VIDEO ANALYSIS SYSTEM

Capt. Adnan
Capt. Athar
GC Mohsin

Submitted to the Faculty of Computer Software Engineering Department
National University of Sciences and Technology, Islamabad
in partial fulfilment for the requirements of a B.E Degree in
Computer Software Engineering
May 2019

CERTIFICATE OF APPROVAL

This is to officially state that the thesis work contained in this report

"Video Analysis System"

Is carried out by:

Capt Adnan, Capt Athar and GC Mohsin

under my supervision and that in my judgment, it is fully ample, in scope and excellence, for the degree of Bachelor of Computer Software Engineering from National University of Sciences and Technology (NUST).

Approved By:

Signature: ________________________
Supervisor: Asst Prof Mobeena Shahzad
MCS, Rawalpindi

ABSTRACT

Security today is a sensitive issue. At present in MCS, surveillance is done manually but with our application developed and installed a lot of stuff can be automated. Our algorithm will be generalizable so given enough data and equipment it could be generalized for detecting other suspicious stuff in the video. It is a key step towards an era of surveillance.

Video Analysis System is a python based desktop application which is used to analyze live camera feed to detect the anomalies before hands. It includes cross platform support and can be executed on any three of known Operating Systems.

VAS will take video feed as input and will tell (through interface) what objects are existing in current frame. A suspicious object includes a **person trying to breach a wall** or a **tool i.e. knife**. This has been done using a deep learning algorithms which is being trained on 4 multiple datasets. First dataset was being made using a CCTV camera on the wall of MCS, Second Dataset was being scrapped from Internet (due to insufficient info in Custom dataset) Images of weapons were being scrapped from the internet. The third dataset was made using a mobile phone with intention to classify any person who is trying to breach the wall. Last dataset used was MS-COCO, which is a labeled dataset for almost 80 different objects

Using these datasets, we almost made 8-10 deep learning models. Including the renown architectures such as Mobile-Net, YOLO, Simple CNN. The final product consists of two working models which classifies all classes other than weapons.

DECLARATION OF ORIGINALITY

We hereby declare that the work contained in this report and the intellectual content of this report are the product of our work. This thesis report has not been formerly published in any structure nor does it include any verbatim of the published resources which could be treated as violation of the international copyright decree. We also affirm that we do recognize the terms ‘plagiarism’ and ‘copyright’ and that in case of any copyright infringement or plagiarism established in this thesis, we will be held fully accountable of the consequences of any such violation.

ACKNOWLEDGEMENTS

All glory goes to Almighty ALLAH who led me to this extent. May all glory, honor and Adoration be unto thy Name.

My special thanks go to my supervisors **Asst Prof Mobeena Shahzad** for their guide line, in completing this project successfully. I wish to add my acknowledgment to the teachers and fellows from MCS, where it all started.

A real and heart full gratitude towards **Dr. Adnan Ahmed Khan** (Head of Computer Software Department) for his guidance and facilitation for the Project. He helped us in deciding our project and later he also helped us in defining the direction of our research

This project is a joint effort of our group along with some exceptional teachers at MCS. Dr. Zaki helped us a lot with the deep learning weapons model and introduced us to concepts like transfer learning. Dr Ahemd Muqeem Sheri on the other hand helped us with making the dataset for wall climbing model. Along with that he helped us with finding the flaws in our research. Moreover, he refined our idea and introduced us to many relevant concepts of Computer Vision.

TABLE OF CONTENTS

Contents

Chapter 1

Introduction

Video Analysis System is a python based deep learning tool which can analyze your feed in real time and detect up to 80 objects in real time. This tool requires python3 to be installed on your system.

We made a video analysis system, installed on the boundary wall of MCS which will take video feed as input and will tell (through interface) whenever a suspicious activity happens. A suspicious activity includes a **crowd appearing** or a **tools and Vehicles** coming near the wall. Placed a CCTV camera on MCS wall which will get the view of road and whenever the incidents mentioned above happen. The main technologies we are using are CCTV for capturing the live feed, an interfacing mechanisms (probably in python) in order to interact with the system. Furthermore, an advanced deep learning solution for frame classification using Convolutional Neural Network.

This system is based on deep learning algorithms which process in real time. The results are shown on a screen immediately

1.1 Intended Audience and Reading Suggestions

The thesis report of VAS is meant for all the stake holders.

1. **Project Supervisor:** It will help to supervise the project and guide the team better.
2. **Development Team:** It will help the developer to develop the product and to trace back the functional requirements.
3. **Testing Team:** It will help the testers to understand the constraints.
4. **UG Project Evaluation Team:** Evaluation committee which will evaluate the progress of UG Projects.
5. **Students:** Any student who require any reference or help can read this report.

1.2 Motivation

Due to increase tensions, there exist a need for such system which can do real time analysis on the video feed and must be able enough to detect the weapons and other motioned classes. So, to the best of our knowledge we have tried to design a system using deep learning algorithms which on run time is able to detect the classes like bags, knifes, person climbing the wall and vehicles.

1.3 Project Scope

The scope of Video Analysis System is pretty much clear. We are trying to build a system that is capable of detecting any weapon or a suspicious crowd given the CCTV video feed. Furthermore, we will be using some advanced machine learning methods to train our system in order to improve the precision of the model. This system can be implemented anywhere on the CCTV keeping in mind the classifying classes. Furthermore, we will be using an automated alarm technique which will notify presence of weapon

1.4 Objective

Primary Objectives:

1. Automating the current video analysis system.
2. Detecting the false classes without any human effort.
3. To enhance the security of MCS.

Academic objectives:

1. To understand the basics of deep learning.
2. Interfacing in python
3. To practice the concepts of computer vision.

1.5 Deliverables

1. Complete working project
2. Documentation

Chapter 2

Literature Review

2.1 Introduction

Security today is a sensitive issue. At present in MCS, surveillance is done manually but with our application developed and installed a lot of stuff can be automated. Our algorithm will be generalizable so given enough data and equipment it could be generalized for detecting other suspicious stuff in the video. It is a key step towards an era of surveillance.

This problem can be solved using deep learning algorithms. For that you will have to see the basic concepts. We can basically divide Deep learning into few main architectures.

CNN: Mostly used for image processing/classification problem

RNN: Time series problem

GAN: Used for Image Generation

DNN: Dense Neural Networks are mostly used for numeric data or weak Image Classification problems.

Later in this chapter we will discuss the CNN and the important and relevant derivatives of CNN which we will be using in our problem.

2.2 Problem Domain

1. Real time object detection
2. Object Tracking
3. Image Processing
4. Computer Vision

2.3 Related Work

- **CNN**

Convolutional Neural network is a deep learning architecture which is mostly used in the field of image processing. On increase in the popularity of deep learning CNN has been used in various applications and along with that it has proved to be the best deep learning algorithm. There are more derived architectures on this model such as VGG nets, YOLO, Mobile-Net

- **Mobile Net**

Mobile Net is a type of CNN which is known to be a light tier architecture as it requires fewer processing power and has been used is light enough to be processed on an android device

- **YOLO**

YOLO (You only look once) is a deep learning segmentation model which has been under discussion on its improved results in objects segmentation. There exist 2 dark-net variants of YOLO and three version for each type. YOLO v3 is the best variant and we have used it MS COCO.

2.4 Shortcomings/issues

1. Low processing power
2. Lack of recourses.
3. No existing dataset for most classes

2.5 Work Flow

This problem was initially covering three major classes. First Weapon second bags and third suspicious activity. We initially worked on getting the deep learning dev skills. Which almost took 40% of the total time as the resources online are misleading and one problem is stated in multiple manners.

Initially we gathered a dataset of almost one lakh images comprising of three main classes

- Weapon
- Suspicious person
- Bags

The dataset was equally divided among the group and then later it was being labeled by all members. After that on studding the real time and low attributes of Mobile Net we used Keras and made our default structure of mobile net. The train accuracy on data was 98% but testing accuracy was 52%

Later we used transfer learning and scrapped 1000 weapon images from the internet and then used them in combination with our custom data in order to make the classifier work better. Which also failed.

Then we used the pre-trained YOLO v3 model and used it to classify segment our classes. The code of Yolo3 was written by using a helper blog on medium with the title “Making YOLO v3 From Scratch”

Later we formed another dataset which included the people who were trying to cross the wall. Later we labeled the data and classified it with training accuracy of 100%

2.6 Deliverables

Software Requirement Specification (SRS)

The purpose of the document is to present a detailed description of the Video Analysis System. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, its entire process, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for both the stakeholders and the developers of the system. It will explain how the system will particularly help disaster management authorities and concerned agencies to collaborate and cooperate with each other.

Software Design document

This software design specification doc describes the internal working of our model, the video analysis system we are trying to implement using different Object-Oriented mythologies. The document will cover in depth all the models under the study in order to make the system work in the maximum efficient manner. The document is majorly targeting 2 types of audience. First one, the stake holders of the design and the document along with the developers in order to maintain and develop the hardware and software of the purposed video analysis system. Furthermore, this document is another source of conveying the current status of our project along with the additions in our idea. The strategic changes we have made in our project. Such as, we are now not using the cloud services as we think it won't be necessary. The cloud service was being involved to serve the purpose of making a live detection system. While extending our research, we found this addition to be unnecessarily weighting on the overall cost and demanding a consistent bandwidth. Hence, we are looking out for a conventional method which might lower the cost while not affecting the efficiency.

Software Testing Document

This document has testing modules in which there are certain test cases which depicts the correctness and accuracy of the project.

Chapter 3

System Requirements Specification

3.1 Product Perspective

Security today is a sensitive issue. At present in MCS, surveillance is done manually but with our application developed and installed a lot of stuff can be automated. Our algorithm will be generalizable so given enough data and equipment it could be generalized for detecting other suspicious stuff in the video. It is a key step towards an era of surveillance.

3.2 Product Features

The features our program will provide are as follows: -

1. Detecting large crowd outside of MCS wall.
2. Detecting armed individual coming near MCS wall
3. Detecting bags left on the road near MCS.
4. Ringing the siren whenever above-mentioned things are detected.
5. If human eye can detect, the system will also detect

Just like any other System/product some limitation would exist

1. It'll always detect objects on which it'll be trained
2. Feed should be in such a quality that a human eye can detect things with high accuracy i.e. it shouldn't be blurred and preferably in 480p quality at least.
3. It won't be able to detect in the following circumstances

1. If the feed is blur
2. Video feed involves fog, rain, sandstorm, mist.
3. Weapon is small that it can't even be detected by human eye

3.3 User Classes and Characteristics

Our program will have just one type of user because it is not a social application.

VAS Operator: -

Having this user is not necessary. In case of false positives this user will be there to identify false positive where nothing suspicious was detected but still the siren wailed. Our accuracy will be high
but it won't be 100% (like any other system) so a user seeing the tape will be good but not necessary.

3.4 Operating Environment

1. Ubuntu OS: Since artificial intelligence libraries and codes are easy to setup on ubuntu, so we will use it for production and for deployment also.

2. GPU: We will train our algorithm on Google cloud because real time video detection requires extensive computation to be done. Since the production code will be written to run on cloud, it can only be deployed on a GPU or Google cloud environment. It can run on CPU but getting a video frame, detecting stuff and then giving output will take some time in which the activity would have happened, and the code will go in vain.

3. 8-16 GB RAM: The algorithm will require extensive computations so 8 GB RAM will be the minimum requirement.

4. 8 GB HDD storage: Because videos will be given as input to the algorithm at least 8GB free space will be required so that videos can be managed on the disk.
5. A good internet connection will be required because videos will have to be uploaded constantly to google cloud where the algorithm will be running.

Technology Platform:

Desktop Based CLI TOOL:

Applications Front-end would be developed for All Desktop OS, providing the users with the interface to get registered on the feed and able to run the feed in three different ways.

Programming Environment

- VS Code
- Anaconda

Libraries

- Keras
- OpenCV
- PyQT
- Pandas
- Numpy
- Matplotlib
- Pickle

3.5 Design and Implementation Constraints

CO-1: It requires High computing power

CO-2: System will not be able to do other tasks

CO-3: Interface is kept simple to support cross platform

CO-5: Use of English language as the only means of communication in the system.

CO-6: System must adhere to MCS Security Regulations.

3.6 User Documentation

UD-1: Final release will be accompanied with a user guide to inform users how to use VAS.

3.7 Assumptions and Dependencies

UD-1: Final release will be accompanied with an online user guide to inform users how to use Mechanical Transport Automation System (MAAS). User documentation that would be delivered along with the final product.

AS-1 Processing power of this system on a GPU is 7 FPS

AS-2: The admin will be shown the interface so that he doesn’t have to be attentive all the time

AS-3: Labeled class will be shown in abounding box

AS-4: The climbing model will work in the background in order to detect any climber.

AS-5: Users must know how to operate CLI

D-1: System must have installed the required modules

D-2: System will have proper cooling system as this processing might cause system to burn out if conditions are not met

3.8 External Interface Requirements

3.8.1 User Interfaces

VAS consists of an interface used by the VAS operator. The learning curve for this will be gradual, to make the operator feel at ease while learning about it.

3.8.2 Hardware Interfaces

Video Analysis System will have the following Requirements.

Python Support

VAS can be executed on all three OS (Windows, Linux and Mac OSX). It requires python3 support on the system and along with that it requires the python interfacing modules to be installed before execution.

Hardware Interfaces

- System shall have keyboard input.
- System shall have mouse input.
- System shall have a monitor.
- System shall have a working internet connection and the hardware requirements that come with it (Network card, Modem, Ethernet Port, etc.)
- System shall have RAM greater than or equal to 16 GB for real time results.

Programming Interface

Programming interfaces for project are:

- Visual Studio Code
- Anaconda Jupyter Notebook Environment

3.8.3 Software Interfaces

Primary Operating System supported by VAS Interface will be Windows 7, Mac OSX and Linux

3.9 Communications Interfaces

System will not be connected to web API from any interface because we don’t have to get or send data to any interface. The whole algorithm will be present on google cloud. The siren will be connected to our python driver program which will be programmed automatically to ring whenever the program detects something anomalous. There will be an option present whenever the siren wails that whether to stop it or keep wailing to prevent from false positives.

System Features

3.10 Schedule creation

3.10.1 Description and Priority

The system will enable the Operator to carryout remote surveillance of the boundary wall of MCS 24/7. In case of something fishy it would notify the operator immediately. The priority of this system feature is high.
95 % confidence level.

3.10.2 Stimulus/Response

Input. A video frame is given to our model.

Output. Model will tell siren to veil or stay quite

3.10.3 Functional Requirements

1. Detecting large crowd outside of MCS wall.
2. Detecting weapons in that area outside of MCS.
3. Detecting bags left on the road near MCS.

4. Ringing the siren whenever above-mentioned things are detected.

3.11 Login / access rights

Detecting weapons in video

Description and Priority

The system will detect weapons i.e. pistol, revolver, rifle from the video feed. And this feature will have quite a high priority so even if the model has little confidence in detection, still the siren will go off.
Hence employing an operator 24/7 to detect false positives.

Stimulus/Response Sequences

Input:

A weapon appears in the video stream being monitored by our algorithm.

Output:

The siren goes off

Functional Requirements

Algorithm will detect weapons in video.

Detecting bags

Description and Priority

If a bag is lying on the ground for more than five minutes, then algorithm will take it as a suspicious activity and siren will wail.

Stimulus/Response Sequences

Input:

A bag is lying on the ground for more than five minutes in the video feed.

Output:

The siren goes off.

Functional Requirements

System will notify whenever a bag stays on the ground for more than 5 minutes.

Description and Priority

If there is a large crowd outside the MCS wall, then the model ring the siren. This feature will have a slightly less priority because there is a school nearby MCS so, if it is a high priority feature then there will be lot of false positives.

Stimulus/Response Sequences

Input:

A large crowd comes near the MCS wall.

Output:

Algorithm wails the siren.

Functional Requirements

Model will notify if a large crowd comes near MCS.

Other Non-functional Requirements

5.1 Performance Requirements

It'll take the video, turn it into frames and then detect the objects.

5.2 Safety Requirements

SF-4: Identity of Staff using the system should be disclosed to the security officer.

Software Quality Attributes

Quality attributes of VAS are described below. By following these attributes, the quality of VAS will be improved.

Runtime System Qualities

At runtime VAS must provide its user with functionalities so that they can publish and search for the desired services. Some of the runtime qualities that should be considered in the development of VAS are described here.

Availability

VAS should be available 24/7. If system is down (in case of power outage or a hardware problem) the operating system will take about 5 minutes to start the

VAS again. It'll work on Google cloud so there's less possibility that system will be down. We've to pay monthly to avail the services.

Usability

Usability is an important criterion in the development of VAS. The system should present all functionalities in such a way that nothing is missed by the user. The graphical user interface of app is to be designed with usability as the priority. The app will be presented and organized in a manner that is both visually appealing and easy for the user to navigate.

5.3 Non-Runtime System Qualities

These are qualities of VAS which are required to make this software useful for further enhancements. It will also be helpful in future development as well as extending system to different environments.

Modifiability

VAS must support modifiability, so any further improvements or features are easy to incorporate.

Portability

VAS should be able to run in different computer environments. VAS will be on google cloud and it could be run in any system if it has working internet connection and access to video feeds.

Testability

Different quality tests should be performed so that VAS is free from faults and perform according to requirements.

Chapter 4

System Design Specifications

4.1 Work Breakdown Structure:

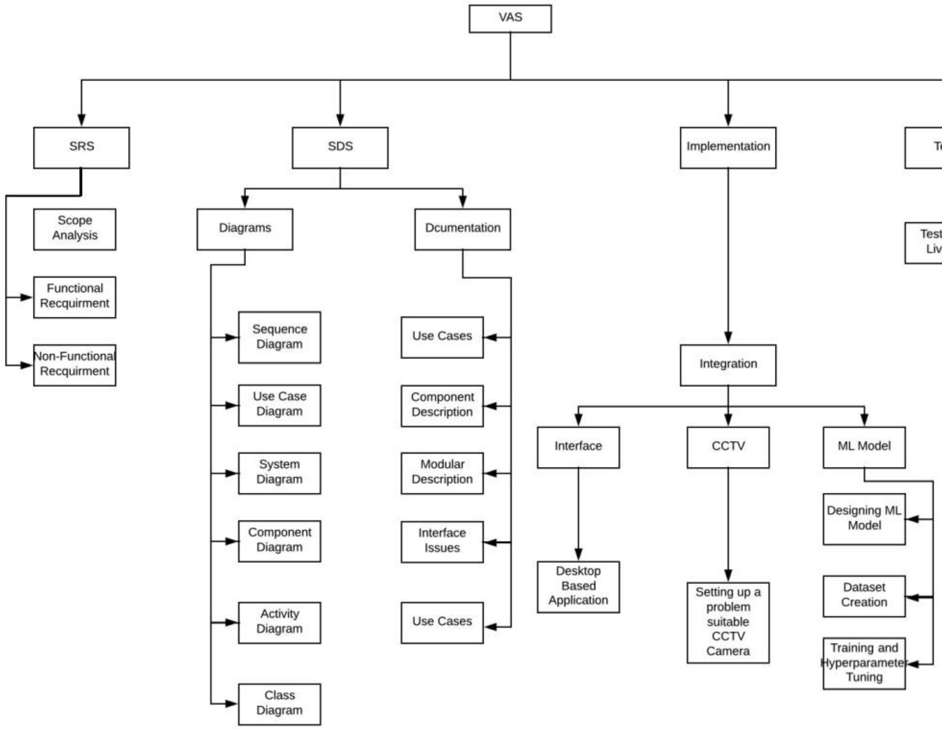


Fig 1.6

4.2 System architecture description

This part will cover in detail the description of our architecture and the software engineering principals we will be following in this project. Our System is focused on four main entities which are described in depth in the subparts below.

2.1 Overview of modules / components

This part of the document will be covering the major entities of VAS. VAS can be divide into four main modules.

2.1.1 Interface

This module will act as the driver module of the VAS. This module will be taking input feed from the CCTV and then will be responsible for feeding the input to the ML Model. Furthermore, this model will also be responsible for maintaining the output of the model and generating the alarm on the required occasions.

2.1.2 CCTV

This module is responsible for providing the live video feed to our system. Given that the lighting is according to the model requirements. The frames from the CCTV will be preprocessed and then will be given input to the module ML Model as input.

2.1.3 ML Model

This module will act as the backbone for the VAS system. This module will take live input frames from the interface module and then process the frames through machine learning algorithm in order to classify the state of the feed. Then the output of the algorithm will be sent back to the interface.

2.1.4 Alarm Module

Alarm module will get the commands from the interface module. In order to be triggered when an unwanted frame has been classified by the VAS.

2.2 Structure and relationships

This part of the document will be covering all the technical details of VAS. This module will show the coordination between multiple modules and entities.

2.2.1 System Block Diagram

System block diagram is showing the relation and the possible interaction between all modules of the system.

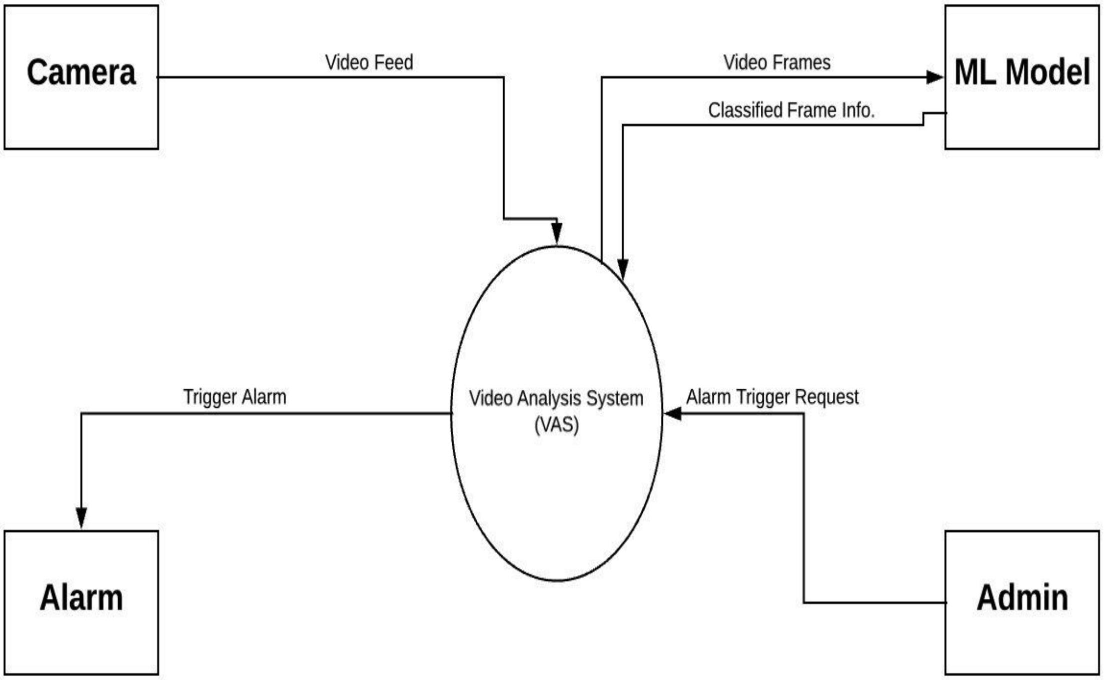


Fig 2.2.1

2.2.2 Component Diagram

Four main components of our model are:

1. Machine Learning Model
2. Alarm
3. CCTV Camera
4. Graphical User Interface

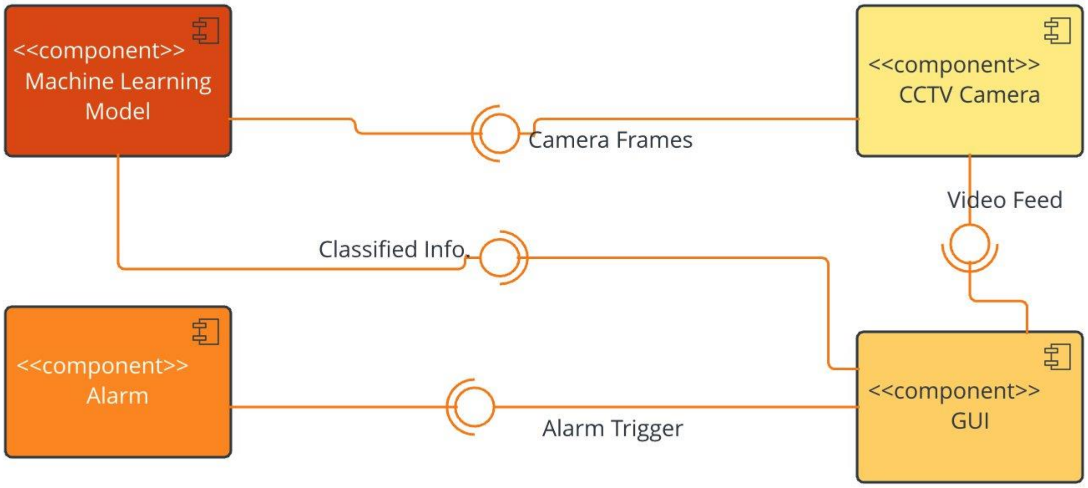


Fig 2.2.2

2.2.3 User Case Diagram

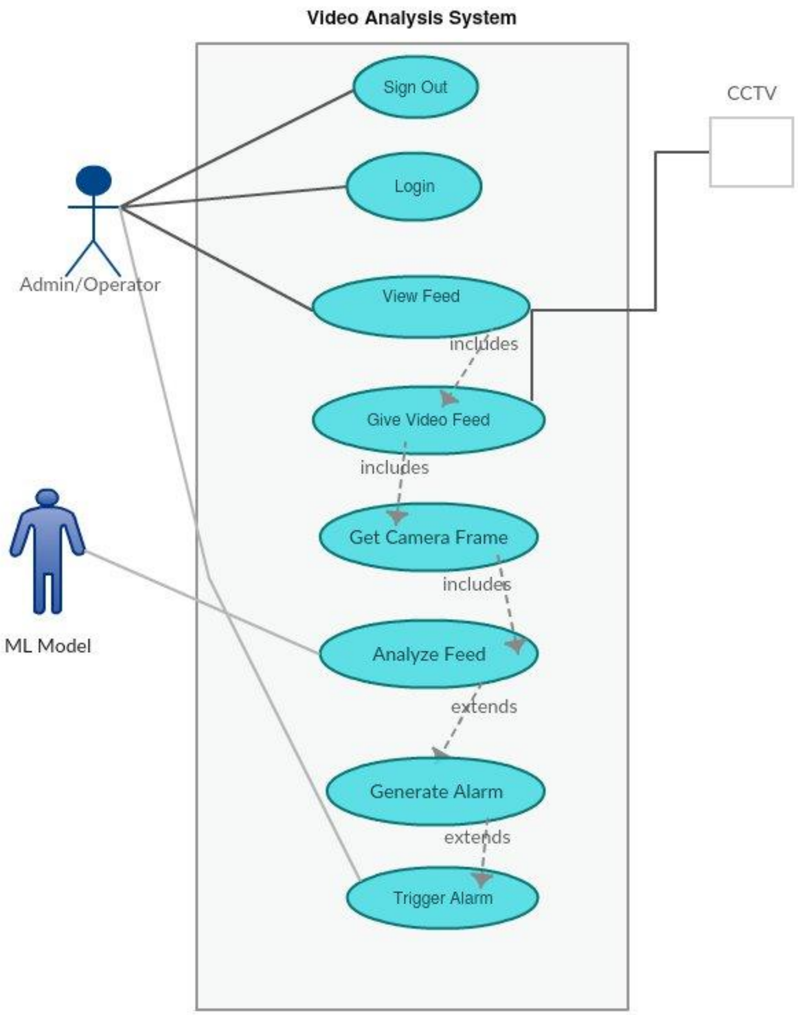


Fig 2.2.3

Actors

5. Admin/Operator **Primary**
6. CCTV **Primary**
7. ML Model **Secondary**

2.2.4 Use Case Description

All use cases have been defined along with the alternative scenarios if needed.

4.1

4.1.1 Log In

Identifier	UC-1	
Purpose	Log In	
Priority	High	
Actors	**Admin/Operator**	
Pre-conditions	**None**	
Post-conditions	**Admin/Operator is logged In and redirected to main page**	
Typical Course of Action		
S#	**Actor Action**	**System Response**
1	Enter User ID and password	
2	Press Login button	System will verify the email and password and login the user
Alternate Course of Action (Invalid username/password)		
S#	**Actor Action**	**System Response**
1		System will show an error message

4.2 2.2.4.2 Sign Out

Identifier	UC-2
Purpose	Sign Out
Priority	High
Actors	**Admin/Operator**
Pre-conditions	**User must be logged into the system**

<table>
<tr><td>**Post-conditions**</td><td colspan="2">**Admin/Operator is logged off and redirected to authentication page.**</td></tr>
<tr><td colspan="3">Typical Course of Action</td></tr>
<tr><td>**S#**</td><td>**Actor Action**</td><td>**System Response**</td></tr>
<tr><td>**1**</td><td>Press Logout button</td><td></td></tr>
<tr><td>**2**</td><td>Press Login button</td><td>System will Logout the user</td></tr>
</table>

4.3 2.2.4.3 Trigger Alarm

<table>
<tr><td>**Identifier**</td><td colspan="2">UC-3</td></tr>
<tr><td>**Purpose**</td><td colspan="2">Trigger Alarm</td></tr>
<tr><td>**Priority**</td><td colspan="2">High</td></tr>
<tr><td>**Actors**</td><td colspan="2">**Admin/Operator**</td></tr>
<tr><td>**Pre-conditions**</td><td colspan="2">**Actor must be logged in**</td></tr>
<tr><td>**Post-conditions**</td><td colspan="2">**Alarm module is being notified to be triggered**</td></tr>
<tr><td colspan="3">Typical Course of Action</td></tr>
<tr><td>**S#**</td><td>**Actor Action**</td><td>**System Response**</td></tr>
<tr><td>**1**</td><td>Press the Trigger Alarm Button</td><td>System will trigger the Alarm</td></tr>
</table>

4.4 2.2.4.4 Analyze Frame

Identifier	UC-4
Purpose	Analyze Frame
Priority	High
Actors	**Admin/Operator**
Pre-conditions	**None**

Post-conditions	**Admin/Operator is logged In and redirected to main page**	
Typical Course of Action		
S#	**Actor Action**	**System Response**
1	Actor will pass the vide frames to the Machine learning algorithm	
2		Algorithm will classify the state of the frame as normal
3	Actor pass the frame state to the interface	
Alternate Course of Action (System Classify frame as Armed)		
S#1	**Actor Action**	**System Response**
2		Algorithm will classify the state of the frame as Armed
3	Actor pass on the alarm notification to interface module	
Alternate Course of Action (System Classify frame as crowd)		
S#2	**Actor Action**	**System Response**
2		Algorithm will classify the state of the frame as Crowded
3	Actor pass on the alarm notification to interface module	

4.5 2.2.4.5 View Feed

Identifier	UC-5
Purpose	View Feed
Priority	High
Actors	**Admin/Operator**
Pre-conditions	**Actor must be logged in**
Post-conditions	**Feed is shown to the actor**

Typical Course of Action		
S#	**Actor Action**	**System Response**
1		System will show the feed if actor is logged in

4.6 2.2.4.6 Get Camera Frame

Identifier	UC-6	
Purpose	Get Camera Frame	
Priority	High	
Actors	**CCTV, ML Model**	
Pre-conditions	**CCTV must be working, Admin Must be logged in**	
Post-conditions	**Frames passed to ML model**	
Typical Course of Action		
S#	**Actor Action**	**System Response**
1		System will fetch the frames from the CCTV and pass it to interface
2	Interface will then pass the frames to ML model	

4.7 2.2.4.7 Get Camera Frame

Identifier	UC-6	
Purpose	Get Camera Frame	
Priority	High	
Actors	**CCTV, ML Model**	
Pre-conditions	**CCTV must be working, Admin Must be logged in**	
Post-conditions	**Frames passed to ML model**	
Typical Course of Action		
S#	**Actor Action**	**System Response**

1		System will fetch the frames from the CCTV and pass it to interface
2	Interface will then pass the frames to ML model	

2.2.5 Activity Diagram

Activity Diagram for 3 different tasks is being drawn, only including the activities that has the interaction with the interface.

2.2.5.1 Sign In

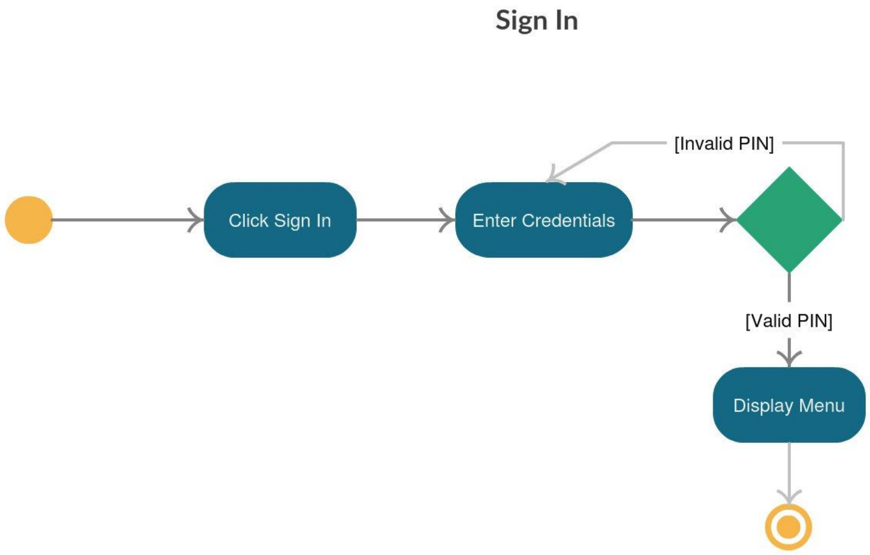


fig 2.2.5.1

2.2.5.2 Display Feed

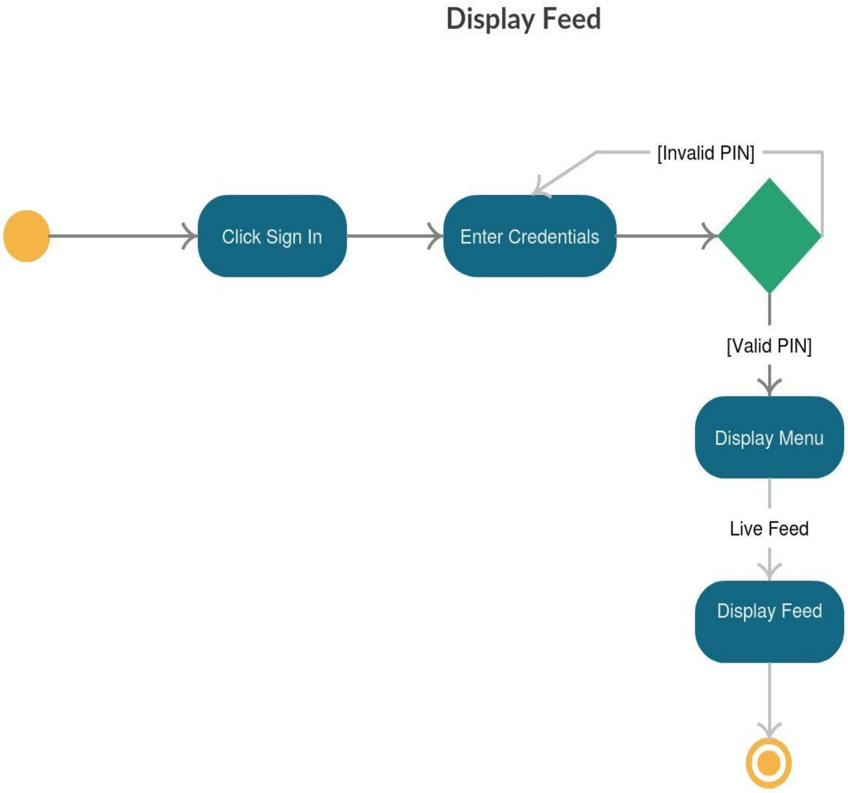


fig 2.2.5.2

2.2.5.3 Trigger Alarm

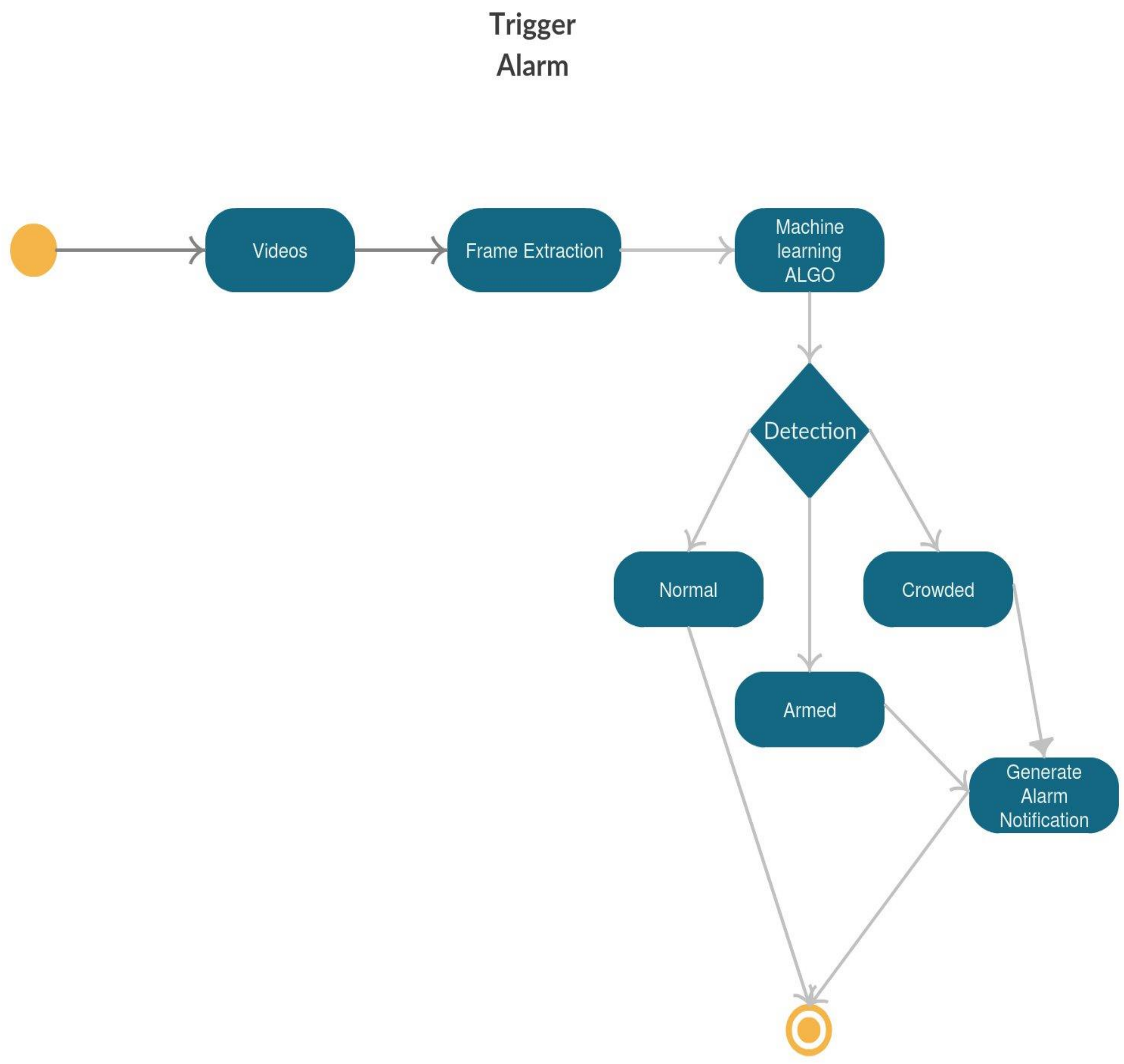


fig 2.2.5.3

2.2.6 State Diagram

State diagram of three different tasks are drawn. Only including the interaction that are most relevant and direct to object interaction

2.2.5.1 Sign In

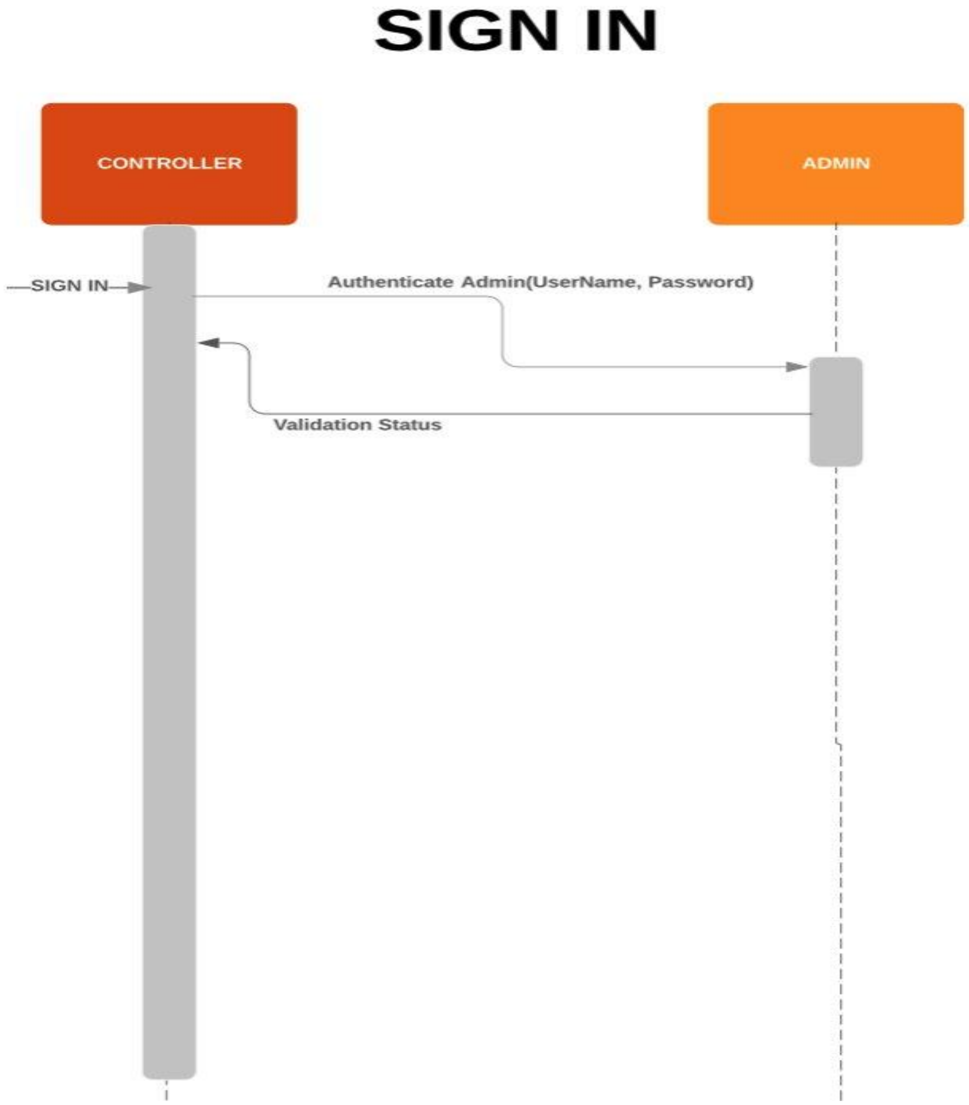


Fig 2.2.6.1

2.2.6.2 View Feed

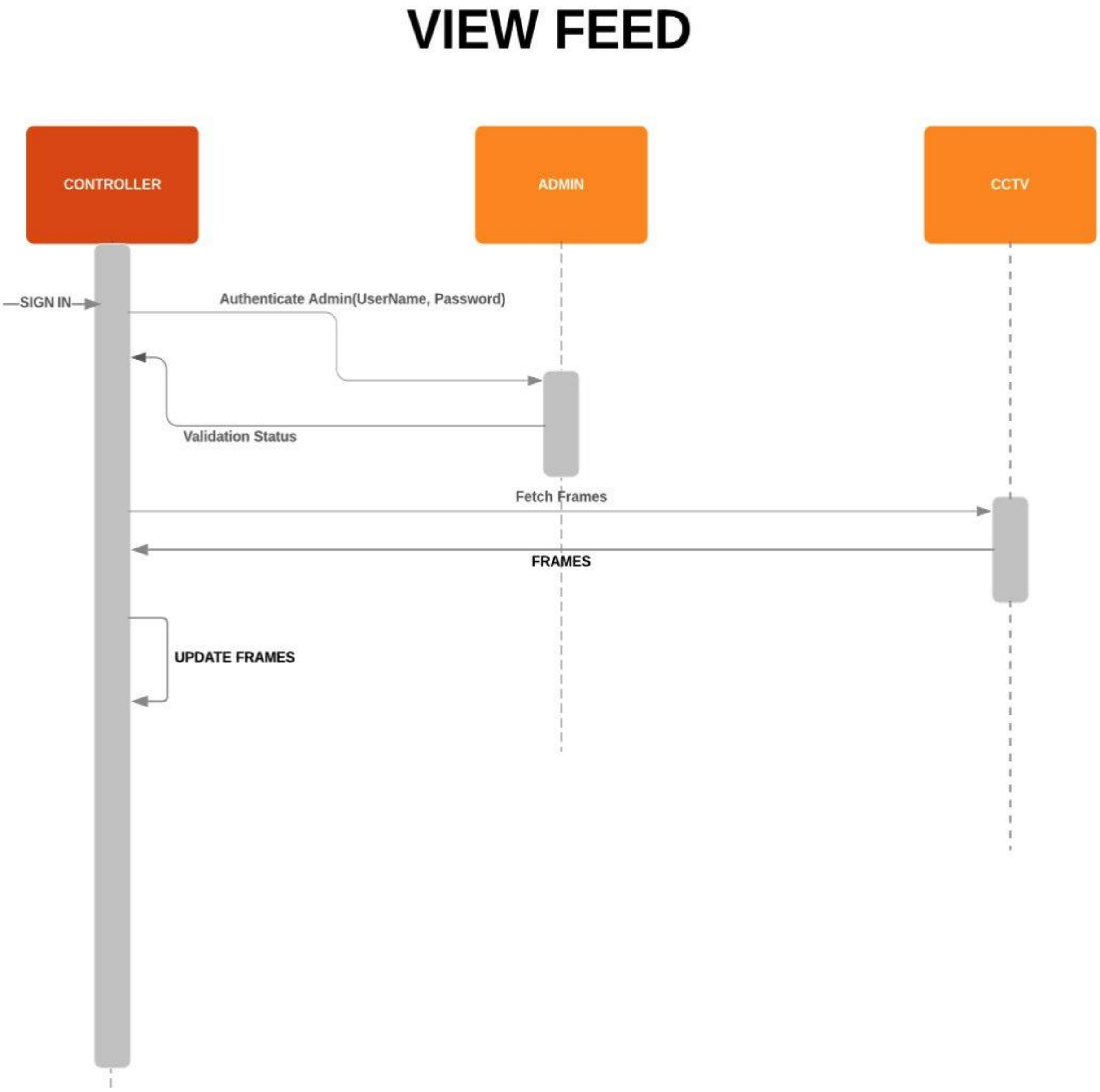


Fig 2.2.6.2

2.2.6.3 Trigger Alarm

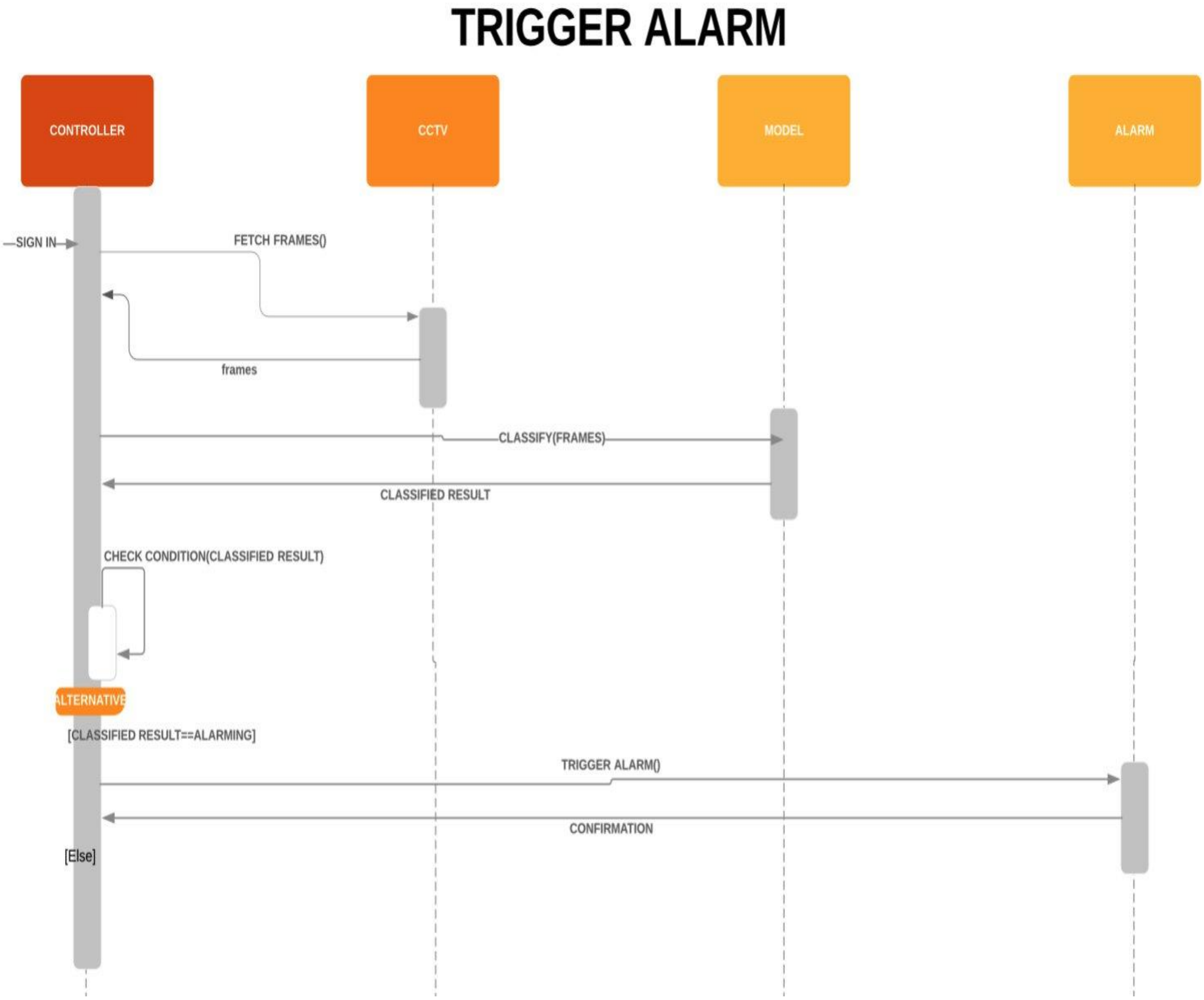


Fig 2.2.6.3

2.2.7 Class Diagram

Class diagram depicts the important classes involved in our Video Analysis System. All Classes along with their attributes and major functionality.

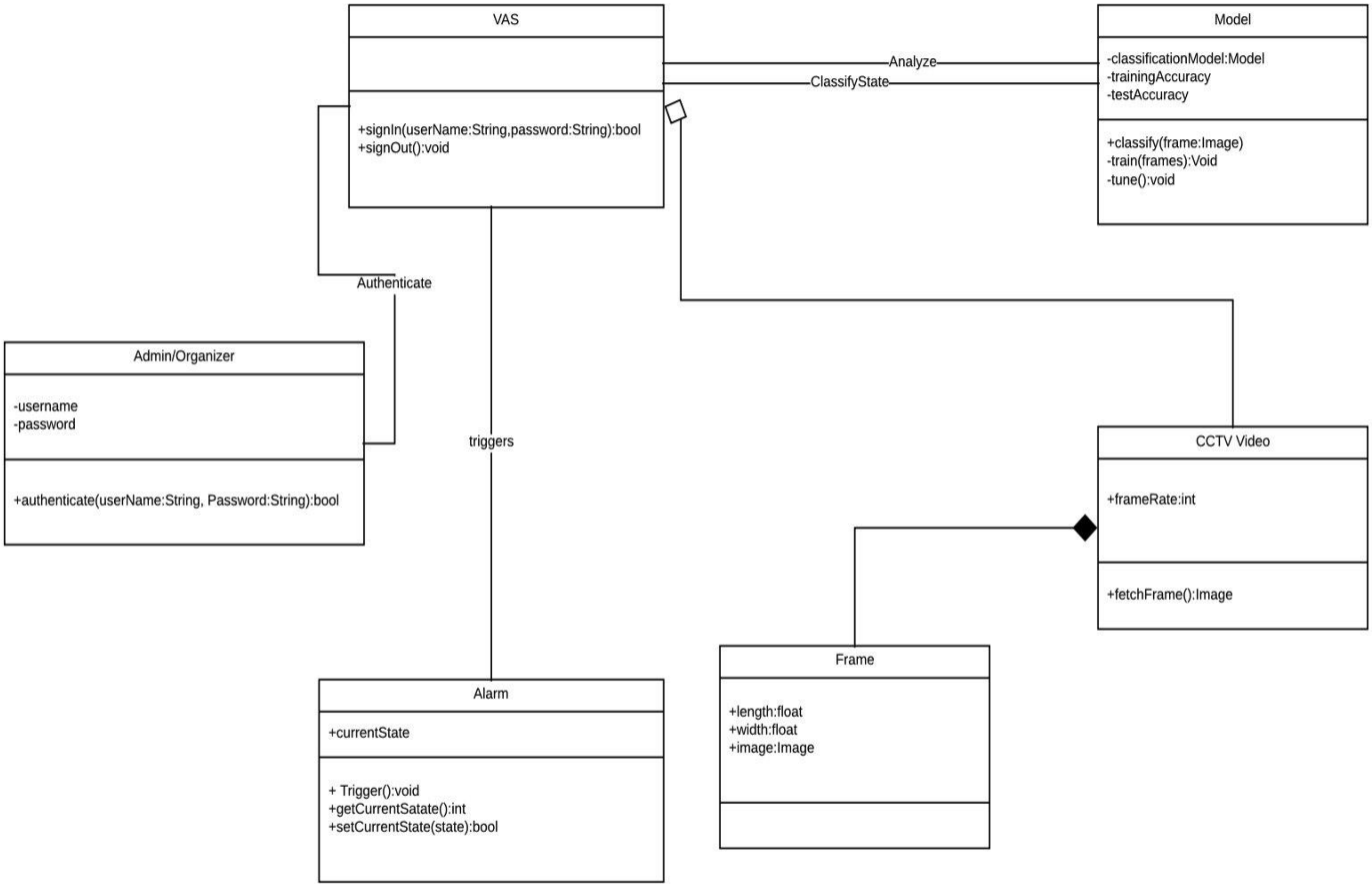


Fig 2.2.7.1

Class Name	Attribute	Description
VAS	Nil	VAS is a controller class which controls the interaction of multiple classes
Admin/ Operator	UserName	UserName assigned to the Admin of VAS
Admin/Operator	Password	Password recquired for log into the VAS
Alarm	CurrentState	In order to trigger the alarm, we keep its state
CCTV Video	Frame Rate	No. of Frames CCTV is generating in a single second (FPS)
Frame	length	Length of the image frame

Frame	width	Width of the image frame
Frame	Image	Image is a frame which further gets combined to form a video
Model	Classification Model	An ML model coded in KERAS using Python 3.0
Model	Training Accuracy	Train Accuracy is the score which best describe our model design
Model	Test Accuracy	Test Accuracy is the reliability score of the Model

2.3 User interface issues

This part of the document will cover the interfacing and how the admin will access the system in order to perform the basic operations like viewing the feed or triggering the alarm. In order to retain the interfacing rules, we have minimized the load on the screen and make the user interface so descriptive that a novice user can easily redirect to our system. Interfacing of VAS involves three screens.

8. Login Screen
9. Invalid Password dialog screen
10. Main Menu

3. Detailed description of components

We have divided our whole project into 4 modules based on the components

Identification	Interface
Type	Component
Purpose	Enable the interaction of user and system
Function	Authenticate the user and show system state
Subordinates	This component has no sub components
Dependencies	In order to receive correct output, CCTV must be working
Interfaces	GUI
Resources	Output Unit and CCTV feed
Processing	Receive user input and authenticate user, Process the output from the model and manage the state of the alarm
Data	User input and CCTV feed

Identification	ML Model
Type	Module
Purpose	Classification of frames
Function	Analyze the current frame, Train, Test
Subordinates	None
Dependencies	Frames must be in good resolution with required lighting condition
Interfaces	None required
Resources	CCTV and GPU
Processing	Frame Analysis
Data	Frame

Identification	ML Model
Type	Module
Purpose	Classification of frames
Function	Analyze the current frame, Train, Test
Subordinates	None
Dependencies	Frames must be in good resolution with required lighting condition
Interfaces	None required
Resources	CCTV and GPU
Processing	Frame Analysis
Data	Frame

Identification	CCTVV
Type	Module
Purpose	Provide Video Feed
Function	Give Frames
Subordinates	GUI
Dependencies	Hardware must be working
Interfaces	GUI
Resources	CCTV
Processing	Frame rate must be set according to the requirement of the system
Data	Frame Input

4. Reuse and relationships to other products

There is no such software that provides you live video analysis of the situation witch the generalized classes. This idea is unique on certain grounds. However, we will be using some default libraries in order to implement the machine learning solution.

4.1 Part of reuse in our design

Any component of this application will not be used from the previous developed models but it will be developed for the scope of this project.

4.2 Reuse in your product implementation

While implementing the machine learning solution we will be using the stock or best present model for the problem domain. So far, we are planning to implement our solution in **KERAS** a Python 3.1 library which allows to implement models in abstract/high level logics. Furthermore, the design of the model may or may not resemble to the existing implementation depending on the accuracy of VAS.

4.3 Not reusing material that is available

We are trying to make the best use of available resources in order to support the reusability principal. Unfortunately, there is no help for us other than the stock implementation of a machine learning model.

5.0 Design decisions and tradeoffs

VAS is only designed for Video analysis, so we have designed the interface in such a professional manner so that the user can understand the screens and perform the task without any kind of complexity. We have followed **Shneiderman's** "Eight **Golden Rules of Interface Design**" which helps the user to minimize the memory load while using the interface furthermore, we have use consistent color designing across the screens in order to support the consistency of the user. Furthermore, we will be using MVC pattern which restricts the direct access of the model to the user.

MVC detailed working is being described in the diagram given below.

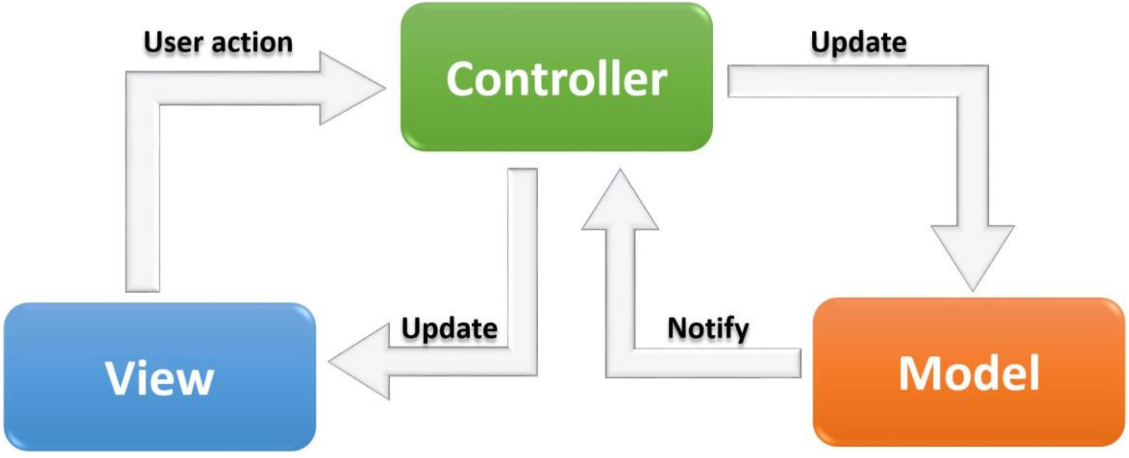


Fig 5.0

6.0 Pseudo code for components

6.1 Sign In

```
//Wait till user press the Sign In button

OnclickListnerLogInButton()
{
   bool Admin.authenticate(username,password)
   if(true)
   {

      VAS.redirect();
   }
   else
   {
      showErrorScreen();
   }
}
```

6.2 VAS controller

```
Bool Login();
if(Sucess)
{
   redirect to the main screen with live feed;
   fetchFrames();
   CurrentStatus=Model.PassFrames(Frames);
   if(CurrentStatus!=Normal)
   {
      TriggerAlarmNotification();
   }
}
else
```

```
{
   redirect to the same page
   //show error dialogue
}
```

6.3 Model

```
Classify(frames);
if(classification!=normal)
{
   LastCall=RenderThreeMoreTrails();
   if(LastCAll!=Normal)
   {
      VAS.CallForNotification();
   }
```

Chapter 5

System Implementation

Technology Used

Programming Language Used

This system is built on Python programming language. Modules including KERAS, Tenser flow, PYQT and basic modules like pandas and NUMPY was used.

Development Tools

Using ANACONDA's Environment Jupyter Notebook for the developing and training of Deep Learning Algorithms along with that Microsoft Visual Studio Code was used for Interfacing.

Operating System

This System was tested on all three benchmark OS, AS we have avoided any sort of cross platform clash. So, this system can be running on:

1. Microsoft Windows
2. MAC OSX
3. Linux

MAC OSX Interface

Following are the sketches of a possible UI implementation for the Start up screen/ Main Menu:

Login Screen

The very first interface screen of this system is the login screen which will take the username and password for the admin.

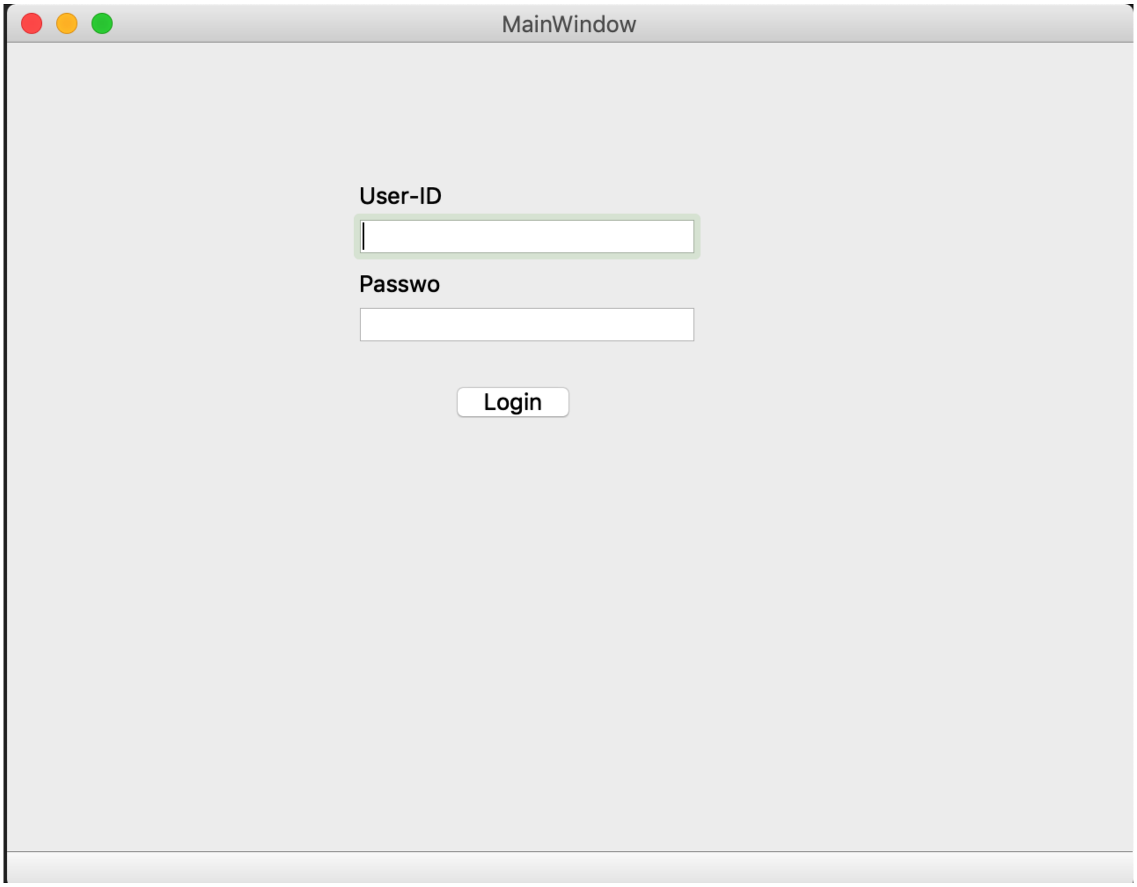

MainWindow
User-ID
Passwo
Login

Figure 1 Login Screen

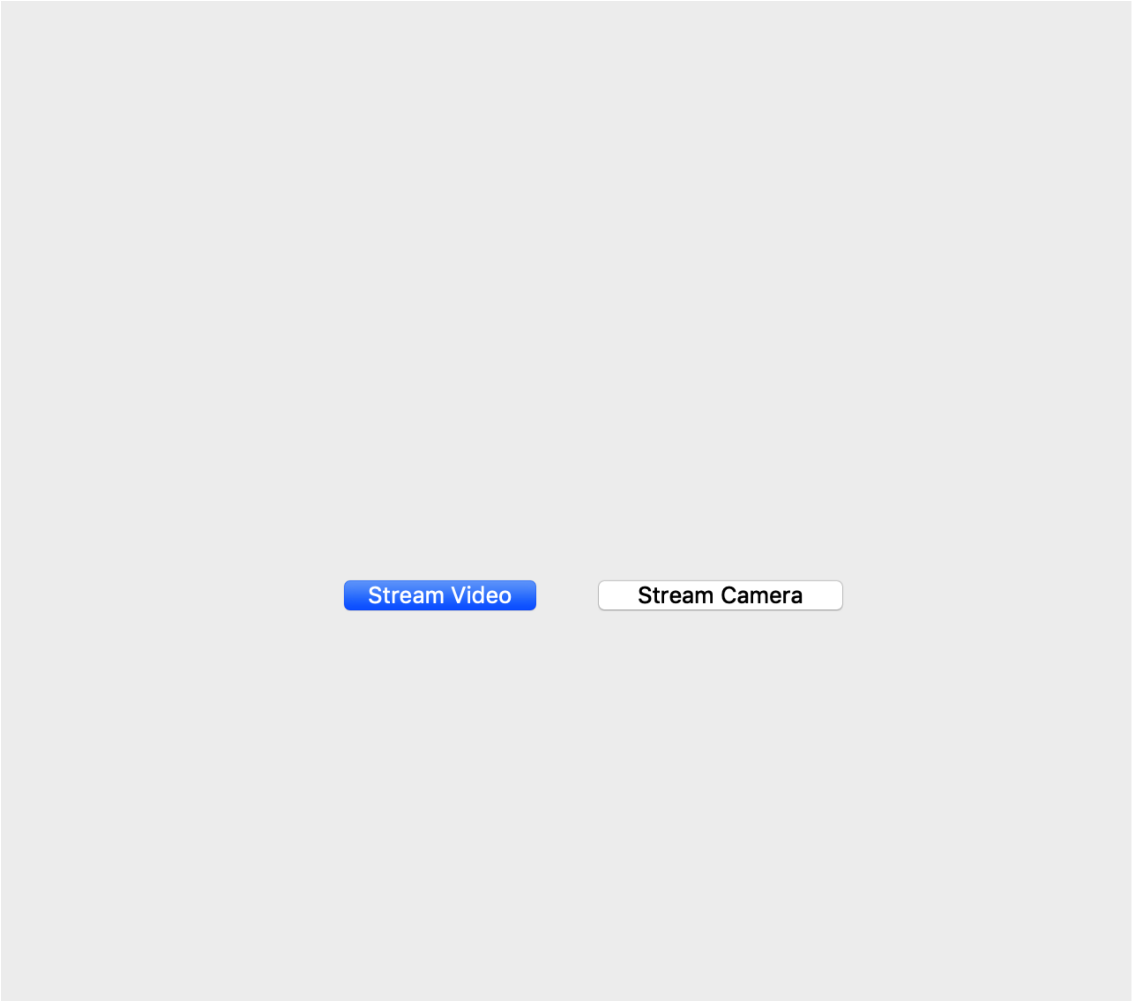


Figure 13 Selection Screen

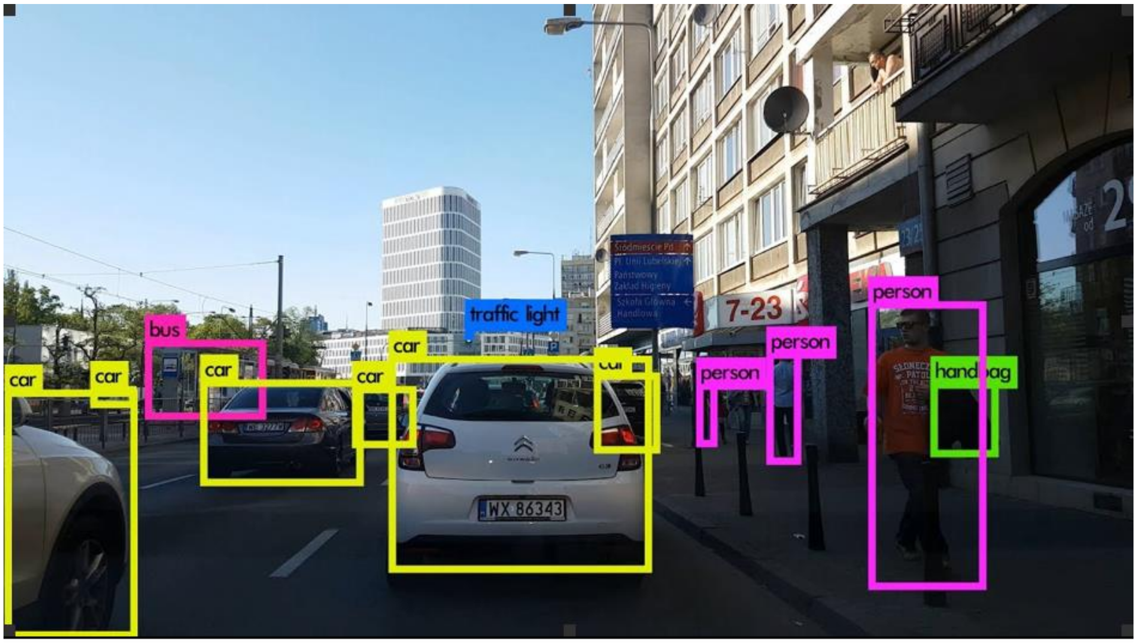


Figure 14 Model Interface

Chapter 6

Testing

6.1 Overview

Testing of software projects involve different levels of testing to make sure that the software which is being developed is error and fault free. The different levels at which testing was done is discussed here.

Item Pass/Fail Criteria

Details of the test cases are specified in the section Test Deliverables. Following the principles outlined below, a test item would be judged as pass or fail.

1. Preconditions are met
2. Inputs are carried out as specified
3. The result works as what specified in output => Pass
4. The system doesn’t work or not the same as output specification => Fail

Suspension Criteria and Resumption Requirements

Identification of any bug in any phase of testing, the required developer be reported regarding the bug. This makes the developer responsible to resolve the error on priority. In case of a larger number of bugs they will be prioritized and resolved in order of their importance.

Test Deliverables

Following are the test cases:

Test case name	**Camera File Upload**
Test Case Number	1
Description	The system is able to read the CCTV file.
Testing Technique used	Unit Testing

Preconditions	File must exist
Input	File must be in .avi/.mp4 format
Steps	1. Run python script 2. Specify the location of the file
Expected output	File being read by the system
Alternative Path	1. **Cause:** 1. File Doesn't exist **Corresponding Output:** Error Message Displayed
Actual output	Confirmed

Test case name	**Model Classification**
Test Case Number	2
Description	Model will classify the frame as any given class
Testing Technique used	Unit Testing
Preconditions	Model must be trained.
Input	Image frame
Steps	1. Run python script 2. Specify the location of the file 3. Give input to the model
Expected output	Audio file uploaded.
Alternative Path	4. **Cause:** 5. Frame is not an image **Corresponding Output:** Error Message Displayed
Actual output	Confirmed

Test case name	**Sounding an Alarm**
Test Case Number	3
Description	This will allow the admin module to sound the alarm if anomaly is detected
Testing Technique used	Unit testing
Preconditions	1. Permission to access hardware granted

Input	Audio file
Steps	1. Generate the command to sound alarm
Expected output	Urdu text will be displayed.
Alternative path	**Cause:** Permission Denied. **Corresponding Output:** Error Message Displayed.
Actual output	Alarm sound

Test case name	**Clear**
Test Case Number	5
Description	Testing System as a whole
Testing Technique used	Integration Testing
Preconditions	All requirements mentioned in above case must be met
Input	N/A.
Steps	1. Run the driver class in anaconda jupyter notebook
Expected output	System will start classifying the current state.
Alternative Path	**Cause:** 1. Any pre-conditioned missed. **Corresponding output:** message displayed accordingly
Actual output	Confirmed

Environmental Needs

6.1 Hardware

1. Computer

2. 8GB RAM.

3. CCTV Camera

6.2 Software

1. Anaconda
2. Jupyter Notebook

6.3 Libraries

1. Keras
2. Matplotlib
3. Cv2
4. TQDM

Responsibilities, Staffing and Training Needs

6.4 Responsibilities

Testing in such projects is technical as one missing parameter to the model can cause you a greater damage in terms of accuracies. So, in order to test such project we need to make sure that we learn the right testing techniques beforehand and perform the testing in right manner

6.5 Staffing and Training Needs

Basic knowledge of testing strategies and techniques is needed for the testing of project. Techniques such as Black Box testing, integration testing should be known to developers.

All the developers will be testing each other's work and will be actively participating in the development and testing of the project simultaneously.

Risk and Contingencies

Extensive efforts have been made to make the model work perfectly. Self-labeling has been done for almost 100000 images in order to make the classes distinguishable for the machine learning model.

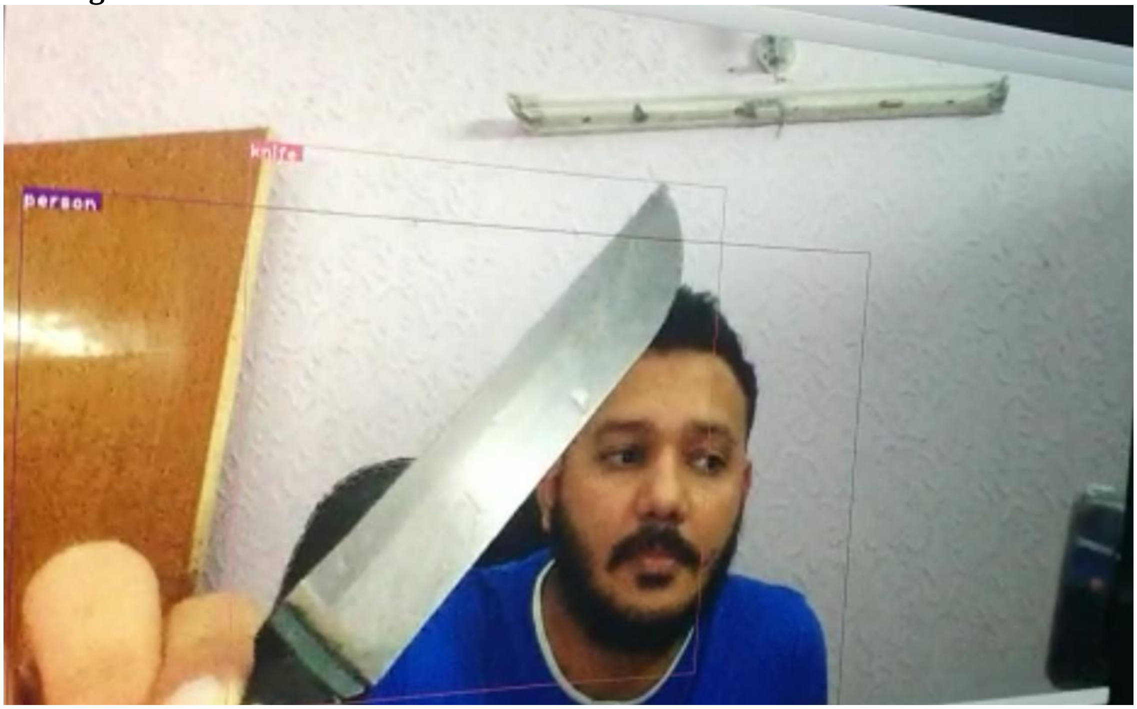
6.6 Schedule Risk

As we are facing problems regarding accuracies This project might not be able to achieve its objectives or we will have to make another dataset in order to solve the problem.

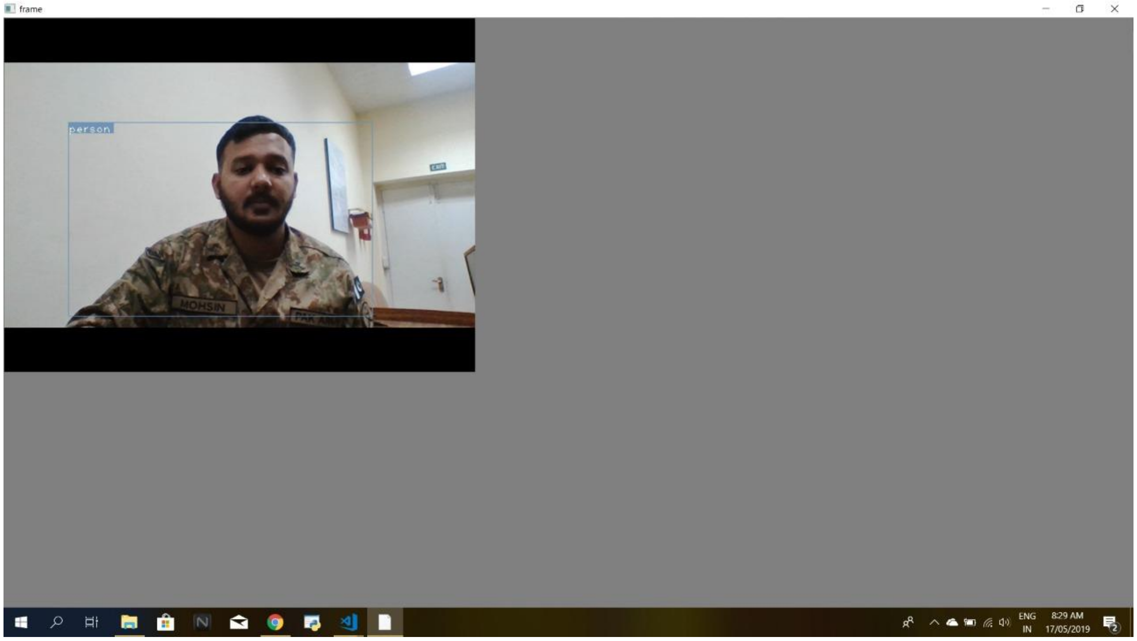
6.7 Model Testing

In order to test the working of model we used some classes labelled, which includes person, book, tool and handbag

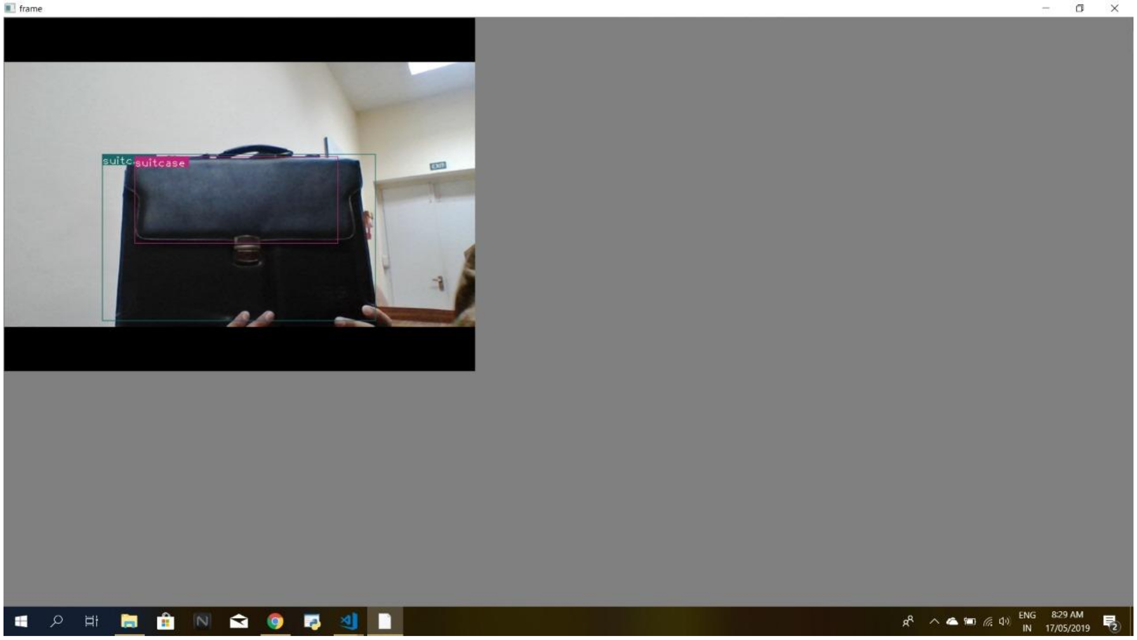
Testing knife



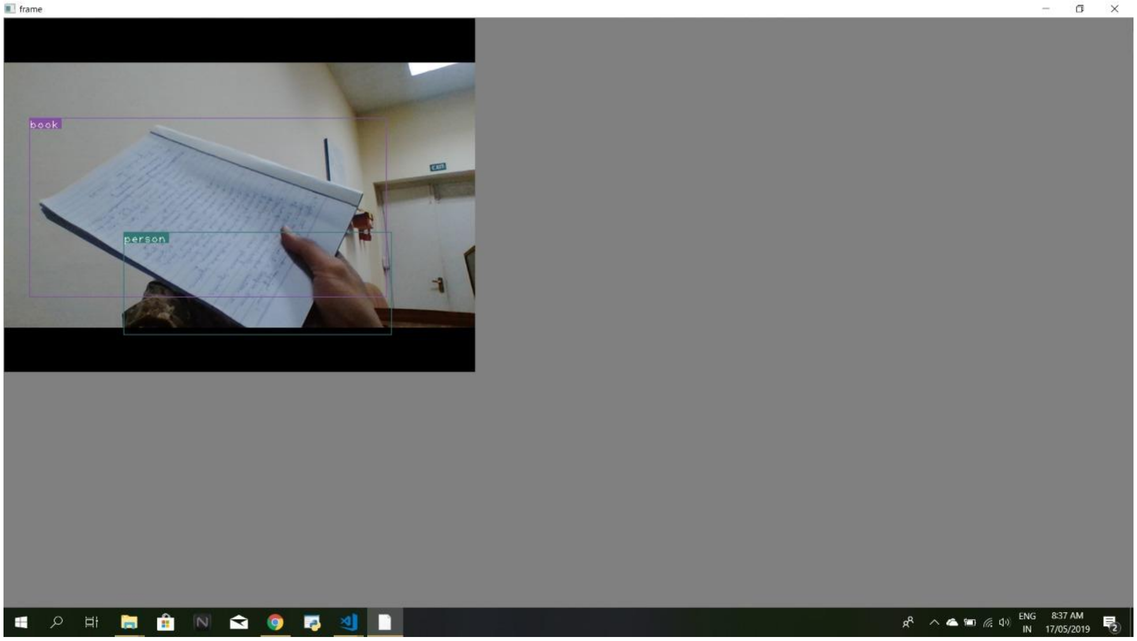
Testing Person



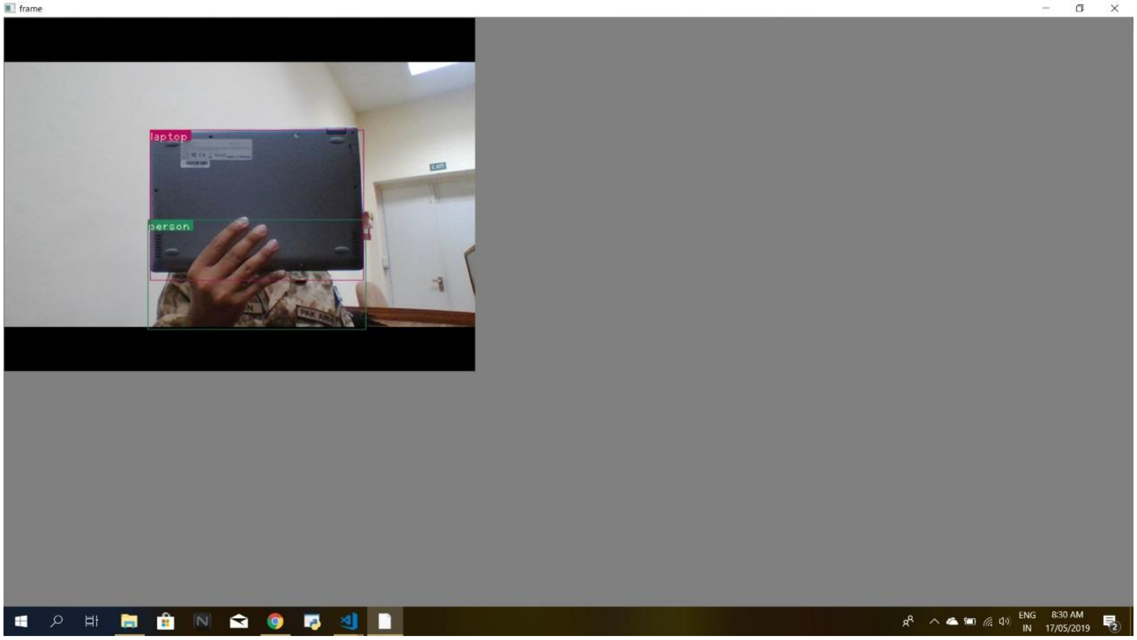
Testing Suitcase



Testing Book



Testing Laptop



Chapter 7

Conclusion and Future Work

Conclusion

In result to our effort we are able to develop a fully functional real-time application which detects the classes like bags, mobile phone, person, Vehicle and other classes. This project has helped us to learn many deep learning frameworks along with that the concepts like transfer learning and distribution of dataset. This research and development project has opened a new avenue of research for us. Which includes the challenge of segmentation of weapons in real time. This study in conclusion provided us development skills along with a fully functional system that is able to analyze the video feed in real time.

Future work

Despite limited time and resources, we did our best to make a system best for automation of video feed analysis. But we were not able to classify the weapons in true manner.

Glossary

API	Application Programming Interface
App	Application
AS	Assumption
Black box Testing	Testing emphasizes on the external behavior of the software entity
CO	Constraints
App	Application
OS	Operating System
REQ	Requirement
SQL	Structured Query Language
SE	Security Requirements
SR	Safety Requirements
SRS	Software Requirements Specification
UD	User Documentation

UML	Unified Modeling Language
White Box Testing	Testing emphasizes on the internal behavior of the software entity
VAS	Video Analysis System
CNN	Convolutional Neural Network
ANN	Artificial Neural Network
CV	Computer Vision
Transfer Learning	Training Model on 2 datasets
Segmentation	Detecting the location of the object in frame
FPS	Frame per Second

Table 1 Glossary

Bibliography

[1]"Coursera | Online Courses & Credentials by Top Educators. Join for Free", *Coursera*, 2019. [Online]. Available: https://www.coursera.org/. [Accessed: 14- Jun- 2019].

[2]"W3Schools Online Web Tutorials", *W3schools.com*, 2019. [Online]. Available: https://www.w3schools.com. [Accessed: 14- Jun- 2019].

[3]"Basic of RFID System - Types and Working Example of RFID Application", *ElProCus - Electronic Projects for Engineering Students*, 2019. [Online]. Available: https://www.elprocus.com/rfid-basic-introduction-simple-application/. [Accessed: 14- Jun- 2019].

[4]"MFRC522 RFID Reader with Arduino Tutorial | Random Nerd Tutorials", *Random Nerd Tutorials*, 2019. [Online]. Available: https://randomnerdtutorials.com/security-access-using-mfrc522-rfid-reader-with-arduino/. [Accessed: 14- Jun- 2019].

[5]"Medium – a place to read and write big ideas and important stories", *Medium*, 2019. [Online]. Available: https://www.medium.com. [Accessed: 14- Jun- 2019].

[6]"Towards Data Science", *Towards Data Science*, 2019. [Online]. Available: https://www.towardsDatascience.com. [Accessed: 14- Jun- 2019].

[7]"Image Processing in OpenCV — OpenCV-Python Tutorials 1 documentation", *Opencv-python-tutroals.readthedocs.io*, 2019. [Online]. Available: https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_table_of_contents_imgproc/py_table_of_contents_imgproc.html. [Accessed: 14- Jun- 2019].

For CNN implantation is KERAS

[8]"Convolutional Layers - KERAS Documentation", *Keras.io*, 2019. [Online]. Available: https://keras.io/layers/convolutional. [Accessed: 14- Jun- 2019].

For UML Diagram syntax

[9]"Examples of UML diagrams - use case, class, component, package, activity, sequence diagrams, etc.", *Uml-diagrams.org*, 2019. [Online]. Available: https://www.uml-diagrams.org/index-examples.html. [Accessed: 14- Jun- 2019].

ORIGINALITY REPORT

12%	2%	0%	12%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Higher Education Commission Pakistan Student Paper	7%
2	www.scribd.com Internet Source	1%
3	於2012-06-13提交至Higher Education Commission Pakistan Student Paper	1%
4	Submitted to RDI Distance Learning Student Paper	1%
5	manualzz.com Internet Source	<1%
6	Submitted to Colorado Technical University Online Student Paper	<1%
7	Submitted to University of Technology, Sydney Student Paper	<1%
8	Submitted to Engineers Australia Student Paper	<1%
9	www.mcs.nust.edu.pk Internet Source	<1%
10	Submitted to Regis University Student Paper	<1%

11	dias.library.tuc.gr Internet Source	<1%
12	Submitted to University of Southampton Student Paper	<1%
13	Submitted to Misr International University Student Paper	<1%
14	Submitted to Ngee Ann Polytechnic Student Paper	<1%
15	Submitted to University of Bradford Student Paper	<1%
16	prezi.com Internet Source	<1%
17	Submitted to Uttar Pradesh Technical University Student Paper	<1%
18	Submitted to Bradford College, West Yorkshire Student Paper	<1%
19	Submitted to University of London External System Student Paper	<1%

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		