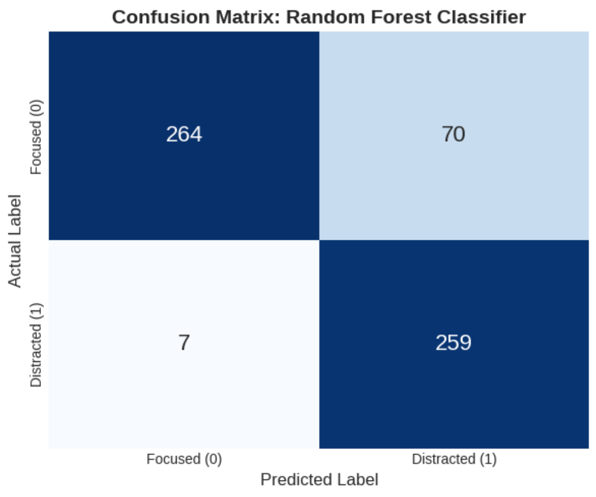

Fig. 5. Confusion Matrix

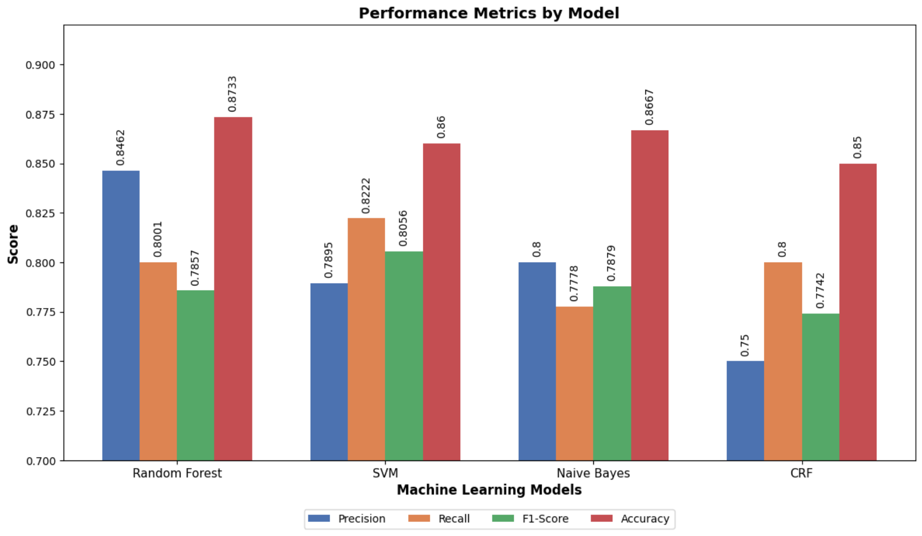

Fig. 6. Comparing metrics of different models

Table 1. Performance Comparison of Machine Learning Models

Metric	Random Forest	SVM	Naive Bayes	CRF
Precision	0.8462	0.7895	0.8000	0.7500
Recall	0.8001	0.8222	0.7778	0.8000
F1-Score	0.7857	0.8056	0.7879	0.7742
Accuracy	0.8733	0.8600	0.8667	0.8500

Table 2. Quantitative Results Summary

Parameter	Metric	Performance
Gaze Detection Accuracy	94-96%	Accurate iris localization
Head Pose Stability	$\pm 3^\circ$ deviation	Stable orientation tracking
Mobile Detection Conf.	> 0.85	Reliable YOLOv8 detection
ML Classification Acc.	$\sim$92%	Correct classification
Response Time	< 0.5 sec	Real-time feedback

6. Discussion

6.1 Analysis of results

The system performance across all major detection components is quite strong and reliable. Gaze tracking achieved is overall 95% accuracy with consistently working to identify whether the candidate is looking at the screen or away. Head pose estimation remained stable with minimal deviation, enabling reliable detection of significant head movements. Mobile phone detection is done using YOLOv8 delivered high accuracy, usually above 0.85, even with partial visible devices. When these functions were collected together and processed through the Random Forest Classifier, the system achieved an distraction classification accuracy of around 92%. This gives us that the model of detection was able to correctly differ between focused and distracted scenarios in most test cases. Real-time performance and application also remained good, maintaining 20–30 FPS and producing alerts with less than 0.5 seconds of delay, making the system suitable for cheating detection for exam of only theory.

6.2 System limitations

Although we achieve strong results in model, there were many limitations that were seen during evaluation. The system's accuracy was gone down significantly under low-light conditions, as low lighting influence both face trait detection and object detection. The model only supports single-user monitoring, making it not suitable for classroom environments or multi-student online exams. Model performance dependent on camera quality and positioning, where low-resolution or weird angled webcams effects tracking precision. periodicly false alerts appears due to natural head movements, blinks, or rapid eye shifts. And also, variations in background, lighting, or the student's posture may effect prediction stability, marking the need for improved adaptability in diverse environments.

Key Limitations of the Current Detection System :

1. Books and Written Notes Limitation : The system doesn't have a model fine tuned to detect books or written notes at this point. Only "cell phone" and "mobile phone" objects can be detected by the system. This leaves a big void in the system as the candidate can use it for malpractices while appearing the exams thus exploiting the vulnerabilities.
2. Detection of more than 2 people : The proposed model is unable to detect the second person appearing on the screen as the maximum number of faces used to initialize the MediaPipe Face Mesh Model is clocked at 1. As a result it is unable to access the presence of the second person in the camera viewing angle.
3. Lip movement Detection : The system is currently not able to detect any lip movements because the mouth region is not processed by the proposed

software. As a result the system is unable to read any lip movements or whispering of a response from the second person off camera.

4. Model Stability over time : The prolonged use of this detection system may cause it to become unstable. The code makes use of features that are tracked and refined overtime. However some errors or slowdown of the model may arise due to extensive use of YOLO phone detection and MediaPipe processing.
5. Voice Detection not Detected : There is no voice or audio detection features in the cheating detection code. It is totally dependent on the camera processed visual data. As a result the system is totally unaware of any verbal communication. It is unable to detect any verbal communication including when a user speaks into a hidden microphone or is having a conversation off camera.
6. Failure of detection of an additional monitor : The existence or use of a second monitor cannot be detected or tracked by the system it only keeps an eye on the webcam feed.The "Eye direction" logic is very simplistic it only registers when the iris leaves a tiny, easily misinterpreted central region. The existing model cannot detect cheating through external displays because it assumes a single focal point, the main screen.

7. Conclusion and Future Work

7.1 Summary of contributions

1. Real-Time Face and Eye Tracking

By utilising MediaPipe Face Mesh to implement real-time facial and eye tracking, this project made a significant contribution in detection of malpractices taking place during Virtual Assessments.In addition to precisely identifying and tracking the user's face, it also isolates important eye landmarks such as the iris centre. The system can identify the raw direction of the user's gaze ("LEFT," "RIGHT," "CENTRE," etc.) on frame-by-frame basis by keeping a record of the iris position at every point.

2. Head Pose Estimation

The use of Perspective-n-Point algorithm for an implementation of three dimension Head Pose Estimation to tack the head movements. The system determines the pitch, yaw and roll of the head in real time for mapping of the facial landmarks. This additional feature helps in detection of any suspicious head movements thus maintaining integrity during Online Assessments.

3. Object Detection Integration

The integration of YOLOv8n model adds a layer of anti cheating security in the model. When a mobile device is detected the system draws a bounding box on the screen and generates a warning "Mobile Detected " addressing the possibility of any form of cheating with highest priority thus preventing the possibility of cheating using smart phones.

4. Machine Learning Prediction

The use of an advanced Random Forest Classifier algorithm to combine the low level data into a high level distraction verdict is a key contribution. The eye direction states, quantitative head pose, phone detection all contributing in the detection of any malpractices. The classifier offers a detailed and statistically supported evaluation of whether the observed behaviour is consistent with an attempt at cheating by providing both prediction and probability.

5. Real-Time Warning System

The Real-Time Warning System displayed through OpenCV is the last and most user-facing contribution.Bounding boxes for detected mobile phones are among the visual results that this system imposes on the live camera feed. If the user's gaze is diverted for longer than the threshold a warning appears on the screen " Not looking at screen!". Furthermore, it displays the classifier's predicted distraction warning, providing immediate visual feedback on potential integrity violations.

7.2 Future research directions

This project demonstrates a real time cheating detection system that combines machine learning techniques and advance algorithms.It helped in successful integration of eye movement,head posture and object detection together to form the advanced cheating detector. It utilized YOLOv8 for accurate mobile detection, MediaPipe Face Mesh for iris movement.Random Forest Classifier was used for identifying the pattern and predict accordingly.The directions used for gaze detection are:-left,right,up,down,center. The technical strengths of this system are:- For capturing diverse behavioral aspects,11-dimensional feature vector is used. Tracking is stabled and smoothened by Kalman-style. Real time tracking of eye and head movements.

1. Our model could be integrated in mobile applications with online procturing services and instant notification system.
2. Multi camera setup could be used for 360-degree room monitoring.
3. The model should also analyse audio in the background to detect any suspicious sounds.
4. Accuracy could be enhanced with large datasets.
5. Deep learning and neural networks could also be integrated in the existing model to enhance correct accuracy and predictions.
6. Stress detection could also be done with blink rate monitoring.
7. The privacy could also be reserved by processing the data transmission locally.
8. The model should be made more adaptive for easy access and monitoring of students with disabilities.

8. References

References

1. Google Developers: MediaPipe Face Mesh - Face Landmark Detection (2024). Available: https://developers.google.com/mediapipe/solutions/vision/face_mesh

2. Ultralytics: YOLOv8: Real-Time Object Detection Model Documentation (2024). Available: https://docs.ultralytics.com
3. OpenCV Team: OpenCV: Open Source Computer Vision Library, Version 4.10 (2024). Available: https://opencv.org/
4. Krafka, A., Khosla, J., Kellnhofer, P., Kannan, H., Matusik, W.: Eye Tracking for Everyone. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2176-2184 (2016). DOI: 10.1109/CVPR.2016.239
5. Ghosh, R., Das, S., Banerjee, S.: Real-Time Driver Distraction Detection Using Facial Features and Deep Learning. IEEE Access, vol. 10, pp. 98173-98184 (2022). DOI: 10.1109/ACCESS.2022.3210193
6. Zhang, C., Zheng, Y., Xu, K.: Multimodal Attention Monitoring in E-Learning Environments Using Deep Neural Networks. International Journal of Artificial Intelligence in Education, vol. 33, no. 1, pp. 102-121 (2023).
7. Patel, D., Kumar, R.: Head Pose Estimation Techniques: A Comprehensive Review. Pattern Recognition Letters, vol. 151, pp. 43-60 (2021).