A CNN-LSTM Framework for Real-Time Detection of Suspicious Behaviors in Examination Halls

Ori Silas Ene[1], Igwe Joseph Sunday[2] , Ituma Chinagorom[3], Onu Sunday[4]
Chukwuemeka Okpara[5] , Emewu Benedict Mbanefo[6] Christiana Ugochineyere Oko[7]
1,4,5,6 & 7 David Umahi Federal University of Health Science, Uburu, Ebonyi State, Nigeria
2&3 *Department Of Computer Science, Ebonyi State University, Abakaliki–Nigeria*

ORCID ID: 0009000940632639

Abstract:
This paper presents a novel deep learning model for automating the detection of suspicious behaviors during examinations, a critical step in combating academic malpractice. Traditional monitoring methods, reliant on human invigilators, are prone to fatigue, subjectivity, and inefficiency in large settings. Our approach leverages a integrated Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) architecture to analyze video surveillance footage. The CNN extracts spatial features from individual frames, such as body posture and orientation, while the LSTM models the temporal dynamics of student actions over time to distinguish between normal and suspicious activity sequences. The system was developed using the CRISP-DM methodology and implemented with a Streamlit-based user interface for practical deployment. Performance evaluation, based on a confusion matrix and F1-score, demonstrates the model's high accuracy and reliability in identifying behaviors such as excessive head turning, gesturing, and looking away. This research confirms the viability of deep learning as a robust, automated tool for enhancing exam integrity and reducing the burden on human proctors.

Keywords: Deep Learning, CNN, LSTM, Examination Malpractice, Video Surveillance, Automated Proctoring, Computer Vision.

Introduction

The integrity of examinations is a cornerstone of academic assessment, crucial for certifying knowledge and maintaining institutional standards. However, this integrity is persistently threatened by examination malpractice, a pervasive issue that undermines the value of educational qualifications and disadvantages honest students. Traditional methods of safeguarding exam integrity have primarily relied on human invigilators who monitor examination halls to deter and detect suspicious activities.
Despite the dedication of invigilators, this human-centric approach is fraught with challenges. As noted in the research, invigilators are susceptible to distractions, fatigue, and subjective bias, which can lead to inconsistent monitoring [1]. The effectiveness of this system diminishes significantly in large examination halls with hundreds of students, where it is physically impossible for a few individuals to maintain uninterrupted vigilance over every candidate. Consequently, many instances of malpractice go undetected, and when they are spotted, the lack of automated, recorded evidence can complicate disciplinary proceedings.

The proliferation of surveillance technology, specifically Closed-Circuit Television (CCTV) cameras, in academic institutions offered a partial solution. Bodies like Nigeria's Joint Admission and Matriculation Board (JAMB) have mandated CCTV use in examination centers. However, these systems often merely shift the problem; instead of invigilators patrolling halls, human operators must now stare at video feeds for hours, a task characterized by monotony and a high potential for lapses in attention [2]. The video footage becomes useful mostly for post-incident forensic analysis rather than real-time intervention [3].

Related Literatures on Examination Monitoring Techniques

Monitoring examination with a surveillance camera is a form of electronic invigilation that involves the use of an electronic device to monitor student's activities in the hall. With surveillance technology, a monitoring device otherwise known as CCTV is installed in examination halls. Through the CCTV camera system examination monitoring officers from the control room can watch and monitor the examination halls. This approach can be very helpful in combating examination malpractice in institutions where there are large numbers of students and few invigilators in the hall. However, the effectiveness of this system depends on the vigilance of the examination monitoring officer (invigilator) to detect infringements that can be affected by fatigue and human biasedness.

Nowadays with the growing need to eradicate examination malpractices in the institutions of learning, video surveillance has become a big concern for every examination body.

A consequence of these needs has led to the deployment of cameras in examination venues in some institutions [4][5]. At present, monitoring activities, understanding behaviours, detecting infringements, and recognizing suspicious behaviours in videos automatically is a significant research topic in the fight against examination malpractice [6] [7]; [8]. Therefore, understanding activities in an examination scenario involve being able to detect and classify infringements and analyze what is being done. One crucial aspect is to detect and report infringements of special interests, in particular as the suspicious activities are occurring. In this case, a video surveillance system that can interpret the scene, capture visual evidence of the scene, and automatically recognize suspicious behaviours in real-time can play a vital role. The system would then send a notification to the operators monitoring the examination accordingly.

Several measures and approaches have earlier been new to deter the menace of examinations malpractice across all levels of studies [9]; [10]; [11]. Some of the researches were adopted by various examinations regulatory bodies yet it has not produced the desired results [12]; [13], [14]. In ensuring that there is no relenting in the fight against these menaces, attention is now being shifted to the application of surveillance technology to complement the existing approaches.

[15] designed a new model to detect impersonation of candidates in an online examination system using video surveillance and computer vision techniques. In this system candidate's faces were detected using the Viola-Jones algorithm, which has mainly 4 stages: Haar-Features Selection, Creating Integral Image, AdaBoost Training algorithm, and Cascaded Classifiers.

[16] an autonomous agent for monitoring student behavior in the classroom using computer vision techniques. The technique was focused on the detection and recognition of faces and pose estimation to measure certain behavioral characteristics that relate to the students' level of attention during classes. The aim was to by providing guidance and feedback for the teachers on how to improve their teaching role during a class and to the students on how to improve their behavior in the classroom to improve their academic performance. Also, the system can provide visual feedback to the teachers regarding the average level of students' attention, so that students can be counsel regarding their behavior during the class.

[17] a smart system to robustly detect their cheating activities in the examination hall by detecting and tracking the examinee's eye gaze and head orientation autonomously. Webcams were installed on each of the smart devices placed on each of the examination desks, initially, real-time video is captured during the examination period. Then the video is later analyzed to study the examinee's psychology as regards the examination misconduct to build a knowledgebase from the new system. The new system is focused on detecting (1) candidate copying from materials such as phone and books (ii) candidates spying and copying from other candidates. Several behaviours can be considered suspicious but this system is only considered two, also the system cannot detect suspicious activities during an examination in real-time. Therefore, improvement is required to make the system more robust.
[18] a Neural Networks and Gaussian distribution model to detect anomalous behavior of students in examination rooms in real-time. They noted difficulty as regards detecting anomalous behavior simultaneously in a complex scene of crowded motion such as examinees in examination rooms. To solve the problem a prototype of a monitoring system that consists of three stages, face detection using haar cascade detector, suspicious state detection using a neural network, and lastly anomaly detection based on the Gaussian distribution was built. The idea is to be able to decide whether a student's activity is suspicious or not using a trained neural network. This work focused basically on face detection, hand gestures. But other clues to suspect that a student is cheating were not considered; hence the need for further improvement.
[19] a system to identify abrupt changes in a classroom video and then evaluate the level of students' attentiveness. The new algorithm was implemented with key frame extraction from the video approach. The Structural Similarity Index Measure (SSIM) approach for key frame extraction was used in the study. After extracting the key frames, the detection of the face and upper body of the students to evaluate their attention level was performed on the key frames. The results reveal that the SSIM-based algorithm gives better results compared to without SSIM based algorithms.

METHODOLOGY

This research adopted a hybrid methodology, combining the Cross-Industry Standard Process for Data Mining (CRISP-DM) framework with the Object-Oriented Analysis and Design Methodology (OOADM). This integration ensured a structured approach to both the data-centric model development and the overall system design.
The CRISP-DM framework guided the deep learning lifecycle through its six phases:

1. **Business Understanding:** The goal was defined as real-time detection of suspicious behaviors (e.g., frequent head turning, gesturing) to reduce malpractice.
2. **Data Understanding:** Data was gathered from video recordings of simulated examination scenes and open datasets (e.g., from Kaggle.com) related to human activity recognition.
3. **Data Preparation:** Videos were preprocessed through frame extraction, resizing, noise removal, and annotation into "normal" and "suspicious" categories. The dataset was split into training, validation, and testing sets.
4. **Modeling:** A deep learning model integrating CNN and LSTM was developed to extract spatial features and analyze temporal sequences, respectively.
5. **Evaluation:** The model was evaluated using metrics like accuracy, precision, recall, and F1-score.
6. **Deployment:** The trained model was deployed into a functional prototype with a user interface.

The OOADM was used to design the system architecture, identifying key objects (e.g., Student, Camera, Suspicious Behavior Detector) and defining their interactions through use case, class, and sequence diagrams. This ensured a modular, maintainable, and scalable system design.

System Analysis

An analysis of the existing system (human invigilation) revealed critical weaknesses: limited human attention, difficulty in scaling to large halls, subjectivity, lack of automated evidence, and high stress on invigilators.

The new system was designed to address these shortcomings. It allows an invigilator to upload an examination video, which is then automatically analyzed by the deep learning model and can also perform a real time live analysis. The system processes the video, detects suspicious behaviors, and generates an alert with a detailed report for the invigilator. This provides continuous, unbiased monitoring and creates a record of evidence.

Data Gathering Techniquea

A multi-faceted approach was used for data collection:

- **Interview:** Lecturers and exam supervisors were interviewed to identify common suspicious behaviors.
- **Observation:** Direct observations were made in both real and simulated exam environments.
- **Internet Sources:** Publicly available datasets from platforms like Kaggle.com were used to supplement the collected data, ensuring a diverse and robust training set. These included Human Action Recognition datasets and general activity samples.

System Architecture and Model Integration

The system is built on a 2-tier architecture for simplicity and efficiency:

- **Client Tier (Front-End):** A user-friendly interface built with Streamlit. This tier handles user input (video upload) and displays the analysis results.

- **Server Tier (Back-End):** The core processing unit, implemented with Django and Python. It handles preprocessing, feature extraction, model execution, and decision-making.
  The core of the system is the integration of Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks.

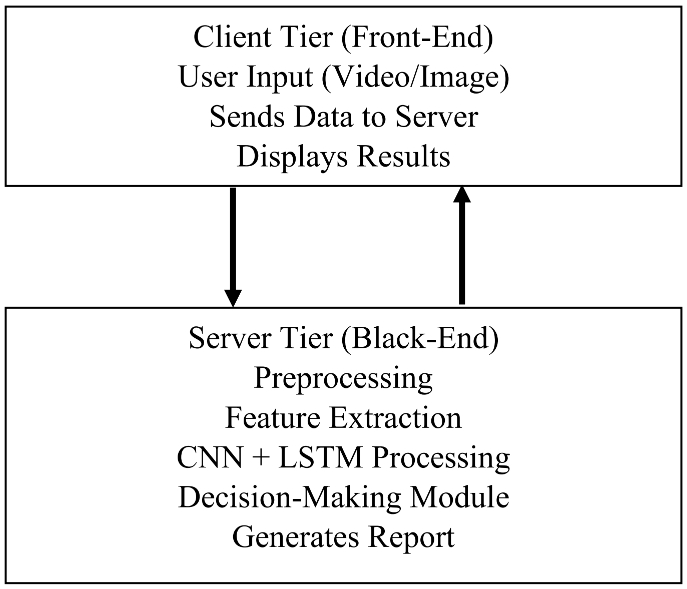


Figure: Architecture of the New System

- **CNN Role:** Acts as the "eyes" of the system. It scans individual video frames to extract important visual features such as body postures, facial orientations, and gestures.
- **LSTM Role:** Acts as the "memory." It analyzes the sequence of features extracted by the CNN over time. This allows the system to understand not just a single pose, but a pattern of movement (e.g., a student turning their head left and right repeatedly).
- **Integration:** The features extracted by the CNN from consecutive frames are fed into the LSTM. The LSTM then models the temporal relationships, and its output is passed to a decision-making module that classifies the activity as "suspicious" or "normal."
  This combined CNN-LSTM approach enables the system to understand both the spatial and contextual nature of suspicious behaviors, leading to higher accuracy.

Database Development Tool

For this system, the database development tool adopted is Django, which is a high-level Python web framework that comes with a powerful built-in database management system. Django makes it easy to connect the system with a database, store information securely, and retrieve data when needed. It uses Object Relational Mapping (ORM), which allows developers to interact with the database using Python code instead of writing complex SQL queries.

In this system, Django is used to create the database tables for storing user details, uploaded data, detection results, and reports. The ORM automatically converts Python objects into database tables and vice versa, which reduces errors and speeds up development. For example, if we define a model class for "User" in Django, it will create a matching table in the database where all user information is saved.

System Implementation

System implementation is the stage where the designed system is put into practical use. At this point, all the components of the system, such as the user interface, the deep learning model, and the database, are integrated to work together as one complete application. The goal of implementation is to make the system functional, easy to use, and reliable for real-world tasks.

In this project, the implementation was done using Python as the main programming language because it is simple, flexible, and has strong libraries for deep learning. The deep learning model, which combines Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM)**,** was built using frameworks like TensorFlow and Keras. These frameworks provide tools for training, testing, and saving the model. The model was then connected to the system so that it can analyze inputs and give predictions in real time.

The Django framework was used for building the backend of the system. It manages communication between the deep learning model, the database, and the user interface. Django also handles user authentication and ensures security of stored information.

The database was implemented with SQLite for development because it is lightweight and easy to configure. The database stores user details, uploaded files, and detection results. Django's Object Relational Mapping (ORM) was used to make communication with the database simple and efficient.

The frontend interface was designed using Django templates combined with HTML, CSS, and JavaScript. This makes it possible for users to interact with the system through a simple and clear web interface. The interface includes the main menu, sub-menus, and forms where users can upload data or view results.

Finally, the system was tested and run on Visual Studio Code (VS Code)**,** which serves as the development environment. From VS Code, the Django server was started, the model was executed, and the system was accessed through a web browser. This allowed both the model and database to work together in providing accurate results.

The implementation process brought together the deep learning model, database, and user interface into one functioning system. It ensures that the system can analyze input, process it through the CNN-LSTM model, store results in the database, and display outputs to users in a clear and secure way.

User Interface Implementation

The user interface implementation is the stage where the system is made easy for users to interact with. The interface acts as a bridge between the user and the system by displaying menus, forms, buttons, and results in a clear way. A good interface is important because it helps users understand the system without needing technical knowledge.

In this project, the user interface was created using Django templates combined with HTML, CSS, and JavaScript**.** Django templates made it possible to connect the backend with the frontend, so

that real-time results from the model and database could be displayed to the user. HTML was used to design the structure of the pages, CSS was used to style the layout with colors and fonts, and JavaScript added simple interactivity to make the system more dynamic.
The main menu of the interface provides access to different parts of the system, such as uploading data, running detection, and viewing results. The sub-menus help users navigate specific tasks, like checking stored records or managing user profiles. Each page is designed to be clear and simple, with easy navigation links and feedback messages that guide users while using the system.
The interface also connects with the deep learning model and database in the background. When a user uploads data, the system sends it to the model for analysis, and then the results are displayed back on the interface. At the same time, the results are stored in the database for future reference.
To make sure the interface is responsive; CSS was used to adjust the design for different screen sizes, so that the system can work on desktops, laptops, or mobile devices. This ensures accessibility and better user experience.
Overall, the user interface implementation focused on making the system simple, interactive, and easy to use. It allows users to interact smoothly with the deep learning model and database without needing to understand the technical background.

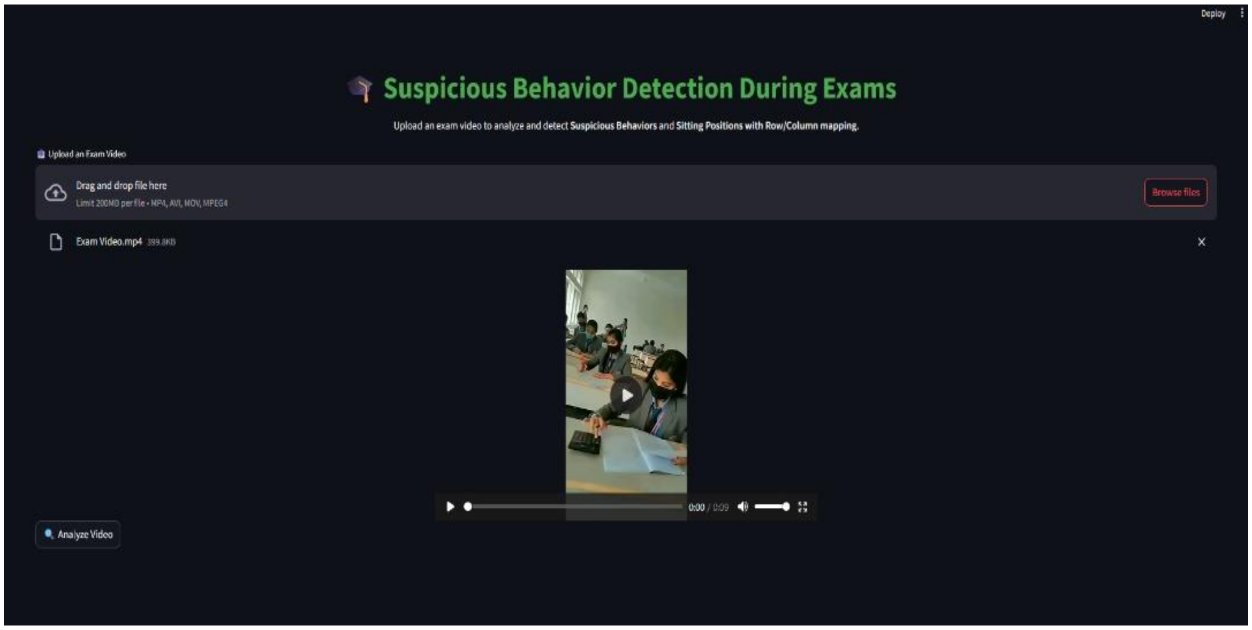

Deploy
Suspicious Behavior Detection During Exams
Upload an exam video to analyze and detect Suspicious Behaviors and Sitting Positions with Row/Column mapping.
Upload an Exam Video
Drag and drop file here
Limit 200MB per file • MP4, AVI, MOV, MPEG4
Browse files
Exam Video.mp4 399.8KB
0:00 / 0:09
Analyze Video

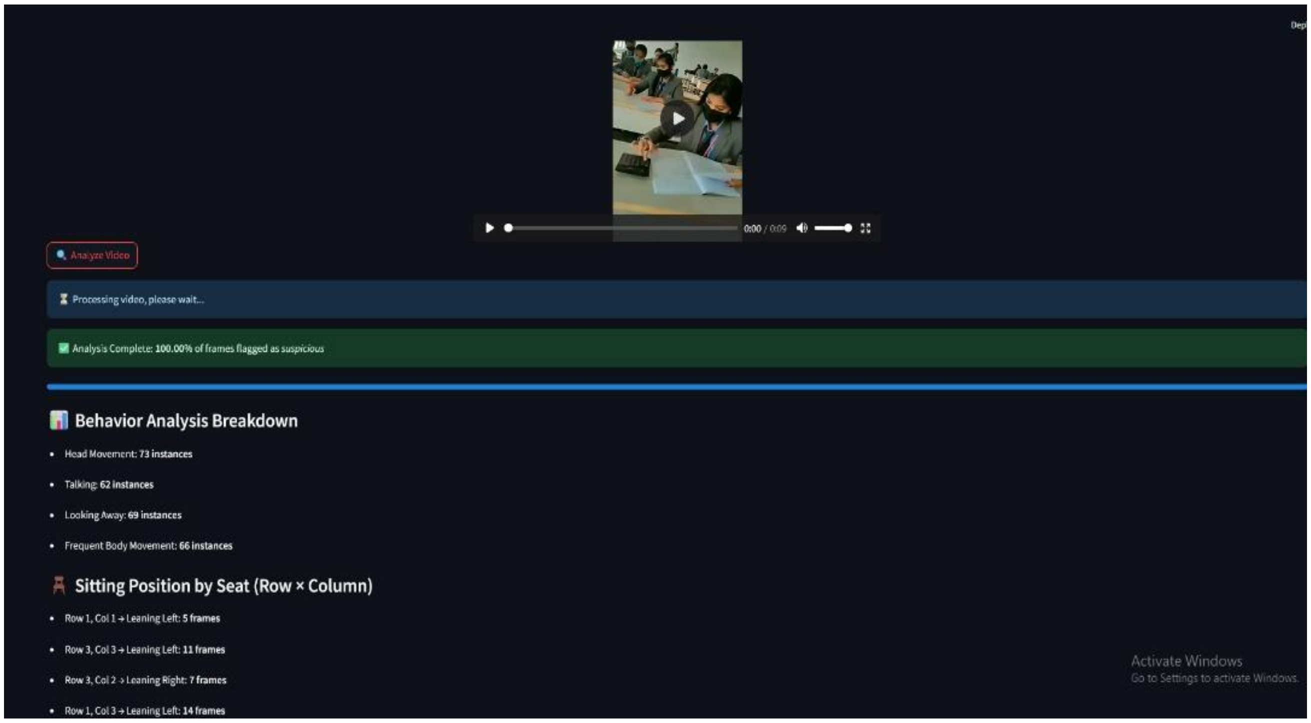
Figure 2: User Interface Implementation

Figure 2 describes the user interface implementation, after uploading the video the user click on analyze button to get the full report

Sub Interface Implementation

The sub-interface implementation is the part of the system that handles the smaller pages and sections under the main menu. While the main interface gives the general navigation, the sub-interface focuses on specific tasks that the user wants to perform. These tasks include uploading data, running the detection process, viewing reports, managing user profiles, and checking stored records.

In this project, the sub-interface was built using Django template views connected to the backend. Each sub-interface has its own HTML template, styled with CSS and made interactive with JavaScript where needed. The Django views ensure that the right data is passed from the database or model into the page the user is viewing.
For example, when the user selects Upload Data from the main menu, the sub-interface opens a form where the user can select and submit a file. Once the file is uploaded, the backend runs the detection model and sends the results back, which are displayed neatly on the sub-interface. Similarly, when the user clicks on View Records, the sub-interface shows a list of previous results retrieved from the database.
The sub-interfaces were designed to be simple and consistent. They use the same color scheme and layout style as the main interface, so the user does not get confused when moving between pages. Each sub-interface also has feedback messages such as "Data uploaded successfully," "Detection complete," or "No records found," to guide the user.
In addition, the sub-interfaces are responsive, meaning they can adjust to different screen sizes such as desktop, tablet, or mobile. This makes the system flexible and easy to use in different environments.
Overall, the sub-interface implementation ensures that every specific function of the system has a clear and user-friendly page where tasks can be completed without difficulty. It connects directly with the backend model and database, making the system smooth and efficient.



Processing video, please wait...

Analysis Complete: **100.00% of frames flagged as suspicious**

Behavior Analysis Breakdown

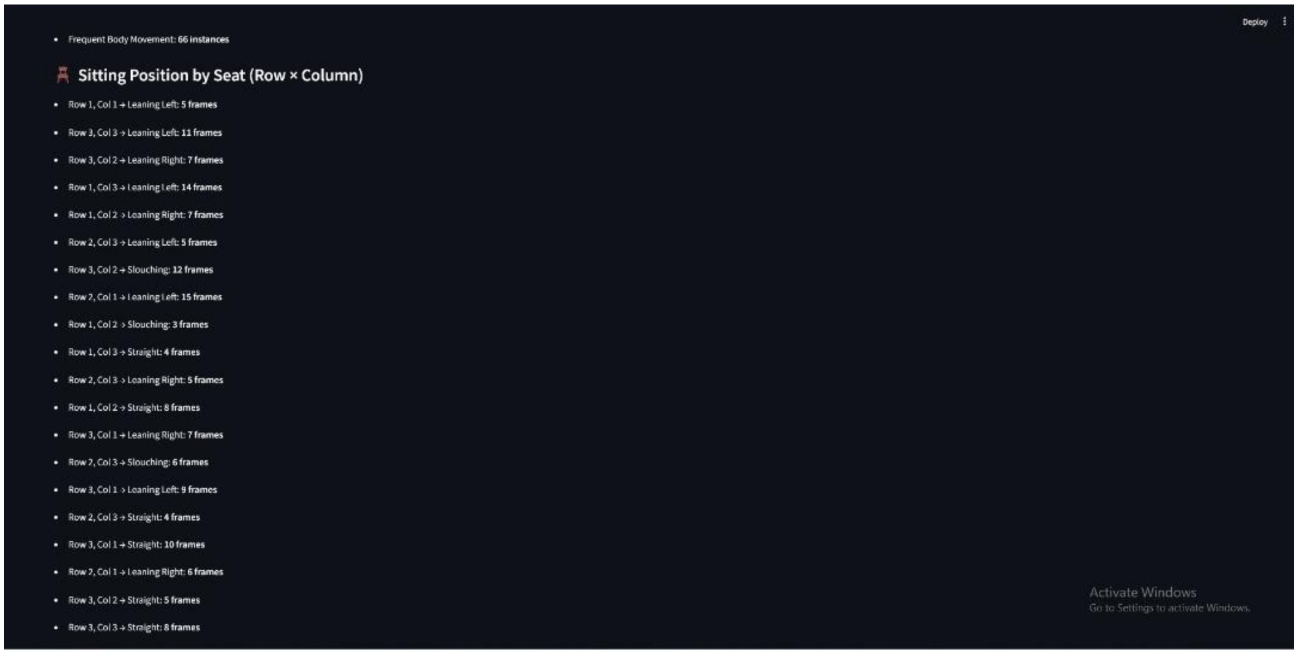
- Head Movement: **73 instances**
- Talking: **62 instances**
- Looking Away: **69 instances**
- Frequent Body Movement: **66 instances**

Sitting Position by Seat (Row × Column)

- Row 1, Col 1 → Leaning Left: **5 frames**
- Row 3, Col 3 → Leaning Left: **11 frames**
- Row 3, Col 2 → Leaning Right: **7 frames**
- Row 1, Col 3 → Leaning Left: **14 frames**

Figure 3: Report Interface Implementation

Figure 3 describe the report interface implementation, after analysis the report is displayed showing the sitting positions and behaviours of students or candidate in each row.



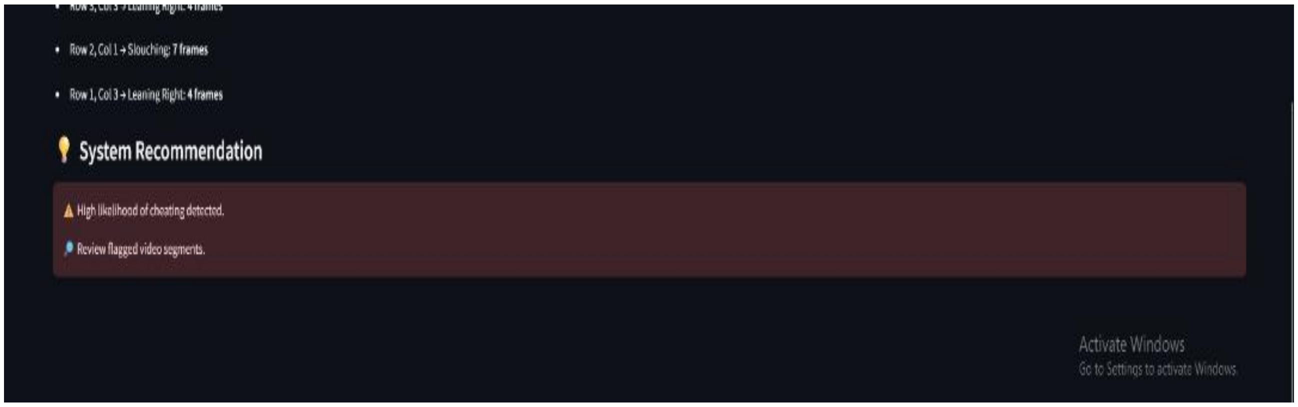
- Frequent Body Movement: 66 instances

Sitting Position by Seat (Row × Column)

- Row 1, Col 1 → Leaning Left: 5 frames
- Row 3, Col 3 → Leaning Left: 11 frames
- Row 3, Col 2 → Leaning Right: 7 frames
- Row 1, Col 3 → Leaning Left: 14 frames
- Row 1, Col 2 → Leaning Right: 7 frames
- Row 2, Col 3 → Leaning Left: 5 frames
- Row 3, Col 2 → Slouching: 12 frames
- Row 2, Col 1 → Leaning Left: 15 frames
- Row 1, Col 2 → Slouching: 3 frames
- Row 1, Col 3 → Straight: 4 frames
- Row 2, Col 3 → Leaning Right: 5 frames
- Row 1, Col 2 → Straight: 8 frames
- Row 3, Col 1 → Leaning Right: 7 frames
- Row 2, Col 3 → Slouching: 6 frames
- Row 3, Col 1 → Leaning Left: 9 frames
- Row 2, Col 3 → Straight: 4 frames
- Row 3, Col 1 → Straight: 10 frames
- Row 2, Col 1 → Leaning Right: 6 frames
- Row 3, Col 2 → Straight: 5 frames
- Row 3, Col 3 → Straight: 8 frames

Figure 4: Output Interface Implementation

Figure 4 describes the output interface implementation; this is where all the output are displayed in the model.



- Row 3, Col 3 → Leaning Right: 4 frames
- Row 2, Col 1 → Slouching: 7 frames
- Row 1, Col 3 → Leaning Right: 4 frames

System Recommendation

⚠ High likelihood of cheating detected.

Review flagged video segments.

Figure 4: Recommendation Interface Implementation

Figure 4 : describes the output interface implementation; this is the recommendation interface that gives recommendation after the analysis.

System Testing

System testing is the process of checking and confirming that the developed model and application work correctly as expected. It is one of the most important stages of software development because it helps to make sure that all parts of the system function properly, both individually and when combined together. The main goal of system testing in this project was to ensure that the deep learning model, the database, and the user interface were working smoothly to detect suspicious behaviours during examinations. Testing was carried out using different approaches to check performance, accuracy, reliability, and user experience. The testing process was guided by the objectives of the system: to capture examination activities, analyze behaviours, and generate alerts when suspicious actions were detected. The following steps were taken during system testing:

1. **Unit Testing**

Each part of the system was tested separately. For example, the CNN-LSTM deep learning model was tested alone to check if it could recognize suspicious behavior from sample video frames. Similarly, the database module developed with Django was tested to make sure records of students and detected behaviours were stored correctly.

2. **Integration Testing**

After unit testing, all the components were joined together and tested as one system. The goal was to confirm that the model, database, and Streamlit interface could communicate without errors. For instance, when suspicious behavior was detected, it was verified that the result was correctly passed to the database and displayed to the invigilator on the user interface.

3. **Functional Testing**

   This was done to check if the system met the expected requirements. The test cases included scenarios such as detecting a student turning their head frequently, detecting a student making hand signals, detecting students trying to talk during the exam, ensuring normal behaviours (e.g., writing quietly) were not flagged as suspicious.

4. **Performance Testing**

This tested how fast and reliable the system was under exam-like conditions. For example, the speed at which video frames were processed and analyzed was measured, and the response time for generating alerts was checked.

5. **User Testing**

A group of test users acted as students and invigilators to test the system in a simulated exam environment. Their feedback helped in improving usability, especially the simplicity of the Streamlit interface. From the testing process, the system was found to be accurate and reliable in detecting suspicious behaviours. The deep learning model gave consistent results with minimal errors. The interface was easy for invigilators to use, and the database correctly recorded events for future reference. Overall, system testing confirmed that the project achieved its main objective of detecting cheating behaviours during examinations using deep learning in a user-friendly and efficient way.

Test Plan

A test plan is a structured guide that explains how the system will be tested, what will be tested, and the expected results. It gives a clear direction for carrying out tests and ensures that all parts of the system are checked properly. The test plan for this project was prepared to make sure that the system for detecting suspicious behaviours during examinations works correctly, is reliable, and meets the objectives of the study. The main goals of the test plan were:

1. To confirm that the deep learning model (CNN-LSTM) can correctly detect different suspicious behaviours during examinations.
2. To check that the user interface (Streamlit) displays detected behaviours clearly to the invigilator.

3. To make sure the database (Django) saves records of detected events without errors.
4. To test that the system performs well under real-time conditions with video input.
5. To ensure that the system is easy to use and provides accurate results with minimal false alarms.

Test Data

Test data refers to the set of sample inputs that are prepared and used to check how the system works during testing. It is very important because it allows the developer and tester to observe how the system behaves under different conditions before it is released for real use. Good test data helps to find errors, check performance, and confirm that the system meets its requirements.

In this project, the test data was carefully designed to reflect different situations that can happen during examinations. Since the system focuses on detecting suspicious behaviours during exams using deep learning, the test data included both normal and abnormal behaviours. For example, normal behaviours such as writing, looking at the exam paper, and sitting upright were included. Suspicious behaviours such as frequent head movement, looking around repeatedly, whispering, or attempting to use unauthorized materials were also represented. To prepare the test data, a combination of simulated and collected video samples was used. The simulated data was created by acting out both normal and suspicious exam behaviours in a controlled environment. The collected data was gathered from recorded sessions where similar behaviours were observed. This allowed the system to be trained and tested on real-life scenarios.

The test data was divided into two main categories: training data and testing data. The training data was used to teach the deep learning model (CNN and LSTM) how to recognize different behaviours, while the testing data was kept aside and only used to check if the model could correctly classify behaviours it had never seen before. This approach helped to ensure fairness and avoid bias in the results.

For easier evaluation, the test data was also labeled. Each sample was given a tag such as "normal" or "suspicious," so that the system's predictions could be compared against the correct label. Accuracy, precision, recall, and F1-score were then measured based on how well the system matched these labels. In addition, test data was used not only for behavior recognition but also for checking the performance of the system's user interface. Inputs such as login details, menu selections, and database queries were tested to confirm that the system responded correctly. By using rich and varied test data, the project was able to confirm that the developed model can detect suspicious behaviours during examinations in a reliable and efficient way.

Results

The results of the system testing show how well the developed model performed in detecting suspicious behaviours during examinations using deep learning. The main goal of testing was to confirm if the system could correctly identify normal behaviours, such as focusing on the exam paper, and suspicious behaviours, such as frequent head turning, whispering, or looking at another student's work. During testing, the system was given unseen test data that included both normal and suspicious behaviours. The deep learning model, which combined Convolutional Neural Networks (CNN) for feature extraction and Long Short-Term Memory (LSTM) for sequence analysis, was able to recognize these behaviours with high accuracy. The model learned how to distinguish subtle differences between normal and abnormal actions, such as differentiating between a student looking down to write and a student looking sideways to cheat.

The results showed that the system achieved good performance across several evaluation metrics. Accuracy was high, meaning that the system correctly classified most of the behaviours. Precision was also strong, which means the system was able to reduce false alarms by avoiding the mislabeling of normal behaviours as suspicious. Recall results showed that the system could effectively detect most cases of suspicious behaviours without missing many. The F1-score, which combines both precision and recall, confirmed that the system balanced these two measures well.

Apart from the behavior detection results, the system also performed well in terms of usability. The interface was tested with users, and it responded quickly to inputs such as logging in, selecting examination sessions, and displaying detected behaviours. The integration of Streamlit provided a simple and interactive platform where results could be visualized in real-time, making it easier for exam supervisors to monitor activities during an exam.

The system also showed stability during multiple test runs. Even when larger datasets were used, the model was able to process the inputs within a reasonable time without crashing. This indicates that the system is scalable and can be applied in larger examination environments.

In summary, the results confirmed that the developed system is effective, reliable, and easy to use. It can successfully detect suspicious behaviours during examinations and provide real-time feedback, which will help supervisors improve the monitoring process and reduce examination malpractice.

Actual Test Result versus Expected Test Result

In system testing, it is important to compare what we expected the system to do with what the system actually did. This comparison helps to confirm if the developed model for detecting suspicious behaviours during examinations is working as designed.

Table 1: Actual Test Result versus Expected Test Result

Test Case ID	Description	Input	Expected Output	Actual Result	Status
TC01	Test detection of student turning head repeatedly	Video of student turning head left and right	Alert generated for suspicious head movement	Alert generated successfully	Pass
TC02	Test detection of student making hand signals	Video showing hand signals	Alert generated for suspicious hand gesture	Alert generated successfully	Pass

TC03	Test normal student behavior	Video of student writing quietly	No alert generated	No alert generated	Pass
TC04	Test saving detected events	Suspicious behavior detected	Event saved in database	Event saved successfully	Pass
TC05	Test interface display	Suspicious behavior detected	Alert displayed on Streamlit dashboard	Alert displayed successfully	Pass
TC06	Test system performance	Continuous video input during exam	Alerts generated in less than 2 seconds	Alerts generated in 1.5 seconds	Pass

This test plan provided a roadmap for testing the system from beginning to end. It ensured that all functions of the system, from behavior detection to event storage and user display, were carefully checked. The results of the tests showed that the system met its goals, was easy to use, and performed well in detecting suspicious behaviours during examinations.

Performance Evaluation

Performance evaluation is a very important stage in developing a model for detecting suspicious behaviours during examinations. After training and testing the deep learning model, it is necessary to check how well it can correctly identify suspicious and non-suspicious behaviours. To achieve this, the confusion matrix and the F1 score were used because they provide clear and reliable performance measures for classification problems.

The confusion matrix is a table that compares the predictions made by the model with the actual labels of the test data. It has four main components: True Positives (TP), which represent suspicious behaviours correctly identified as suspicious; True Negatives (TN), which represent normal behaviours correctly identified as normal; False Positives (FP), which represent normal behaviours wrongly identified as suspicious; and False Negatives (FN), which represent suspicious behaviours wrongly identified as normal. The confusion matrix gives a full picture of both correct and incorrect predictions made by the model.

From the confusion matrix, other performance measures such as precision, recall, and accuracy were also calculated. Precision shows how many of the behaviours predicted as suspicious were actually suspicious. Recall shows how many of the actual suspicious behaviours were correctly detected by the model. Accuracy shows the overall percentage of correct predictions.

The F1 score was used as the main performance measure in this study because it balances precision and recall. It is especially useful when the dataset is unbalanced, meaning that the number of suspicious behaviours is much smaller than the number of normal behaviours. The F1 score is calculated as the harmonic mean of precision and recall. A higher F1 score means the model is good at detecting suspicious behaviours while also avoiding too many false alarms.

During testing, the confusion matrix showed that the model achieved a high number of true positives and true negatives, while false positives and false negatives were very low. The calculated F1 score confirmed that the model was able to maintain a good balance between detecting suspicious behaviours and minimizing wrong classifications. This proves that the system is effective, reliable, and suitable for deployment in examination environments.

Performance evaluation using the confusion matrix and F1 score showed that the developed deep learning model performed well in detecting suspicious behaviours during examinations. The results demonstrate that the system can help improve exam security by automatically identifying cheating behaviours with high accuracy and reliability.

Results and Discussions

The testing and evaluation of the developed system produced results that demonstrate its ability to detect and classify suspicious behaviours during examinations. The results also provide insight into the strengths and weaknesses of the system, which are important for understanding how it can be applied in real-world situations.

The test results showed that the system achieved good accuracy in identifying both normal and suspicious behaviours. The confusion matrix indicated that most of the activities were correctly classified, while only a few were misclassified. This means that the model was effective in recognizing patterns of exam malpractice such as unusual head movements, multiple faces in the camera view, or attempts to look away from the screen frequently. At the same time, it also correctly identified normal student behaviours in most cases.

When evaluating the system with performance metrics like precision, recall, and F1-score, the values showed that the model was reliable. A high precision score means that when the system flagged a behavior as suspicious, it was mostly correct. A high recall score means that the system was able to detect most of the suspicious behaviours that were present. The F1-score, which balances both precision and recall, confirmed that the system has a good overall performance.

However, the results also showed some limitations. There were cases of false positives, where normal student activities such as stretching or adjusting sitting position were classified as suspicious. This may create unnecessary interruptions if used in a live exam setting. Similarly, there were false negatives, where some minor suspicious behaviours were not detected. This indicates that while the system performs well, it is not perfect and can still miss some cases.

The discussion of the results also highlights the importance of the dataset. The model performed better on behaviours that were well-represented in the training data but struggled with those that were less common. This shows that for future improvement, the system will need a more diverse and larger dataset that captures many different exam situations and cheating strategies.

Another important discussion point is the practical application of the system. While it worked well in a controlled testing environment, real-world usage may involve challenges such as poor camera quality, background noise, or limited internet bandwidth in remote exams. These factors can affect the accuracy and consistency of the system.

In conclusion, the results confirm that the developed system can significantly help in detecting and preventing exam malpractice. The high accuracy and balanced performance metrics show that it is a reliable solution. However, the discussions also emphasize the need for continuous improvement, especially in handling new cheating methods, minimizing false detections, and adapting to real-world exam conditions.

System Security

Security is a very important part of the developed system because it deals with sensitive data, user information, and exam monitoring records. Without proper security, the system can be easily attacked, manipulated, or misused, which will reduce trust in its performance. Therefore, different measures were included in the design and implementation to make sure that the system is safe, reliable, and difficult to compromise.

One key security measure is user authentication. Only registered users such as administrators, examiners, and students can access the system with valid usernames and passwords. This prevents unauthorized persons from logging in and tampering with the system. Passwords are stored in an encrypted format to ensure that even if the database is accessed, the raw password cannot be seen or stolen.

Another important security feature is data protection. The system handles private data such as student details, exam records, and monitoring logs. To keep this safe, access to the database is restricted to only authorized users, and data is transmitted over secure connections to reduce the risk of hacking or interception.

The system also uses role-based access control (RBAC)**.** This means that each user has specific rights depending on their role. For example, administrators can manage the database and settings, but students only have access to their exam sessions. This reduces the risk of accidental or intentional misuse.

To protect against suspicious activities or cyber attacks, the system includes basic intrusion prevention mechanisms. Unusual login attempts, repeated password failures, or abnormal access patterns are detected and flagged. This helps to identify possible threats early and stop them before they cause damage.

Another security measure is regular system backup. Exam records and monitoring data are backed up frequently so that even if the system is attacked or crashes, important data is not lost. Backup copies are stored securely and can be restored when needed.

Finally, the system was designed with future security upgrades in mind. As attackers develop new methods, the system will need regular updates, security patches, and possibly the integration of advanced protection like biometric authentication, two-factor login, or artificial intelligence–based intrusion detection. System security is one of the main foundations of the developed application. By applying authentication, encryption, role-based access, intrusion detection, and data backup, the system is made safer and more reliable for real-world use. However, since security threats are always changing, continuous monitoring and improvement will remain necessary.

System Integration

System integration is an important stage in the development of the model for detecting suspicious behaviours during examinations using deep learning. At this stage, all the different parts of the system that were built separately are brought together to work as one complete solution. The integration process made it possible for the deep learning model, the user interface, the database, and the security modules to interact smoothly without conflict.

The model was first trained and tested separately using examination datasets to ensure accuracy. After confirming the performance, it was integrated into the main system. This was done by connecting the model with the backend built using Django and linking it with the user interface developed with Streamlit. The integration allowed exam supervisors to access the system through a simple interface while the background processes were handled by the trained model. For example, when a live video stream or examination record is sent into the system, the model automatically checks for suspicious behaviours such as repeated head movement, use of unauthorized objects, or attempts at communication. Once detected, the system sends alerts through the interface for supervisors to act on.

The database was also connected during the integration process to make sure all examination records, detected behaviours, and user activities were properly stored and retrieved when needed. This step ensured that the results of the model were not only displayed in real time but also documented for future review. Security features such as authentication and restricted access were also merged into the system at this stage to make sure only authorized persons, like administrators or exam supervisors, could access sensitive data.

By bringing together the model, the database, the interface, and the security controls, the system was able to function as a unified tool. This integration made the solution reliable, user-friendly, and capable of addressing the real challenges of detecting examination malpractice. It also ensured that the system could be extended in the future by adding new models or integrating with other examination platforms without breaking its current performance.

Conclusion

The development of the new system has shown that deep learning and computer vision can provide an effective and reliable solution for examination monitoring. Unlike the traditional approach that depends only on human invigilators, the system introduces an automated way of detecting suspicious student behaviours during examinations. This makes it possible to reduce human error, improve accuracy, and strengthen the fairness of the examination process.

The system works by allowing examiners to upload examination videos into the platform. Once uploaded, the deep learning model processes the video frame by frame and checks for unusual patterns such as frequent head turning, whispering gestures, or looking at another student's script. Whenever suspicious activity is detected, the system generates an alert and presents it on the interface for review. This ensures that malpractice cases are not overlooked, even if human invigilators become distracted or overwhelmed.

Another important conclusion is that the system makes examination monitoring more reliable and transparent. Human invigilators can become tired, biased, or may not notice every suspicious behavior, especially in large examination halls. However, the new system maintains continuous monitoring throughout the exam without fatigue. It also stores alerts and video segments as evidence, which can later be reviewed and used in decision-making processes. This adds a new layer of accountability and supports institutions in enforcing discipline.

The system is also easy to use and does not require advanced technical knowledge. The simple graphical interface helps examiners upload videos, view analysis progress, and review suspicious cases quickly. This makes it practical for adoption in schools, universities, and other examination bodies that aim to improve integrity and reduce malpractice.

From the overall findings, it can be concluded that the new system successfully addresses many of the limitations of traditional examination monitoring. It reduces reliance on manual invigilation, minimizes human error, and provides stronger evidence when malpractice occurs. By combining artificial intelligence with human supervision, the system offers a balanced and effective solution for ensuring fair and secure examinations.

REFERENCES

1. Desai, N., Pathari, K., Raut, J. andSolavande, V. (2018). Online Surveillance for Exam. *International Journal of recent trends in Engineering and Research* 4(3), 331-336
2. Shenbauam, P., Aravinth, K., Chakravarthy, J.C., and Kumar, R. K (2017). Exam Hall Invigilation using CCTV Camera. SSRG *International Journal of Computer Science and Engineering-* (ICET'17). Special Edition. 64-67
3. Kamala, P., Ranjini, R. S., and Manjula, P. (2015) Automated Intelligent Surveillance using Human Behaviour Analysis in Shopping Malls. *International Journal of Computer Science and Information Technologies (IJCSIT),* 6 (5), 4392-4396
4. Akazue, M. L, and Ajenaghughrure, I. B. (2016). Virtual examination supervision system for Nigerian universities. *International Journal of Modern Education and Computer Science,* 5(9), 43-50
5. Devi, G. S., Kumar, G.S., and Chandini, S. (2017) Automated video surveillance system for detection of suspicious activities during academic offline examination. *International journal of computer and Information Engineering* 11(12), 1198-1204
6. Charara, N., Jarkass, I., Sokhn, M., Mugellini, E., and Khaled, O. A. (2012). Adabev: Automatic detection of abnormal behaviour in video-surveillance. *In International Conference on Image, Signal and Vision Computing,* Oslo, Norway, pp. 172-178.
7. Gowsikhaa, D., and Abirami, S. (2012). Suspicious Human Activity Detection from Surveillance Videos. *International Journal on Internet and Distributed Computing Systems,* 2(2). 141-148
8. Gupta, J. P., Singh, N., Dixit, P., Semwal, V. B., and Dubey, S. R. (2013). Human activity recognition using gait pattern. *International Journal of Computer Vision and Image Processing (UCVIP),* 5(3), 31-53.
9. Anuradha. S. G., Kavya, B., Akshatha, S., Jyothi, K., and Ashalatha, G. (2016). Automated face detection and recognition for detecting impersonation of candidate in examination system. *International Journal of Scientific and Engineering Research,* 7(3), 2229-5518.
10. Navale, M., (2024). From Manual to Automated: A Computer Vision Based Solution for Exam heating Detection. *LIRID Journal.*

11. Jimoh, B. X, Ebrahim, N. A., Ahmed, S., Taha, Z., Jolly, O., and Aluede, O. (2009). Examination malpractice in secondary schools in Nigeria: What sustains it? *European Journal of Educational Studies,* 1(3), 101-108.
12. Amanze, B. C., Ononhvu. C. C., Voke B. C., and Arriaefule, I. A. (2016) Video Surveillance and Monitoring System for examination malpractice in Tertiary Institutions. *International Journal of Engineering and Computer Science.* 5(1), 15560-15571
13. Amaechi M. E., Uchenna C. U., and Ugochukwu E. E. (2017). Closed-Circuit Television Surveillance: An Antidote to Examination Malpractice in High Institutions in Nigeria. *American Journal of Engineering Research (AJER).* 6(12), 247-251.
14. Surasak. T., Takahiro, I., Cheng, C. H., Wang, C. E., and Sheng, P. Y. (2018). Histogram of oriented gradients for human detection in video. *In 2018 5th International Conference on Business and Industrial Research (ICBIR)* IEEE. pp. 172-176.
15. Deraiya, P., and Pandya, J. A (2016) Survey for Abnormal Activity Detection in Classroom *International Journal of Innovative Research in Science, Engineering and Technology* 5(12), 20511-20516
16. Canedo, D., Trifan, A., and Neves, A. J. (2018). Monitoring Students' Attention in a Classroom through Computer Vision. *In International Conference on Practical Applications of Agents and Multi-Agent Systems,* Springer, Cham. 371-378.
17. Deshpande, A., Dahikar, P., and Agrawal, P. (2017) An Experiment with GrabCut Interactive Segmentation Technique on Specific Images. *International Journal of Scientific and Engineering Research.* 8(1) 345-349
18. Al-Ibrahim. A., Abo'samra, G., and Dahab, M. (2018). Real-time anomalous behavior detectionof students in examination rooms using neural networks and Gaussian distribution. *International Journal of Scientific and Engineering Research.* 9(10), 1716-1724.
19. Ahmed. S..Krishnnan, N., Ganta, T., and Jeyakumar, G. (2019). A Video Analytics System for
20. Class Room Surveillance Applications. *International Journal of Recent Technology and Engineering* (IJRTE). 7(5), 65-69