AUTOMATED REMOTE PROCTORING SYSTEM

A report submitted in partial fulfillment of the requirements for the Award of Degree of

BACHELORS DEGREE
in
COMPUTER SCIENCE AND ENGINEERING
by

AMRUTHA KORUMILI
ID NO: B151633

SHAIK, MUBEEN
ID NO:B151743

August 18, 2020
Under the supervision of

Dr.Suryakanth V. Gangashetty
Assistant Professor at IIIT Hyderbad
(Duration : 06/06/2020 to 31/07/2020)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
Rajiv Gandhi University of Knowledge and Technologies
Basar,Nirmal,Telangana,INDIA.

July 2019-2020

Rajiv Gandhi University of Knowledge and Technologies
Basar,Nirmal,Telangana,INDIA.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Project work entitled **"Automated Remote Proctoring System"** is a bonafide work carried out and submitted by **K.Venkata Bhavani Amrutha(B151633),Shaik Mubeen(B151743)** in partial fulfilment of the requirements for the degree of Bachelor of Technology in Computer Science and Engineering , Basar during the academic year 2019-20 at International Institute of Information Technology, Hyderbad.The results embodied in this report have not been submitted to any other University or Institution for the award of any diploma or degree

Signature of the
Department Coordinator

Signature of Supervisor

Signature of the
HEAD OF DEPARTEMNT

Acknowledgements

First we Would like to thank management of RGUKT-Basar for giving us the opportunity to do an internship within the organization.

We also would like to thank all the people who worked along with us RGUKT-Basar, with their patience and openness they created an enjoyable working environment.

It is indeed with a great sense of pleasure and immense of gratitude that we acknowledge the help of these individuals.

We am highly indebted to Vice-Chancellor and Administrative Officer, for the facilities provided to accomplish this internship.

We would like to thank our Head of the Department ,CSE for her constructive criticism throughout my internship.

We would like to thank our supervisor Dr.Suryakanth V Gangashetty for his constructive criticism throughout my internship.

We would like to thank our department internship coordinator Mr.K.RaviKanth, Assistant Professor,CSE for their support and advices to get and complete internship in above said organization.

We am extremely great full to my department staff members, family members and friends who helped us in successful completion of this internship.

Amrutha Korumilli

Shaik Mubeen

Abstract

Massive open online courses (MOOCs) and other forms E-Learnings continue to increase in popularity and reach.The rapid growth of the e-learning industry has created needs for various supporting technologies.One area that is gaining significance is the virtual proctoring space. The ability to efficiently proctor remote online examinations is an important limiting factor to the scalability of this next stage in education.

Currently,human proctoring is the most common approach of evaluation, by either requiring the test taker to visit an examination center, or by monitoring them visually during exams via a webcam. However, such methods are labor-intensive and costly. Our project, is a automated analytics system which performs online exam proctoring.

The system hardware includes a webcam, for the purpose of monitoring the visual environment of the test location. The system includes five basic components which continuously estimate the key behavior cues: user authentication, active window detection, multiple people detection,constant check on who is giving the test and phone detection. By combining the continuous estimation components, we design a webpage which classify whether the test taker is cheating at any moment during the test. Henceforth, Assessment providers, educational institutions and a MOOC's can become valuable partners by enabling assessments to be administered cheaper and faster. Also this helps test-takers to give exam at convenience of their homes at a suitable time.

Keywords: Online exam proctoring (OEP), user authentication, active window detection, multiple people detection,face recognition and phone detection

List of Tables

2.1 Existing Technologies 22

6.1 Week -1 Report 64
6.2 Week -2 Report 66
6.3 Week -3 Report 68
6.4 Week -4 Report 70
6.5 Week -5 Report 72
6.6 Week -6 Report 74
6.7 Week -7 Report 76
6.8 Week -8 Report 78

List of Figures

1.1 Statistics of cheating behavior in an OEP 12
1.2 Flowchart of the OEP system 17

3.1 Image of a person 27
3.2 Taking each pixel of the image 27
3.3 Looking at just this one pixel and the pixels touching it, the image is getting darker towards the upper right. 28
3.4 Gradients of the image 28
3.5 The original image is turned into a HOG representation that captures the major features of the image regardless of image brightnesss. 28
3.6 Comparision the HOG face pattern of others and our image . 28
3.7 Face Landmarks 29
3.8 Triplet Training 29
3.9 128 measurements generated from input image 30
3.10 Calculating Probability 31
3.11 Locating Bounding Boxes 31
3.12 YOLOv3 Darket Architecture 35

4.1 User Verification Output Console 38
4.2 User Verification Output 38
4.3 Phone Detection output 39
4.4 Phone Detection Alert Output 39
4.5 Multiple Person Detection Output 40
4.6 Multiple Person Alert Output 40
4.7 Codesnippet 46
4.8 Output in terminal 47
4.9 Flask Application 48
4.10 User Authentication: Case 1 result 50

4.11 User Authentication: Case 2 result 51
4.12 Face Verification: Case 1 result 52
4.13 Face Verification: Case 2 result 52
4.14 Active Window Detection: Case 1 result 53
4.15 Active Window Detection:Case 2 result-1 54
4.16 Active Window Detection:Case 3 result-2 54
4.17 Active Window Detection:Case 4 result-3 55
4.18 Active Window Detection:Case 5 result-4 55
4.19 Phone Detection: Case 2 result 56
4.20 Multiple People Detection: Case 2 result 57
4.21 Unauthorized Examinee Detection: Case 2 result 58

Contents

1 INTRODUCTION **9**
- 1.1 Introduction . . . 9
- 1.2 Aim and Objective of the Project . . . 10
- 1.3 Motivation . . . 11
- 1.4 Literature Survey . . . 13
- 1.5 Technical Approach . . . 16
- 1.6 Applications . . . 18
- 1.7 Organization of Report . . . 19

2 PROPOSED MODEL **21**
- 2.1 Introduction . . . 21
- 2.2 Existing Technologies . . . 21
- 2.3 Proposed Model . . . 23

3 RELATED WORK **26**
- 3.1 Introduction . . . 26
- 3.2 Face Recognition . . . 26
  - 3.2.1 Face Detection . . . 27
  - 3.2.2 Encoding Faces . . . 28
  - 3.2.3 Recognize face from Encodings . . . 30
- 3.3 Object Detection . . . 30
  - 3.3.1 Method . . . 30
  - 3.3.2 Features . . . 32
  - 3.3.3 Network Design . . . 35

4 IMPLEMENTATION **36**
- 4.1 Introduction . . . 36
- 4.2 Hardware Components . . . 37

4.3 Features Extraction 37
4.3.1 User Authentication 37
4.3.2 Phone Detection 38
4.3.3 Multiple Person Detection 39
4.3.4 Invalid Examinee Detection 40
4.3.5 Active Window Detection 40
4.4 Web Application 44
4.4.1 Introduction 44
4.4.2 UI Design 44
4.4.3 Working 44
4.4.4 How are the deep-learning models integrated into the backend of the web pages? 45
4.5 Database Collection 48
4.5.1 Dataset for Face Recognition Task 48
4.5.2 Dataset for Object Detection Task 48
4.5.3 Database for User Credential Verification 49
4.6 Result 49
4.6.1 Introduction 49
4.6.2 Feature Analysis and Cheat Detection 49

5 CONCLUSION **59**

6 FUTURE SCOPE **60**

Chapter 1

INTRODUCTION

1.1 Introduction

Massive open online courses (MOOCs) offer the potential to greatly expand the reach of today's educational institutions, both by providing a wider range of educational resources to enrolled students and by making educational resources available to persons who cannot access a campus due to location or schedule constraints. Instead of taking courses in a typical classroom on campus, now students can take courses anywhere in the world using a computer, where educators deliver knowledge via various types of multimedia content.It is stated that 70% of higher education institutions believe that online education is a critical component of their long-term strategy.

Exams are a critical component of any educational program, and online educational programs are no exception. In any exam, there is a possibility of cheating, and therefore its detection and prevention is important. Educational credentials must reflect actual learning in order to retain their value to society.But the academic cheating activity is on the rise.When exams are administered in a conventional and proctored classroom environment, the students are monitored by a human proctor throughout the exam. In contrast, there is no convenient way to provide human proctors in online exams. As a consequence, there is no reliable way to ensure against cheating. Without the ability to proctor online exams in a convenient, inexpensive, and reliable manner, it is difficult for MOOC providers to offer reasonable assurance that the student has learned the material, which is one of the key

outcomes of any educational programs, including online education.

The common testing procedure for online learners is that the, Students come to an on-campus or university-certified testing center and take an exam under human proctoring.New emerging technologies like Kryterion and ProctorU allow students to take tests anywhere as long as they have an Internet connection. However, they still rely on a person "watching" the exam-taking. For example, Kryterion employs a human proctor watching a test taker through a webcam from a remote location. The proctor is trained to watch and listen for any unusual behaviors of the test taker, such as unusual eye movements, or removing oneself from the field of view. He can alert the test taker or even stop the testing.

In this project, we introduce a web based system to perform automatic and continuous online exam proctoring (OEP) . The overall goal of this system is to maintain academic integrity of exams, by providing real-time proctoring to detect the majority of cheating behaviors of the test taker.

1.2 Aim and Objective of the Project

Aim:

To design a web application that provide real-time proctoring to detect cheating behaviours of the test taker.

Objective:

1. To take necessary action when the authorized test taker is not the one who is giving the exam.
2. To take necessary action when the test taker cheat with help of electronic gadgets like mobile
3. To take necessary action when the test taker try to cheat by accessing other websites or files from the local computer.

1.3 Motivation

To demonstrate and maintain academic integrity, some institutions require proctor supervision of online exams. However, proctoring can be very expensive. Costs to students can include fees at testing centers, costs to purchase the Remote Proctor, time to find an approved proctor, and effort required to coordinate a time for the exam. Costs to the institution include salaries of staff to administer a proctoring process, approval of proctors, maintaining testing centers, and potential loss of enrollments and revenue since not all institutions require proctors for online exams.

Problems with offline proctored tests or online tests without proctoring:

1. In the current global scenario, it is nearly impossible to do offline proctoring due to the travel ban and lockdown in multiple countries.

2. Providing a proctored exam center near the location of the test taker is a significant challenge for most organizations administering any forms of tests.

3. Qualified Proctors are hard to find, and it's hard to ensure the quality of proctoring. And there are no records available to cross check whether the proctor did his/her job properly.

4. A limited supply of test centers or proctors also leads to extended test schedules.

5. In online test without proctoring, cases are often reported of impersonation and cheating. Students either ask some else to take the test on their behalf or use methods of cheating like referring to a textbook, using smartphones or other devices to search for answers online or taking help from a friend.

And the need of preventing students from cheating and maintain academic integrity is also a part of motivation. Before that,how does students actually cheat during exams?

How students cheat In the absence of good online exam control procedures, how do online students cheat?

In some cases, students can obtain exam questions or even exam answers before they take the exam. Some instructors actually make their exams available online for a week so students can take the exam at their convenience. Students then conspire with their network of classmates. A superior student takes the exam first, records the answers, and/or copies the questions. Then the questions are researched, answered, and distributed to the remaining students. If instructors do not periodically revise exams, then student groups develop files for their current and future classmates to use. Students can also illicitly obtain publishers' test banks and related solutions manuals from university libraries, faculty, or underground sources. Online exams that remain open (available for access) for extended periods of time permit one student to take the exam while receiving help from other conspiring students who then take the exam at a later time. There are many other methods of cheating during online exams. Cheating, as in fraud, seems limited only by one's imagination. Henceforth, we propose a less costly, non-proctor alternative to promote academic honesty and thus reduce the likelihood of cheating by students.

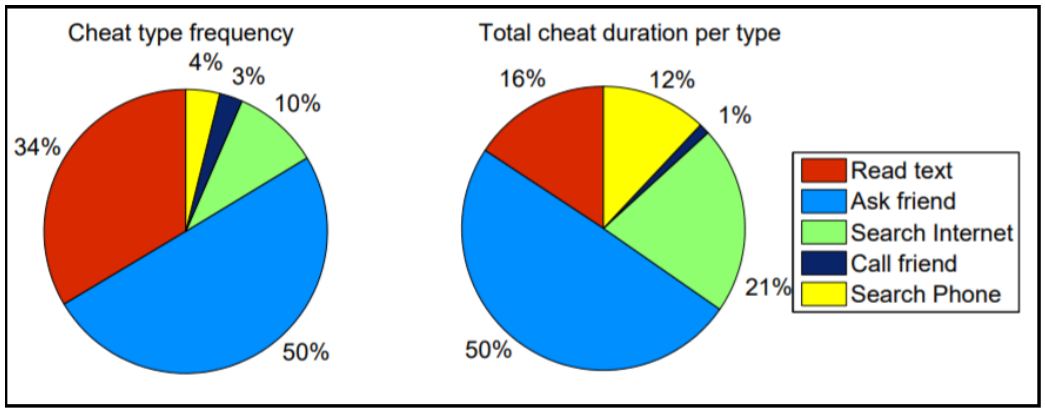


Figure 1.1: Statistics of cheating behavior in an OEP

1.4 Literature Survey

Literature review was done for various works which was essential to know how serious the action of cheating is considered and what all work have been done to minimize them.

The paper [1] present a conceptually simple, flexible, and general framework for object instance segmentation. Our approach efficiently detects objects in an image while simultaneously generating a high-quality segmentation mask for each instance. The method, called Mask R-CNN, extends Faster R-CNN by adding a branch for predicting an object mask in parallel with the existing branch for bounding box recognition. Mask R-CNN is simple to train and adds only a small overhead to Faster R-CNN, running at 5 fps. Moreover, Mask R-CNN is easy to generalize to other tasks, e.g., allowing us to estimate human poses in the same framework. We show top results in all three tracks of the COCO suite of challenges, including instance segmentation, bounding-box object detection, and person keypoint detection. Without bells and whistles, Mask R-CNN outperforms all existing, single-model entries on every task, including the COCO 2016 challenge winners. We hope our simple and effective approach will serve as a solid baseline and help ease future research in instance-level recognition.

The real-time object detection model is discussed in [2] is an improvement over previous YOLO detection networks. Compared to prior versions, it features multi-scale detection, stronger feature extractor network, and some changes in the loss function. As a result, this network can now detect many more targets from big to small. And, of course, just like other single-shot detectors, YOLO V3 also runs quite fast and makes real-time inference possible on GPU devices. Well, as a beginner to object detection, you might not have a clear image of what do they mean here. But you will gradually understand them later in my post. For now, just remember that YOLO V3 is one of the best models in terms of real-time object detection as of 2019.

The paper [3] discuss different ways of preventing cheating. The authors propose a less costly, non-proctor alternative to promote academic honesty, using eight control procedures that enable faculty to increase the difficulty and thus reduce the likelihood of cheating by students.

The paper [4] is a multimedia wearable automated online proctoring system.Proposes a fully automated online exam proctoring system with visual and audio sensors for the purpose of maintaining academic integrity. Designs a hybrid two-stage multimedia analytics approach where an ensemble of classifiers extracts middlelevel features from the raw data, and transforming them into high-level features leads to the detection of cheating. Collects a multimedia dataset composed of two videos and one audio for each subject, along with label information of all cheating behaviors. This database is publicly available for future research.

The paper [5] discuss about Massive Open Online Courses (MOOCs) enable everyone to receive high-quality education. However, current MOOC creators cannot provide an effective, economical, and scalable method to detect cheating on tests, which would be required for any certification. In this paper, we propose a Massive Open Online Proctoring (MOOP) framework, which combines both automatic and collaborative approaches to detect cheating behaviors in online tests. The MOOP framework consists of three major components: Automatic Cheating Detector (ACD), Peer Cheating Detector (PCD), and Final Review Committee (FRC). ACD uses webcam video or other sensors to monitor students and automatically flag suspected cheating behavior. Ambiguous cases are then sent to the PCD, where students peer-review flagged webcam video to confirm suspicious cheating behaviors. Finally, the list of suspicious cheating behaviors is sent to the FRC to make the final punishing decision. Our experiment show that ACD and PCD can detect usage of a cheat sheet with good accuracy and can reduce the overall human resources required to monitor MOOCs for cheating.

In this paper[6], the authors present a face recognition system called FaceNet. This system uses a deep convolutional neural network which optimizes the embedding, rather than using an intermediate bottleneck layer. The authors state that the most important aspect of this method is the end-to-end learning of the system. The team trained the convolutional neural network on a CPU cluster for 1,000 to 2,000 hours. They then evaluated their

method on four datasets. Notably, FaceNet attained an accuracy of 99.63% on the famous Labeled Faces in the Wild (LFW) dataset, and 95.12% on the Youtube Faces Database.

The paper [7], As of the writing of this article, current embedding methods used for face recognition are able to achieve high performance in controlled settings. These methods work by taking an image of a face and storing data about that face in a latent semantic space. However, when tested in fully uncontrolled settings, the current methods cannot perform as well. This is due to instances where facial features are absent from or ambiguous in the image. An example of such a case would be face recognition in surveillance videos, where the quality of the video may be low. To help address this issue, the authors of this paper propose Probabilistic Face Embeddings (PFEs). The authors propose a method for converting existing deterministic embeddings into PFEs. Most importantly, the authors state that this method effectively improves performance in face recognition models.

The article, Modern Face Recognition with Deep Learning by Davis King and Adam Geitgey, discuss how does deep learning and face recognition work. The secret is a technique called deep metric learning. If you have any prior experience with deep learning you know that we typically train a network to- Accept a single input image, And output a classification/label for that imageHowever, deep metric learning is different. Instead, of trying to output a single label (or even the coordinates/bounding box of objects in an image), we are instead outputting a real-valued feature vector.

1.5 Technical Approach

In this work, we aim to develop a multimedia analysis system to detect a wide variety of cheating behaviors during an online exam session. Our proposed online exam process includes two phases, the preparation phase and exam phase.

Preparation Phase:

In the preparation phase, the test taker has to authenticate himself before beginning the exam, by using a password and face authentication. Further, the test taker learns and verbally acknowledges the rules of using the OEP system, such as, no second person is allowed in the same room, the test taker should not leave the room during the exam phase, etc. In the preparation phase, if the test taker is not the authorized user then he/she is not allowed to take the exam.

Exam Phase:

In the exam phase, the test taker takes the exam, while under the "monitoring" of our OEP system for real-time cheating behavior detection. Once as he begins to take the exam the webcam gets activated and continuously keeps a track of the surroundings.If forbidden objects like cell phones or any other person beyond the test taker are detected then the test taker is warned a few times if the same continues to happen the test taker will be forcefully logged out of the exam. The system also keeps a track whether the test taker is the authorized user or not until the end of the test so that no other could take his place and cheat. The test taker is again warned when he tries to cheat by accessing a different browser or files from the computer. The active flow of the tab is under surveillance.

Flow Chart

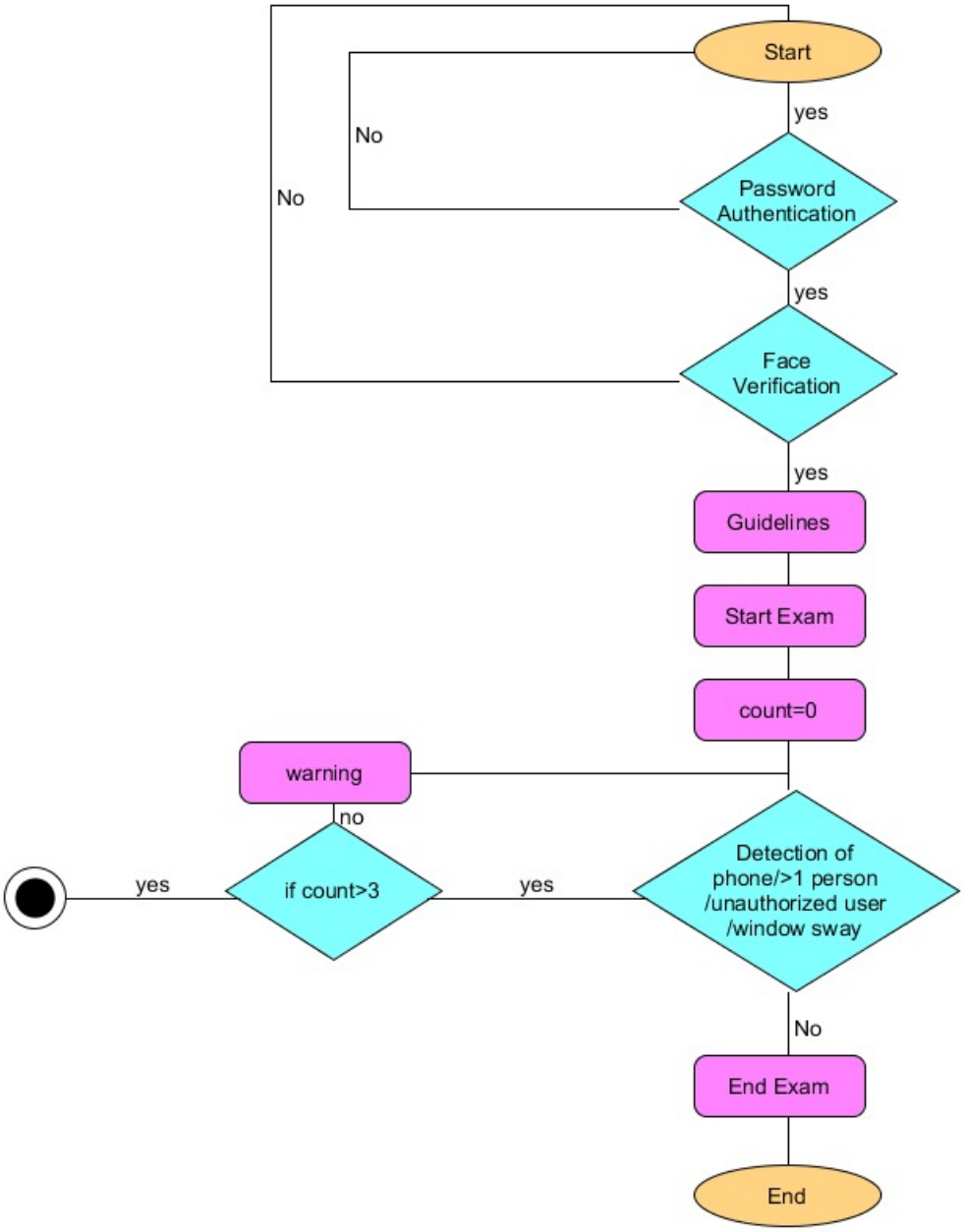


Figure 1.2: Flowchart of the OEP system

1.6 Applications

Many can get beneficial by the use of Online Exam Proctoring like:

1. Assessment Providers

Assessment Providers who provide ready-made assessments or provide test engine to clients who have their content can leverage proctoring for the following benefits:

1. Become a valuable partner by enabling assessments to be administered cheaper and faster.
2. Generate additional revenue opportunities.
3. Improve usage per client by providing On-Demand tests anytime-anywhere
4. Take business global by enabling remote proctoring

The use cases can be as varied as recruitment, college admission test, certifications, promotions, etc. Assessment providers like Pearson, IBM Kenexa, provide online proctoring.

2. Online Education Providers- Universities, MOOCs

Online Education Providers, schools, and MOOCs, are being challenged by how to verify the validity and quality of their online programs effectively. The two major concerns of the customer are cheating and student authentication.

1. It is clear that trusted proctoring solutions have become essential tools in online education for authenticating the identities of online students, assuring that students are taking exams without cheating, and assessing student performance effectively.
2. The Higher Education Opportunity Act of 2008 in the US requires colleges and universities to verify the identity of students to ensure those who register for an online course are the ones who participate.

More than 500 universities in the US including Arizona State University, California State University use online proctored exams. MOOCs like edX and Coursera also make use of online video proctoring.

3. Certifying Agencies- Oracle, MS, etc

Certifying agencies are using online proctoring for the following benefits:

1. Scale business by enabling students to complete certification from home
2. Become more competitive by bringing down the cost of certification by eliminating the need for proctored test centers
3. Maintain visual records for audits on demand

Many organizations which provide certifications like Microsoft and Salesforce provide an online video proctoring option.

1.7 Organization of Report

Chapter 1:

This chapter gives a brief introduction to our project along with the approach and applications of the project.

Chapter 2:

This chapter discusses about the already existing technologies and how is our system different from others.

Chapter 3:

This chapter discusses about the deep-learning algorithms utilised in the implementation of the system and their working.

Chapter 4:

This chapter discusses about the step wise process taken to build up an automated online proctoring system and integrating into a web application with the help of flask.

Chapter 5:

This chapter discusses about the results obtained while testing our web application.

Chapter 6:

This chapter present the final conclusion of the automated proctoring system.

Chapter 7:

This chapter discusses about the ' Future Scope' of the project and what features can be integrated so as to make it more efficient and user-friendly.

Chapter 8:

This chapter present the references made which helped in the implementation of the project.

Chapter 2

PROPOSED MODEL

2.1 Introduction

Online proctoring, sometimes called remote proctoring, generally refers to proctors monitoring an exam over the Internet through a webcam. It includes as well the processes, occurring at a distance, for authenticating the examinee as the person who should be taking the exam. Adding to the definition, online proctoring includes any automated processes that help to secure a test administration event.

The term, online proctoring, is more descriptive than and preferable to remote proctoring. It emphasizes the critical use of the Internet and automated processes to produce a secure solution in monitoring test takers. Remote proctoring, on the other hand, is a term that can refer to any proctoring that occurs in a situation remote from a standard testing location (e.g., testing center or school). In particular, the popular "find your own proctor" model, which is often correctly referred to as remote proctoring, has been a less-than-ideal, non-technology-based solution for monitoring exam administration for distance education courses for several decades.

2.2 Existing Technologies

Online proctoring using human proctors in an effective way was first introduced and championed by Kryterion in 2006, and began large-scale operations in 2008. Several other organizations have following Kryterion's lead.

Table 2.1: Existing Technologies

Online Proctoring Organizations	Description
Kryterion Inc.	Launched in 1999; a Drake International company (founder of Prometric in 1990)
Software Secure	Long-term provider of services; known for integrated camera a fingerprint device
ProctorU	Founded in 2008; associated with Andrew Jackson University
B Virtual	Member of B Wyze Group, a leader in remote workplace innovation.
Tegrity	Grew out of Tegrity lecture capture technology; a CTB-McGraw Hill company
ProctorCam	Founded in 2007 and based in Boston. Massachusetts
Respondus	Assessment applications for elearning market; entering the online proctoring market space
Loyalist Exam Services	A division of Loyalist College in Ontario, Canada

These include Software Secure, ProctorU, Tegrity, Respondus , ProctorCam, B Virtual, and Loyalist. These will be compared and contrasted in following sections of this paper.

What is the value of online proctoring? Why is it becoming a viable option for monitoring exam administrations? First, many in the testing industry finally acknowledge the security weaknesses of traditional proctoring. As an example, it is hard to miss the reports of cheating from educational statewide assessment programs. This educational example illustrates that much of the cheating that occurs at the lower grades is by the proctors, usually by teachers or other school officials. Local, on-site proctors for any test may know the students being tested and therefore, have a stake in the outcome of the tests making the tests vulnerable to compromise. Standards (e.g., ISO's 17024 standards for certification) that require proctors be independent of the testing outcomes are often ignored in favor of cost savings, convenience, and resource availability.

Second, on-site proctors are generally considered on par with "volunteers," meaning they are not paid (or poorly paid), relatively unmotivated, and poorly-trained. There are few models in the high-stakes testing industry where attention is paid to high-quality proctoring.

A third reason is that technology-based alternatives, such as online proctoring, are becoming more capable and are gaining attention. Indeed, there are online proctors who are better trained, may be on career paths, and are able to detect cheating at least as well as onsite proctors (see research section below). Technology-based aides, such as computer/system lockdowns,

keystroke monitoring, the ability to stop/start a test, and many other assistive proctoring processes have been relatively easy to integrate into the monitoring process.

2.3 Proposed Model

The utmost main feature that make our model different from the existing ones is that, all the existing proctoring systems are software that need to installed into the local computer to take the exam and these softwares usually comes with a specific price tag.While the model we proposed is a web application that has integerted all the main features that a automated proctoring software has, this makes is easier to access and free of cost. Our model is similar to any other webpage on the internet but it has deeplearning models integrated into it to take care of the cheating issue.

The model mainly has six features:

1. Username and Password Authentication
2. Face Verification
3. Phone Detection
4. Active Window Detection
5. Multiple People Detection
6. Continous survilleance on whether or not the test taker is authorized until the end of the exam.

1. User Verification

One of the major concerns in online exams is that the test taker solicits assistance from another person on all or part of an exam. An OEP system should be able to continuously verify whether the test taker is who he claims to be throughout the entire exam session. The test taker is also expected to take the exam alone without the aid of another person in the room. While there are various options for continuous user authentication, such as

keystroke dynamics, we decide to use face verification due to its robustness. There are a number of challenges for user verification in OEP. First, face detection under various lighting and poses is difficult. Although face detection has improved substantially over the years, occasional miss detection and false alarm is inevitable, and how to handle this is another challenge. We propose to overcome these challenges by using an approach integrating both face encodings. We use the python face recognition module which has an approx accuracy of 98%.Once the credentials are identified to be authorized the webcam capture a face shot and veify it. Only if the user is authorized then he/she is allowed to take the exam.

2. Active Window Detection

The Internet and computers are an open gateway to valuable information for answering exam questions. It indicate that cheating from the Internet is the most frequent among e-learners. We approached this issue by keeping a track of the activeness of the tab in which the website is opened.We use use a combination of the 3 methods: W3C Visibility API, then focus/blur and user activity methods in order to reduce the false positive rate. This allows to manage the following events:

Changing browser tab to another one (100% accuracy, thanks to the W3C Page Visibility API)

Page potentially hidden by another window, e.g. due to Alt+Tab (probabilistic = not 100% accurate)
User attention potentially not focused on the HTML page (probabilistic = not 100% accurate) this reacts according to the current user visibility state, this could be to pause a playing video when the user ALT+TABs to a different window, tracking stats about how the users interact with our application, how often does him switch to a different tab, how long does it take him to return and a lot of performance improvements that can benefit from this kind of API. Whenever a sway is detected the test taker is warned and if this happens more than specific count of times despite of warning a necessary action will be taken like a force log out.

3. Phone Detection

Our online exam rule prohibits the use of any type of mobile phones. Therefore, the presence of a mobile phone in the testing room can be an indication of potential cheating. With advancements in mobile phone technology, there are many ways to cheat from them, such as reading saved notes, text messaging friends, browsing the Internet, and taking a snapshot of the exam to share with other test takers. Phone detection is challenging due to the various sizes, models and shapes of phones (a tablet could also be considered a type of phone). Some test takers might have large touch screens while others might use a button-based flip phones. Moreover, cheating from a phone is usually accompanied with various occlusions, such as holding the phone under the desk, or covering part of the phone with their hand. To enable this capability, we utilize Yolov3 object detection algorithm which specifically identifies the phone class and generates a warning and if this is detected more than specific amount of times a force log out is done.

4. Multiple People Detection

Until the exam end the test location is under continous survillieance of the webcamera and the count of the persons is kept with the help of person class of yolov3 and whenever there is a raise in count the warning is generated and if this is detected more than specific amount of times a force log out is done.

5. Detection of unauthorized test taker

There is a subtle possibility of the test taker swap once after the face authentication is done, to deal with is issue we keep track of the test taker until the end of the exam verifying continuously whether he is authorized or not. This is done by taking the face detected in the caputuring video and continuously comparing it with the one that was taker during successful login. This helps in preventing any unauthorized user swap in with the test taker.

Chapter 3

RELATED WORK

3.1 Introduction

Over the years, the demand for online learning has increased significantly. Researchers have proposed various methods to proctor online exams in the most efficient and convenient way possible, yet still preserve academic integrity. These methods can be categorized into three categories: (a) no proctoring [3],[11], (b) online human monitoring [12],[13] and (c) semi automated machine proctoring [14],[15] .No proctoring does not mean that test takers have the freedom of cheating, instead cheating is minimized in various ways.

Online human monitoring is one common approach for proctoring online exams. The main downside is that it's very costly in terms of requiring many employees to monitor the test takers. Among all prior work, the most relevant work to ours is Massive Open Online Proctoring framework which combines both automatic and collaborative approaches to detect cheating behaviors in online exams. In our approach, we have adapted two main methodologies to detect cheating behaviors using neural network design.

3.2 Face Recognition

Face recognition is the general task of identifying and verifying people from photographs of their face. It can be described with two main modes as:

Face verification: A one-to-one mapping of a given face against a known identity (e.g is this the person?)

Face Identification: A one-to-many mapping for a given face against a database of known faces (e.g who is this person?)

To start with any of the above modes, one has to find a face in a photograph/ video ie. Face detection.

3.2.1 Face Detection

To find faces in an image, we use a method called Histogram of Oriented Gradients. We input a grayscale image(fig.1) and look at every single pixel in our image one at a time.(fig.2)

Figure 3.1: Image of a person

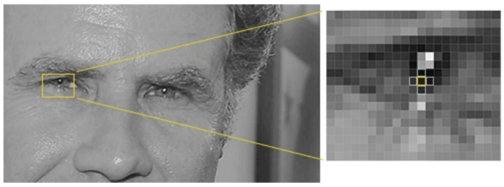


Figure 3.2: Taking each pixel of the image

For every single pixel, we want to look at the pixels that directly surrounding it and figure out how dark the current pixel is compared to the pixels directly surrounding it and draw an arrow showing in which direction the image is getting darker. (fig.3)Repeating the process for every single pixel in the image replaces pixel with arrow. These arrows are called gradients. But

saving the gradient for every single pixel gives us way too much detail. (fig.4)

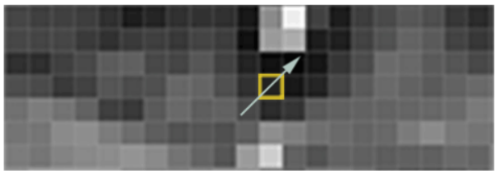


Figure 3.3: Looking at just this one pixel and the pixels touching it, the image is getting darker towards the upper right.

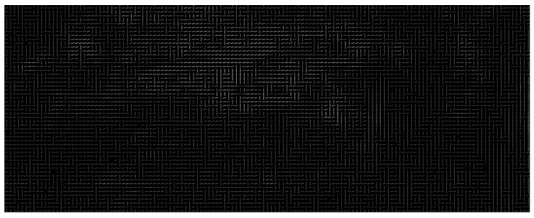


Figure 3.4: Gradients of the image

To see a basic pattern of the image, it is broken into small squares of 16*16 pixels each. In each square, we'll count up how many gradients point in each major direction. Then we'll replace that square in the image with the arrow directions that were the strongest. (fig.5) To find faces in this HOG image, all we have to do is find the part of our image that looks the most similar to a known HOG pattern that was extracted from a bunch of other training faces.(fig.6)

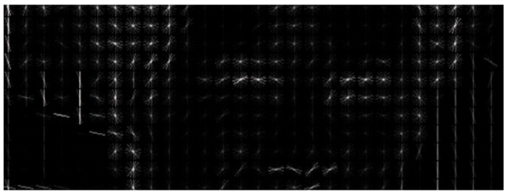


Figure 3.5: The original image is turned into a HOG representation that captures the major features of the image regardless of image brightnesss.

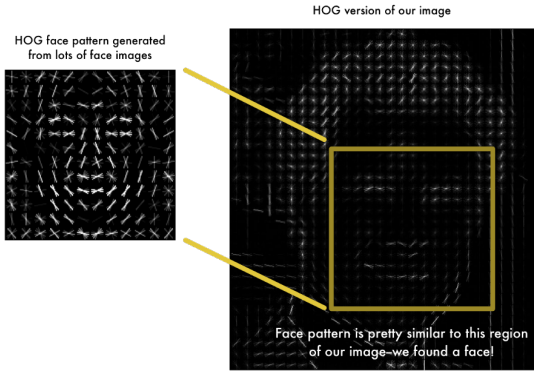


Figure 3.6: Comparision the HOG face pattern of others and our image

3.2.2 Encoding Faces

After a face is detected in a picture/video, given the unknown image compare it with the known images.This comparision requires extraction of few

basic measurements from each face (fig.8). Then, we could measure our unknown face the same way and find the known face with closest measurements.

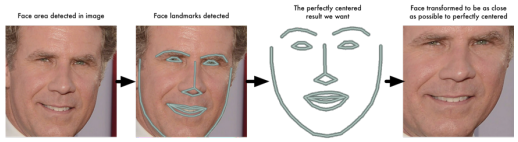


Figure 3.7: Face Landmarks

We train each face to generate 128 measurements using Deep Convolutional Neural Network. This process works by looking at 3 face images at a time called Triplet Training.(fig.7)

1. Load a training face image of a known person
2. Load another picture of the same known person
3. Load a picture of a totally different person

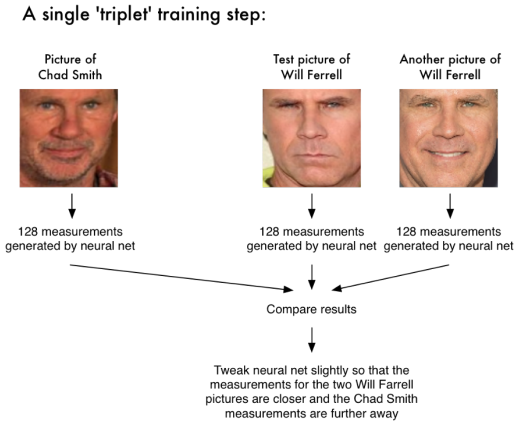


Figure 3.8: Triplet Training

Figure 3.9: 128 measurements generated from input image

The 128 measurements of each face is called as Embedding. (fig.9)

3.2.3 Recognize face from Encodings

We need to train a classifier that can take in measurements from a new test image and output which known person is the closest match.

3.3 Object Detection

Object detection encompasses the task of both image classification and object localization. It takes an image with one or more objects and outputs one or more bounding boxes and a class label for each bounding box.

Though there have been many object detection algorithms over the years, we stick to YOLOv3 algorithm as it works very well in real-time processing compared to other algorithms.

3.3.1 Method

YOLO algorithm takes an image as input and divides it into G×G grids. Each grid is responsible for object detection. Each grid cell now can estimate number of boundary boxes that can be used for an object. In the Fig.10 there are two objects. So, YOLO does it by taking the midpoint of each of two objects and then assigns the object to the grid cell containing it.

Each grid cell has an output called Y, where Y is defined as a vector of five elements.(fig.11)

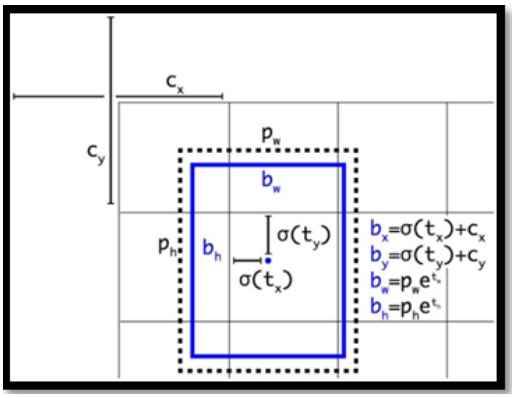


Figure 3.10: Calculating Probability

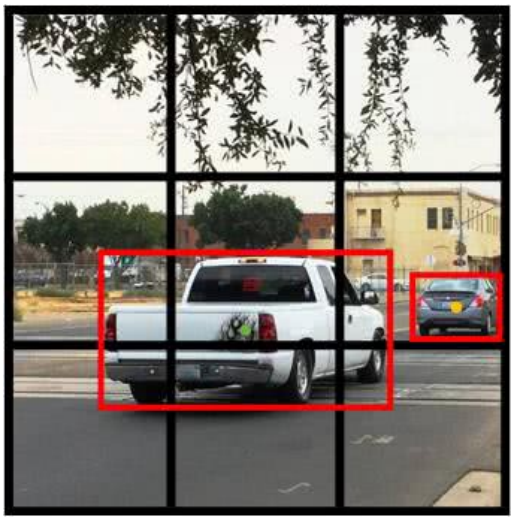


Figure 3.11: Locating Bounding Boxes

$$
\begin{aligned}
Y &= (p_c, b_x, b_y, b_h h, b_w) \\
b_x &= \sigma(t_x) + c_x \\
b_y &= \sigma(t_y) + c_y \\
b_h &= p_w * e^{t_w} \\
b_w &= p_h * e^{t_h}
\end{aligned}
\tag{3.1}
$$

$$
Probability(object) * IoU(b, object) = \sigma(t_o) \tag{3.2}
$$

$$
Probability(object) = \begin{cases} 1, & \text{if object exists, confidence score= IoU} \\ 0, & \text{otherwise.} \end{cases} \tag{3.3}
$$

where,

$$
\begin{aligned}
t_x, t_y, t_w, t_h &\text{ are predictions made by YOLO} \\
c_x, c_y &\text{ is the top left corner of the grid cell of the anchor} \\
p_w, p_h &\text{ are the width and height of the anchor} \\
c_x, c_y, p_w, p_h &\text{ are normalized by the image width and height} \\
b_x, b_y, b_h, b_w &\text{ are predicted boundary box} \\
\sigma(t_o) &\text{ is the box confidence score}(p_c)
\end{aligned}
\tag{3.4}
$$

3.3.2 Features

YOLO has many features that vary from version to version i.e every new version of YOLO comes with advanced features that makes the detection more easy and fast. YOLO v1 is simple and easy to implement as it doesn't have much features in it. It takes input image grid by gird and predicts the output as mentioned above. YOLO v2 and YOLO v3 came up with new features in addition to the working of the algorithm.

Intersection over Union(IoU)

It is used to evaluate the object detection algorithm using object localization. For suppose, if the image contains one object and it detects two bounding boxes (say A and B), then how can we predict the correct bounding box? Here comes the picture of IOU.

$$IoU = \frac{A \cap B}{A \cup B} \tag{3.5}$$

In our experiment we have used the threshold as 0.5, i.e.

$$IoU = \begin{cases} valid, & \text{if IoU}>= 0.5 \\ invalid, & \text{if IoU}<0.5 \end{cases} \tag{3.6}$$

$$\text{If they overlap} IoU = 1 \tag{3.7}$$

Non-Max Suppression

YOLO algorithm may output many bounding boxes in which most of them are irrelevant and redundant. So, to filter out the unwanted boxes, non-max suppression came into picture. It takes the probabilities of all the boundary boxes and filter out with a threshold value. This further uses IoU concept and suppresses the box with maximum IoU.

Anchor Boxes

An image can have multiple objects in it. According to YOLO basic algorithm, only one grid cell can detect only one object. But if there is more than one object sharing the same midpoint of the grid then the detection fails. To overcome this problem, anchor boxes are used. We define anchor

boxes of different shapes and sizes pass it through the image. Here the image is assigned to the grid cell that contains the object midpoint and highest IOU among the anchor boxes. Anchor boxes can be chosen manually or by K- means algorithm.

In our experiment the threshold value for IOU is 0.5 and K-means algorithm is used to define anchor boxes.

Loss Function

YOLO can predict more than one bounding box in one grid cell. To know the loss for the true positive, the one with highest IoU with the ground truth is selected. Loss function has:

Classification loss:

$$\sum_{i=0}^{s^2} 1_i^{obj} \sum_{c \in classes} (p_i(c) - \hat{p}_i(c))^2 \tag{3.8}$$

where,

$$\begin{aligned} &1_i^o bj = 1\text{if an object appears in cell i,otherwise 0} \\ &\quad \hat{p}_i(c) \text{ denotes the conditional class probability for class c in cell i.} \end{aligned} \tag{3.9}$$

Localization Loss:

$$\lambda \sum_{i=0}^{s^2} \sum_{j=0}^{B} 1_i^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] + \lambda \sum_{i=0}^{s^2} \sum_{j=0}^{B} 1_i^{obj} [\sqrt{(h_i)} - \sqrt{(\hat{h}_i)})^2 + (\sqrt{(h_i)} - \sqrt{(\hat{h}_i)})^2] \tag{3.10}$$

where,

$$\begin{aligned} &1_i^o bj = 1\text{if the jth boundary box in cell i is responsible for detecting the object, otherwise 0} \\ &\quad \lambda_c oord \text{ increase the weight for the loss in the boundary box coordinates} \end{aligned} \tag{3.11}$$

Confidence Loss:

If an object is detected in the box ,

$$\sum_{i=0}^{s^2}\sum_{j=0}^{B} 1_i^{obj}(C_i - \hat{C}_j)^2 \tag{3.12}$$

where,

$1_i^obj = 1$if the jth boundary box in cell i is responsible for detecting the object, otherwise 0

$\hat{C}_i$ is the box confidence score of the box j in cell i

(3.13)

If object is not detected in the box,

$$\lambda_n oobj \sum_{i=0}^{s^2}\sum_{j=0}^{B} 1_i^{noobj}(C_i - \hat{C}_j)^2 \tag{3.14}$$

where,

$1_i^n oobj$ is the complement of$1_i^o bj$

$\hat{C}_i$ is the box confidence score of the box j in cell i (3.15)

λ_{noobj}weights down the loss when detecting background

Loss:

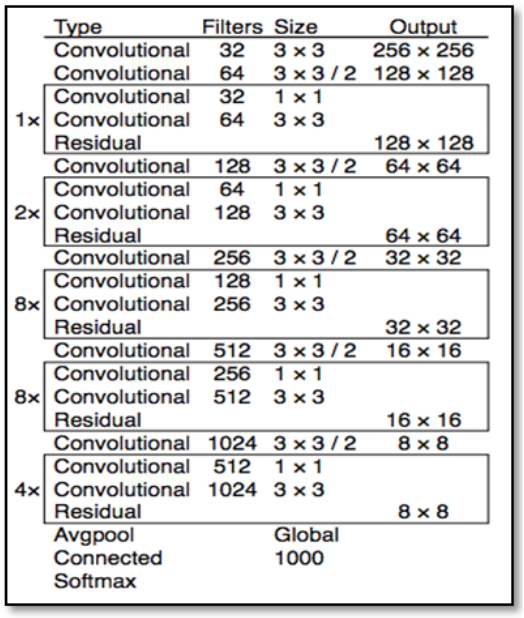
The final loss is adding all the three together.

$$\lambda \sum_{i=0}^{s^2}\sum_{j=0}^{B} 1_i^{obj}[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] + \lambda \sum_{i=0}^{s^2}\sum_{j=0}^{B} 1_i^{obj}[\sqrt{(h_i)} - \sqrt{(\hat{h}_i)})^2 + (\sqrt{(h_i)} - \sqrt{(\hat{h}_i)})^2] +$$
$$\lambda_{noobj} \sum_{i=0}^{s^2}\sum_{j=0}^{B} 1_i^{noobj}(C_i - \hat{C}_j)^2 +$$
$$\sum_{i=0}^{s^2}\sum_{j=0}^{B} 1_i^{obj}(C_i - \hat{C}_j)^2 +$$
$$\sum_{i=0}^{s^2} 1_i^{obj} \sum_{c \in classes} (p_i(c) - \hat{p}_i(c))^2 \tag{3.16}$$

3.3.3 Network Design

In our experiment we have used Darknet architecture to detect objects .YOLO v3 uses a Darknet-53 network with 53 convolution layers. It uses filter size of 3*3 and 1*1 along with skip connections as shown in (Fig.12). In YOLO v3 we use logistic classifier for each class.

Each convolution layer outputs a feature map that is obtained by the product of input image and the filter/kernel.



	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 3.12: YOLOv3 Darket Architecture

Given,

$$\begin{aligned} inputSize &= n_h * n_w * n_c \\ filterSize &= f * f * n_{\hat{c}} \\ featureMapSize &= (n - f + 1) * (n - f + 1) * n_{\hat{c}} \end{aligned} \tag{3.17}$$

It is applied when there is no stride and padding.
In our experiment we have stride=2 and padding =0. Then ,

$$featureMapSize = ((n + 2p - f)/s + 1) * ((n + 2p - f)/s + 1) * n_{(\hat{c})} \tag{3.18}$$

Chapter 4

IMPLEMENTATION

4.1 Introduction

In this project, we aim to develop a multimedia analysis system to detect a wide variety of cheating behaviors during online exam session. Our proposed online exam process includes two phases, the preparation phase and exam phase. In preparation phase, the test taker has to authenticate himself before beginning the exam, by using a password and face authentication. Further the test taker learns and verbally acknowledges the rules of OEP system, such as no second person allowed in the same room, the test taker should not leave the room during exam phase etc.

In the exam phase, the test taker takes the exam while under the monitoring of our OEP system for real- time cheating behavior detection. We use a sensor i.e. a webcam to capture the visual cues of the exam environment and the test taker. The sensed data is first processes using five components to extract middle- level features. These components are : user verification, active window detection, multiple person detection, invalid examinee detection, phone detection where the first component is the preparation phase and remaining four are exam phase. The working of each component is discussed in the further sections.

4.2 Hardware Components

During an exam, the test taker may cheat by viewing forbidden information or making another to attend the exam. Therefore, the OEP system hardware should be designed in a way to see what the test taker sees. This leads to our design of hardware component: a webcam. The webcam is mounted on top of the monitor/ in-built to monitor facing the test taker and serves multiple purposes, e.g., knowing who the test taker is, what is he doing and where is he looking to detect "the viewing-based" cheating behaviors such as reading from smart phones. This design is not only motivated by the need to see what the test taker sees, but also the growing popularity and decreasing cost of wearable cameras.

4.3 Features Extraction

4.3.1 User Authentication

One of the major concerns in online exams is that the taker solicits assistance from another person on all or part of an exam. An OED system should be able to continuously verify whether the test taker is who he claims to be throughout the entire exam session.

Whenever the person registers to an exam online, the system takes one image of the test taker and saves it in the database and also in the cloud in a folder with test taker name. All the pictures with given known names are trained using the method HOG, by finding the faces in the images and extracting encodings from it. These 128 embeddings are saved and compared with unknown face given at the time of login. When the test taker logins in with his username and password , then the webcam captures detected face, finds the encodings and then compares it with the all the known encodings created at the time of training.

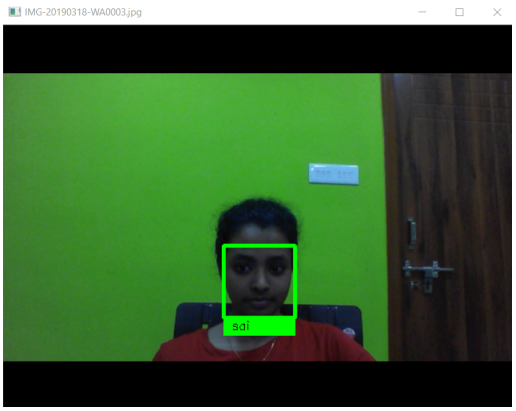


Figure 4.1: User Verification Output Console

Figure 4.2: User Verification Output

The known encoding that is close enough to the unknown encoding, outputs the corresponding name of the person. This name is matched with test taker's name in the database. If he is the valid user, then authentication is successful. If he is not the valid user, the webcam waits for 10seconds to find the user, and then warn the examinee saying "INVALID EXAMINEE".

4.3.2 Phone Detection

Our online exam rule prohibits the use of any type of mobile phones. Therefore, the presence of a mobile phone in the testing room can be an indication of potential cheating. With advancements in mobile phone technology, there are many ways to cheat from them, such as reading saved notes, text messaging friends, browsing the internet, and taking a snapshot of the exam to share with other test takers.

Phone detection is challenging due to various sizes, models and shapes of phones. Some test takers might have large touch screens while others might use a button-based flip phones. Keeping all the challenges in mind, the object detection YOLOv3 model is trained with the COCO dataset on GPU. It consists of more than 200K labeled images with 80 class labels including mobile phone and person.

After the test taker logins in successfully and starts the exam, there will be continuous video capturing from webcam of what the test taker is doing.

The neural network takes in the input frame by frame from the webcam. Using the weights that are generated and saved during training, it detects if the current frame has a phone or not. If a phone is detected, the system warns the test taker of using a phone. If the test taker is found using the phone more than thrice, then the system automatically comes out the exam.

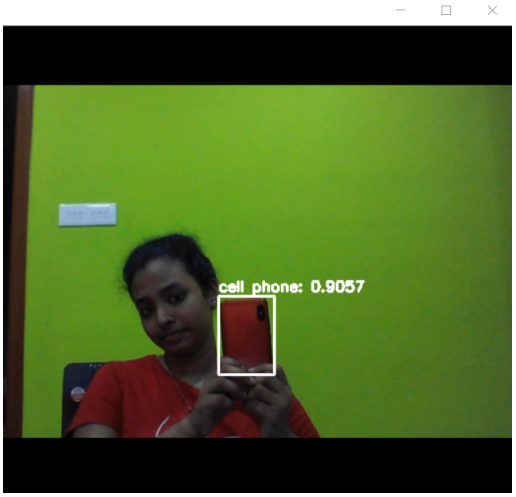


Figure 4.3: Phone Detection output

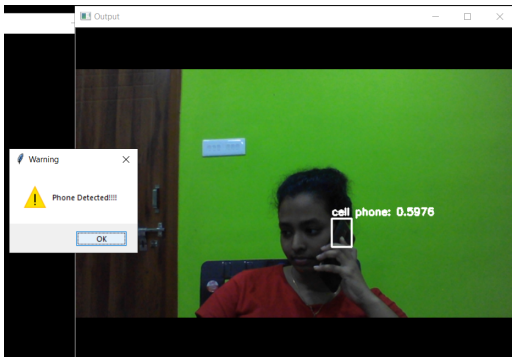


Figure 4.4: Phone Detection Alert Output

4.3.3 Multiple Person Detection

Solving the problem of usage of phones, can't alone be an effective OEP system. There are many more factors that should be taken into account and one of them is "asking help from another person". The test taker might seek assistance with another person present in the room. This is also a concern to be solved.

As we have discussed earlier, the YOLOv3 model is trained to detect mobile phone as well as person. Along with phone detection, when the neural network takes frame by frame input from webcam, it also keeps track of number of persons in the frame. If the count of persons is more than one, then it warns the test taker to take the exam alone with another person. As in phone detection, it is a continuous process and if the warning comes up more than thrice then, the system comes out the exam immediately.

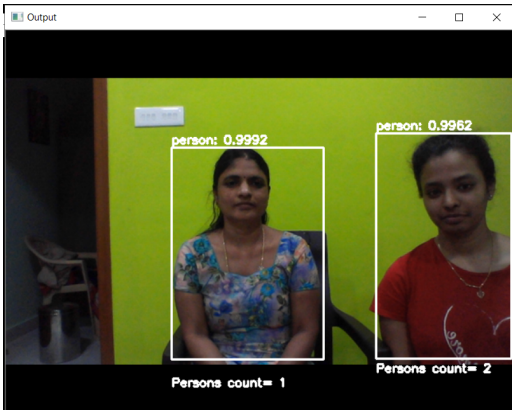


Figure 4.5: Multiple Person Detection Output

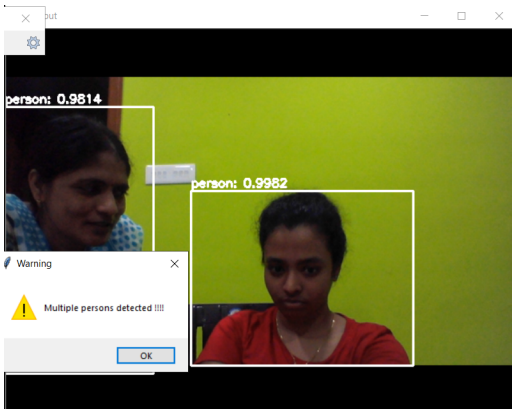


Figure 4.6: Multiple Person Alert Output

4.3.4 Invalid Examinee Detection

A rare but an important challenge that should be take care in online exam is the change of test taker after authentication or in between the exam. Most of the current OEP systems don't have this feature which might be an advantage to cheaters.

If test taker is writing the exam alone, with any help from gadgets and another person, it doesn't completely mean that he is not cheating. The person taking the exam might not be the authorized one. When, the test taker successfully logins in after authentication, then there will be continuous video capturing of the current frame. This detection is a two step process.

First, using YOLOv3 model, it detects if there is any person in the frame. If there is only one person, then the test taker's image is captured and saved. Second, the image saved is then send for the face recognition task. It finds the 128 encodings of the current test taker image and compares it with the original test taker image. If the both are same, then exam continues without any interruption. If not, the test taker can't attend the exam.

4.3.5 Active Window Detection

During an online test there is a huge possibility of the test taker accessing internet or the filest from the local computer to cheat. This is one of the major issue and mostly used method of cheating. So, we came up with a

solution that continuously keeps a track of the tab's activeness in which the website is opened and warn the user whenever he sway away from the tab or access a new tab or window this also includes when the user sway away in order to access files from the local computer or while using a split screen.

In any kind of javascript application we develop there may be a feature or any change in the application which reacts according to the current user visibility state, this could be to pause a playing video when the user ALT+TABs to a different window, tracking stats about how the users interact with our application, how often does him switch to a different tab, how long does it take him to return and a lot of performance improvements that can benefit from this kind of API.

The Page Visibility API provides us with two top-level attributes: document.hidden (boolean) and document.visibilityState (which could be any of these strings: "hidden", "visible", "prerender", "unloaded"). This would not be not good enough without an event we could listen to though, that's why the API also provides the useful visibilitychange event.

Dealing with vendor issues

Some of the implementations on some browsers still need that the attributes or even the event name is vendor-prefixed, this means we may need to listen to the msvisibilitychange event or check for the document.webkitHidden or the document.mozHidden attributes. In order to do so, we should check if any vendor-prefixed attribute is set, and once we know which one is the one used in the current browser (only if there's the need for a prefix), we can name the event and attributes properly.

Other issues

There is a challenging issue around the "Page Visibility" definition: how to determine if the application is visible or not if the window focus is lost for another window, but not the actual visibility on the screen? what about different kinds of visibility lost, like ALT+TAB, WIN/MAC key (start menu / dash), taskbar/dock actions, WIN+L (lock screen), window minimize, window close, tab switching. What about the behaviour on mobile devices?

There's lots of ways in which we may lose or gain visibility and a lot of possible interactions between the browser and the OS, therefore I don't think there's a proper and complete "visible page" definition in the W3C spec. This is the definition we get for the document.hidden attribute:

Hidden Attribute

On getting, the hidden attribute MUST return true if the Document contained by the top level browsing context (root window in the browser's viewport) [HTML5] is not visible at all. The attribute MUST return false if the Document contained by the top level browsing context is at least partially visible on at least one screen.

If the defaultView of the Document is null, on getting, the hidden attribute MUST return true. To accommodate accessibility tools that are typically full screen but still show a view of the page, when applicable, this attribute MAY return false when the User Agent is not minimized but is fully obscured by other applications.

I've found several inconsistencies on when the event is actually fired, for example (Chrome 41.0.2272.101 m, on Windows 8.1) the event is not fired when I ALT+TAB to a different window/program nor when I ALT+TAB again to return, but it IS fired if I CTRL+TAB and then CTRL+SHIFT+TAB to switch between browser tabs. It's also fired when I click on the minimize button, but it's not fired if the window is not maximized and I click my editor window which is behing the browser window. So the behaviour of this API and it's different implementations are still obscure.

A workaround for this, is to compensate taking advantage of the better implemented focus and blur events, and making a custom approach to the whole "Page Visibility" issue using an internal flag to prevent multiple executions, this is what I've come up with:

Three typical methods used to determine if the user can see the HTML page, however none of them work perfectly:

1. The W3C Page Visibility API is supposed to do this (supported since: Firefox 10, MSIE 10, Chrome 13). However, this API only raises events

when the browser tab is fully overriden (e.g. when the user changes from one tab to another one). The API does not raise events when the visibility cannot be determined with 100% accuracy (e.g. Alt+Tab to switch to another application).

2. Using focus/blur based methods gives you a lot of false positive. For example, if the user displays a smaller window on top of the browser window, the browser window will lose the focus (onblur raised) but the user is still able to see it (so it still need to be refreshed).

3. Relying on user activity (mouse move, clicks, key typed) gives you a lot of false positive too. Think about the same case as above, or a user watching a video.

In order to improve the imperfect behaviors described above, I use a combination of the 3 methods: W3C Visibility API, then focus/blur and user activity methods in order to reduce the false positive rate. This allows to manage the following events:

1. Changing browser tab to another one (100% accuracy, thanks to the W3C Page Visibility API)

2. Page potentially hidden by another window, e.g. due to Alt+Tab (probabilistic = not 100% accurate)

3. User attention potentially not focused on the HTML page (probabilistic = not 100% accurate)

This is how it works:

When the document lose the focus, the user activity (such as mouse move) on the document is monitored in order to determine if the window is visible or not. The page visibility probability is inversely proportional to the time of the last user activity on the page: if the user makes no activity on the document for a long time, the page is most probably not visible. The code below mimics the W3C Page Visibility API: it behaves the same way but has a small false positive rate. It has the advantage to be multibrowser (tested on Firefox 5, Firefox 10, MSIE 9, MSIE 7, Safari 5, Chrome 9).

4.4 Web Application

4.4.1 Introduction

A web application or dynamic website generates content based on retrieved data (most of the time is a database) that changes based on a user's interaction with the site. In a web application, the server is responsible for querying, retrieving, and updating data. This causes web applications to be slower and more difficult to deploy than static websites for simple applications.

4.4.2 UI Design

The UI part of the project is a web application designed using HTML,CSS, javascript,MYSQL The web application is majorly divided into 5 pages:

1. **Home:**
   Home page provides necessary information of the web application.
2. **Register:**
   A new user should get registered first to access the services of the web application.
3. **Login:**
   Only after a successful login the test taker will be redirected to the test page.
4. **Guidelines:**
   Provides the test taker with the necessary guidelines, do's and dont's.
5. **Test:**
   The test page is where the test taker takes the exam.

4.4.3 Working

The opening page 'Home' provides with the necessary information about the web application then the user needs to register himself for the exam this takes in the credentials along with a picture of the user. These credentials pictures will be stored into the database. Only after the user get registered he is considered authorized. Then comes the process of login where the test

taker needs to login with authorized credentials once the credentials are considered correct by verifying them with the ones in the database,then the webcam automatically takes a snap of the test taker and face verification is done by comparing the picture in the database and the picture clicked this is done by the face recognition algorithm integrated in the backend of the login page. After the login is successful the test taker is redirected to the guidelines page which provides the do's and dont's during the exam. Once the test taker clicks 'START EXAM' a continuous surveillance of the test location begins to take place until the end of the exam this keeps a track of whether the test taker is cheating or not and detects for phone,multiple people or unauthorized user until the end of the exam, if found any warnings are generated till a specified count if that exceeds the test taker is forcefully logged out of the exam.

4.4.4 How are the deep-learning models integrated into the backend of the web pages?

Introduction

Flask is a small and lightweight Python web framework that provides useful tools and features that make creating web applications in Python easier. It gives developers flexibility and is a more accessible framework for new developers since you can build a web application quickly using only a single Python file. Flask is also extensible and doesn't force a particular directory structure or require complicated boilerplate code before getting started.
As part of this tutorial, you'll use the Bootstrap toolkit to style your application so it is more visually appealing. Bootstrap will help you incorporate responsive web pages in your web application so that it also works well on mobile browsers without writing your own HTML, CSS, and JavaScript code to achieve these goals. The toolkit will allow you to focus on learning how Flask works.

Flask uses the Jinja template engine to dynamically build HTML pages using familiar Python concepts such as variables, loops, lists, and so on. You'll use these templates as part of this project. In this tutorial, you'll build a small web blog using Flask and SQLite in Python 3. Users of the application can view all the posts in your database and click on the title of a

post to view its contents with the ability to add a new post to the database and edit or delete an existing post.

Installing Flask

You need to activate your Python environment and install Flask using the pip package installer.

How Does a Flask App Work?

The code lets us run a basic web application that we can serve, as if it were a website.

```
from flask import Flask

app = Flask(__name__)

@app.route("/")
def home():
    return "Hello, World!"

if __name__ == "__main__":
    app.run(debug=True)
```

Figure 4.7: Codesnippet

This piece of code is stored in our main.py.

Line 1: Here we are importing the Flask module and creating a Flask web server from the Flask module.

Line 3: name means this current file. In this case, it will be main.py. This current file will represent my web application. We are creating an instance of the Flask class and calling it app. Here we are creating a new web application.

Line 5: It represents the default page. For example, if I go to a website such as “google.com/” with nothing after the slash. Then this will be the

default page of Google.

Line 6–7: When the user goes to my website and they go to the default page (nothing after the slash), then the function below will get activated.

Line 9: When you run your Python script, Python assigns the name "main" to the script when executed.

If we import another script, the if statement will prevent other scripts from running. When we run main.py, it will change its name to main and only then will that if statement activate.

Line 10: This will run the application. Having debug=True allows possible Python errors to appear on the web page. This will help us trace the errors.

Let's Try Running main.py

In your Terminal or Command Prompt go to the folder that contains your main.py. Then do py main.py or python main.py. In your terminal or command prompt you should see this output.

Figure 4.8: Output in terminal

The important part is where it says Running on http://127.0.0.05:5000/. 127.0.0.05 means this local computer. If you do not know the meaning of this (like I didn't when I started — this article is really helpful), the main idea is that 127.0.0.05 and localhost refer to this local computer.
Go to that address and you should see the following:

Figure 4.9: Flask Application

4.5 Database Collection

4.5.1 Dataset for Face Recognition Task

To train the face recognition model we tried to obtain the dataset from the process of registration which ask the user for their pictures which are stored in separate folders and used to train the face recognition module used. We came across the valid question of minimal dataset but to make our system work efficiently and make it real time this is the most subtle solution of obtaining test taker authorized pictures to train the system or the second choice is the educational institution providing the pictures of the test takers. The minimal dataset issue is solved by using "ONE SHOT LEARNING". The idea here is that we need to learn an object class from only a few data and that's what One-shot learning algorithm is. In face recognition systems, we want to be able to recognize a person's identity by just feeding one picture of that person's face to the sytem.This can be achieved by using "Siamese neural network" which is convenient to retrain the model everytime we add a picture of a new person into the database.

4.5.2 Dataset for Object Detection Task

YOLOv3 model is trained with the COCO dataset on GPU."COCO is a large-scale object detection, segmentation, and captioning dataset."Common Objects in Context (COCO) literally implies that the images in the dataset are everyday objects captured from everyday scenes. This adds some "context" to the objects captured in the scenes.COCO provides multi-object labeling, segmentation mask annotations, image captioning, key-point detection and panoptic segmentation annotations with a total of 80 categories, making it

a very versatile and multi-purpose dataset.It consists of more than 200K labeled images with 80 class labels including mobile phone and person.We had extracted the required classes from the dataset and used them to train our object detection model.

4.5.3 Database for User Credential Verification

phpMyAdmin is a free and open source software that lets you handle the administration of MySQL over the web. You can easily manage the database through a graphic user interface known as phpMyAdmin in this case. phpMyAdmin is written in PHP and has gained a lot of popularity in terms of web-based MySQL management solution. You can perform operations on MySQL via phpMyAdmin user interface while you can still directly execute SQL queries. And it lets you carry out operations like editing, creating, dropping, amend MySQL database, alter fields, tables, indexes, etc. In fact, which user should be given what privileges, you can manage that too. phpMyAdmin has huge multi-language community support. We used phpmyadmin to create tables and store user credentials data in it.

4.6 Result

4.6.1 Introduction

The system once designed is tested on various scenarios to check its efficiency, performance and whether the system was successful to fulfil all the objectives or not.This chapter presents all the results of the implementation of the automated proctoring system in detail by considering a feature analysis, its working and the outputs obtained in every possible case and scenario.

4.6.2 Feature Analysis and Cheat Detection

As mentioned, there are six main features namely:

1. User Authentication
    (a) If the user is registered, he/she needs to login to access test.
    (b) If the user is not registered, he/she needs to get registered first then login.

2. Face Verification
3. Active Window Detection
4. Phone Detection
5. Multiple People Detection
6. Unauthorized Examinee Detection

User Authentication

During Login Process:
Case 1: If the user logins with correct credentials he/she would be considered for face verification process.

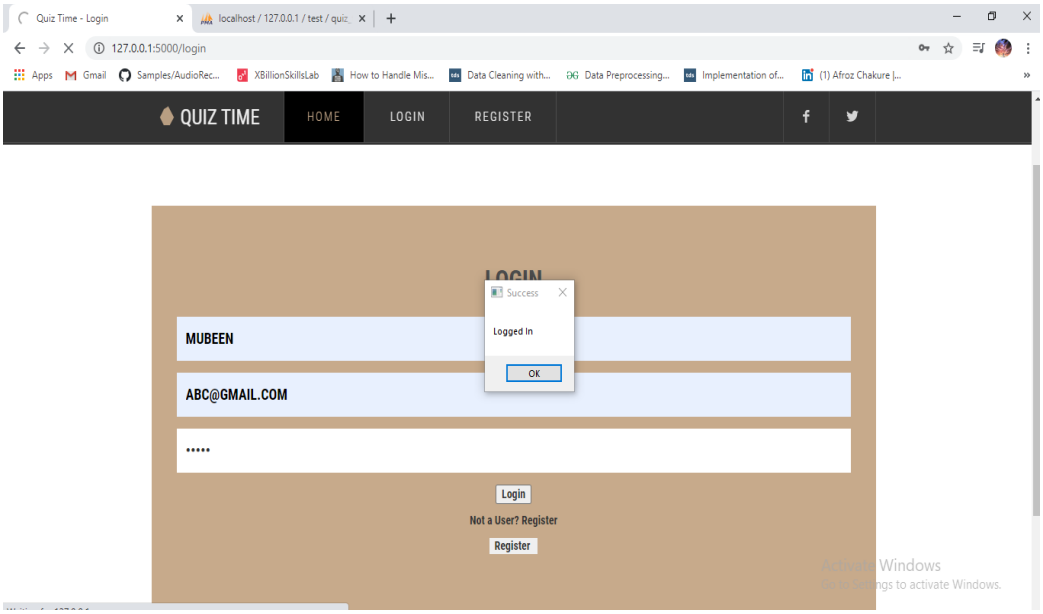


Figure 4.10: User Authentication: Case 1 result

Case 2: If the user tries to login with incorrect credentials a warning is generated.

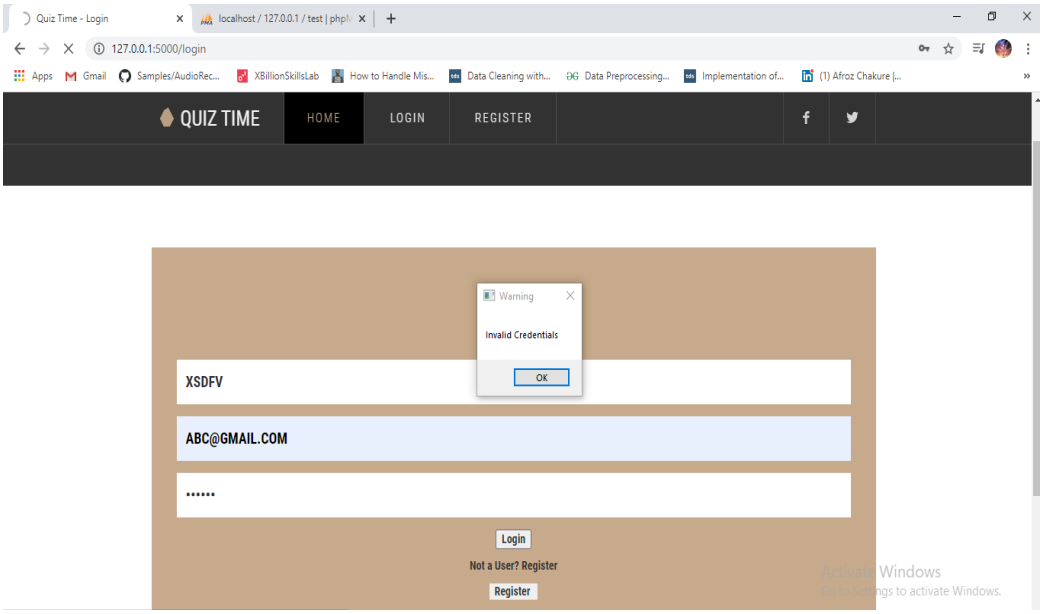


Figure 4.11: User Authentication: Case 2 result

Face Verification

Once after the user types in correct credentials the webcamera automatically capture user's face to verify it.
Case 1: If the user is authorized he/she would be redirected to Guidelines page.

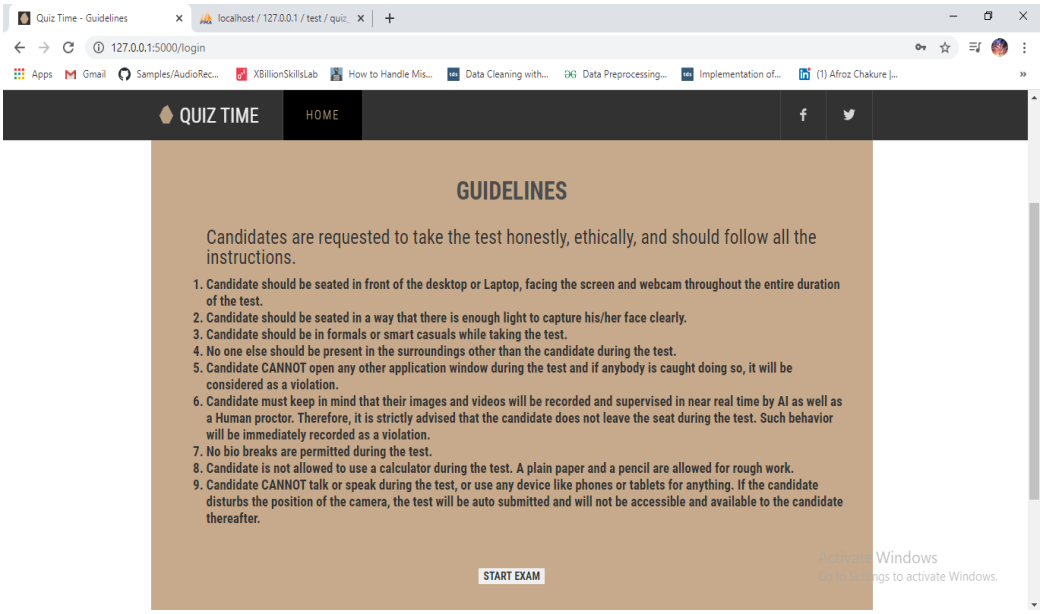


Figure 4.12: Face Verification: Case 1 result

Case 2: If the user is unauthorized, a warning is generated.

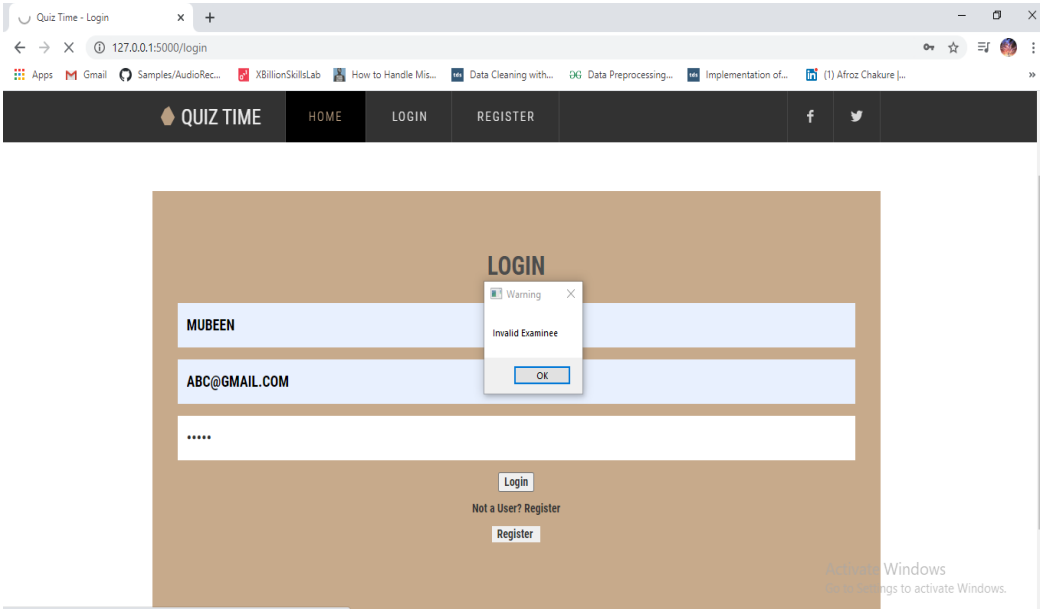


Figure 4.13: Face Verification: Case 2 result

Active Window Detection

Once after the user starts to take the test a continuous track is kept on the activeness of the tab or window.
Case 1: If the user stays on the same tab until the end of the exam, no warning is generated.

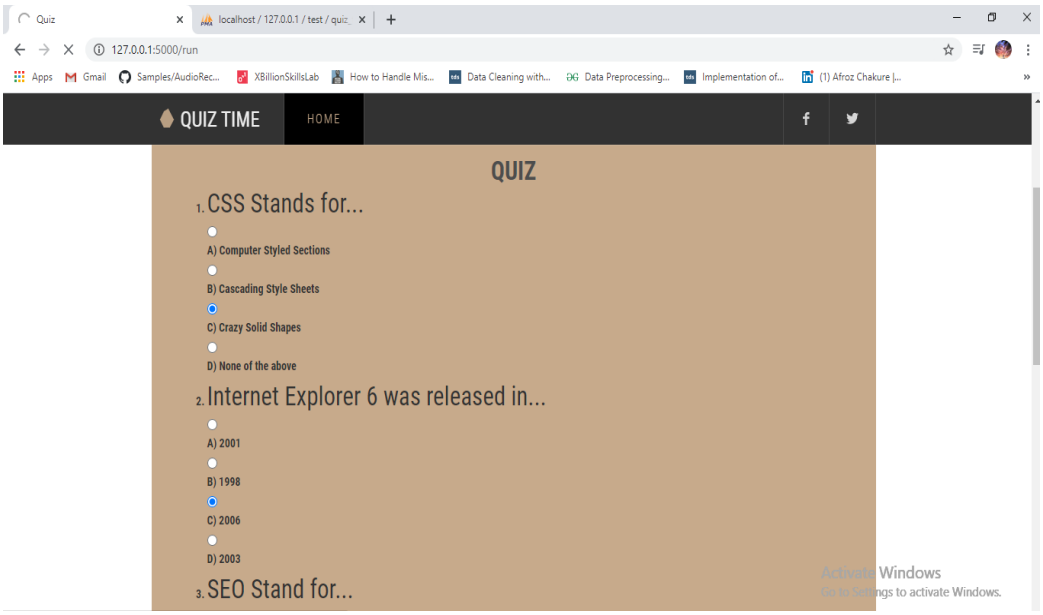


Figure 4.14: Active Window Detection: Case 1 result

Case 2: If the user tries to sway away from the tab, open a new tab, access local files, uses split screen etc then a warning is generated.

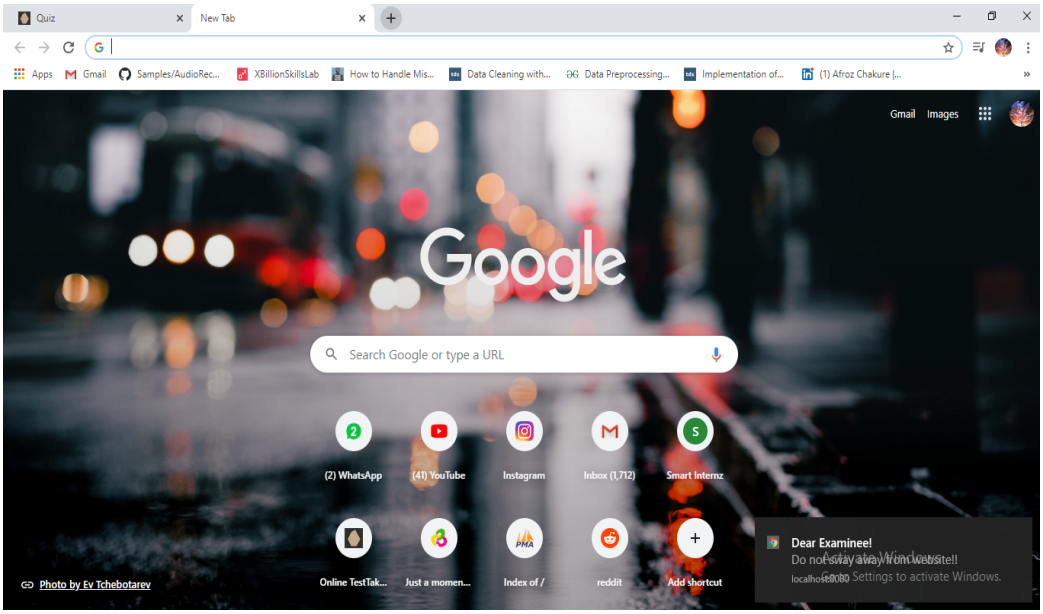


Figure 4.15: Active Window Detection:Case 2 result-1

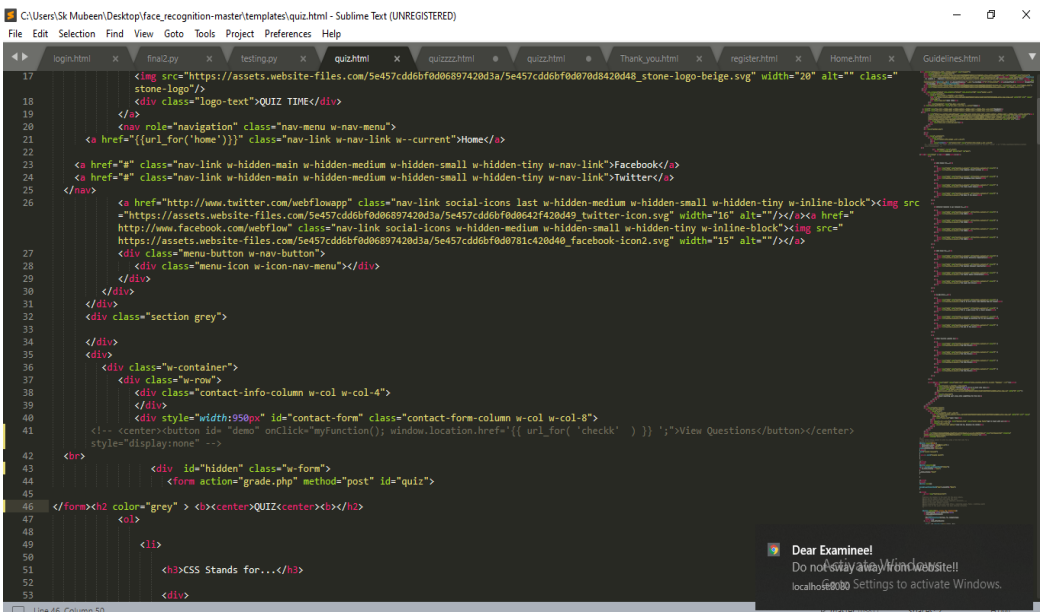


Figure 4.16: Active Window Detection:Case 3 result-2

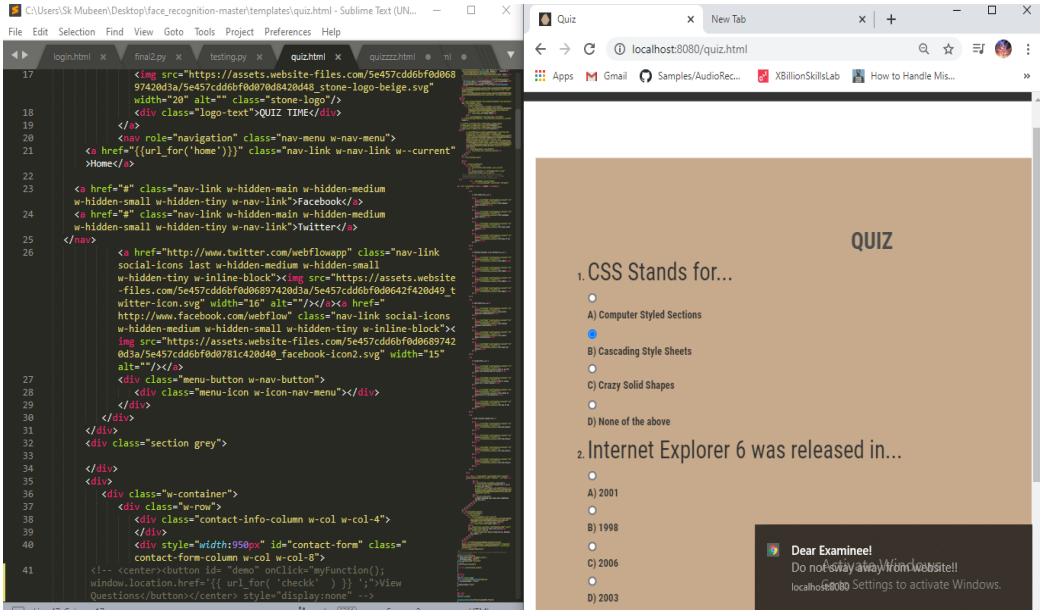


Figure 4.17: Active Window Detection:Case 4 result-3

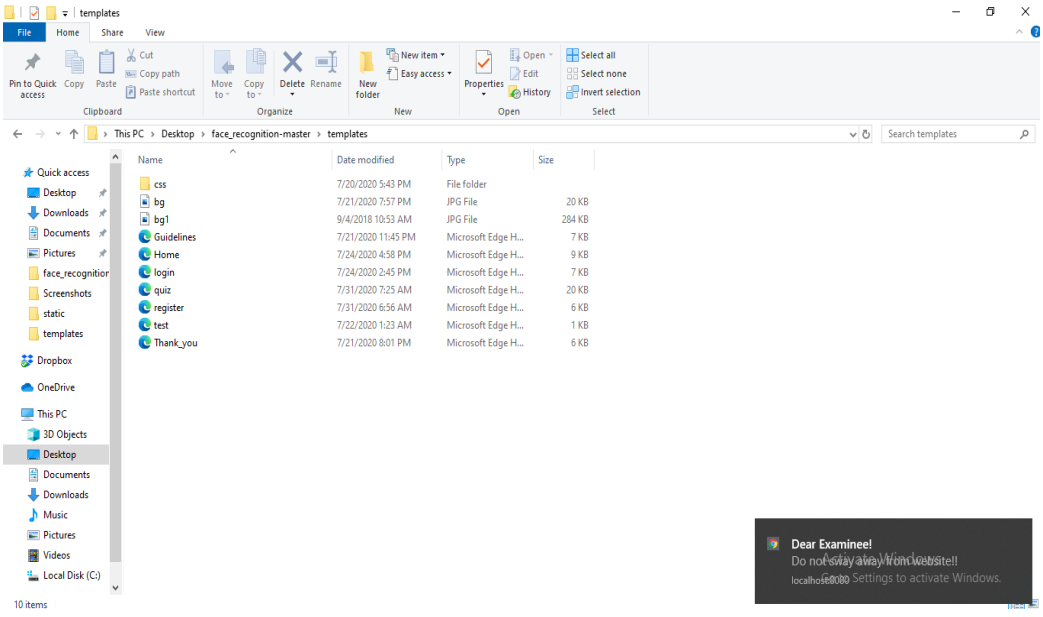


Figure 4.18: Active Window Detection:Case 5 result-4

Phone Detection

During the test,
Case 1: If the user takes the exam without any electronic gadgets nearby, no warning is generated.
Case 2: If the user tries to use mobile then a warning is generated.

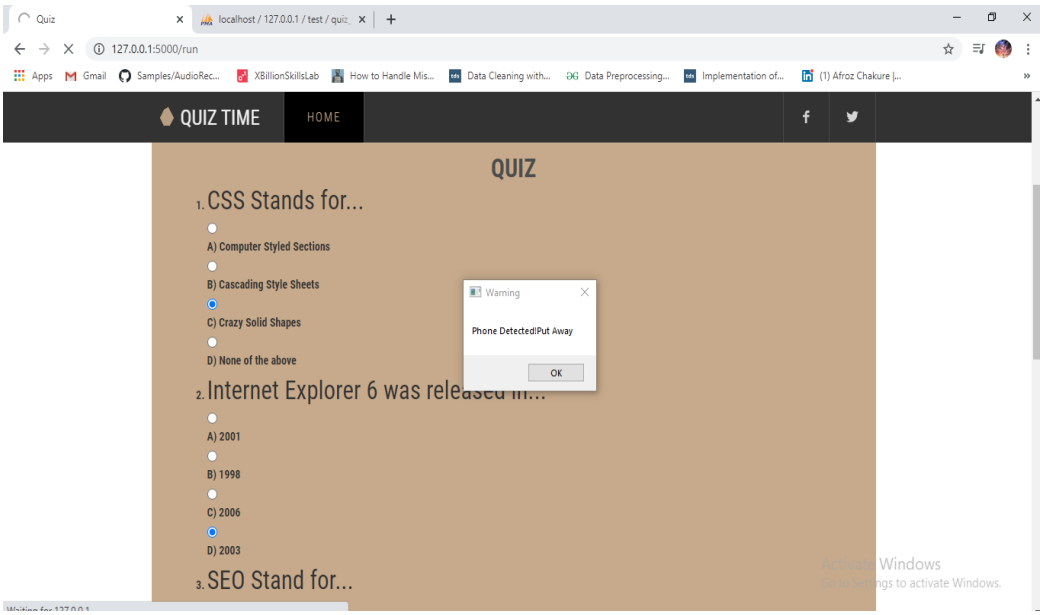


Figure 4.19: Phone Detection: Case 2 result

Multiple People Detection

During the test,
Case 1: If the user is the only one in the test location then no warning is generated.
Case 2: If the test location consists of any other rather than the test taker a warning is generated.

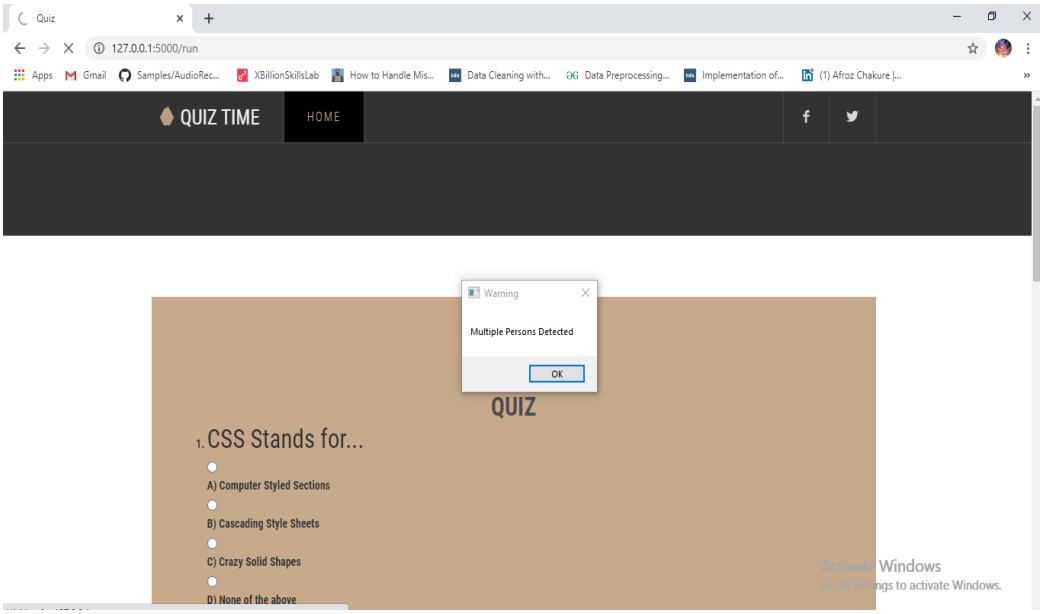


Figure 4.20: Multiple People Detection: Case 2 result

Unauthorized Examinee Detection

During the test,
Case 1: If the authorized user takes the test until the end, no warning is generated.
Case 2: If the user tries to swap his place or makes someother take the test at any instance then a warning is generated.

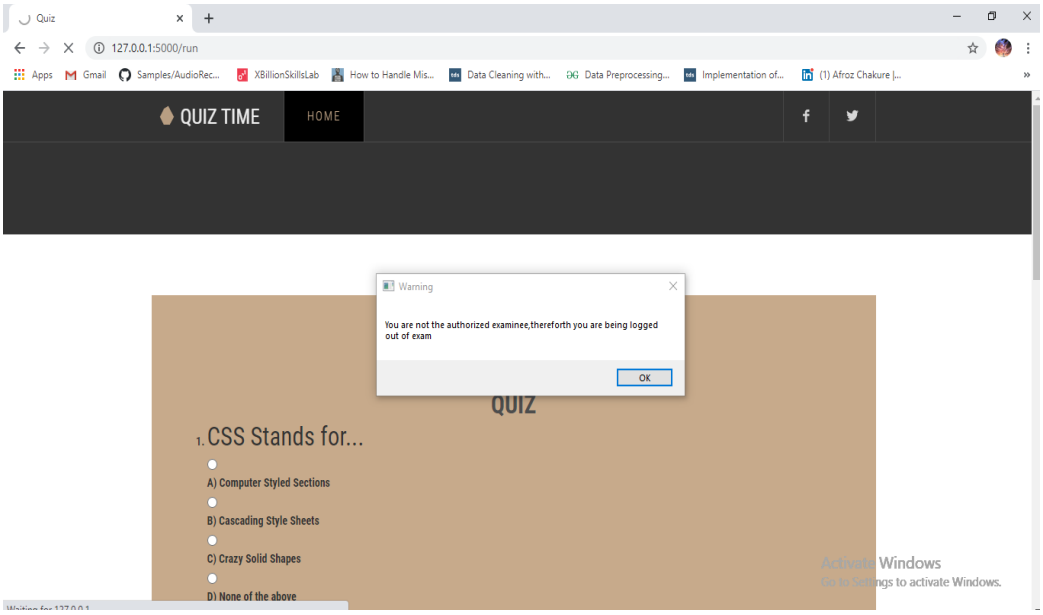


Figure 4.21: Unauthorized Examinee Detection: Case 2 result

Henceforth these are the results or outputs obtained during all possible scenarios.

Chapter 5

CONCLUSION

This project presents a automated analytics system for online exam proctoring, which aims to maintain academic integrity in e-learning. The system is affordable and convenient to use from the text taker's perspective, since it only requires to have a webcamera. With the captured videos, we extract low-level features from six basic components: user verfication, phone detection, active window detection, multiple people detection, unauthorized test taker detection. These features are then processed in a temporal window to acquire high-level features, and then are used for cheat detection. Finally, with the features integrated the common methods of cheating are prevented upto a high level this makes the automated online proctoring efficient and being a web application it is free of cost. These promising results warrant further research on this important behavior recognition problem and its educational application.

Chapter 6

FUTURE SCOPE

Remote proctoring tools is going to become the mainstay for online courses. It might soon impact other types of assessments too. The current trends include:

1. Enhancing the test taker's authentication by the use of biometric inputs from devices like smartwatches and fitness monitors.
2. Smartwatches and fitness monitors may also be employed to detect changes in pulse and temperature and send such data to proctoring software to serve as malpractice cues.
3. Facial recognition with sound and background noise detection is already used by Talview to avoid impersonation. Keyboard behavior analysis is also in use. In the future, touchscreen behavior analysis might be utilized as additional checks.
4. Head movement and position and illumination analysis are already in use by Talview for cheating. The tone of voice, facial expressions, etc. can be used in future.

Bibliography

[1] *Mask R-CNN.* Kaiming He, Georgia Gkioxari, Piotr Dollár, Ross Girshic.

[2] Redmon, Joseph and Farhadi, Ali. *YOLOv3: An Incremental Improvement.* , (2018).

[3] G. Cluskey Jr, C. R. Ehlen, and M. H. Raiborn. *Thwarting online exam cheating without proctor supervision.* Journal of Academic and Business Ethics, 4:1–7, 2011.

[4] Y. Atoum, L. Chen, A. X. Liu, S. D. H. Hsu and X. Liu, *Automated Online Exam Proctoring,* in IEEE Transactions on Multimedia, vol. 19, no. 7, pp. 1609-1624, July 2017, doi: 10.05109/TMM.2017.2656064.

[5] Xuanchong Li, Kai-min Chang, Yueran Yuan, and Alexander Hauptmann. 2015. *Massive Open Online Proctor: Protecting the Credibility of MOOCs certificates.* In Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work and Social Computing. Association for Computing Machinery, New York, NY, USA, 1129–1137.

[6] F. Schroff, D. Kalenichenko and J. Philbin, *FaceNet: A unified embedding for face recognition and clustering* ,2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, 2015, pp. 815-823.

[7] Yichun Shi ,Anil K. Jain and Nathan D. Kalka, *Probabilistic Face Embeddings*, CoRR,abs/1904.09658,2019

[8] Sang, J.; Wu, Z.; Guo, P.; Hu, H.; Xiang, H.; Zhang, Q.; Cai, B. *An Improved YOLOv2 for Vehicle Detection.* Sensors 2018, 18, 4272.

[9] Zhao, L.; Li, S. *Object Detection Algorithm Based on Improved YOLOv3.* Electronics 2020, 9, 537.

[10] M. Parchami, S. Bashbaghi and E. Granger, *Video-based face recognition using ensemble of haar-like deep convolutional neural networks,* 2017 International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, 2017, pp. 4625-4632, doi: 10.05109/IJCNN.2017.7966443.

[11] A. Wahid, Y. Sengoku, and M. Mambo. *Toward constructing a secure online examination system.* In Proc. of the 9th Int. Conf. on Ubiquitous Information Management and Communication, page 95. ACM, 2015.

[12] P. Guo, H. feng yu, and Q. Yao. *The research and application of online examination and monitoring system.* In IT in Medicine and Education, 2008. IEEE Int. Sym. on, pages 497–502, 2008

[13] I. Jung and H. Yeom. *Enhanced security for online exams using group cryptography.* Education, IEEE Trans. on, 52(3):340–349, 2009.

[14] X. Li, K.-m. Chang, Y. Yuan, and A. Hauptmann. *Massive open online proctor: Protecting the credibility of moocs certificates.* In ACM CSCW,pages 1129–1137. ACM, 2015.

[15] W. Rosen and M. Carr. *An autonomous articulating desktop robot for proctoring remote online examinations.* In Frontiers in Education Conf.,2013 IEEE, pages 1935–1939, 2013.

WEEKLY REPORTS

Week 1

Table 6.1: Week -1 Report

S.no	Date	Description of the work	Supervisor signature
1.	6-6-2020	Foundations of Convolutional Neural Networks: Learn to implement the foundational layers of CNNs (pooling, convolutions) and to stack them properly in a deep network to solve multi-class image classification problems.	
2.	7-6-2020	Deep convolutional models: case studies Learn about the practical tricks and methods used in deep CNNs straight from the research papers.	
3.	8-6-2020	Object detection Learn how to apply your knowledge of CNNs to one of the toughest but hottest field of computer vision: Object detection.	
4.	9-6-2020	Practical Implementation of Detection Algorithms.	
5.	10-6-2020	Special applications: Face recognition & Neural style transfer Discover how CNNs can be applied to multiple fields, including art generation and face recognition.Implement your own algorithm to generate art and recognize faces!	
6.	11-6-2020	Practical Implementation of Special applications: Face recognition & Neural style transfer.	

Name of the Supervisor	:	Prof.Suryakanth V Gangashetty
Supervisor's Sign	:	
Date	:	11-6-2020
Remarks by Supervisor	:	

Week-1 Summary

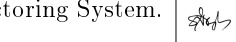
Duration: From 06-06-2020 to 11-06-2020

Learning Outcomes:

Got familiar with cnn topics such as,Computer Vision, Edge Detection Example, One Layer of a Convolutional Network, Simple Convolutional Network Example, Pooling Layers, CNN Example, Why Convolutions? Strided convolutions, ResNets, Why ResNets Work, Networks in Networks and 1x1 Convolutions, Inception Network Motivation, Inception Network, Using Open-Source Implementation, Transfer Learning, Data Augmentation, State of Computer Vision, Object Detection, Yolo Algorithm, What is face recognition?, One Shot Learning, Siamese Network, Triplet Loss, Face Verifiction and Binary Classification, What is neural style transfer?, What are deep ConvNets learning?, Cost Function

Week 2

Table 6.2: Week -2 Report

S.no	Date	Description of the work	Supervisor signature
1.	12-6-2020	Introduction to Domain: Deep Learning Project: Automated Remote Proctoring System. A brief look around of the project, its working and applications.	
2.	13-6-2020	Project Inspection: Requirements Data Collection,Programming language and Framework selection.	
3.	14-6-2020	Read related research papers: Made notes, understood the concepts and looked for new features.	
4.	15-6-2020	Worked on existing models with the help of github and understood the implementations.	
5.	16-6-2020	Chose efficient deep learning algorithm based upon their performaces and desired features required of the project.	
6.	17-6-2020	Planned the front-end	

Name of the Supervisor	:	Prof.Suryakanth V Gangashetty
Supervisor's Sign	:	
Date	:	17-6-2020
Remarks by Supervisor	:	

Week-2 Summary

Duration:From 12-06-2020 to 17-06-2020

Learning Outcomes:

1.Obtained a good knowledge about convolutional neural networks and gained a good idea about the project by referring to many research papers and existing models, had hands-on the existing technologies and were able to figure out the requirements and deep learning algorithms required to implement the given project. 2.Also got familiar with how and why research papers are written and their importance.

Week 3

Table 6.3: Week -3 Report

S.no	Date	Description of the work	Supervisor signature
1.	18-6-2020	Dataset collection for face recognition task	
2.	19-6-2020	Implemented of Mask R-CNN model for Face recognition task.	
3.	20-6-2020	Implemented of FaceNet model for Face recognition task.	
4.	21-6-2020	Implemented of python face recognition module using One-shot learning	
5.	22-6-2020	Implemented of python face recognition module using One-shot learning	
6.	23-6-2020	Finalized python face recognition module	

Name of the Supervisor	:	Prof.Suryakanth V Gangashetty
Supervisor's Sign	:	
Date	:	23-6-2020
Remarks by Supervisor	:	

Week-3 Summary

Duration: From 18-06-2020 to 23-06-2020

Learning Outcomes:

1.This week we worked on the face recognition algorithm required to implement the face verification feature in the project. 2.In the process, we were able to categorize algorithms based on their performance and our requirements then implemented the algorithms we thought would be a good fit. 3.Every algorithm had it's own approach while some where too dense to understand and some were adequate. 4.By the end of the week we were able to finalize a face recognition algorithm which stood out to be a good fit for our project.

Week 4

Table 6.4: Week -4 Report

S.no	Date	Description of the work	Supervisor signature
1.	24-6-2020	Dataset collection for Object detection task	
2.	25-6-2020	Implemented of Fast R-CNN algorithm for Object detection task.	
3.	26-6-2020	Implemented of YOLOv2 algorithm for Object detection task.	
4.	27-6-2020	Implemented of YOLOv3 algorithm for Object detection task	
5.	29-6-2020	Implemented of YOLOv3 algorithm for Object detection task	
6.	30-6-2020	Finalized YOLOv3 algorithm for Object detection	

Name of the Supervisor	:	Prof.Suryakanth V Gangashetty
Supervisor's Sign	:	
Date	:	30-6-2020
Remarks by Supervisor	:	

Week-4 Summary

Duration: From 24-06-2020 to 30-06-2020

Learning Outcomes:

1.This week we worked on the implementation of object detection feature which was responsible to detect cheating done by phone, help of others or a swap. 2.To find a algorithm which would satisfy our needs we tried understanding and implementing a few algorithms which are considered good with help of articles later we finalized Yolov3 as a good fit and worked on to created functions which would help us track out phone and persons instead of every object in test location. 3.After this, we created workaround which would identify multiple people and also unauthorized test taker by integrating face recognition task into it. 4. By the end we were able to solve three cheating issues.

Week 5

Table 6.5: Week -5 Report

S.no	Date	Description of the work	Supervisor signature
1.	1-7-2020	Searched and Implemented for programming methods to keep track of active window.	
2.	2-7-2020	Worked on page visibility API	
3.	3-7-2020	Worked on browser vendor issues	
4.	5-7-2020	Designed a webpage and implemented page visibility API in javascript	
5.	6-7-2020	Experimented with jQuery and blur and focus functions.	
6.	7-7-2020	Combined page visibility API with blur & focus functions and solved window sway issue	

Name of the Supervisor	:	Prof.Suryakanth V Gangashetty
Supervisor's Sign	:	
Date	:	7-7-2020
Remarks by Supervisor	:	

Week-5 Summary

Duration: From 02-7-2020 to 07-7-2020

Learning Outcomes:

1.This week we worked on active window detection. 2.To start with, we searched for all the possible methods that would keep a track of active window or tab and warn when swayed away from it. 3.After few implementations in python and javascript we finalised to use javascript integrated in a HTML webpage. 4.We utilised the page visibility API and implemented it. 5.Later in the week, we were able to solve the bugs which were not managed by the API by using functions like blur and focus. This helped us detect the activeness of the window at every instant of the time. 6. By the end of the week we were able to solve the active window detection problem.

Week 6

Table 6.6: Week -6 Report

S.no	Date	Description of the work	Supervisor signature
1.	8-7-2020	Came across the issue of multiple persons and unauthorized test taker in mid of the test.	
2.	9-7-2020	Created a workaround using object detection class label to solve multiple people problem	
3.	10-7-2020	Solved the problem of unauthorized test taker by integrating face recognition task while object detection is running with the help of "person" class label	
4.	12-7-2020	Integrated the previous works into the object detection algorithm.	
5.	13-7-2020	Worked and fixed bugs.	
6.	14-7-2020	Optimized the code.	

Name of the Supervisor	:	Prof.Suryakanth V Gangashetty
Supervisor's Sign	:	
Date	:	14-7-2020
Remarks by Supervisor	:	

Week-6 Summary

Duration: From 8-7-2020 to 14-7-2020

Learning Outcomes:

1.In this week, we worked on solving issues like multiple people and unauthorized test taker. 2.The implementation turned out to be a little tricky after a lot of brainstroming.3. We decided to make workarounds on the object detection algorithm which would solve this issues. 4.We used to the person label of COCO dataset and maitained its count continously until the end of the exam. This helped us detect multiple people. 5.Later in the week, we integrated face recognition whenever a single was detected in the test location and kept a track of whether the test taker is authorized or not. This solved the second issue.

Week 7

Table 6.7: Week -7 Report

S.no	Date	Description of the work	Supervisor signature
1.	16-7-2020	Chose Flask framework to integrate front end with python and got familiar with flask concepts.	
2.	17-7-2020	Designed User Interface with the help of HTML and CSS.	
3.	19-7-2020	Integrated JavaScript and solved window sway issue	
4.	20-7-2020	Connected the web application to Xampp database	
5.	21-7-2020	Intergrated the deep learning algorithms into flask	
6.	22-7-2020	Connected HTML pages with flask	

Name of the Supervisor	:	Prof.Suryakanth V Gangashetty
Supervisor's Sign	:	
Date	:	22-7-2020
Remarks by Supervisor	:	

Week-7 Summary

Duration:From 16-7-2020 to 22-7-2020

Learning Outcomes:

1.This week we worked on the front end of the automated proctoring system. 2.We designed web pages, using HTML and CSS and connected them. 3.For authentication purposes we connected the web application with Xampp Database. 4. Later, the javascript was integrated to implement active window detection. 5.Then, we decided on to use Flask to connect the front end with python. 6.The rest of the days we worked on integrating the deep learning models as the backend of web application.

Week 8

Table 6.8: Week -8 Report

S.no	Date	Description of the work	Supervisor signature
1.	23-7-2020	Tested and fixed bugs	
2.	24-7-2020	Tested and fixed bugs	
3.	26-7-2020	Tested and fixed bugs	
4.	27-7-2020	Report making	
5.	30-7-2020	Evaluation	
6.	31-7-2020	Evaluation	

Name of the Supervisor	:	Prof.Suryakanth V Gangashetty
Supervisor's Sign	:	
Date	:	31-7-2020
Remarks by Supervisor	:	

Week-8 Summary

Duration: From 23-7-2020 to 31-7-2020

Learning Outcomes:

1.This week, we continously tested the system and worked on fixing the bugs. 2.The made us familiar with the real-time problems that a real-time web application faces.

To be filled by the internal internship supervisor at the time of report submission:

Rubrics for Industrial Training- Report (CIE 1 & CIE II)

CRITERIA	Excellent (8-10)	Good (4-7)	Poor (1-3)	Points
Lay-out & Organization (CIE 1 & 2)	Informative summary; Table of contents in logical sequence; Page numbering; Suitable sub-titles.	Inadequate summary; Table of contents not in sequence; Page numbering; acceptable sub-titles.	No summary; No Table of contents; No page numbering; unsuitable sub-titles.	10/10
	Excellent (8-10)	**Good (4-7)**	**Poor (1-3)**	
Graphics (CIE 1 & 2)	Effective use of pictures, models, diagrams, charts, tables and graphs.	Some appropriate use of pictures, models, diagrams, charts, tables and graphs.	No use of pictures, models, diagrams, charts, tables and graphs.	9/10
	Excellent (8-10)	**Good (4-7)**	**Poor (1-3)**	
Conventions (CIE 1 & 2)	Generally error free in regards to sentence structure, punctuation, capitalization, spelling, and standard usage.	Sentence structure, punctuation, capitalization, spelling, and standard usage errors are noticeable, but do not seriously impair readability.	Errors in sentence structure, punctuation, capitalization, spelling, and standard usage impair readability.	10/10
	Excellent (8-10)	**Good (4-7)**	**Poor (1-3)**	
Understanding of defined problem (CIE 1)	Clear understanding of Problem defined by the industry is reflected.	Moderate understanding of problem defined by the industry is reflected.	Poor understanding of problem defined by the industry is reflected.	10/10
	Excellent (8-10)	**Good (4-7)**	**Poor (1-3)**	
Study of existing system. (CIE 2)	Detailed explanation and specifications of the existing system.	Moderate explanation and specifications of the existing system.	Poor explanation and specifications of the existing system.	10/10
	Excellent (8-10)	**Good (4-7)**	**Poor (1-3)**	
Observations (CIE2)	Key observations clearly mentioned with excellent understanding.	Observations clearly mentioned with good understanding.	No observations are mentioned. existing system.	10/10

Maximum Marks : 59