VioDet

(Video Violence Detection)

Authors

Muhammad Fahad Habib (NUST Reg. No. 201372)

Suleman Akhtar (NUST Reg. No. 175513)

Shahid Hussain (NUST Reg. No. 199887)

Supervisor

Dr. Naima Iltaf

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

July 2020

VioDet

(Video Violence Detection)

Authors

Muhammad Fahad Habib (NUST Reg. No. 201372)

Suleman Akhtar (NUST Reg. No. 175513)
Shahid Hussain (NUST Reg. No. 199887)

Supervisor
Dr. Naima Iltaf

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

July 2020

In the name of Allah, the Most Beneficent, the Most Merciful

CERTIFICATE OF CORRECTIONS & APPROVAL

Certified that work contained in this thesis titled " VioDet – Violence Detection in Videos", carried out by Muhammad Fahad Habib, Suleman Akhtar and Shahid Hussain under the supervision of Dr. Naima Iltaf for partial fulfillment of Degree of Bachelors of Computer Software Engineering, in Military College of Signals, National University of Sciences and Technology, Islamabad during the academic year 2019-2020 is correct and approved. The material that has been used from other sources it has been properly acknowledged / referred.

Approved by

Supervisor

Dr. Naima Iltaf

Department of CSE, MCS

_____

Date

_____

DECLARATION

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

PLAGIARISM CERTIFICATE (Turnitin Report)

This thesis has been checked for Plagiarism. Turnitin report endorsed by Supervisor is attached.

Signature of Student

Suleman Akhtar

Registration Number

175513

Signature of Supervisor

Dr. Naima Iltaf

ACKNOWLEDGEMENTS

There is no success without the will of ALLAH Almighty. We are grateful to ALLAH, who has given us guidance, strength and enabled us to accomplish this task. Whatever we have achieved, we owe it to Him, in totality.

We are also grateful to our parents and family and well-wishers for their admirable support and their critical reviews.

We are extremely grateful to our project supervisor Asst. Professor Dr. Naima Iltaf, for her continuous guidance and motivation throughout the course of our project. Without her help, we would have not been able to accomplish this feat.

We are highly thankful to all the faculty and staff of Computer Software Department of Military College of Signals, NUST for their support and training throughout our coursework. Their guidance helped us to carry out this project.

In the end, we would like to acknowledge the efforts of all our friends, colleagues and well-wishers whose prayers and support helped us in achieving this milestone.

DEDICATION

Dedicated to our exceptional parents and adored siblings whose tremendous support and cooperation led us to this wonderful accomplishment.

And to our supervisor, Dr. Naima Iltaf who has given us great support and valuable suggestions throughout the journey of this project.

ABSTRACT

VioDet
(Video Violence Detection)

In the world we live in today, the sheer volume of video content online and around us, is increasing exponentially. As a matter of fact, Cisco stated that by 2022, videos will make up more than 82% of all consumer internet traffic. With such large volumes of video content being generated, it is a reality that a significant portion of it contains some degree of violence.

Numerous psychological studies have shown that exposure to violent video content is a causal risk factor for increased aggressive behavior, aggressive cognition, and aggressive affect and for decreased empathy and prosocial behavior. Many studies can be quoted here, however, the problem is clear. We must detect and stop violent videos. VioDet is the solution.

VioDet is a modern AI-based video violence detection mechanism that uses deep learning to automatically detect violent content in videos and consequently, flag them as either violent or non-violent. With the help of this classification, users will have the knowledge about violent videos and can stop their propagation. Another case where VioDet can be helpful is real-time CCTV monitoring where if violence is detected, it can notify the concerned authorities.

After selecting a base model that could detect violence in videos accurately, we applied 'transfer learning' to get new model that met the requirements of our specific use case. In order to run inference, the model was made accessible through a REST API (built using Flask) and the API was connected to a web application built using React.

Key Words: - Artificial Intelligence, Transfer Learning, Computer Vision

Table of Contents

List of Figures

Table of Tables

Chapter 1. Introduction

1.1 Overview

VioDet will help security camera operators as well as general public to automatically detect violence in videos. Exposure to violent content has numerous side effects on a person's mental health and behavior. Furthermore, continuous monitoring of CCTV feeds is a tedious task. VioDet tackles both issues by providing a web service that allows detection of violence in live camera feeds as well as complete videos. VioDet deploys a custom-trained model that makes use of deep learning and computer vision techniques to perform the task of automatic violence detection in videos. The model was trained on over 3000 violent and non-violent videos that led to a robust and accurate solution.

1.2 Problem Statement

VioDet address two main problem areas.

Firstly, it is the tedious and inefficient task of continuously monitoring mostly static CCTV feeds. Considering the amount of static video feeds, a security operator must watch daily it is understandable that after just twenty minutes, their attention span significantly decreases, meaning most of that video is never watched, let alone recognizing culprits in the footage. By deploying an AI-based solution we eliminate the need to constantly monitor CCTV footages and can help the operator to focus on attention-requiring matters only.

Secondly, it is the exposure to violent video content which, according to numerous psychological studies, is a causal risk factor for increased aggressive behavior, aggressive cognition, and aggressive affect and for decreased empathy and prosocial behavior. In a 2009 Policy Statement on Media Violence, the American Academy of Pediatrics said, "Extensive research evidence indicates that media violence can contribute to aggressive behavior, desensitization to violence, nightmares, and fear of being harmed."

1.3 Approach

A modular approach was adopted to accomplish this project, so it was divided into four main phases. First phase was focused on the selection of suitable model and collection of all publicly available datasets regarding video violence. In the second phase, transfer learning was implemented with the help of the collected datasets and inference was successfully executed. Third phase revolved around the development of the application including the backend and the front-end. After successful integration of the work that was done in previous phases, VioDet was fully functional. In the fourth phase, VioDet underwent testing.

1.4 Scope

The scope of this project is limited to video content 'only'. Meaning that, this project would not utilize any associated audio tracks, subtitles and other contextual information for the purpose of detecting violence. In this project, aggressive behavior is the definition of violence rather than the presence of specific features such as blood or fire.

1.5 Objectives

The primary goal of this project is to provide video violence detection web-service with a reasonable accuracy and friendly user interface. Following are the main objectives of this project: -

1. Detect violence with a 'reasonable' accuracy.
2. Timely detection of violence (Less latency).
3. Minimize dependency on human security camera operators.
4. Minimum buffering in the results.
5. Easy-to-use and cost-effective solution.

During this project, all the aspects of software engineering are covered i.e. survey and feasibility analysis, requirement gathering, architectural and detailed design, implementation and testing along with documentation (SRS, SDS, Test Document, Final Report and User manual). Group members are also expected to develop extensive knowledge and technical skills in the fields of Deep Learning, Transfer Learning, Computer Vision, Python Programming (Anaconda, Flask) and Web Development (React).

1.6 Deliverables

Sr	Tasks	Deliverables
1	Literature Review	Literature Survey
2	Requirements Specification	Software Requirements Specification document (SRS)
3	Detailed Design	Software Design Specification document (SDS)
4	Implementation	Project demonstration
5	Testing	Evaluation plan and test document
6	Training	Deployment plan
7	Deployment	Complete application with necessary documentation

Table 1- 1 Deliverables

Chapter 2. Literature Review

In recent years, the problem of human action recognition from video has become achievable using computer vision techniques that leverage human-engineered features. However, despite its potential usefulness, the specific task of violent action detection has been comparatively less studied. A violence detector has immediate applicability in the surveillance domain.

The primary function of large-scale surveillance systems deployed in institutions such as schools, prisons and psychiatric care facilities is for alerting authorities to potentially dangerous situations. However, human operators are overwhelmed with the number of camera feeds and manual response times are slow, resulting in a strong demand for automated alert systems. Similarly, there is increasing demand for automated rating and violence detecting systems that can process the great quantities of video uploaded to the internet. Violence detection is becoming important not only on an application level but also on a more scientific level, because it has particularities that make it different from generic action recognition. For all these reasons the interest in violence detection has been steadily growing, and different proposals are already being published in major journals and conferences. Also, public datasets are becoming increasingly available that are specifically designed for this task.

2.1 Deep Learning for Violence Detection

Violence Detection algorithms are based on Computer Vision techniques only, are capable of extracting features. These features are human engineered, and accuracy and the reliability of the models directly depend on the extracted features and on the methods used for feature extraction.

The difficulty with this approach of feature extraction in frame classification is that the features that are to be considered must be pre-defined. When the number of classes of the classification goes high or the image clarity goes down it’s hard to work with traditional computer vision algorithms.

The main difference in deep learning approach of computer vision is the concept of end-to-end learning. There’s no longer need of defining the features and do feature engineering. The neural network can do extract those by itself. It can simply put in this way.

If a [deep] neural network is to recognize a cat, for instance, it is not told to look for whiskers, ears, fur, and eyes. The neural network is shown thousands and thousands of photos of cats, and eventually it works things out. If it keeps misclassifying foxes as cats, nothing is hard-coded, and the neural network eventually learns itself.

2.2 Violence Detection using hand-crafted features and learning features

2.2.2 Hand Crafted Features

The features designed by human are known as hand-crafted features. Among them, Space Time Interest Points (STIPs) and Improved Dense Trajectories (IDTs) are widely used in action recognition. Especially for violence detection, extreme acceleration patterns are used as the main feature. These kinds of methods based on hand-crafted features are intuitive and efficient for a specific small-scale dataset, but in a big dataset their deficiencies are triggered, leading to low training speed, massive memory usage and inefficient execution.

2.2.3 Learning Features

The features learned by deep learning networks are considered as learning features. With the enhancement of computer power brought by GPU and the collection of large-scale training set, action recognition based on deep learning makes a great progress. Multiple layers of convolutional Neural Networks consisting of spatial and temporal nets can be used. However, these methods are limited by the computational cost.

2.3 Related Work

Some works regarding video violence detection require audio cues for detecting violence or rely on color to detect cues such as blood. In this respect, it is to be noted that there are important applications, particularly in surveillance, where audio and color are not available. Besides, while explosions, blood and running may be useful cues for violence in action movies, they are rare in real-world situations.

Research on human perception of other's actions has shown that the kinematic pattern of movement is enough for the perception of actions. More specifically, empirical studies in the field have shown that relatively simple dynamic features such as velocity and acceleration correlate to emotional attributes perceived from the observed actions albeit the degree of correlation varies for different emotions.

Thus, features such as acceleration and jerkiness tend to be associated to emotions such as anger, happiness, whereas slow and smooth movements are more likely to be sadness.

In this context, this project assumes that violence in video can be reliably detected by such kinematic cues that represent violent motion and strokes. Consequently, VioDet attains better accuracy rates than state-of-the-art action recognition methods at much less computational cost.

Chapter 3. Software Req. Specification (SRS)

3.1 Introduction

The introduction of the Software Requirements Specification (SRS) provides an overview of the entire SRS with purpose, scope, definitions, acronyms, abbreviations, references and overview of the SRS. The aim of this document is to present detailed description of the project VioDet (Violence Detection in Videos) which uses a deep learning model to detect violence in videos, both live and downloaded. The detailed requirements of the VioDet are provided in this document.

3.2 Purpose

This document covers the software requirement specifications for project VioDet (Violence Detection in Videos). The aim of this project is to detect violence in videos using a custom-trained deep learning model deployed on a server.

Proper content filtering of violent media is an important issue nowadays, for its many applications: it can be used in conjunction with surveillance cameras to detect inappropriate behavior; aiding parental control by rating videos of streaming services; protecting users from receiving undesired media via messaging applications; blocking content from being uploaded to websites such as social networks, forums or educational platforms; or preventing it from being shown in specific places such as schools and workplaces. With hundreds of hours of video uploaded every minute through the Internet and becoming a part of our everyday life, comes many violent scenes not suited for people, especially for children. Hence, the demand for automated systems such as VioDet that detect these violent scenes is increasing.

This chapter is meant to outline the features and requirements of VioDet, to serve as a guide to the developers on one hand and a software validation document for the prospective client on the other.

3.3 Document Conventions

This section describes the standards followed while writing this document.

3.3.1 Headings

Heading are prioritized in a numbered fashion, the highest priority heading having a single digit and subsequent headings having more numbers, per their level.

All the main headings are titled as follows: single digit number followed by a dot and the name of the section (Calibri, size 22).

All second level subheadings for every sub section have the same number as their respective main heading, followed by one dot and subsequent sub heading number followed by name of the sub section (Calibri, size 20).

Further subheadings, i.e. level three and below, follow the same rules as above for numbering and naming, but different for font (Calibri, size 18).

3.3.2 Figures

All figures in this document are numbered. Context and flow diagrams are based UML standards.

3.3.3 Reference

All references in this document are provided where necessary, however where not present, the meaning is self-explanatory. All ambiguous terms have been clarified in the glossary at the end of this document.

3.3.4 Links to web pages

All links have been provided with underlined font, the title of the web page is written at the top of the link and the title may be searched on Google to pinpoint to the exact address.

3.3.5 Basic Text

All other basic text appears in regular, size 12 Calibri. Every paragraph explains one type of idea.

3.4 Intended Audience and Guidelines

The intended audiences for the VioDet SRS include the Project Supervisor, FYP Group itself (developers), UG Project Evaluation Team, and other persons at MCS CSE Department.

3.4.1 Project Supervisor

It will help the supervisor to supervise the project and guide the group in a better way. This document can be used by the supervisor to check whether all the requirements have been understood and, in the end, whether the requirements have been properly implemented or not.

3.4.2 BESE 22 FYP Group (Developers, Testers, and Documentation Writers)

For FYP group members, this document will provide the guideline for developing and testing the project.

3.4.3 UG Project Evaluation Team

It will help the evaluation team to evaluate the progress of FYP project. The document will provide the evaluators with the scope, requirements and details of the project to be built. It will also be used as basis for the evaluation of the implementation and final project.

3.4.4 Reading suggestions

The SRS begins with the title and table of contents. All level 1 and level 2 headings are given in the table of contents, but the lower subheadings are not included. Each main heading is succeeded by several subheadings, which are all in bold format. The product overview is given at the start, succeeded by the complete detailed features, including both functional and non-functional requirements. The entire interfaces are also described.

3.5 Product Scope

The scope of this project is limited to video content 'only'. Meaning that, this project would not utilize any associated audio tracks, subtitles and other contextual information for the purpose of detecting violence. In this project, we will be considering aggressive behavior as the definition of violence rather than the presence of specific features such as blood or fire.

3.6 Overall Description

3.6.1 Product Perspective

Nowadays, the amount of public violence has increased dramatically. This can be a terror attack involving one or several persons wielding guns to a knife attack by a single person. This has resulted in the ubiquitous usage of surveillance cameras. This has helped authorities in identifying violent attacks and take the necessary steps in order to minimize the disastrous effects. But almost all the systems nowadays require manual human inspection of these videos for identifying such scenarios, which is practically infeasible and inefficient. It is in this context that this project becomes relevant.

Moreover, video content makes up more and more proportion of the world's Internet traffic at present. Video service represented by short video clip and live streams have

become the new trends of the Internet. However, Internet video content is filled with some violent videos, which are seriously harming the construction of the network ecology. Furthermore, monitoring sudden violence in time creates tremendous challenges for video surveillance.

Thus, violent video detection is of vital importance.

3.6.2 Product Function

The major functionalities of VioDet are highlighted below:

1. To be able to accept a video from the user / Uploaded by the user.
2. To analyze the video such that it is divided into frames and then those frames are classified whether there was any violent content in them.
3. Performing the above-mentioned function using a custom-trained Deep learning model which can distinguish between violent and non-violent videos with good accuracy, precision and precision.
4. Based on the classification of frames, the application can determine whether the video is violent or not.

3.6.3 User Classes and Characteristics

The following section describes the types of users of the VioDet. There are explanations of the user followed by the interactions the user(s) shall be able to make with the software.

Security Companies/ Law Enforcement Agencies

Surveillance systems are often ineffective due to insufficient numbers of trained supervisors watching the footage and the natural limits of human attention capabilities. This is understandable, when considering the huge numbers of cameras that require supervision, the monotonic nature of the footage, and the alertness required to pick up on events and provide an immediate response. The security companies / LEAs can use this software in conjunction with their surveillance cameras to automatically detect violence or any aggression without the need for human supervision.

Social Media/Online Platforms Page Administrators

Online Platforms such as Facebook, Twitter, Reddit, Instagram and many others have pages on which people post and share hundreds of videos daily. These pages can be used by people with malign intentions to spread violent content which can have negative effects on other people’s mind. Administrators of these pages can use this software to check videos before uploading them.

3.6.3 Operating Environment

Hardware

The VioDet Application would require GPU workstation for training the model and a standard for the client to run the application on:

- **GPU Workstation (for training)**: - Processor i7 + NVIDIA GTX 1060(or better).
- **Standard Desktop (client usage): -** Processor i3 or advanced and 4GB+ RAM

Software

- Anaconda (Python 3.6)
- Deep Learning Framework: - Keras + Tensorflow
- Latest NVIDIA GPU Driver for GTX 1060.
- CUDA 6.1, cuDNN 7.6
- IDE: - Jupyter Notebook
- Python Libraries: - numpy, sklearn, opencv-python, keras, tensorflow-gpu, pillow.
- For Application Dashboard: - HTML, CSS, JS, Bootstrap, ReactJS, JSON.

3.6.4 Design and Implementation Constraints

Constraints of the product are given below:

- VioDet will only be able to detect violence form videos without taking any audio or other contextual information into consideration.
- Cameras used as the source of videos need to still/fixed.
- Input may contain noise along with data.
- Certain friendly gestures such as hugging may get labelled as violent.

3.6.5 User Documentation

A user manual will be provided to the users in which separate instructions will be given according to the user i.e. Regular user and the admin, developers and testers. It will include the details of the system's working. Help documents will also be a part of the system.

The project report will also be available for the users which will highlight the system features, working and procedures.

3.6.6 Assumptions and Dependencies

The video uploaded by the client is of standard quality and length. For instance, if a video is of too poor quality then the model will not be able to extract the requisite features to determine whether the video is violent or not.

3.7 System Features

This section illustrates organizing the functional requirements for the project VioDet by system features: -

- Video Acquisition
- Video Processing
- Checking for Violence
- Giving out final classification whether video is violent or not.

Following is the component Diagram:

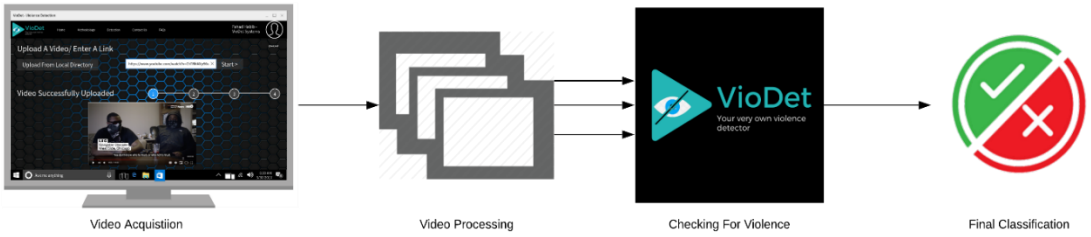

Figure 3 – 1 System Overview

3.7.1 Video Acquisition

Description

This feature enables the system to acquire video from the user who will upload it into the application. This video will be fed into the system for further processing.

Stimulus/Response Sequences

Normal Path: Video Successfully Uploaded
Preconditions • The video is selected from the local directory and uploaded by the user via the application.
Interactions • The captured video is sent to the system for processing.
Post conditions • Success Message displayed.

Categorization • **Criticality**: High • **Probability of Defects**: Medium • **Risk**: High

Table 3- 1 Video Acquisition (Normal Path)

Exceptional Path: Error Message Displayed
Preconditions • The video was not of the correct format or it was too long (max. video length is yet to be decided).
Interactions • An error message is displayed telling the user what the supported video specs are including format and length.
Post conditions • The video upload prompt reappears.
Categorization • **Criticality**: High • **Probability of Defects**: Medium • **Risk**: High

Table 3- 2 Video Acquisition (Exceptional Path)

Functional Requirements

1. The system shall be able to acquire videos from the directory's address or link entered by user.

2. If successful, it shall display a Success Message.

3. If not successful, then it shall display a Failure Message and re show the upload prompt.

3.7.2 Video Pre-Processing

Description

This feature involves sending the video to the model where it is divided into frames.

Stimulus/Response Sequences

Normal Path: Successfully divided into frames
Preconditions • The video is sent to the system/model for processing.
Post conditions • These frames are, then, fed into the trained model.
Categorization • **Criticality**: Medium • **Probability of Defects**: Medium • **Risk**: Medium

Table 3- 3 Video Pre-Processing

Functional Requirements

1. System shall be able to effectively divide video in to frames as per the specified fps parameter.

2. System shall be able to transmit those video frames to the model on the server.

3.7.3 Check for Violence

Description

The video frames obtained from previous stage will be analyzed and features data will be extracted from these frames.

Stimulus/Response Sequences

Normal Path: Model can successfully extract feature data from frames.
Preconditions • The frame images are successfully loaded into the model.
Interactions • Spatio-temporal and global features are extracted frame by frame.
Post conditions • The model continues to process the next frames.
Categorization • **Criticality**: High • **Probability of Defects**: High • **Risk**: High

Table 3- 4 Check for Violence (Normal Path)

Exceptional Path: Model is unable to extract feature data from frames
Preconditions • The frame images are successfully loaded into the model.
Interactions • The model cannot extract features from the frames due to poor image quality or noise in image.
Post conditions • An error notification is generated.
Categorization • **Criticality**: High • **Probability of Defects**: Low • **Risk**: High

Table 3- 5 Check for Violence (Exceptional Path)

Functional Requirements

1. The model shall be able to receive and load video frames into the model.
2. The model shall be able to extract feature data for each frame.
3. In case the frame image quality is poor, then it shall generate an error notification.

3.7.4 Display Results

Description

Based on the outputs from the previous stage, the system shall give a final classification whether the video is violent or not.

Stimulus/Response Sequences

Normal Path: Video is non-violent
Preconditions • Features data per frame is generated from the previous stage and analyzed.
Post conditions • The user is notified that the video does not contain any violent scenes/actions.
Categorization • **Frequency**: High • **Criticality**: High • **Probability of Defects**: Medium • **Risk**: High

Table 3- 6 Final Classification (Normal Path)

Exceptional Path: Video is violent.
Preconditions • Features data per frame is generated from the previous stage and analyzed.
Post conditions • The user is notified that the video contains violent scenes/actions and is, therefore, violent.
Categorization • **Criticality**: High • **Probability of Defects**: Low • **Risk**: High

Table 3- 7 Final Classification (Exceptional Path)

Functional Requirements

1. The model shall be able to assess the features data and based on it, give a final classification that whether the video is violent or not.
2. The model shall be able to transmit the results/error back to the application.

3.8 External Interface Requirements

3.8.1 User Interfaces

The VioDet Dashboard will provide its user with a clear statement stating whether the uploaded video or live stream of video is violent. The video being played would be shown on the dashboard along with the frames in which violence is detected. Here are two mockups of how our UI will look like.

Home Page

This screen will allow user to login into the web application.

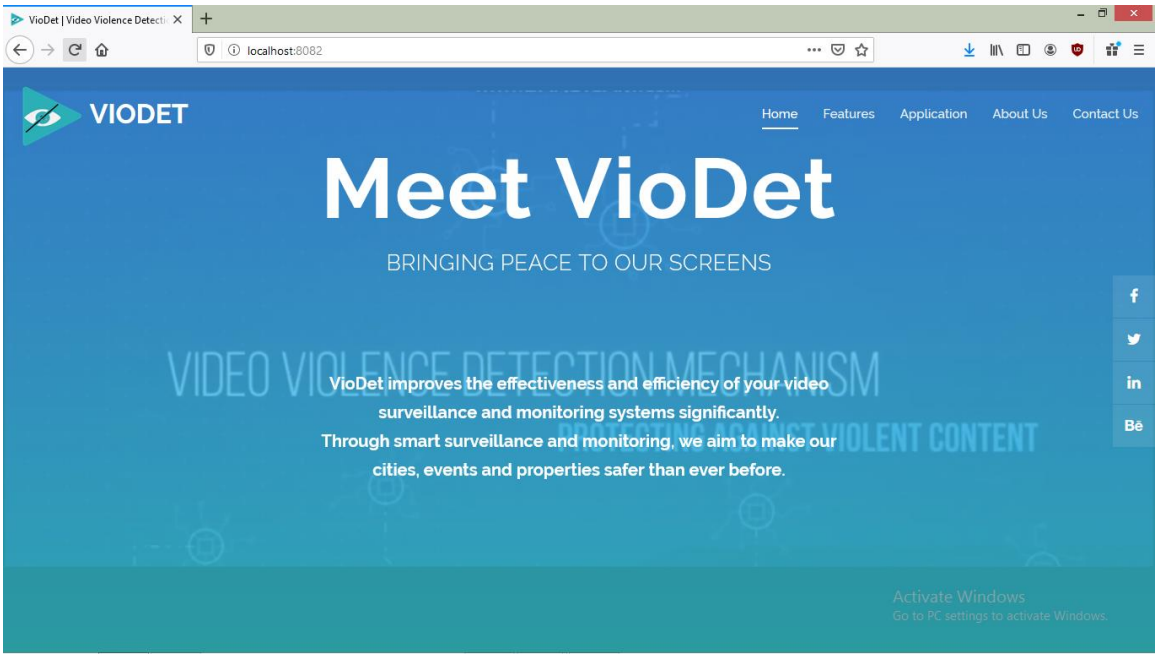

Figure 3 – 2 Home Page

Detection Page

On this page, the user uploads a video from his/her local directory or gives access to the system's local camera feed. After that, the video is processed, and violence detection is performed on it. Finally, the results are displayed on this screen.

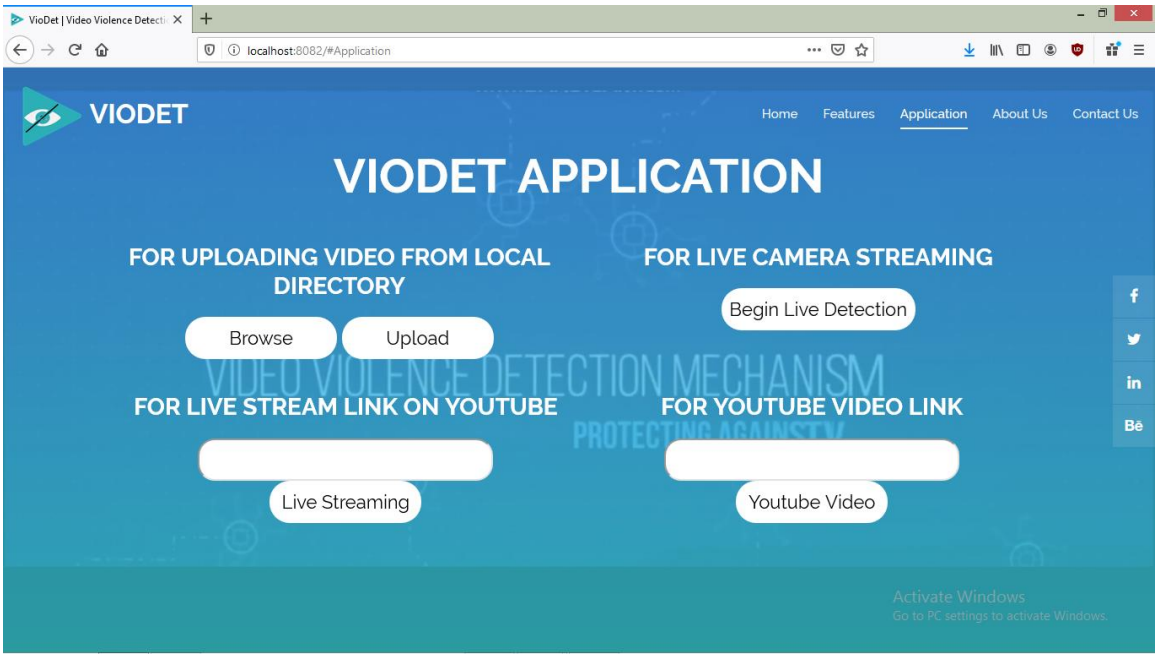

Figure 3 – 3 Detection Page

3.8.2 Hardware Interfaces

1. Computers/Laptops/Mobile Phones with Internet Connections

3.8.3 Software Interfaces

1. Operating System: - Windows 10.
2. Frontend Dashboard: -
3. Deployment of Trained Model on Server (Flask).

3.8.4 Communications Interfaces

Wi-Fi will be used by the client to connect to the server on which the trained model is present. Using this connection, the client can send the videos to that server and get results.

3.9 Other Nonfunctional Requirements

3.9.1 Performance Requirement

The performance requirement for VioDet is that the results of violence detection is accurate, and another aspect is that after how much time the results are shown after the video is uploaded. This time should be minimum possible however it depends on the network connection speed as well.

3.9.2 Security

System shall be secure enough not to breach any security that could comprise the service. Videos and their classifications should not be compromised.

3.9.3 Legal

System should follow customer privacy policy strictly.

3.9.4 Software Quality Attributes

Reliability

The application should run perfectly with all the features mentioned above available. It should be tested and debugged completely. All exceptions should be well handled.

User Friendliness/Simplicity

The graphical user interface of system is to be designed with usability as the priority. The application will be presented and organized in a manner that is both visually appealing and easy for the user to navigate.

Accuracy

To ensure reliability and correctness, the system must attain an acceptable level of accuracy with regards to the classification of videos as violent/nonviolent. Acceptable Range (85% +).

3.9.5 Business Rules

1. The user must upload a video that exists and is in the specified directory.
2. The video which the user is uploading must not be more than the supported length.
3. The video which the user is uploading must be of minimum or higher quality.
4. The user must provide a valid path where the detected video is to be stored.

Chapter 4. Design and Development

4.1 Introduction

The introduction of the Software Design Specification (SDS) document provide overview of the entire SDS with purpose, scope, definitions, acronyms, abbreviations, references and overview of the SDS. The aim of this document is to present, in detail, the functional and non-functional aspects of the project VioDet which uses image processing techniques and deep learning to detect violence in videos. The detailed descriptions and visualizations of the VioDet are provided in this document.

4.1.1 Purpose

This software design document describes the architecture and system design of project VioDet, Version 1.0. The document is meant to detail the design of features and requirements of VioDet, to serve as a guide to the developers on one hand and a software validation document for the prospective client on the other. Document includes classes and their inter-relationships, use cases with detailed descriptions, sequence diagrams and various flow charts.

4.1.2 Project Scope

The scope of this project is limited to video content 'only'. Meaning that, this project would not utilize any associated audio tracks, subtitles and other contextual information for the purpose of detecting violence. In this project, we will be considering aggressive behavior as the definition of violence rather than the presence of specific features such as blood or fire.

The primary goal of this project is to provide video violence detection service with a reasonable accuracy and friendly user interface.

This project can be of benefit to each one of us who uses the internet and to security companies as well. The ubiquitous usage of surveillance cameras has led to a requirement of human inspectors who must remain continuously alert for identifying violent scenarios, which is practically infeasible and inefficient. It is in this context that this project becomes relevant.

Moreover, video content makes up more and more proportion of the world's Internet traffic at present.

Video service represented by short video clip and live streams have become the new trends of the Internet.

However, Internet video content is filled with some violent videos, which are seriously harming the construction of the network ecology. Furthermore, monitoring sudden violence in real time creates tremendous challenges for video surveillance.

4.1.3 Abbreviations/Definitions

SDS: Software Design Specification.

OpenCV: - Python library used for image manipulation.

Flask: - Python-based micro web framework.

TensorFlow: - Machine Learning platform.

4.1.4 Overview of Document

This document is about the detailed architectural design of VioDet. For simplicity the document is divided into various sections. Section 1 introduces the document and provides overview for executive purposes. Section 2 includes detailed description of the system with various diagrams and charts. This section includes all the architectural details of system under development. Section 3 describes all the modules and components of the system in detail one by one. Section 4 compares this product to various other similar products available in market. Section 5 throws light on the design decisions and tradeoffs. In the last section pseudo code of all the components in provided.

This document is intended for developers, testers, users, documentation writers, project clients, project supervisor and project evaluators. A copy of this document will be made available to all stakeholders.

4.2 System Architecture Description

This section provides detailed system architecture of VioDet. Overview of system modules, their structure and relationships are described in this section. User interfaces and related issues are also discussed.

4.2.1 Overview of Modules

VioDet requires several modules to work. These modules can be divided into two parts, the Front-End Application and Back-End Server. Following is the brief overview of all these modules. Detailed descriptions of these modules are presented in section 3.

1. **Front-End Application Modules: -**

   1.1. Video Acquisition and Transmission Module

   This is the module from where the major functioning of application initiates. This module takes a video from the user which could be in the form of an uploaded video or link to a live stream and pass it to the server.

   1.2. Notification/Result View Module

   The notification and resultant are on display by this module for the user to see.

2. Back-End Server Modules: -

2.1 Video Acquiring Module: -

The task of this module is to load the video or access the live stream, which was received at the endpoint.

2.2 Preprocessing Module: -

The video transmitted by the application is received and it is divided into frames at specified rate which would be set by the user (fps).

2.3 Violence Detection Module: -

When preprocessing module has completed its work and a series of frames has been produced, this module would run those frames through the custom-trained deep learning model. Based on the spatio-temporal features of each frame, the frame would be given a final classification.

2.4 Notification Module: -

If a 'specific' amount of violence is detected, the video is termed as violent. In case it is violence is detected less than the threshold, the video is termed as non-violent. The generated notification is sent to the server along with the frame details. The server will then notify the web application about the results.

4.3 Structure and Relationships

This section covers the overall architectural description of VioDet. It encompasses the high-level and low-level descriptions of the project including block diagrams of the application and the deep learning model. Moreover, a complete object-oriented description which includes class diagrams, sequence diagrams and others. Finally, the rationale for the design pattern is provided.

4.3.1 Architectural Design

The architectural design has been divided into two portions; one covers the application architecture while the other depicts the architecture of the deep learning model used.

4.3.1.1 System Block Diagram (VioDet Application)

This diagram shows the higher-level description of the application. It shows all the modules of the system and their associations and flow of data between modules:

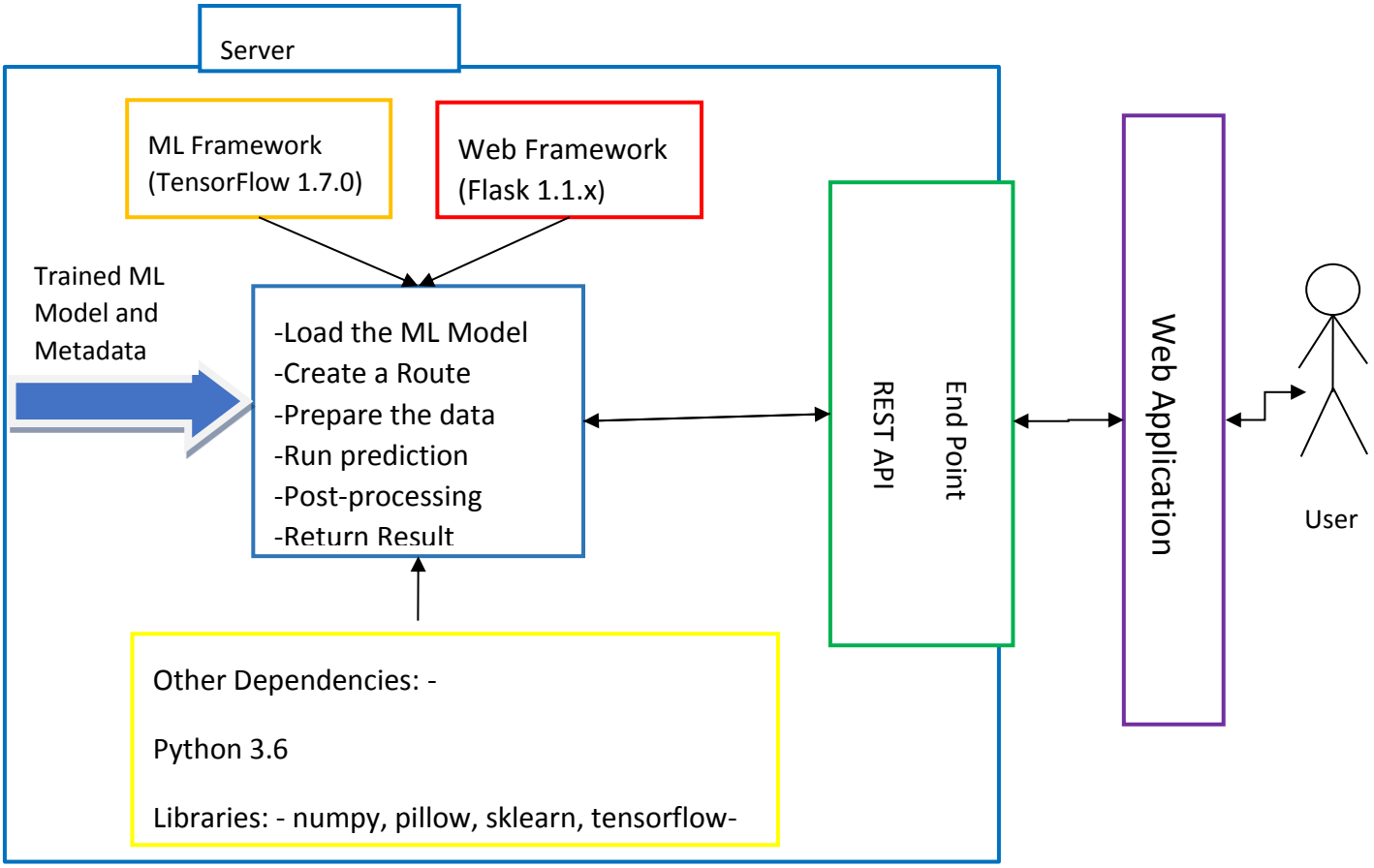

Figure 4 – 2 System Block Diagram

All the modules mentioned above are incorporated in this system block diagram.

First, we have the deep learning model which we have trained on datasets of videos. Then, we wrap that model using Flask into a web service so that the user can use the model to run predictions.

We construct a server that provides the model with the environment it needs to execute including all the dependencies such as TensorFlow, Python, and Python Libraries. This Server would communicate with the Web Application via the End Point based on REST API.

Now, all that is left, is for the user to upload a video and/or provide a link to a live stream.

4.3.1.2 System Block Diagram (DL model)

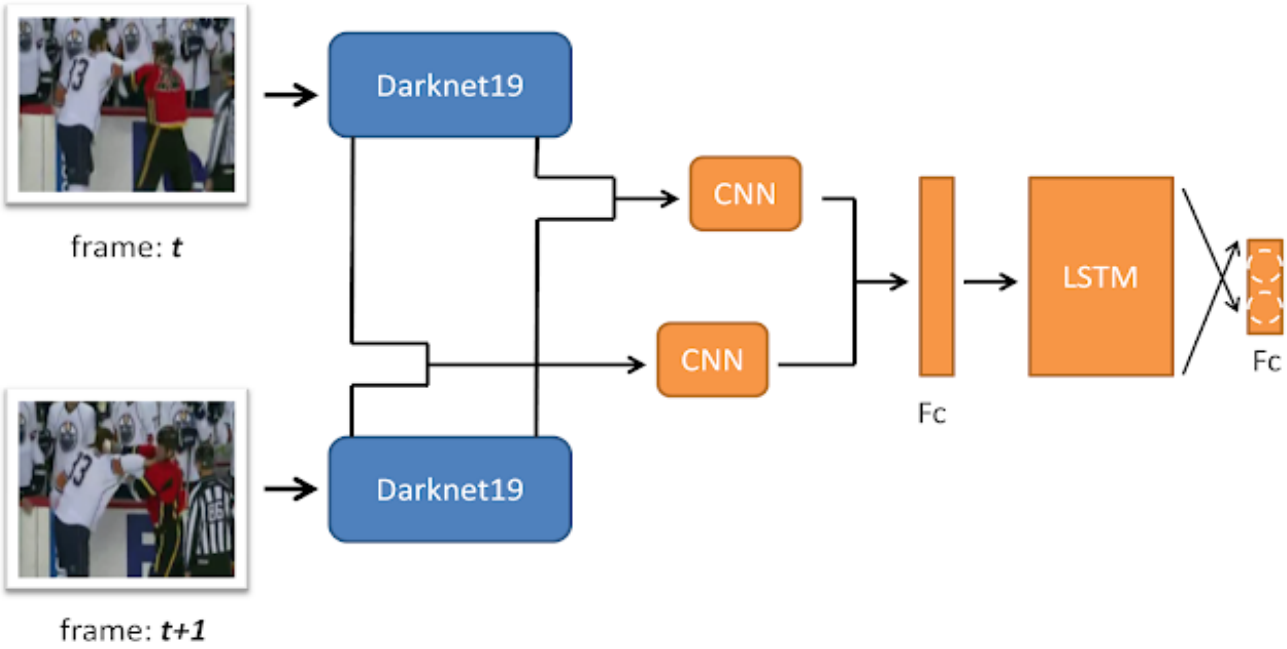

Figure 4 – 3 Model Block Diagram

In this model, following steps are being performed: -

1. Two consecutive frames are taken as input. They are processed separately but in parallel by the two pretrained CNNs (Darknet19). Output from the bottom layer of the Darknet19 gives us low-level features while output from the top layer of the Darknet19 gives us high-level features.

2. The low-level feature outputs from Darkent19 are concatenated and fed into one of the additional CNNs (which are not pretrained). The additional CNN is supposed to learn the local motion features as well as the appearance invariant features by comparing the two frames feature map.

3. The high-level feature outputs from Darknet19 are concatenated and fed into the other additional CNN. Here, the high-level features of the two frames are compared.

4. Output from both additional CNNs are concatenated and passed to a fully connected layer and the LSTM cell to learn the global temporal features.

5. Finally, the outputs of the LSTM cell are classified by a fully connected layer which contained two neurons that represent the two categories (violent and non-violent), respectively.

4.4 Decomposition Description

4.4.1 Sequence Diagrams

Following sequence diagrams show the sequence of activities performed in all of the use cases.

Sequence Diagram 1 shows the sequence of actions involved in the input of the video.

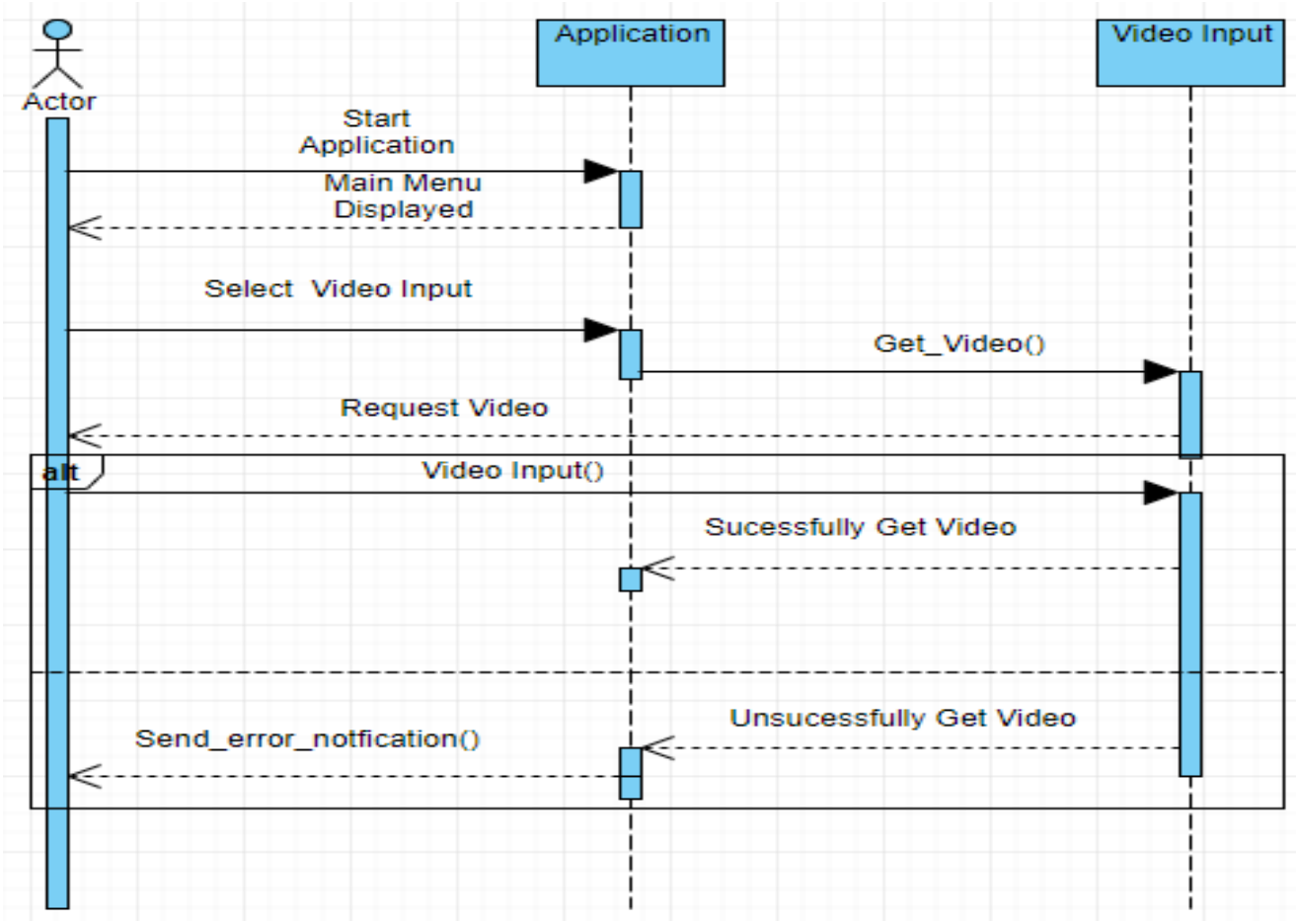

Figure 4 – 4 Sequence Diagram (Video Input)

Sequence Diagram 2 shows the sequence of actions up till the detection of violence.

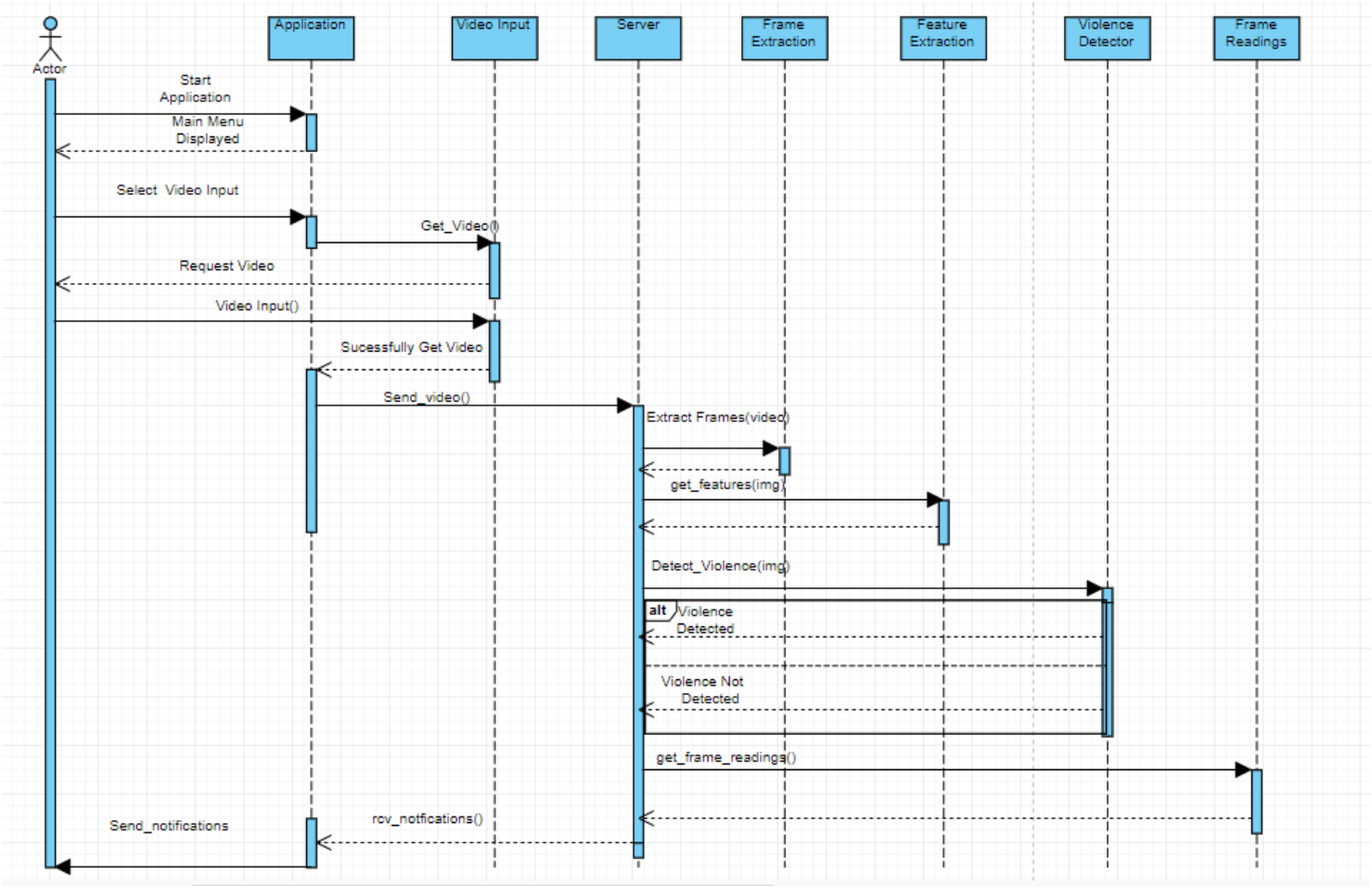

Figure 4 – 5 Sequence Diagram (Violence Detection)

Sequence Diagram 3 shows the sequence of actions of the complete cycle of VioDet system.

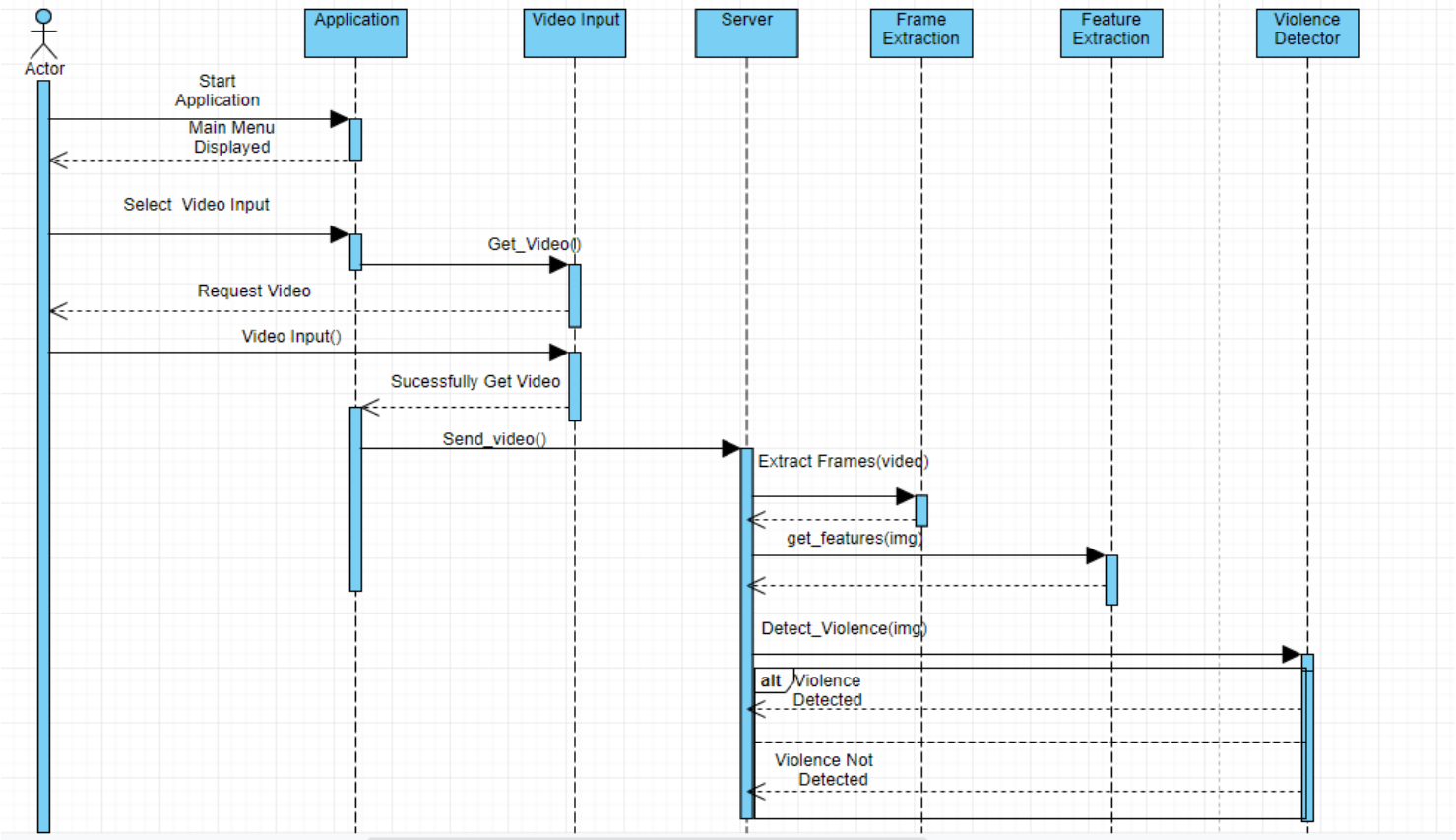

Figure 4 – 6 Sequence Diagram (Complete Sequence of Actions)

4.4.2 Implementation View (Class Diagram)

Class diagram shows all the classes of system and their relationship with one another. Following is the class diagram by following the MVC design pattern

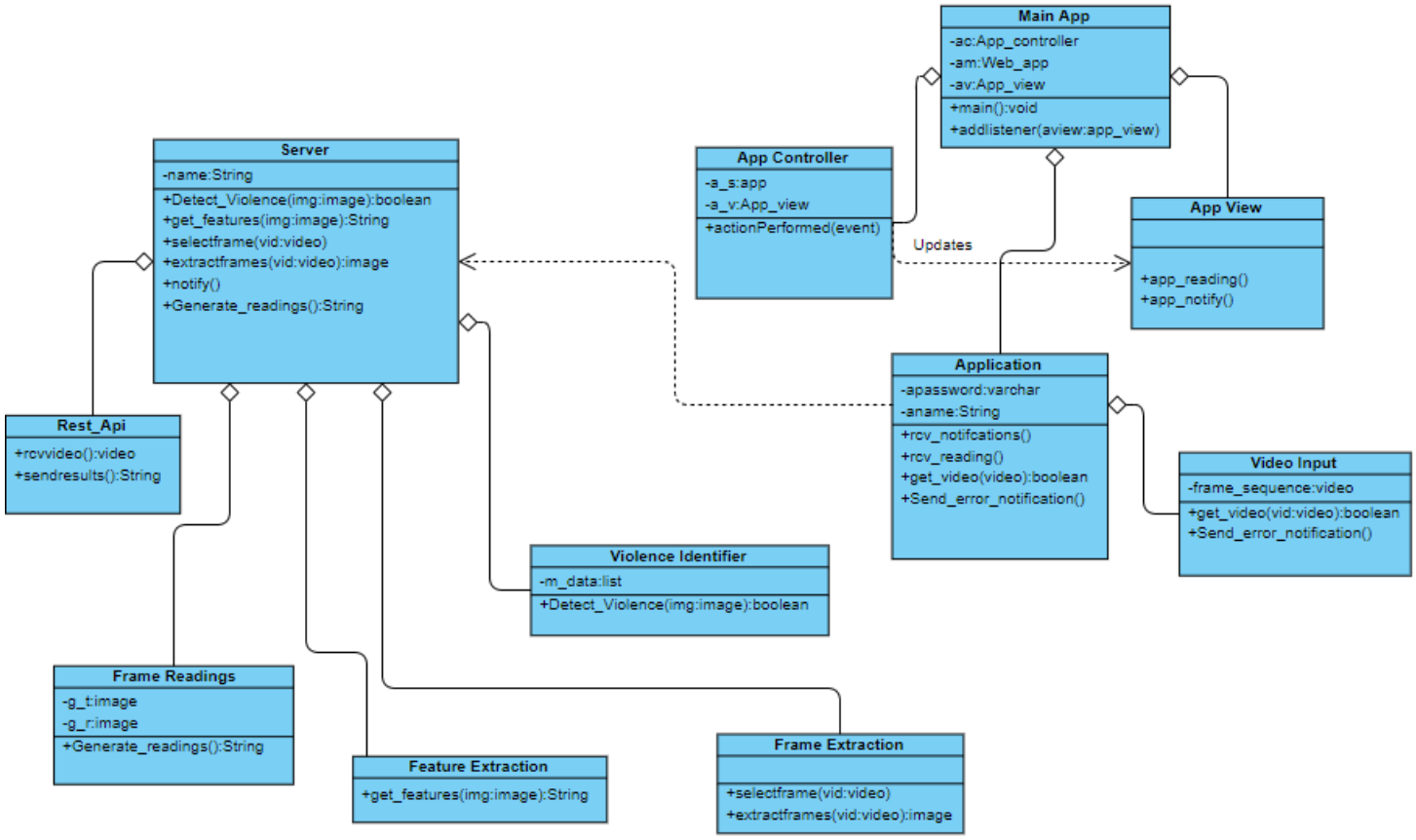

Figure 4 – 7 Class Diagram

Class	Description
Application	It is main class that starts when the user interacts with the system. This is the model class in applications' MVC pattern. It receives notification and readings from server whether the video was violent or non-violent.
Video Input	This class gets video from the user.
App controller	It plays the role controller in app's MVC and has objects of view and model
App view	It is the view part in MVC of application and generates view of application and get updated by controller class.
Main app	This main class of application has objects of model, view and controller classes of application's and implements MVC pattern in app.
Feature Extraction	This class uses extracted frames and performs feature selection technique.
Frame Extraction	This class extracts frames from the video which will be then further used.
Violence Identifier	This class will classify the video as violent or not.
Frame Readings	This class gets readings about the violent and non-violent frames in the video and saves the data.
Server	It has all the methods performed by server and it receives video from application through REST API and then using it methods determines whether the video was violent or not.
REST API	It receives video from the application through the API and then sends it to the server.

Table 4- 1 Class Descriptions

4.4.3 Dynamic View (Activity Diagram)

In activity diagram, the dynamic view of the system is shown. All the activities are shown concurrently with their respective start and end states.

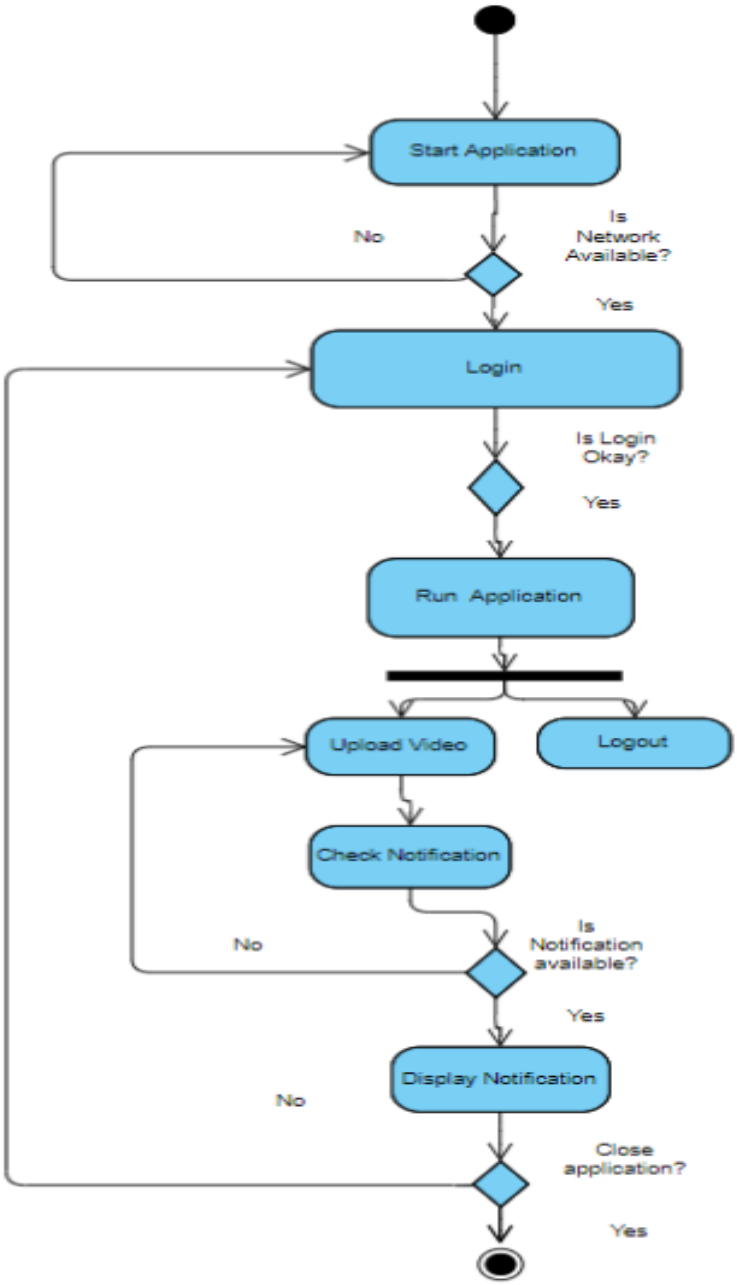

Figure 4 – 8 Activity Diagram (Video Input)

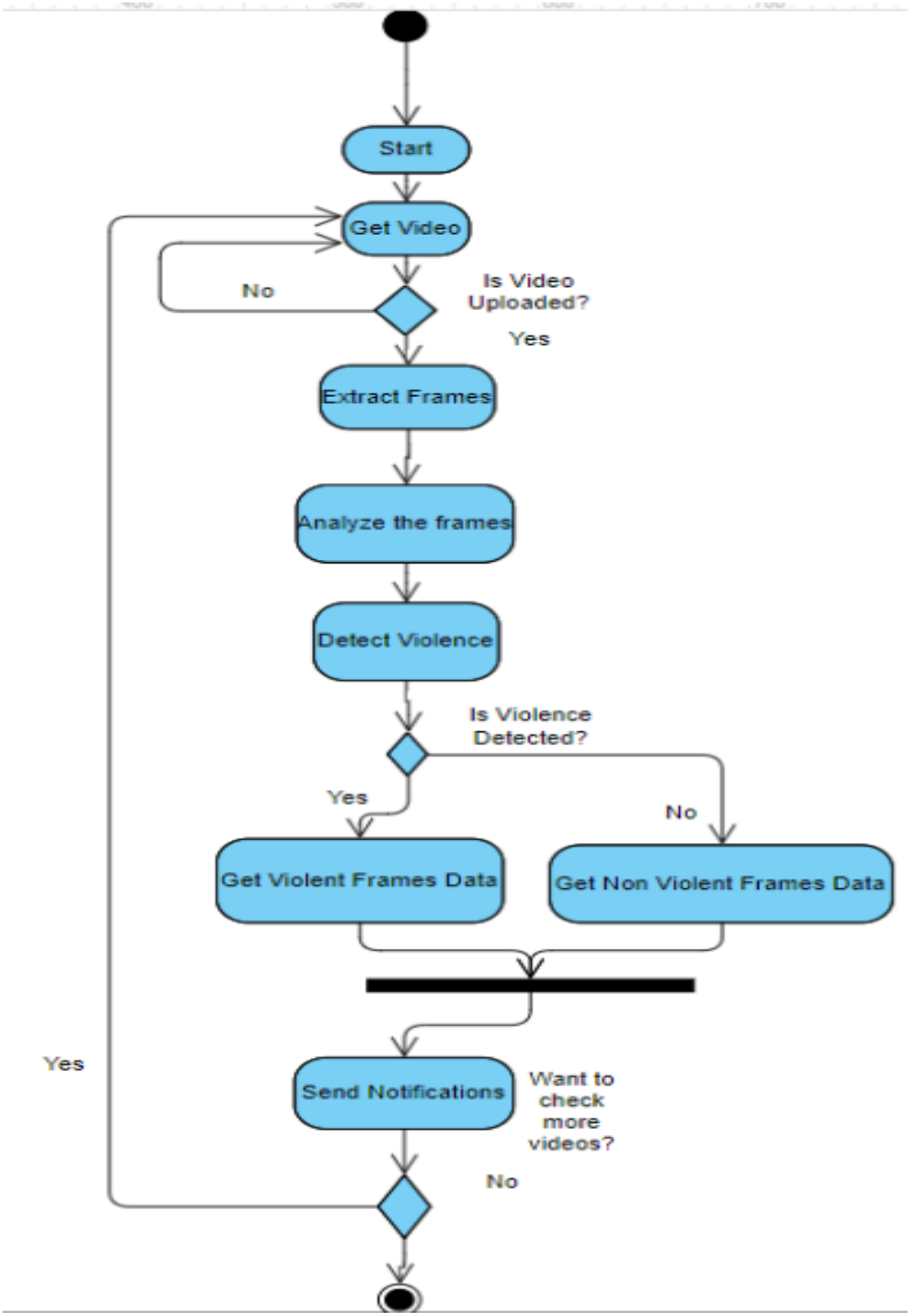

Figure 4 – 9 Activity Diagram (Violence Detection)

4.4.5 Logical View (State Diagram)

Following is the state diagram of VioDet showing all the states that the system has during action.

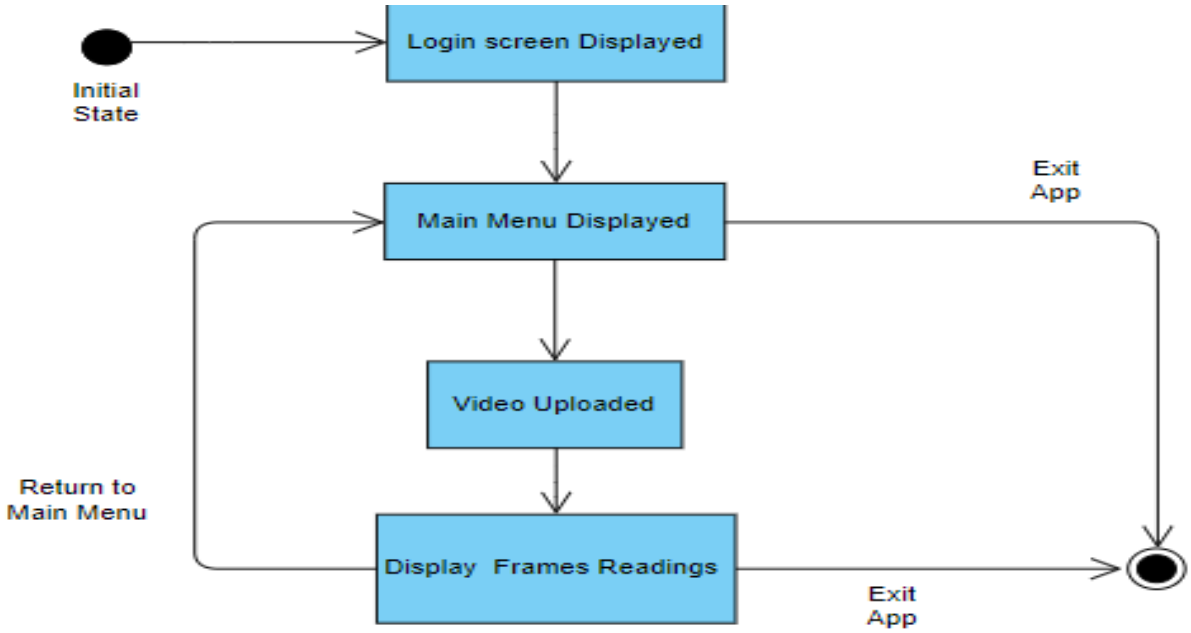

Figure 4 – 10 State Diagram (Video Input)

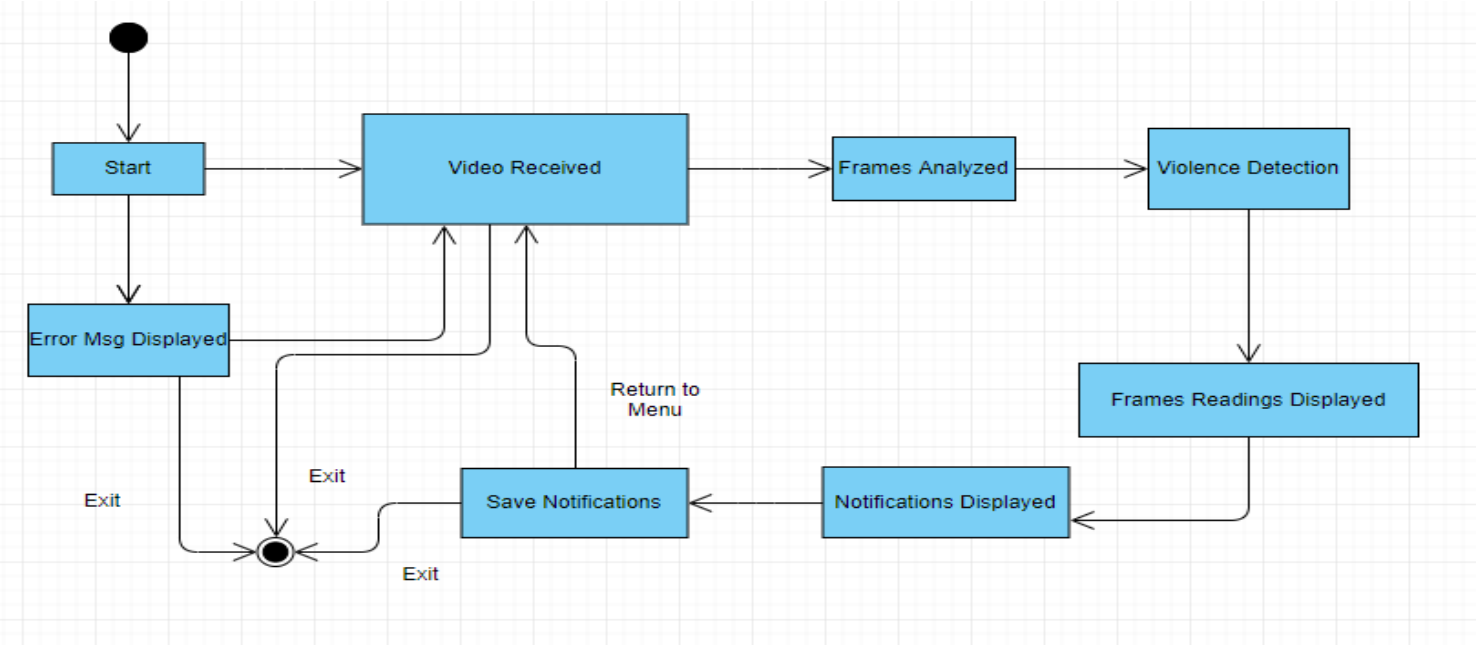

Figure 4 – 11 State Diagram (Violence Detection)

4.4.6 Structure Chart

In activity diagram, the dynamic view of the system is shown. All the activities are shown.

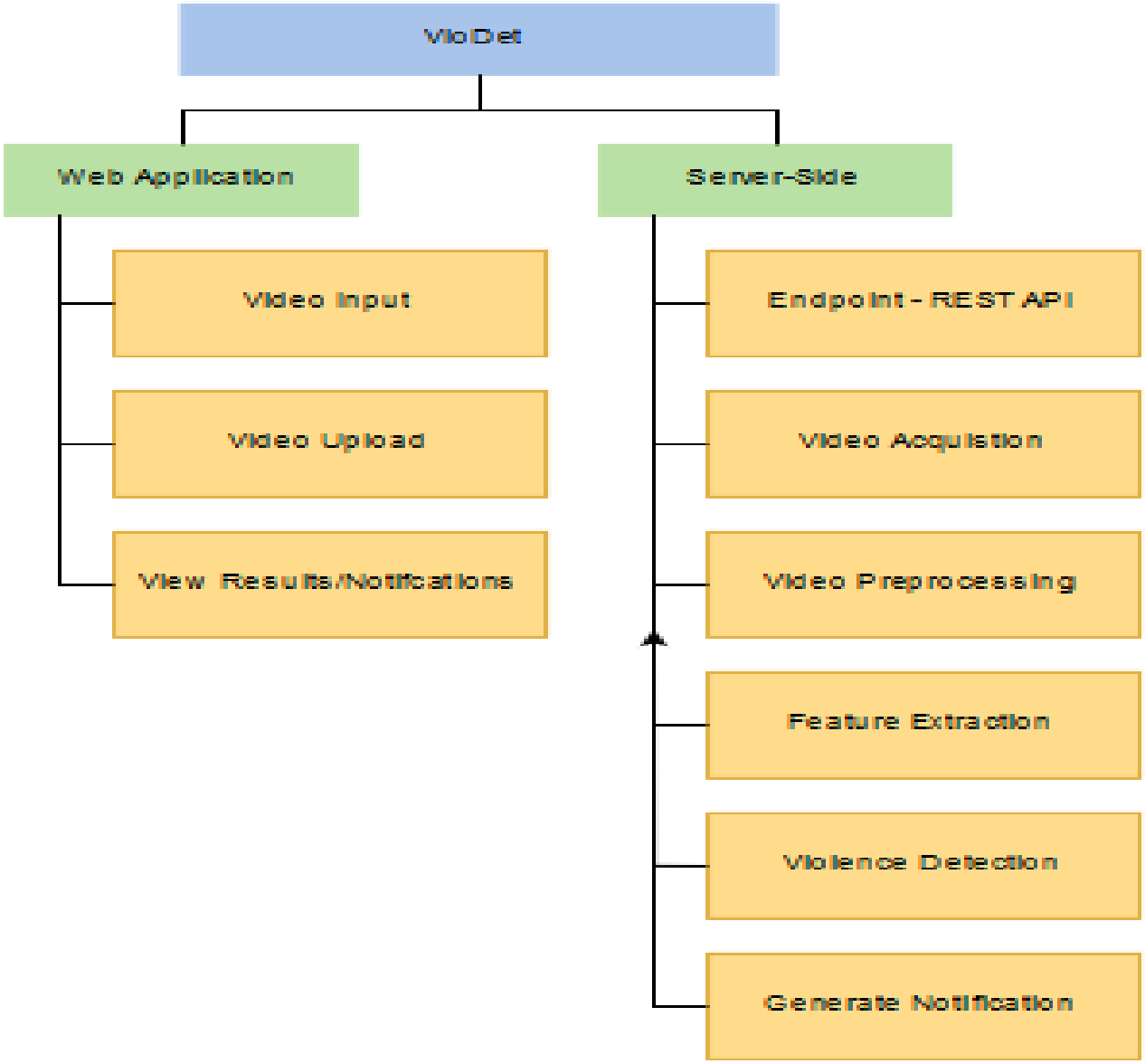

Figure 4 – 12 Structure Chart

4.4.7 User View (Web Application Use Case Diagram)

In this section, the major use cases of the VioDet from user's perspective as well as from the system's perspective, are covered.

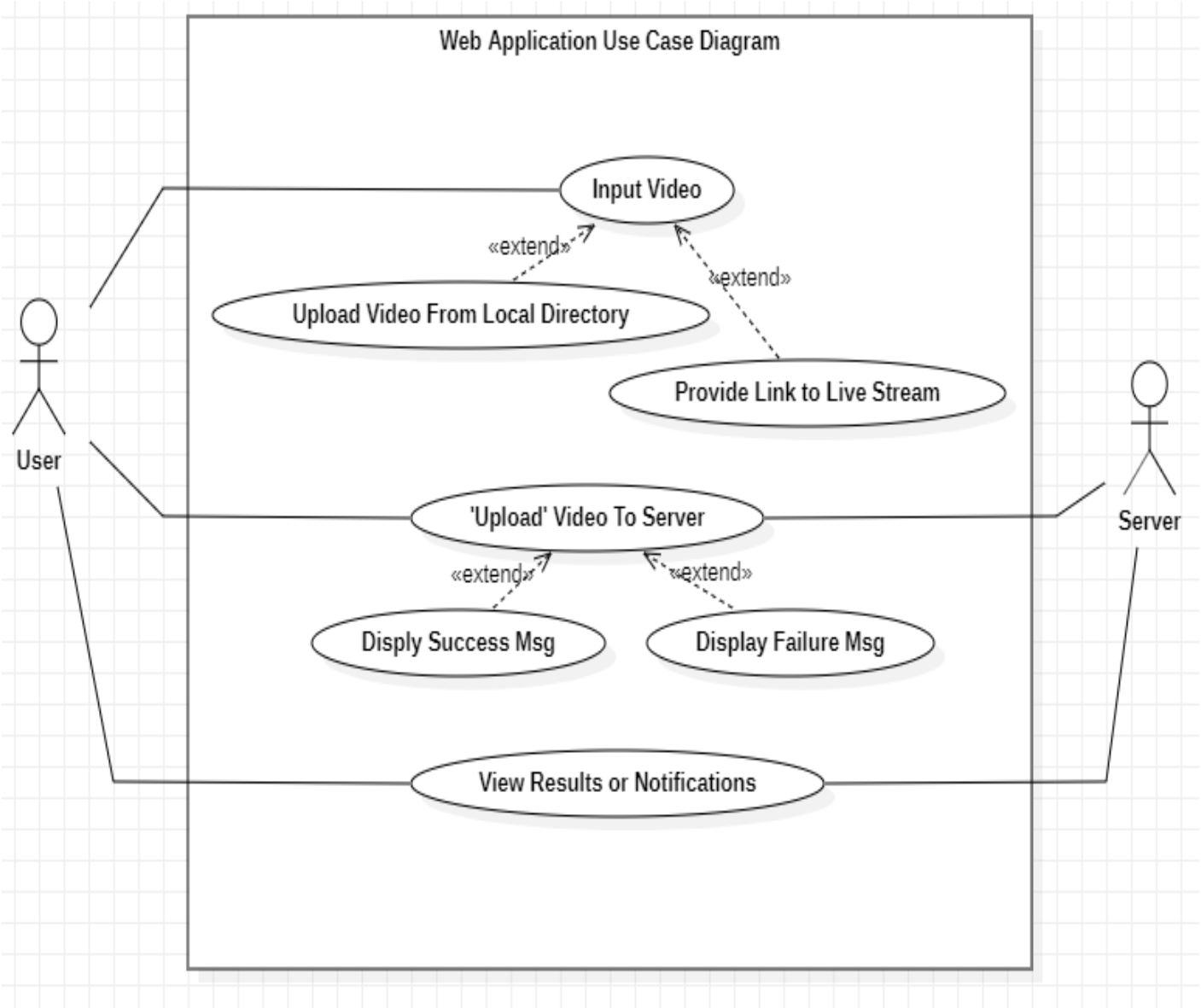

Figure 4 – 13 Use Case Diagram (Web Application)

Uses cases shown in Fig 4-13 are described below in detail.

Use Case 1(Input Video)

Use case name	Input Video
Primary actor	User
Secondary actor	N/A
Normal course	- User navigates to the Detection page on the web page.
	- User selects how he would like to provide input i.e. through live stream link/web cam access or uploading a video from local directory. - Based on the selected option, a prompt appears for the user to provide path - Video is successfully acquired from the provided path.
Pre-Condition	User must ensure that the link to the live stream is valid, the specified video is present in the local directory, the application has access to a functional web cam (whichever applicable).
Post-Condition	Internet connection is available to connect to the server.
Alternate course	The application was unable to acquire video from the input information provided by the user.
Pre-condition	The link to live stream was not valid or the specified video was not present in the local directory.
Post-condition	Error notification is displayed.
Extends	Upload Video From Local Directory/Camera Feed
Assumptions	- Videos being inputted are of the minimum supported resolution i.e. 480p. - The video which the user is uploading must not be more than the supported length.

Table 4- 2 Use Case 1 (Video Input)

Use Case 2 (Upload Video from Local Directory)

Use case name	Upload Video From Local Directory
Primary actor	User
Secondary actor	N/A
Normal course	- User navigates to the Detection page on the web page.
	- User selects 'Upload Video' Option - A local browse prompt is displayed where user selects the target video. - The target video is uploaded to the application.
Pre-Condition	User must ensure that the path specified for the target video is valid and that the video is present there.
Post-Condition	The application successfully in gaining access to the video and uploading it.
Alternate course	The application was unable to acquire video from the local directory path provided by the user.
Pre-condition	The target video was not present in the local directory.
Post-condition	'Video Not Found' notification is displayed.
Assumptions	- Videos being inputted are of the minimum supported resolution i.e. 480p. - The video which the user is uploading must not be more than the supported length. - The video selected is of supported formats (.mp4, .avi)

Table 4- 3 Use Case 2 (Upload Video from Local Directory)

Use Case 3 (Provide Link to Live Stream)

Use case name	Provide Link to Live Stream
Primary actor	User
Secondary actor	N/A
Normal course	- User navigates to the Detection page on the web page.
	- User selects 'Live Stream' Option - User enters the link to the live stream. - The target live stream is successfully located by the application.
Pre-Condition	User must ensure that the link being inputted is valid.
Post-Condition	The application successfully gains access to the live stream.
Alternate course	The application was unable to capture the live stream from the link provided
Pre-condition	The link provided was not valid or no live stream was found at that link.
Post-condition	'Live Stream Not Found' notification is displayed.
Include	N/A
Assumptions	- Videos being inputted are of the minimum supported resolution i.e. 480p. - The live stream is currently running, that is, it has not finished.

Table 4- 4 Use Case 3 (Live Stream)

Use Case 4 (Upload Video to Server)

Use case name	Upload Video to Server
Primary actor	User
Secondary actor	Server
Normal course	- After inputting video information, user clicks on the 'Start Detection' Button.
	- Application transmits the video/link to the server. - Application goes into 'waiting' state as server processes the information and transmits back results/notifications.
Pre-Condition	The application has successfully captured the target video or live stream.
Post-Condition	Internet connection for connecting with the server. "Successfully Transmitted. Awaiting Results." Message displayed.
Alternate course	The application was unable to start violence detection on the target video/live stream.
Pre-condition	The application could was unable to transmit the video information due to no internet connection or offline server.
Post-condition	'No Internet Connection/Server Down' notification is displayed.
Include	N/A
Assumptions	- Videos being inputted are of the minimum supported resolution i.e. 480p. - The live stream is currently running, that is, it has not finished.

Table 4- 5 Use Case 4 (Upload Video to Server)

Use Case 5 (View Results / Notifications)

Use case name	View Results/Notifications
Primary actor	User
Secondary actor	Server
Normal course	- After the application has transmitted information about the target video, the user and the application wait for the response.
	- Application receives the processed video which has undergone the violence detection procedure at the server end. - Application also receives a notification mentioning that 'violence was detected'. - Application displays the resultant video along with the notification for the user to see.
Pre-Condition	The application has a working internet connection.
Post-Condition	Resultant video and 'violent' notification displayed.
Alternate course	The application receives the processed video in which no violence was detected.
Pre-condition	The application has a working internet connection.
Post-condition	Resultant video and 'non-violent' notification displayed.
Extends	N/A
Assumptions	- Videos being inputted are of the minimum supported resolution i.e. 480p. - The live stream is currently running, that is, it has not finished. - The user is waiting for the results.

Table 4- 6 Use Case 5 (View Results/Notifications)

4.4.8 User View (Server-Side Use Case Diagram)

Following diagram shows course of events that take place on the server-side i.e. when the server receives the video from the application and processes the video undergoes:

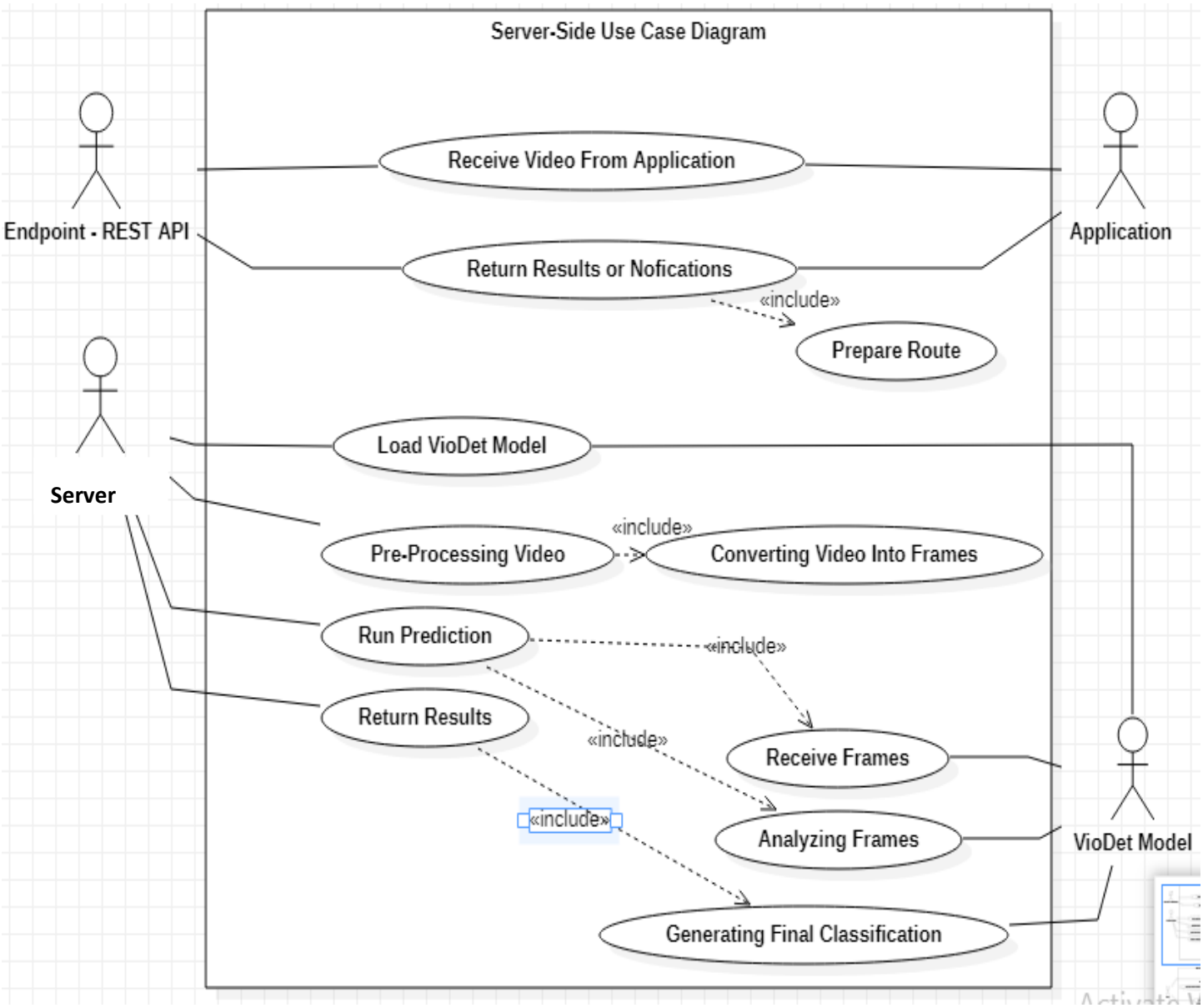

Figure 4 – 14 Use Case Diagram (Server)

Use Case 1 (Receive Video from Application)

Use Case Name	Receive Video/Frames from Application
Primary Actor	Endpoint – REST API
Secondary Actor	Application
Normal Course	- Successfully receive the http get/post request from the application. - Interpret the message body received from the application to determine whether it's video or frames. - Return a JSON response that 'video information has been successfully received'
Pre-Condition	The endpoint received a properly formatted http request.
Post-Condition	The target video/live stream has been successfully accessed at the server-side. 'Success' message sent to the application side.
Alternate Course	The endpoint receives the http request but is unable to interpret it due any inconsistencies in the request.
Pre-Condition	The get/post request was not properly formatted.
Post-Condition	'Failure to interpret request' message transmitted to the application.
Assumptions	- Videos being inputted are of the minimum supported resolution i.e. 480p. - The live stream is currently running, that is, it has not finished.

Table 4- 7 Use Case 1 (Receive Video From Application)

Use Case 2 (Return Results/Notifications)

Use Case Name	Return Results/Notifications
Primary Actor	Endpoint – REST API
Secondary Actor	Application
Normal Course	- Assuming that violence was detected, receive the consequent results/notifications from the Server. - Generate a properly formatted JSON response that would include the resultant video and its classification. - Transmit that JSON response to the application.
Pre-Condition	The endpoint received a violent classification result.
Post-Condition	The results are successfully transmitted and received.
Alternate Course	No violence was detected in the target video. Consequently, a 'non-violent' JSON notification along with the resultant video are sent to the application.
Pre-Condition	The endpoint received a non-violent classification.
Post-Condition	The resultant video and its classification have been successfully transmitted.
Extends/Includes	Prepare Route
Assumptions	- Videos, after processing, do not exceed the maximum transmission size.

Table 4- 8 Use Case 2 (Return Results/Notifications)

Use Case 3 (Load VioDet Model)

Use Case Name	Load VioDet Model
Primary Actor	Server
Secondary Actor	N/A
Normal Course	- Server loads the latest version of the VioDet Model, which has been set by the developer. - Dependencies for the running the model, including TensorFlow, Flask, Python and its libraries, are loaded.
Pre-Condition	The developer provided the correct path to the model.
Post-Condition	The Server gains access to the specific version of the model.
Alternate Course	Server was not able to gain access to the VioDet model.
Pre-condition	The developer did not provide the correct path.
Post-Condition	The VioDet model could not be accessed and consequently, violence detection could not take place.
Extends/Includes	N/A
Assumptions	- Developer provides path to the VioDet model which is most accurate in its performance.

Table 4- 9 Use Case 3 (Load VioDet Model)

Use Case 4 (Pre-Processing Video)

Use Case Name	Pre-Processing Video
Primary Actor	Server
Secondary Actor	N/A
Normal Course	- The video/link provided has been successfully accessed. - The video is successfully divided into frames based on the pre-defined fps parameter. - In case of live videos, a buffer storage is used to store few frames which are then processed.
Pre-Condition	The Server successfully gained access to the video/live stream.
Post-Condition	The video has been divided into frames.
Alternate Course	Server was not able to divide the video/link into frames.
Pre-Condition	The Server was unable to gain access to the videos as it maybe in an unsupported format or the live stream has gone offline.
Post-Condition	Server could not divide the videos into frames.
Assumptions	- Video/Link has been received at the endpoint.

Table 4- 10 Use Case 4 (Pre-Processing Video)

Use Case 5 (Run Prediction)

Use Case Name	Run Prediction
Primary Actor	Server
Secondary Actor	VioDet Model
Normal Course	- The VioDet model receives and loads the frames. - Global and spatio-temporal features are generated. - These features are then fed into the neural network.
Pre-Condition	The VioDet model and its dependencies were successfully loaded by the Server. And the video was successfully divided into frames.
Post-Condition	The video/live stream is undergoing violence detection.
Alternate Course	The VioDet model was unable to run prediction on the provided frames due to poor quality/noise.
Pre-Condition	The VioDet model and its dependencies were successfully loaded by the Server. And the video was successfully divided into frames.
Post-Condition	The target video/live stream did not undergo violence det.
Extends/Includes	Receive Frames, Analyze Frames.
Assumptions	- The environment for running the model was set-up. - The server does not run out of RAM/VRAM.

Table 4- 11 Use Case 5 (Run Prediction)

Use Case 6 (Return Results)

Use Case Name	Return Results
Primary Actor	Server
Secondary Actor	N/A
Normal Course	- The VioDet model is running and checking for violence in the video/live stream. - The model returns the 'violent' classification notification and resultant video. - These results are then transmitted to the Endpoint.
Pre-Condition	The video is undergoing violence detection by the model.
Post-Condition	The video/live stream is classified as violent and the subsequent results are transmitted to the endpoint.
Alternate Course	The VioDet model was unable to find violence in the videos/live stream.
Pre-Condition	The video is undergoing violence detection by the model.
Post-Condition	The target video/live stream did not contain any violence therefore, a 'non-violent' classification note, and video were sent to the endpoint.
Extends/Includes	Generates Final Classification
Assumptions	- The environment for running the model was set-up. - The server does not run out of RAM/VRAM while running predictions.

Table 4- 12 Use Case 6 (Return Results)

Use Case 7 (Receive Frames)

Use Case Name	Receive Frames
Primary Actor	VioDet Model
Secondary Actor	Server
Normal Course	- The VioDet model is successfully loaded and waiting for input.
	- The Server inputs the frames according to the pre-defined batch size. - VioDet model checks if RAM/VRAM enough to start processing.
Pre-Condition	The VioDet model has been successfully loaded.
Post-Condition	The frames were received successfully.
Alternate Course	The VioDet model was unable to accept all of the frames.
Pre-Condition	The VioDet model has been successfully loaded.
Post-Condition	The model runs prediction on a part of the video, and not the entire video.
Extends/Includes	N/A
Assumptions	N/A

Table 4- 13 Use Case 7 (Receive Frames)

Use Case 8 (Analyze Frames)

Use Case Name	Analyze Frames
Primary Actor	VioDet Model
Secondary Actor	Server
Normal Course	- The VioDet model accepts the number of frames as per the batch size. - All the frames are analyzed and feature data for each frame is generated and stored. - This feature data is then fed into the neural network.
Pre-Condition	The model successfully receives the frames.
Post-Condition	The feature data for all of the frames is generated and fed to the model.
Alternate Course	The VioDet model was unable to generate features data for the frames due to poor resolution of frame images.
Pre-Condition	The model successfully receives the frames.
Post-Condition	Feature data could not be generated for the frames.
Extends/Includes	N/A
Assumptions	- The environment for running the model was set-up. - The server does not run out of RAM/VRAM while running predictions.

Table 4- 14 Use Case 8 (Analyze Frames)

4.5 Detailed Description of Components

This section describes in detail all the components of VioDet. Each of these components have separate responsibilities. These components have been separated into two main modules, Web Application and Server Side.

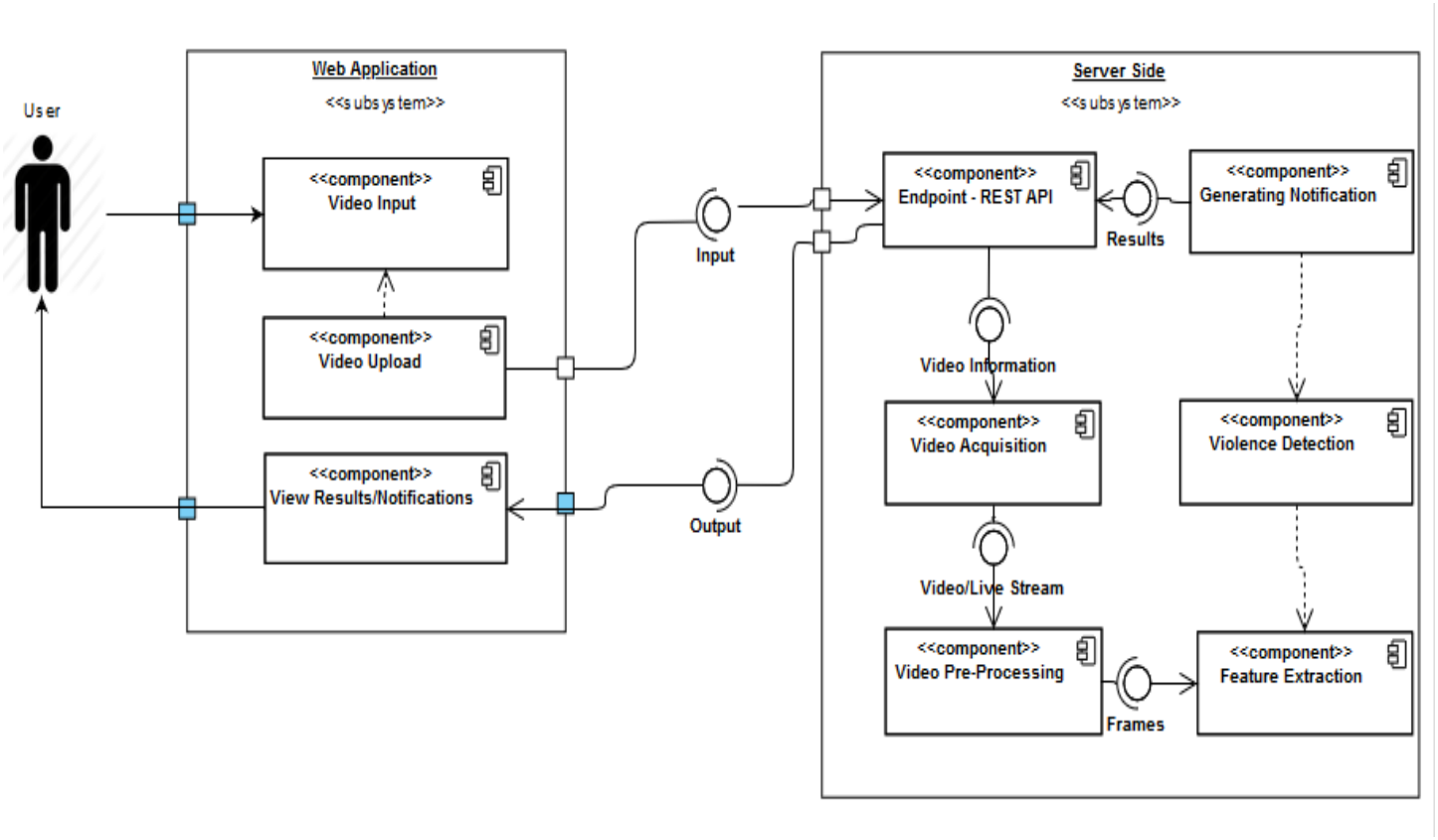

Figure 4 – 15 Component Diagram

4.5.1 Web Application Module

This module performs all the front-end tasks for VioDet system which includes receiving video input from user, uploading it to server and displaying the results. This module provides the base for successful working of the server module.

4.5.1.1 Video Input

Identification	Name: Video Input Location: Web Application Module
Type	Component
Purpose	This component fulfils following requirement from Software Requirements Specification Document: **Video Input** **Requirement** The system shall be able to acquire videos from the directory's address or link entered by user. [4.2.3.1 – SRS] **Description** This feature enables the system to receive video input from the user. This input will form the basis for violence detection when it is sent over to the server.
Function	This component of system interfaces with user to obtain the video/link for further processing.
Subordinates	It has one subordinate: Video Upload: Req. [4.3.3.2]
Dependencies	This component is independent module and runs in parallel to entire application.
Interfaces	This component has following interfaces: **Camera Interface:** For getting video input as live feed from camera.

Resources	**Hardware:** Camera, Desktop Computer, LAN/WiFi connection. **Software:** Web Browser (Chrome, Firefox, IE)
Processing	Video Input component would receive downloaded video/live stream which will be used for further processing.
Data	This component uses following information of the application: - Time of upload by the user.

Table 4- 16 Component Description (Video Input)

4.5.1.2 Upload Video

Identification	Name: Upload Video Location: Web Application Module
Type	Component
Purpose	This component fulfils following requirement from Software Requirements Specification Document: **Upload Video** **Requirement** The system shall be able to successfully transmit the user input (video/link) to the server. [4.3.3.2 – SRS] **Description** This feature is basically the next obvious step of video input. It is responsible for interacting with the REST API to transmit the information.
Function	This component of system interfaces with the REST API to send information with the server.
Subordinates	None.

Dependencies	This component is dependent on:- Video Input Component.
Interfaces	None.
Resources	**Hardware:** Desktop Computer, LAN/WiFi connection. **Software:** Web Browser (Chrome, Firefox, IE)
Processing	Upload video component would be responsible for transmitting the user input to the server.
Data	This component uses following information of the application: - Time of upload by the user.

Table 4- 17 Component Description (Upload Video)

4.5.1.3 View Results/Notifications

Identification	Name: View Results/Notifications. Location: Web Application Module
Type	Component
Purpose	This component fulfils following requirement from Software Requirements Specification Document: **View Results** **Requirement** The web application shall be able to receive the results/notifications from server and display them for the user to see. [4.5.3.2 – SRS] **Description** This feature is responsible for receiving the classification result and the video (after it has undergone detection) from the server and display them.

Function	This component of system interfaces with the REST API to receive the results from the server.
Subordinates	None.
Dependencies	This component is dependent on:- Acquire Video Component.
Interfaces	None.
Resources	**Hardware:** Camera, Desktop Computer, LAN/WiFi connection. **Software:** Web Browser (Chrome, Firefox, IE)
Processing	Display Results/Notifications component would be in charge of displaying the video classification note and the processed video.
Data	This component uses following information of the application: - Time of upload by the user, Time on which results received.

Table 4- 18 Component Diagram (View Results/Notifications)

4.5.2 Server-Side Module

This module performs all the back-end functionalities related to pre-processing of videos, feature extraction and violence detection. Feature extraction is the main input for violence detection component.

4.5.1.1 Endpoint – REST API

Identification	Name: Endpoint – REST API Location: Server Side Module
Type	Component
Purpose	This component fulfils following requirement from Software Requirements Specification Document: **Endpoint – REST** **Requirement** This component is responsible for communicating to and fro with the web application. **Description** This component is the main contact point between the web application and the server-side. It receives the video/link from the application and return the results after processing.
Function	This component of system interfaces with Video Upload and Display Results components of the application.
Subordinates	None.
Dependencies	None.
Interfaces	None.

Resources	**Hardware:** Desktop Computer, LAN/WiFi connection. **Software:** Flask (Web microframework)
Processing	Endpoint – REST API component would receive videos from and send results to the application.
Data	This component uses following information of the application: - Video/Live Stream Link, Classification Note and Processed Video.

Table 4- 19 Component Description (REST API)

4.5.2.2 Pre-Processing Video and Feature Extraction

Identification	Name: Pre-Processing Video and Feature Extraction Location: Server side Module
Type	Component
Purpose	This component fulfils following requirement from Software Requirements Specification Document: **Video Pre-Process and Feature Extraction Requirement** System shall be able to effectively divide video in to frames as per the specified fps parameter. [4.3.3.1 – SRS] The model shall be able to extract feature data for each frame. [4.4.3.2 – SRS] **Description** This component is responsible for the dividing the video into required number of frames so that they undergo feature extraction. This component would basically perform feature extraction on frames. These features will be of global and spatio-temporal nature.

Function	This component of system interfaces with the Video Acquisition module to obtain the video.
Subordinates	Violence Detection.
Dependencies	This component is independent module, however, it depends on input from Video Acquisition component.
Interfaces	None.
Resources	**Hardware:** Desktop Computer, LAN/WiFi connection. **Software:** Docker, Python
Processing	Video Pre-Processing component is responsible for preparing the video for feature extraction stage. Feature Extraction component would process the incoming frames and store the feature data for each frame. This is then fed to the VioDet model.
Data	This component uses following information of the application: - Video/Live Stream, fps.

Table 4- 20 Component Description (Preprocessing Video and Frame Extra.)

4.5.2.5 Violence Detection

Identification	Name: Violence Detection Location: Server side Module
Type	Component
Purpose	This component fulfils following requirement from Software Requirements Specification Document: **Violence Detection** **Requirement** The model shall be able to receive and load the feature data for

	each frame. [4.4.3.1 – SRS] The model shall be able to assess the features data and based on it, give a final classification that whether the video is violent or not. [4.5.3.1 – SRS] **Description** This major component covers two requirements. One is that the model loads the feature data for each of the frames into the neural networks. And assess that data to classify the video.
Function	This component of system interfaces with the Feature Extraction component to obtain the feature data.
Subordinates	Generating Notification
Dependencies	This component is dependent on the Feature Extraction component for the feature data.
Interfaces	None.
Resources	**Hardware:** Desktop Computer, LAN/Wi-Fi connection. **Software:** Docker, Python, TensorFlow.
Processing	Violence Detection is performed by the VioDet model which is basically a set of neural networks that perform inference based on the feature data.
Data	This component uses following information of the application: - Feature Data for Frames, VioDet model.

Table 4- 21 Component Description (Violence Detection)

4.5.2.6 Generating Notification

Identification	Name: Generating Notification Location: Server side Module
Type	Component
Purpose	This component fulfils following requirement from Software Requirements Specification Document: **Violence Detection** **Requirement** The model shall be able to transmit the results back to the application. [4.5.3.2 – SRS] **Description** This last component is responsible for producing the relevant notification based on the classification done by the model. And then send that notification to the application through the Endpoint.
Function	This component of system interfaces with the Violence Detection component to obtain the classification data.
Dependencies	This component is dependent on the Violence Detection component for the classification data.
Resources	**Hardware:** Desktop Computer, LAN/Wi-Fi connection. **Software:** Docker, Python.
Processing	Generating Notification properly formats the response which includes the classification note and the resultant video and sends it to the Endpoint for it to be transmitted to application.
Data	This component uses following information of the application: - Classification data, Resultant video.

Table 4- 22 Component Description (Generating Notification)

4.6 Reuse and Relationships to other Products

VioDet is a mechanism to detect violence in videos. As far as this project is concerned, we have developed VioDet as a standalone service application that can help its users to automatically detect any violent scenes.

However, these days the concept of micro-services is rapidly trending in the market/industry and VioDet can be used as a micro-service on top of a much bigger and more extensive platform related to Video Content Management.

Furthermore, VioDet application can be used as a basis for a browser extension/plugin that would detect violence in any video content displayed on the screen.

4.7 Design Rationale

Mainly, there are two main modules, web application and server-side. Web application is responsible for interacting with the user while server-side handles the backend.

Therefore, it is suitable that we use Model View Control (MVC) as the design pattern. Model is basically the machine learning model that takes a video as input and gives the classification as output. View is basically used for responding to the browser's request. Controller is the main connector component that connects the browser to the model and the model to the view.

Clearly, components can do their work independently, but, in a certain flow (data as well as control). That led us to high cohesion. Moreover, component don't have much interaction, once a component has completed its work, it will communicate its state to the other component. Consequently, that component will come into action. That led us to low coupling.

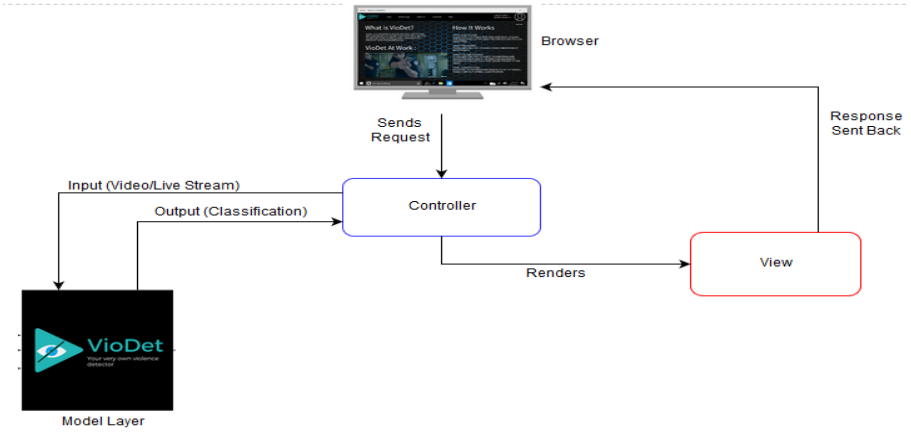

Figure 4 – 16 MVC in VioDet

4.8 Requirement Matrix

Functional Requirements	Components
4.2.3.1 The system shall be able to acquire videos from the directory's address or link entered by user	**Video Input** (This component of system interfaces with user to obtain the video/link for further processing.)
4.3.3.2 The system shall be able to successfully transmit the user input (video/link) to the server	**Upload Video** (This feature is basically the next obvious step of video input. It is responsible for interacting with the REST API to transmit the information.)
4.5.3.2 The web application shall be able to receive the results/notifications from server and display them for the user to see.	**View Results/Notifications** (This feature is responsible for receiving the classification result and the video (after it has undergone detection) from the server and display them.)
4.3.3.1 System shall be able to effectively divide video in to frames as per the specified fps parameter.	**Pre-Processing Video** (This component is responsible for the dividing the video into required number of frames so that they undergo feature extraction.)
4.4.3.2 The model shall be able to extract feature data for each frame.	**Feature Extraction** (This component of system interfaces with the Video Pre-Processing module to obtain the frames.)
4.4.3.1 The model shall be able to receive and load the feature data for each frame. **4.5.3.1** The model shall be able to assess the features data and based on it, give a final classification that whether the video is violent or not.	**Violence Detection** (This major component covers two requirements. One is that the model loads the feature data for each of the frames into the neural networks. And assess that data to classify the video.)

Table 4- 25 Requirement Matrix

Chapter 5. Project Test and Evaluation

5.1 Introduction

This document provides the test documentation for the project VioDet 1.0, that will facilitate the technical tasks of testing including the detailed test cases for black box testing. Each test case specifies who will be performing the test, the preconditions required to execute each test case, the specific item to be tested, the input, expected output or results, and procedural steps where applicable.

5.2 Test Objectives

The objective of this document is to expand on the test plan and provide specific information needed to perform the necessary tests. By providing detailed test information, important modules will not be overlooked, and test coverage will improve. Testers will be able to use each test case provided in this document to move forward and begin testing

5.3 Test Items

Based on the requirements of VioDet (v. 1.0), mentioned in Section 3 of this document, following are the major modules/ functionalities that should be considered during the testing process: -

1. Video Input
2. Upload Video/Transmit Frames
3. Video Preprocessing
4. Feature Extraction
5. Violence Detection
6. Display Results/Notifications

5.4 Features to Be Tested

Following features are being tested:

1. The system shall be able to acquire videos from the directory's address or link entered by user.
2. The system shall be able to successfully transmit the user input (video/link) to the server.
3. System shall be able to effectively divide video in to frames as per the specified fps parameter.
4. The model shall be able to extract feature data for each frame.
5. The model shall be able to assess the features data and based on it, give a final classification that whether the individual frame is violent or not.

6. The server then transmits the classified frames to the web application which should be to successfully receive it.
7. The Web application shall be able to display the results coming in from the server.

5.5 Detailed Test Strategy

The project VioDet is a computationally intensive system that is why systems modules should be developed independently and then these modules should be integrated. Overall strategy comprises of Unit Testing using White Box and Black box testing. Integration testing is performed in order to successfully integrate the system.

5.5.1 Unit Testing

Unit Testing is done at the source or code level for language-specific programming errors such as bad syntax, logic errors, or to test functions or code modules. The unit test cases shall be designed to test the validity of the program's correctness.

5.5.2 White box Testing

In white box testing, the UI is bypassed. Inputs and outputs are tested directly at the code level in functions and the results are compared according to requirements. This form of testing ignores the function of the program under test and will focus only on its code and the structure of that code. The test cases that have been generated shall cause each condition to be executed at least once. To ensure this happens, we are applying Basis (alternative) Path Testing. Because the functionality of the program is relatively simple, this method will be feasible to apply.

5.5.3 Black Box Testing

Black box testing typically involves running through every possible input to verify that it results in the right outputs using the software as an end-user would.

5.5.4 Integration Testing

Integration testing is the part where we will test all the previous tested modules in a way that they are functioning normally when they are combined.

5.5.5 Incremental Testing

There are four primary modules that are required to be integrated. These components, once integrated, will form the complete application testing. The following describes these modules as well as the steps that will need to be taken to achieve complete

integration. We will be employing an incremental testing strategy to complete the integration. The integration testing will be performed by the development team.

1) Video Acquisition and Transmission Module

This is the module from where the major functioning of application initiates. This module is responsible for allowing the user to either upload a video from his/her local directory or to give access to the video feed from one of cameras attached to his device. Once the video source is acquired, the next step is to transmit it to the server. This has two parts to it, a complete video from the local directory is uploaded to the server end in its entirety while a camera feed is continuously transmitted to the server frame-by-frame for as long as the user wants to. This module is developed independently and tested first separately, then it is combined with the Video Processing module.

2) Video Processing Module

Video Processing module comes into action when a complete video has been uploaded to the server. This module accesses that video and breaks it down into frames based on the specified fps parameter. It is necessary because the violence detection model takes input frame-by-frame so for that to be successfully executed, the video needs to be divided into frames. This module is developed independently and tested first separately, then it is combined with the Violence Detection Module.

3) Violence Detection Module

The core functionality of this project takes place at this module which is to detect violence in each video frame. Before detection, feature data is extracted for each frame and feature maps are generated which are then fed to the model. The utilizes this data to determine whether the frame is to be classified as violent or non-violent. The model then returns the result as a frame with red(violent) or green(non-violent) borders. This module is developed independently and tested first separately, then it is combined with the Video Processing, Display Results modules.

4) Display Results/Notifications Module

The output from the model is used as input for this module which is responsible for transmitting the results back to the web application and to display it there. This module must continuously perform this function if the frame images keep coming. This module is developed alongside Violence Detection and tested with it as well.

5.6 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

5.7 Suspension Criteria and Resumption Requirements

Testing will be suspended when a defect is introduced/found that cannot allow any further testing. Testing will be resumed after defect removal.

5.8 Test Deliverables

Following are the test cases

Test Case 1 - Acquire Video/Camera Feed

Test case name	Acquire Video/Camera Feed
Test Case Number	1
Description	This test case is aimed at checking the video acquisition functionality of VioDet. The user should able to choose a video (.mp4,.avi) from local directory or in case of live detection, the application is able to access the video feed from one of the device's camera.
Testing Technique used	White Box Testing
Preconditions	There are some videos downloaded on the device and (or) a camera is attached to it.
Input	User provides path for the video or selects one of the attached

	cameras.
Steps	The videoupload/livedetection function is executed by calling it through buttons on a simple HTML page which are linked to relevant Flask routes
Expected output	The functions return the video that was selected by showing it simply in a video tag on the same web page.
Alternative path	**Cause:** Video was not found or was of a not supported format. In case of live detection, no camera was found. **Corresponding Output:** Error Message Displayed.
Actual output	Confirmed

Table 5- 1 Test Case 1 (Acquire Video/Camera Feed)

Test Case 2 – Video/Frames Transmission

Test case name	Video/Frames Transmission
Test Case Number	2
Description	This test case is designed to check whether server is able to receive a video or to continuously receive frames from the client's device camera.
Testing Technique used	White Box Testing
Preconditions	Test Case 1 is satisfied. Client and Server are connected to the local network/internet.
Input	Video/Frames
Steps	In case of a simple video, it is saved on the server side in a designate folder. For live video feed, frames are being accepted on run-time.

Expected output	The video/camera feed is displayed on a simple web page.
Alternative Path	N/A
Actual output	Confirmed

Table 5- 2 Test Case 2 (Video/Frames Transmission)

Test Case 3 - Video Processing

Test case name	Video Processing
Test Case Number	3
Description	This test case is used to check the functionality of dividing the complete video into frames (1 frame per second).
Testing Technique used	White Box Testing
Preconditions	A complete video is uploaded. Test Cases 1 and 2 are satisfied.
Input	A complete video.
Steps	Using OpenCV's read function, we divide the video into frames.
Expected output	A frame each second.
Alternative Path	N/A
Actual output	Confirmed

Table 5- 3 Test Case 3 (Video Processing)

Test Case 4 – Feature Extraction

Test case name	Feature Extraction
Test case number	4
Description	Generating feature maps for each frame that is to be classified.
Testing Technique used	White Box Testing
Preconditions	Test cases 1, 2 and 3 are satisfied.
Input	Frame Image.
Steps	First each frame image is converted into 224x224x3 (IMAGE_SIZE x IMAGE_SIZE x IMAGE_CHANNELS). Then, we basically extract features such as contours, shapes, colors to generate a feature map.
Expected output	Feature Map
Alternative Path	N/A
Actual output	Confirmed

Table 5- 4 Test Case 4 (Feature Extraction)

Test Case 5 – Violence Detection

Test case name	Violence Detection
Test Case Number	5
Description	This test case checks how the model performs and gives its output.
Testing Technique used	Black Box Testing

Preconditions	Test cases 1, 2, 3 and 4 are satisfied.
Input	Feature Maps (224x224x3). Weight Variables.
Steps	Feature Maps are fed into the model
Expected output	A final classification is generated for each frame (1 for violent and 0 for non-violent).
Alternative Path	In case, a feature map is not of the specified size, that frame is skipped.
Actual output	Confirmed

Table 5- 5 Test Case 5 (Violence Detection)

Test Case 6 – Result Preparation

Test case name	Results Preparation
Test case number	6
Description	Here, the functionality of preparing the video frames to be displayed, is tested.
Testing Technique used	White Box Testing
Preconditions	All previous test cases are satisfied.
Input	Output from model (0 for non-violent and 1 for violent)
Steps	In case of 1, a red border is added to the frame. In case of 0, a green border is added to the frame.
Expected output	Frames with red/green borders.

Alternative Path	N/A
Actual output	Confirmed

Table 5- 6 Test Case 6 (Result Preparation)

Test Case 7 – Result Transmission and Display

Test case name	Result Transmission and Display
Test case number	7
Description	In this test case, the final leg of the system is scrutinized which is the display of the resulting video frames after they have undergone violence detection.
Testing Technique used	White Box Testing
Preconditions	Previous test cases are satisfied. Client and Server are connected to the local network/internet.
Input	Prepared frames (with green/red border)
Steps	Obtain those frames and transmit them using a video streaming route which was developed using Flask.
Expected output	The resulting frames are displayed on the user interface at the client side.
Alternative Path	N/A
Actual output	Confirmed.

Table 5- 7 Test Case 7 (Result Transmission and Display)

Test Case 8 – Calculate Violence Percentage

Test case name	Calculate Violence Percentage
Test Case Number	8
Description	The unsmoothed result array stores a true/false for each frame.
Testing Technique	White Box Testing
Preconditions	Unsmoothed result array is returned each time violence detection is performed.
Input	Unsmoothed result array.
Steps	Calculate the size of the array. Calculate the number of True entries in the array. Determine the percentage using TrueEntries/TotalEntries.
Expected output	Percentage of violent frames in the video.
Alternative Path	N/A
Actual output	Confirmed

Table 5- 8 Test Case 8 (Calculate Violence Percentage)

Test Case 9 – Website Interface

Test case name	Website Interface.
Test Case Number	9
Description	This module is related to the design of dynamic and responsive user interface of android application.
Testing Technique used	Black Box Testing

Preconditions	Internet is available
Input	Program Initiation will be treated as input to the system
Steps	Create and layout the widgets.
Expected output	All widgets and other elements are correctly displaying.
Alternative Path	N/A
Actual output	Confirmed

Table 5- 9 Test Case 9 (Website Interface)

5.9 Environmental Needs

Hardware

The VioDet Application would require GPU workstation for training the model and a standard for the client to run the application on:

- **GPU Workstation (for training)**: - Processor i7 + NVIDIA GTX 1060(or better).
- **Standard Desktop (basic inference): -** Processor i3 or advanced and 4GB+ RAM

Software

- Anaconda (Python 3.6)
- Deep Learning Framework: - Keras + Tensorflow
- Latest NVIDIA GPU Driver for GTX 1060.
- CUDA 6.1, cuDNN 7.6
- IDE: - Jupyter Notebook
- Python Libraries: - numpy, sklearn, opencv-python, keras, tensorflow-gpu, pillow.
- For Application Dashboard: - HTML, CSS, JS, Bootstrap, ReactJS, JSON.

5.10 Responsibilities, Staffing and Training Needs

Responsibilities

All developers of the project are responsible for the completion of all units testing and integration testing tasks.

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.

5.11 Risks and Contingencies

Efforts have been made to remove all and every chance of failure but there are certain unpredictable factors such as network issues, corrupt input data, or system failure that may lead to some issues. Error handling will be applied more deeply to cover all these issues, but unforeseen circumstances may happen.

Schedule Risk

The project might get behind schedule. So, in order to complete the project on time, we will need to increase the hours/day.

Budget Risk

The budget will be compensated by using less costly alternatives to fit the budget requirements.

Chapter 6. Future Work

VioDet is a mechanism to detect violence in videos. As far as this project is concerned, we have developed VioDet as a standalone service application that can help its users to automatically detect any violent scenes. Taking into account the increasing role of AI in our day-to-day lives, one can foresee multiple way in which VioDet can be improved/upgraded: -

1. The accuracy of the system can be improved subject to the availability of public datasets.
2. VioDet can easily be deployed using the cloud so that it can be accessed from anywhere.
3. VioDet can be modified to include a facial recognition feature that would allow troublemakers to be identified.
4. Furthermore, VioDet application can be used as a basis for a browser extension/plugin that would detect violence in any video content being run.
5. Moreover, these days the concept of micro-services is rapidly trending in the market/industry and VioDet can be used as a micro-service on top of a much bigger and more extensive platform such as a Video Management System.

Chapter 7. Conclusion

7.1 Overview

In conclusion, this comprehensive document elaborates all facets of the development of VioDet from initial requirement gathering to final testing. VioDet is a stand-alone web-service that provides us with an effective and accurate video violence detection. VioDet is a product of the latest computation capabilities, transfer learning and web development. It has high potential to be further developed into a solution that offers more features and is more accurate and robust.

7.2 Objectives achieved

1. Using deep learning to detect violence instead of simple computer vision.
2. Detection of violence in videos and real-time camera feeds.
3. Easy and efficient monitoring of security cameras/video content online.
4. Reduces the need for security camera operators.
5. Provides a platform that can be easily integrated with content management solutions.

Bibliography

[1] http://joshua-p-r-pan.blogspot.com/2018/05/violence-detection-by-cnn-lstm.html.

[2] https://dzone.com/articles/video-analysis-to-detect-suspicious-activity-based.

[3] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6169868/.

[4] **Peixoto, Bruno & Avila, Sandra & Dias, Zanoni & Rocha, Anderson. (2018). "Breaking down violence: A deep-learning strategy to model and classify violence in videos."** https://www.researchgate.net/publication/327002778_Breaking_down_violence_A_deep-learning_strategy_to_model_and_classify_violence_in_videos.

[5] **Sudhakaran, Swathikiran, and Oswald Lanz. "Learning to detect violent videos using convolution long short-term memory."** In Advanced Video and Signal Based Surveillance (AVSS), 2017 14th IEEE International Conference on, pp. 1-6. IEEE, 2017. https://ieeexplore.ieee.org/abstract/document/8078468.

[6] **W. Song, D. Zhang, X. Zhao, J. Yu, R. Zheng and A. Wang, "A Novel Violent Video Detection Scheme Based on Modified 3D Convolutional Neural Networks," in *IEEE Access*, vol. 7, pp. 39172-39179, 2019.** https://ieeexplore.ieee.org/document/8669768.

Appendices

Appendix A. Glossary

Transfer Learning:	Transfer learning is the improvement of learning in a new task through the transfer of knowledge from a related task that has already been learned.
Flask:	Python-based micro web framework.
React:	An open-source JavaScript library for building user interfaces.
CNN:	Convolutional Neural Network
LSTM:	Long-Short Term Memory network
API:	Application Programming Interface
OS:	Operating System
App.:	Application

Appendix B. Work Plan

Phase #	Activities	Time Frame	Anticipated Outputs	Reporting
1	• Requirement gathering using different tools • Classifying Functional and non- functional requirements • Discussing the security requirements and the standards to be followed • Model Selection • Dataset Collection	15th Sept 2019-31st Oct 2019	• SRS Document • Base Model • Sufficient Data	• Project Supervisor • FYP Evaluation Committee
2	• Transfer Learning • Running Inference • Designing features and requirements of the system • Logical representation of the system. • Further Training of Model.	1st Nov 2019 – 10th Feb 2020	• Software Design Document (SDD) • Application Dashboard Prototype • Model Inference.	• Project Supervisor • FYP Evaluation Committee
3	• Application Backend Development (streaming server). • Application Frontend Development • Checking model connectivity while on server.	15th Feb 2020-30th Mar 2020	• Streaming Server. • Running inference through web.	• Project Supervisor
4	• Integration of frontend and backend. • Writing test cases. • Unit testing • Integration Testing • System Testing.	1st April 2020-28th May 2020	• VioDet Application (Alpha version)	• Project Supervisor • FYP Evaluation Committee
5	• Project Evaluation (by deploying it) • Verification and Validation • Final Touches • Documentation	1st June 2020 – 15th June 2020	• Thesis • VioDet Application (Beta version)	• Project Supervisor • FYP Evaluation Committee (External)