Proctoring Online Exam Using Eye Tracking

Waheeb Yaqub[1], Manoranjan Mohanty[2] and Basem Suleiman[3]
[1]*School of Computer Science, The University of Sydney, Australia*
[2]*Center for Forensic Science, University of Technology Sydney, Australia*
[3]*School of Computer Science and Engineering, University of New South Wales, Australia*

Keywords: Online Teaching, Online Proctoring, Student's Privacy.

Abstract: Online proctoring is required for online teaching. Typically, third-party video-based crowd-sourced online proctoring solutions are being used for monitoring exam-takers (e.g., students). This approach, however, has privacy concerns as an exam-taker's face is shown to the third-party provider. In this paper, we propose to address this concern using face hiding, and then monitoring the face hidden exam takers via eye (gaze) tracking. The eye tracking is used to detect if the exam-taker is reading from computer screen, e.g., from ChatGPT. The face is hidden by exposing the eyes such that eye tracking is possible.

1 INTRODUCTION

The dramatic increase in online teaching necessitates online proctoring for exam-takers. Proctoring a large number of exam-takers online is daunting for educational organizations, such as universities. As a result, universities are outsourcing the proctoring task to third-party companies like ProctorU. Figure 1 demonstrates how employees remotely monitor exam-takers by reviewing videos of the exam rooms.

Figure 1: A widely used proctoring system, ProctorU, proctors are monitoring exam takers. (from (Dimeo, 2017)).

Outsourcing the proctoring task to a third-party company raises privacy concerns (Nigam et al., 2021; Furby, 2020). Exam-takers' faces and background information in the videos are readily available to the company employees. There is a risk of these videos being leaked to the public, including social media (Balash et al., ; Milone et al., 2017). Consequently, some exam-takers are uncomfortable sharing their videos with third-party proctors.

One way to address privacy concerns is by blurring or masking the face of an exam-taker (Yaqub et al., 2022). However, it is crucial to ensure that proctoring is still possible. Yakub et al. (Yaqub et al., 2022) previously proposed a method to hide the face of an exam-taker while enabling proctoring through observing their body movements. However, their work did not consider cheating by the exam-taker through reading from another computer screen. In this paper, we propose a proctoring approach that detects such cheating by identifying when the exam-taker is reading from another computer screen. We consider this behavior as an anomaly and detect it using eye-tracking. We do not hide the eyes of the exam-taker, only the face.

One major challenge in this area of research is the lack of a public exam-taking video dataset. Similar to Cote et al.'s (Cote et al., 2016) work, we systematically collect an in-house dataset of five exam-taking and cheating-attempting videos for our study. Experimental results demonstrate that the proposed scheme outperforms the work of Yaqub et al. (Yaqub et al., 2022), which is one of the pioneering studies in this field. The proposed work represents an initial attempt to address a new and practical research problem: privacy-preserving online proctoring. Further research is necessary to enhance the results, such as utilizing a larger exam-taking video dataset.

The rest of this paper is organized as follows: Sec-

Yaqub, W., Mohanty, M. and Suleiman, B.
Proctoring Online Exam Using Eye Tracking.
DOI: 10.5220/0012120600003555
In *Proceedings of the 20th International Conference on Security and Cryptography (SECRYPT 2023)*, pages 738-745
ISBN: 978-989-758-666-8; ISSN: 2184-7711

tion 2 discusses related work. Section 3 provides an overview of the proposed method. Section 4 explains the Initialization stage, Section 5 explains face hiding, and Section 6 explains anomaly detection. Section 7 presents experimental results. Section 8 concludes and discusses future work.

2 BACKGROUND AND RELATED WORK

Current online proctoring systems generally fall into three categories: Live Proctoring, Recorded Proctoring, and Automated Proctoring (Hussein et al., 2020). *Live Proctoring* is a real-time system where a crowd-sourced human proctor monitors students' activities during the exam through live webcams, as shown in Figure 1. It resembles on-campus exam proctoring, recording minimal information about the exam taker.

With the rise of computer vision deep learning models, a new form of live online proctoring has emerged, monitoring all the exam takers' movements automatically using a software tool (Conijn et al.,). The heavy reliance on such automated online proctoring solutions has also recently increased due to the COVID-19 pandemic.

Côté et al. (2016) proposed a video summarization method for remote proctoring of online exams. Their solution eliminates the need for a real-time proctor by detecting abnormal behavior through head-pose estimation and a two-state hidden Markov model (HMM). Suspicious snippets are then forwarded to proctors for further review. While addressing students' concerns about invasiveness, this approach raises privacy concerns as snippets expose students without any form of veiling (Balash et al.,).

The system developed by (Atoum et al., 2017) verifies the test-taker's identity by continuously matching their face with a database to prevent substitution during the exam. Text detection ensures the absence of textual resources in the user's surroundings, while speech detection aims to identify audible speech. To detect cheating on the user's computer, tracked windows include those currently opened by the user. Gaze estimation is used to detect anomalous eye movements. However, unlike (Yaqub et al., 2021), who relied solely on the webcam, this system utilizes both the webcam and a wearable camera. The portable camera is also employed to detect mobile phones within the user's field of view.

The work by Masud et al. aimed to develop a fully automated exam proctoring assistance system (Masud et al., 2022). The system relied solely on visual data to detect cheating. The classifier was trained to detect cheating based on a multi-variate time series. To evaluate its performance, they collected 20 non-cheating and cheating behavior sample videos, each a few seconds long and consisting of frames varying from 75 to 250. Since the classifier required uniform-length training data, longer videos were split into shorter ones for evaluation. The videos were grouped based on length and evaluated individually against the classifier. Datasets with shorter videos consistently performed better, achieving at least 80% accuracy. However, as the video length increased, the performance noticeably declined. The dataset with videos of length 250 frames demonstrated accuracies ranging from 60% to 80%. This decline indicates the model's limited ability to handle longer videos as they are more likely to exhibit complex behavior, which was not considered during training or dependent on auditory input.

However, students' privacy has not yet been addressed by researchers. The systems by (Irfan et al., 2021; Masud et al., 2022; Cote et al., 2016; Atoum et al., 2017) and commercial solutions are typically built with the objective of maximizing cheating detection, and without consulting students about their concerns (Selwyn et al., 2021). The list of private information that an exam taker gives up can be unanticipated and intrusive. Some of the examples of information collected during online exams are audio, video, screen sharing, keyboard strokes, room panning videos, etc.

3 PROPOSED METHOD

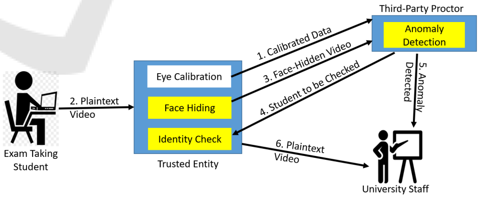


Figure 2: Architecture of the proposed system.

Figure 2 shows the overall architecture of the proposed system. There are four main players: a student (i.e., exam-taker), a trusted entity, an honest-but-curious third-party proctor, and a trusted university staff. We assume that the trusted entity can access the student's information, such as exam videos, photos, ID cards, in plain-text. This entity can either be present at the student-end (such as a trust-zone in student's computing device) or at the university-end (such as a highly secure dedicated machine). The third-party proctor is assumed to be honest-but-curious as it does its task honestly but can be curious to know information without any authorization. Communications between different entities are assumed to be secured.

Workflow: The proposed system consists of two main stages: the one-time initialization stage and the run-time eye-tracking-based anomaly detection stage. In both stages, the exam taker is required to switch on their webcam or selfie camera.

In the initialization stage, the exam taker's identity is first checked by the Identity Check module of the Trusted Entity to determine if they are enrolled for the exam, similar to an offline exam. Proxy exam takers are not allowed to take the exam. Enrolled exam takers then undergo a one-time eye calibration process using the Eye Calibration module before the start of the actual exam. This calibration is essential for understanding how the exam taker will interact with the computer screen during the exam.

After the calibration, the plain-text eye calibration data is sent by the Trusted Entity to the Third-Party Proctor (Step 1). During the live exam stage, the exam taker's live exam-taking video is sent to the Trusted Entity in plain-text (Step 2). The Face Hiding module of the Trusted Entity then hides the facial information to minimize privacy leaks. The face-hidden video is sent to the Third-Party Proctor (Step 3).

The Third-Party Proctor runs anomaly detection on the face-hidden video to detect potential cheating. This anomaly detection tool serves as a triaging tool. If the exam taker is flagged by this tool, they are reported to the Trusted Entity for another round of identity check, which is carried out automatically. This is to ensure that a proxy has not replaced the exam taker after their identity was previously verified. If a proxy is found, their plain-text video clip is sent to University Staff for further action (Step 6).

If a proxy is not found, the flagged exam taker undergoes another round of manual check for malpractice at the third-party proctor's end by reviewing the video clip showing the potential malpractice. Those confirmed by the third-party proctor are reported to the trusted university staff (Step 5). University staff obtain the plain-text video clip from the Trusted Entity (Step 6) and take any further actions.

Our proposed system is made of various modules as shown in Figure 2. In the following sections, the details of the initialisation and anomaly detection will be discussed.

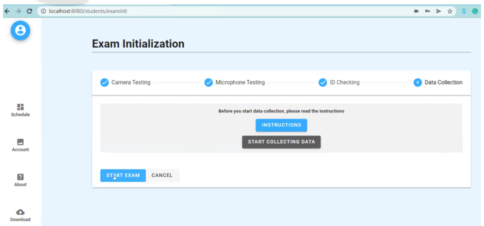
4 INITIALIZATION

4.1 Identity Check

This module can be divided into two parts: face verification and OCR. The face verification part confirms the identity of the examinee and detects surrogate exam-takers before the exam. The OCR part extracts the student ID number to establish a connection between the student sitting for the exam and the ID photos in the database. It also labels the student with their student ID number instead of their full name for anonymization purposes. Figure 4 depicts separate flowcharts for the ID verification part and the OCR part.

The flowchart for ID verification illustrates the process of face verification. First, the images of students' photos are fed into the face detection algorithm - MTCNN - to detect faces in the pictures. If faces are detected, the system proceeds to the next stage; otherwise, the process is terminated. Once the faces are detected, the system utilizes the pre-trained face recognition model - FaceNet - to convert them into 128-dimensional face embeddings. Subsequently, the system employs the Siamese network, trained from scratch, to improve data representation. Finally, the system computes cos-similarity scores to determine whether the examinee is genuine or if there is a surrogate exam-taker.

The flowchart for OCR illustrates the text detection/recognition algorithm. Initially, the images of students' ID cards are input into the text detection and recognition algorithm - EasyOCR - to detect the student IDs on the cards. If the correct IDs are detected, the system confirms the detection; otherwise, it prompts the human proctor to recheck.



localhost:8080/students/examinit

Exam Initialization

Before you start data collection, please read the instructions

Figure 3: The initialisation process in live system on student side interface.

4.1.1 Face Detection

In Figure 4, the system utilizes MTCNN to capture accurate bounding boxes of faces. The pre-trained face recognition algorithm, FaceNet, is then employed to extract face embeddings, which represent the features of faces. However, face embeddings

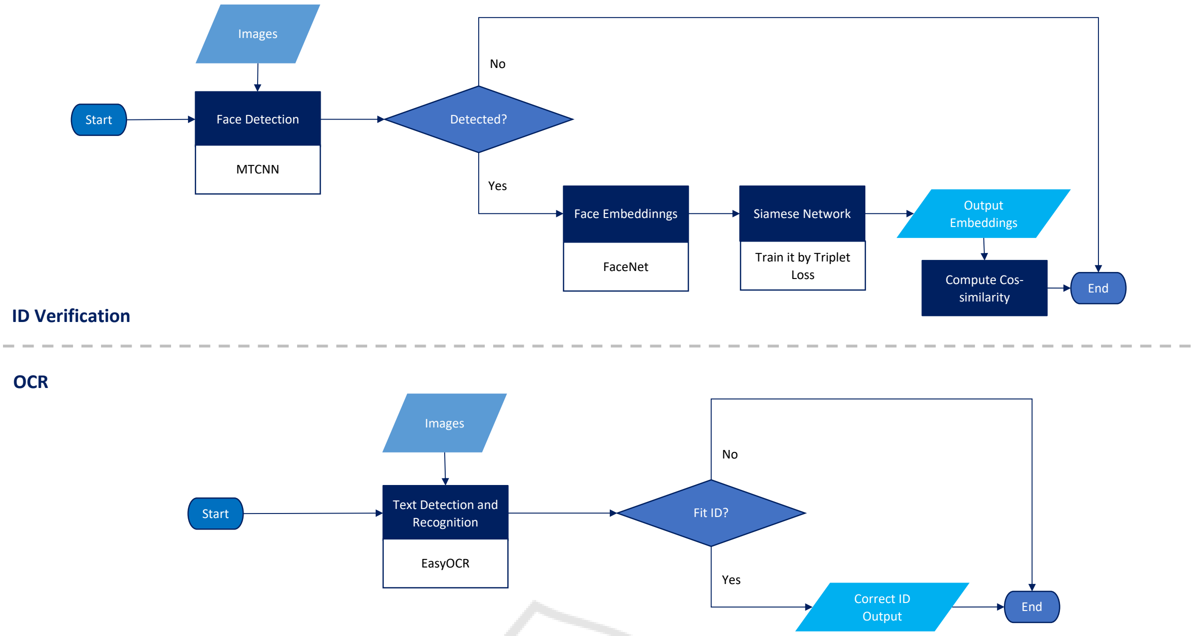


Figure 4: The detailed flowchart of ID verification module.

alone are insufficient for distinguishing different individuals' faces using cosine similarity scores. To improve data representation, a one-shot learning face recognition algorithm is applied to project the face embeddings onto another hyperplane.

The main objective is to establish the Siamese Network. There are three identical layers for every input face embedding, and the face embeddings are concatenated to compare their cosine similarity scores. The weight vectors are used to compare the face verification by computing the cosine similarity score. The Triplet loss training criterion is utilized for randomly selecting a batch of triplets: anchor, positive, and negative. The triplet loss is given by:

$$Triplet\ Loss = \sum_{i=1}^{N} [(f_i^a - f_i^p)^2 + (f_i^a - f_i^n)^2 + \alpha],$$

where f_i^a represents the output of the anchor data, f_i^p refers to the output of the positive data, f_i^n represents the output of the negative data, and α is a hyperparameter that separates the distance between the positive and negative data as much as possible. The hyperparameters are as follows: Input Dimension = 128, Output Dimension = 64, Batch size = 1000, Epochs = 100, Steps per epoch = 10, and $\alpha = 0.2$.

The anchor and the positive embedding must be of the same class, while the negative embedding must be of a different class. The core concept of the triplet loss is to minimize the difference between the weight vectors of the anchor and the positive, and maximize the difference between the anchor and the negative.

4.1.2 OCR

EasyOCR was selected to implement text detection and recognition in our system. The EasyOCR pipeline includes image input, pre-processing, CRAFT for detection models, mid-processing, ResNet + LSTM + CTC for recognition models, and post-processing for text output. However, EasyOCR's efficiency is low due to the long execution period. To address this issue, we implemented a flag mechanism. The fundamental concept of the flag mechanism is to terminate the process once the system detects the correct student ID, and continue capturing until the student's ID is detected if necessary.

The lower half of Figure 4 shows the flowchart of how the OCR system functions. The input is student ID images. After EasyOCR scans the student ID photos, the output data type is text, and the system compares the text with the student ID in the database. If the text matches the student ID in the database, the process will terminate. Otherwise, it will continue to search for the text that matches the student ID using the flag mechanism.

During the initialisation stage, students will use the ID verification UI system to take photos of their faces for ID verification purposes. This process aims to prevent exam surrogate takers from attending exams. The ID verification module will detect any exam surrogate takers present during the exams. Figure 3 illustrates the UI of the module functions during the initialization process.

4.1.3 Performance of ID Verification

The ID verification module is validated using a self-made test dataset to assess the performance of face verification and OCR algorithm. The dataset consists of five distinct student IDs, each with the following sampled frames and lengths: $P1_{id}$ (230 frames, 9 secs), $P2_{id}$ (271 frames, 17 secs), $P3_{id}$ (372 frames, 12 secs), $P4_{id}$ (285 frames, 9 secs), and $P5_{id}$ (304 frames, 10 secs).

4.2 Eye Calibration

The student's sitting pattern is initially recorded through her sitting position and eye interaction with the computer. The correct sitting position is established by displaying a real-time camera view to the student and instructing her to adjust her sitting position. A method similar to the approach proposed by Yaqub et al. (Krafka et al., 2016) is then employed to accurately record the eye interaction.

5 FACE HIDING

This module hides the student's facial information from a video to minimize privacy leaks. We use blurring and masking techniques to conceal the face (Figure 5). In anonymized videos, the eyes remain visible to facilitate anomaly detection, specifically cheating detection. Firstly, the face and eyes are detected, and then blurring or masking is applied. The eye detection and subsequent blurring or masking are performed using Yaqub et al.'s method (Yaqub et al., 2022). The eyes are not concealed to ensure gaze detection (eye tracking).

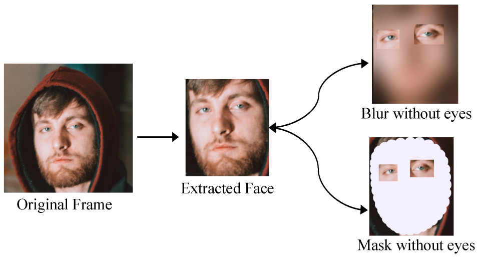


Figure 5: Frame-by-frame blurring or masking (Yaqub et al., 2022).

6 ANOMALY DETECTION

Anomaly detection is used to detect anomalies even when the student's face is blurred or masked. Gaze detection is used for this purpose. In the following section, we discuss gaze detection in detail.

6.1 Gaze Estimation

The gaze detection module determines if the student's gaze is within the screen boundaries. It utilizes the output of the gaze detection model and information about the examinee's screen size to assess if the examinee is looking beyond the physical range of the screen, which is considered anomalous behavior. The gaze estimation module relies on two sub-modules: iTracker and the calibration model.

iTracker submodule was trained using a dataset of single portrait photos taken by an Apple mobile device. The training was done using CNN. The core CNN-based neural network of iTracker is referred to as the iTracker model in the paper. Inputs to the model include left eye, right eye, and face images from the original frame, as well as a face grid calculated based on the spatial position of the face image. The final output of the iTracker model is the coordinate of the estimated gaze point on the photo frame. The spatial features prevented us from using the iTracker module directly. Therefore, we calibrated the output of iTracker for various laptop and desktop screens. The calibration model is a linear model that takes the raw output of the iTracker model as input and provides the calibrated point position as output.

Calibrating Gaze Estimation Module. We introduced a simpler linear calibration model compared to existing Support Vector Regression (Krafka et al., 2016). Unlike traditional iTracker, we expect the screen size of the examinee to vary at most 20 inches in the real world. To overcome this, we further improved performance by developing a personalized calibration model. For this, a single calibration model per examinee was used. We assessed the linear model using 25 pictures collected by us, corresponding to a 5 x 5 grid on the screen. Figures 6 and 7 show the estimated gaze location compared to the ground truth for subjects with and without glasses. All red points represent the raw prediction points of the iTracker, while the corresponding black points represent the ground truth points. We also employed a linear model to reduce incorrect gaze estimations by mapping the pixel locations to the closest original ground truth.

We examined various personalised linear calibration models based on the Euclidean distance metric (Krafka et al., 2016). The five models used to assess the effectiveness of personalised linear calibration are listed below, along with the results in Table 1.

I. *lm*(x, y): $X_f, Y_f \propto Linear(X_p, y_p)$

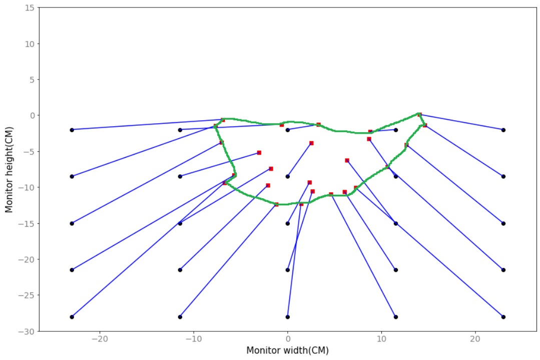


Figure 6: Incorrect estimated gaze location by Itracker because of screen size (subject1) without glasses.

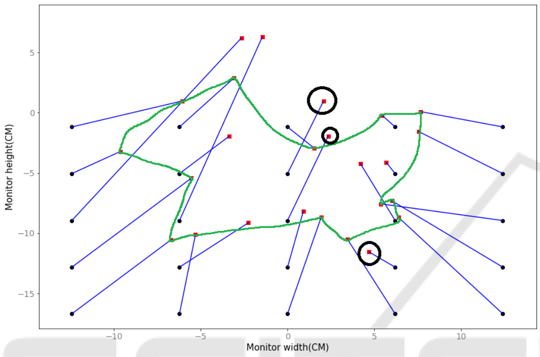


Figure 7: Incorrect and irregularities estimated gaze location by Itracker because of screen size and screen light reflection on glasses (subject1).

II. *lm*(x, y, nx): $X_f, Y_f \propto Linear(X_p, y_p, n_x)$
III. *lm*(x, y, ny): $X_f, Y_f \propto Linear(X_p, y_p, n_y)$
IV. *lm*(x, y, nx, ny): $X_f, Y_f \propto Linear(X_p, y_p, n_x, n_y)$
V. *lm*(x, y, nx, ny, nz): $X_f, Y_f \propto Linear(X_p, y_p, n_x, n_y, n_z)$

Data X_p and Y_p represent the raw output of the iTracker. n_x, n_y, and n_z represent the nose coordinates provided by Mediapipe. Nose coordinates are used to indicate the model's response to the examinee's head turning.

From the table, the most robust model is lm(x, y), which significantly reduces errors for situations with and without glasses-wearing examinee. The adjusted error is, on average, 42% of the raw error, equivalent to 3.97cm on a 14-inch-screen laptop. This error is considered acceptable on a large screen compared to mobile phones.

Gaze Estimation Anomaly Detection Module. The combination of the itracker model and the personalized calibration model serves as the gaze estimation module for the anomaly detection system. The performance of the gaze estimation module is tested using a self-made gaze video, as demonstrated in the upcoming experiment section. We assessed the prediction results of all videos and discovered a predictable pattern in the module. It tends to predict points closer to the coordinate origin, typically located at the top of the screen where the camera is positioned. For example, if the user is looking at the bottom edge of the screen, the predicted gaze point will be higher than the bottom edge. However, if the user is looking at the top edge, there is no such gap. Therefore, we improved the workflow of the gaze estimation module.

Table 1: Performance summary for 3 different anomaly detection modes with 2 different privacy preserving modes. (Acc. for Accuracy, Rec. for Recall, Pre. for Precision).

Black background GUI		Error(cm)	Calibrated error(cm)(Cross-Validation)				
		Raw	**lm(x,y)**	lm(x,y,nx)	lm(x,y,ny)	lm(x,y,nx,ny)	lm(x,y,nx,ny,nz)
Glasses	Subject 1	7.85	4.07	4.24	3.71	4.16	4.89
	Subject 1	7.86	4.3	4.27	5.16	5.21	5.71
	Error	7.855	4.185	4.255	4.435	4.685	5.3
	Error cut		46.72%	45.83%	43.54%	40.36%	32.53%
	Subject 2	4.03	3.02	3.24	2.96	3.05	3.12
	Subject 2	3.78	2.91	2.96	3.09	3.34	3.16
	Error	3.905	2.965	3.1	3.025	3.195	3.14
	Error cut		24.07%	20.61%	22.54%	18.18%	19.59%
	Subject 3	17.17	6.62	5.97	5.73	5.2	5.04
	Subject 3	16.37	5.49	5.07	7.6	6.52	6.58
	Error	16.77	6.055	5.52	6.665	5.86	5.81
	Error cut		63.89%	67.08%	60.26%	65.06%	65.35%
Glasses Free	Subject 1	8.33	4.92	5	5.24	5	16.13
	Subject 1	6.88	4.76	7.17	68.36	73.72	66.11
	Error	7.605	4.84	6.085	36.8	39.36	41.12
	Error cut		36.36%	19.99%	-383.89%	-417.55%	440.70%
	Subject 2	4.4	2.91	4.29	4.35	5.31	5.95
	Subject 2	3.32	3.61	2.35	3.35	3.49	7.41
	Error	3.86	3.26	3.32	3.85	4.4	6.68
	Error cut		15.54%	13.99%	0.26%	-13.99%	-73.06%
	Subject 3	14.25	4.45	6.33	4.28	6.35	6.24
	Subject 3	14.1	5.08	5.35	4.96	5.22	5.16
	Error	14.175	4.765	5.84	4.62	5.785	5.7
	Error cut		66.38%	58.80%	67.41%	59.19%	59.79%
Mean	Error cut		**42.16%**	37.72%	-31.65%	-41.46%	-56.08%

The image dataset is divided into a training set and test set. The calibration model is trained on the training set and applied to the test set to estimate the error. This estimated error is then used to adjust the left, right, and bottom edges of the screen, forming the anomaly decision boundary for the gaze estimation anomaly detection module. By using the anomaly decision boundary, the recall increases from 13% to 88% without affecting the F1-score compared to the results obtained with the raw screen edges.

The practicability of the gaze estimation anomaly detection module in the simulated test video has been discussed. Figure 8 shows the flow chart of this module.

- In the first stage, the training set provided by the examinee is used to build a personalized calibration model. Gaze pictures are processed with their corresponding face and eye bounding boxes provided by the privacy-preserving module. The output of this step is the input for iTracker. The raw coordinate prediction results of iTracker are grouped with the corresponding ground truth labels for training the personalized calibration model of the examinee. The calibration model and the iTracker model are then combined to form the gaze estimation module.

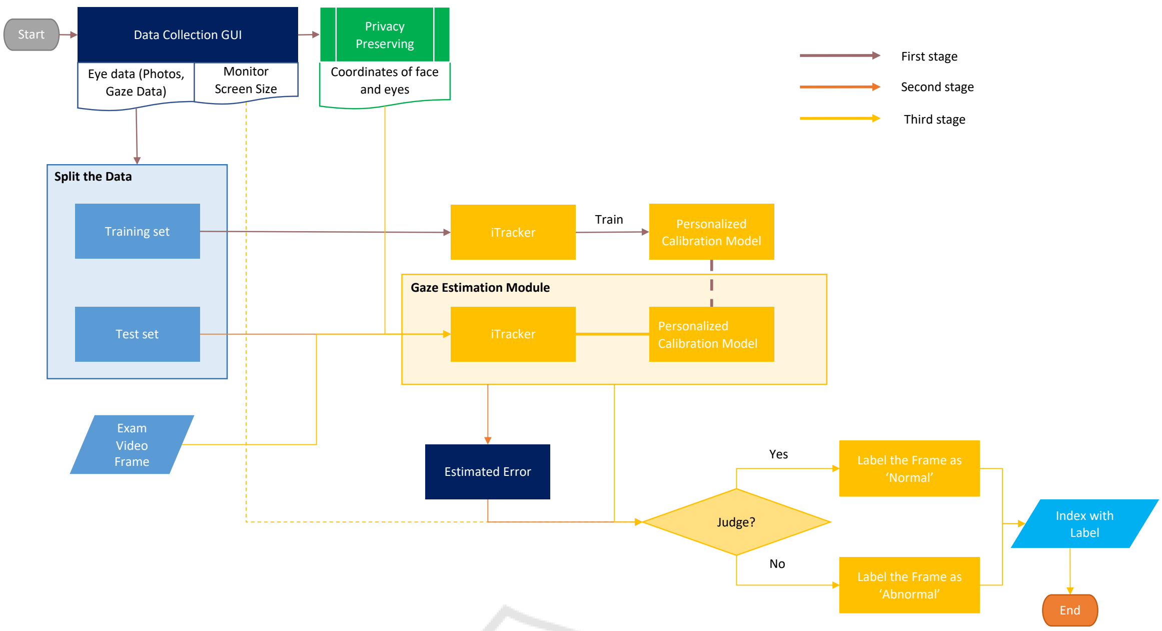


Figure 8: The flowchart of gaze estimation anomaly detection module.

- In the second stage, the post-processed test set passes through the gaze estimation module to obtain the estimated error.
- In the final stage, the estimated error is used to adjust the monitor size to the anomaly decision boundary. Like previous stages, exam video frames are processed with face detection bounding boxes. The data then goes through the gaze estimation module to obtain the calibrated coordinate prediction result. The adjusted prediction result is compared to the decision boundary to determine if the frame is an anomaly and is marked.

7 RESULTS

The experiment used a test dataset comprising three exam-taking videos from different participants to measure the performance of Yaqub et al.'s image-hashing-based anomaly detection method, our gaze detection-based anomaly detection method, and the combined anomaly detection method (both hashing and gaze). Each video had a duration of two to three minutes. Detailed instructions on how to emulate an exam and attempt cheating were provided to each participant. Each frame of the video was manually labeled as either a normal or anomaly pose. The participants were asked to perform the following actions for each direction: left, right, up, and down, to test Yaqub et al.'s image-hashing-based method.

MediaPipe and Dlib were used for face and eye detection, Gaussian blurring, single-white masking, and dHashing-based image hashing with a hash size of 12. The experiment was conducted on a Windows 10 computer with 16 GB RAM and an i7-10710U CPU. Each video frame was processed using MediaPipe and Dlib for face and eye detection, followed by the blurring or masking-based face hiding module, and finally the dHashing-based image hashing module. The obtained anomaly results were compared to the ground truth.

Table 2: Performance summary for 3 different anomaly detection modes with 2 different privacy preserving modes (Acc. for Accuracy, Rec. for Recall, Pre. for Precision).

		Blur			Mask		
	Mode	Acc.	Rec.	Pre.	Acc.	Rec.	Pre.
1	dHash	67.4%	41.2%	83.3%	76.6%	64.7%	83.3%
	Gaze	78.3%	57.6%	96.1%	73.7%	52.9%	88.2%
	Combined	85.1%	**80.0%**	88.3%	85.1%	**87.1%**	83.1%
2	dHash	76.6%	54.1%	95.8%	77.7%	61.2%	89.7%
	Gaze	73.7%	51.8%	89.8%	72.6%	49.4%	89.4%
	Combined	88.6%	**84.7%**	91.1%	85.7%	**83.5%**	86.6%
3	dHash	78.2%	60.0%	92.3%	80.6%	63.8%	94.4%
	Gaze	78.8%	67.5%	85.7%	78.8%	65.0%	88.1%
	Combined	87.3%	**90.0%**	84.7%	90.3%	**92.5%**	88.1%

The proctoring system is designed to perform anomaly detection on anonymized images such as blurred and masked images. It is essential to evaluate the model's performance on anonymized video clips. We propose two main anonymization functions: the image hashing model will be evaluated on blurred and masked images, respectively.

The dHash model maintains its performance on anonymized data, performing even better on masking images compared to blurred anonymization. The running time of the model on anonymized data remains at the same level as the original one. Table 2 presents the performance of the proposed method for three participants. The running time of the entire system was also measured in terms of FPS (frames per second), which are 31 and 35 FPS, respectively. As expected, the masking approach performed better, although both approaches can be easily executed on normal PCs.

8 CONCLUSION AND FUTURE WORK

Online student proctoring is a reality in online exams. In this paper, we proposed a privacy-preserving online proctoring system using gaze-based anomaly detection. Experiments showed promising results. There are several ways this preliminary work can be further improved. The first requirement is creating a large dataset of exam-taking students. Secondly, the proposed method can be improved by exploring other privacy-preserving measures and considering other anomalies such as audio.

ACKNOWLEDGEMENTS

This work is supported by UTS New Faculty Startup Grant 261011.0226628. We also thank Jason, Yang, Mill, Doris, and Vivi for implementing/testing algorithms on their dataset.

REFERENCES

Atoum, Y., Chen, L., Liu, A. X., Hsu, S. D., and Liu, X. (2017). Automated online exam proctoring. *IEEE Transactions on Multimedia*, 19(7):1609–1624.

Balash, D. G., Kim, D., Shaibekova, D., Fainchtein, R. A., Sherr, M., and Aviv, A. J. Examining the examiners: Students' privacy and security perceptions of online proctoring services. In *Seventeenth Symposium on Usable Privacy and Security (SOUPS 2021)*.

Conijn, R., Kleingeld, A., Matzat, U., and Snijders, C. The fear of big brother: The potential negative side-effects of proctored exams. *Journal of Computer Assisted Learning*, n/a(n/a).

Cote, M., Jean, F., Albu, A. B., and Capson, D. (2016). Video summarization for remote invigilation of online exams. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9. IEEE.

Dimeo, J. (2017). Online exam proctoring catches cheaters, raises concerns. https://www.insidehighered.com/digital-learning/article/2017/05/10/online-exam-proctoring-catches-cheaters-raises-concerns.

Furby, L. (2020). Are you implementing a remote proctor solution this fall? recommendations from nln testing services. *Nursing education perspectives*, 41(4):269–270.

Hussein, M. J., Yusuf, J., Deb, A. S., Fong, L., and Naidu, S. (2020). An evaluation of online proctoring tools. *Open Praxis*, 12(4):509–525.

Irfan, M., Aslam, M., Maraikar, Z., Jayasinghe, U., and Fawzan, M. (2021). Ensuring academic integrity of online examinations. In *2021 IEEE 16th International Conference on Industrial and Information Systems (ICIIS)*, pages 295–300.

Krafka, K., Khosla, A., Kellnhofer, P., Kannan, H., Bhandarkar, S., Matusik, W., and Torralba, A. (2016). Eye tracking for everyone. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2176–2184.

Masud, M. M., Hayawi, K., Mathew, S. S., Michael, T., and El Barachi, M. (2022). Smart online exam proctoring assist for cheating detection. In *International Conference on Advanced Data Mining and Applications*, pages 118–132. Springer.

Milone, A. S., Cortese, A. M., Balestrieri, R. L., and Pittenger, A. L. (2017). The impact of proctored online exams on the educational experience. *Currents in Pharmacy Teaching and Learning*, 9(1):108–114.

Nigam, A., Pasricha, R., Singh, T., and Churi, P. (2021). A systematic review on ai-based proctoring systems: Past, present and future. *Education and Information Technologies*, pages 1–25.

Selwyn, N., O'Neill, C., Smith, G., Andrejevic, M., and Gu, X. (2021). A necessary evil? the rise of online exam proctoring in australian universities. *Media International Australia*, page 1329878X211005862.

Yaqub, W., Mohanty, M., and Suleiman, B. (2021). Image-hashing-based anomaly detection for privacy-preserving online proctoring. *arXiv preprint arXiv:2107.09373*.

Yaqub, W., Mohanty, M., and Suleiman, B. (2022). Privacy-preserving online proctoring using image-hashing anomaly detection. In *2022 International Wireless Communications and Mobile Computing (IWCMC)*, pages 1113–1118.