Rare Event Detection in Machine Vision   
using Deep Learning

Software Requirements Specification

Version 1.0

Group 16

Roberta Beaulieu,Rakeem Durand,Thomas Fuller,Ihsan Hashem,  
Karim Rattani

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 02/13//2018 | 1.0 | Added use case diagram, written use case | Karim Rattani |
| 02/13/2018 | 1.0 | Introduction, Overview,Overall Description | Roberta Beaulieu |
| 02/14/2018 | 1.0 | Modified use case diagram | Karim Rattani |
| 02/15/2018 | 1.0 | Added Performance and other requirements | Ihsan Hashem |
| 02/15/2018 | 1.0 | Added requirements and analysis classes | Rakeem Durand |
| 02/15/2018 | 1.0 | Added sequence, object collaboration and object behavior diagrams | Thomas Fuller |
| 02/16/2018 | 1.0 | Modified Formatting | Thomas, Karim |

Table of Contents

[**Introduction**](#_a0vs4fzbbccz) **4**

[Purpose](#_xtklabbcpkas) 4

[Scope](#_d6tq1oh5fzho) 4

[Definitions, Acronyms and Abbreviations](#_9nyfwh90kd7n) 4

[References](#_mo5omnel9sg4) 4

[Overview](#_qfol3m3rpt1f) 5

[**Overall Description**](#_aez8mh292set) **5**

[Use-Case Model Survey](#_9z2qo9lpap4p) 5

[Assumptions and Dependencies](#_b80b80q2uark) 10

[**Specific Requirements**](#_2u1vjcth8dyx) **10**

[External Interface Requirements](#_2hqtthm0uozd) 10

[UML Class Diagram](#_2s766aqqqkup) 11

[Analysis Classes](#_z0pxasrsik5i) 11

[Sequence Diagrams](#_x13ayjwg06mj) 13

[Object Collaboration Diagrams](#_ey41o428lru5) 15

[Object Behavior Diagrams](#_xtuhrmsv11y) 16

[Performance Requirements](#_3ok978zawx96) 17

[**Other Requirements**](#_ti9ode7v6cp6) **17**

[**Supporting Information**](#_zb2st2hmaek1) **17**

[Appendix I - Learning Sources](#_v4hj7mvnfk3m) 17

[Appendix II - Helpful Resources](#_xj9qcc5k9szi) 18

Software Requirements Specification

# Introduction

This section gives a scope description and overview of the information include in this Software Requirements Specification (SRS). As well, this document describes the purpose and includes a table with definitions, acronyms and abbreviations.

## Purpose

Surveillance cameras have become a part of our life. We see them in almost every corner of the street and outside restaurants. However, hiring people to monitor all the surveillance video is expensive. The purpose of our project is to implement an algorithm that will detect abnormal activity such as robbery, accident, etc. and alert the corresponding authorities.

## Scope

The scope of this project is to create a system that will detect anomalous event through images and videos using deep learning. In the subsequent release, we will design a classification method to classify the event. After that, we will design an alert system to alert the corresponding person about anomalous event

## Definitions, Acronyms and Abbreviations

**Table 1- Definitions**

|  |  |
| --- | --- |
| **Term** | **Definition** |
| CNN | Convolutional Neural Network |
| Neural Network | use for solving pattern recognition problem, involves large number of processes operating in parallel and arranged in tiers. |
| CNN | algorithm of Neural Network, analyze visual imagery. |
| Pixel | Smallest unit of picture |
| RGB | Red, Green, Blue Computer |
| Vision | algorithms for understanding of digital image |
| Anomalous | abnormal, unusual |

## References

[1] Kennesaw State University. (2018). Senior project course syllabus. Marietta, Georgia: Shi, Yong.

[2] Sultani, Waqas, et al. “Real-World Anomaly Detection in Surveillance Videos.” *Real-World Anomaly Detection in Surveillance Videos*, University of Central Florida, 12 Jan. 2018. Web.

[3] Stahl, Bernd Carsten. “Identifying Ethical Issues during the Development of a Computer Vision Based 2AmI System: A Case Study.” *University of Salzburg*, 2009.

## Overview

The rest of this document is organized in six sections. The second section provides the overall description of the project. This section also includes the use-case model and few written use cases. Further, this section also mentions the project assumptions and dependencies.

The third section describes in detail all the specification requirements that qualify designers to design a system to satisfy those requirements. Accordingly, this section is organized in these subsections: external interface requirements, analysis of classes, sequence diagrams, object collaboration diagrams, and object behavior diagrams.

The fourth section prioritizes in which performance requirements the system needs to achieve a robust performance.

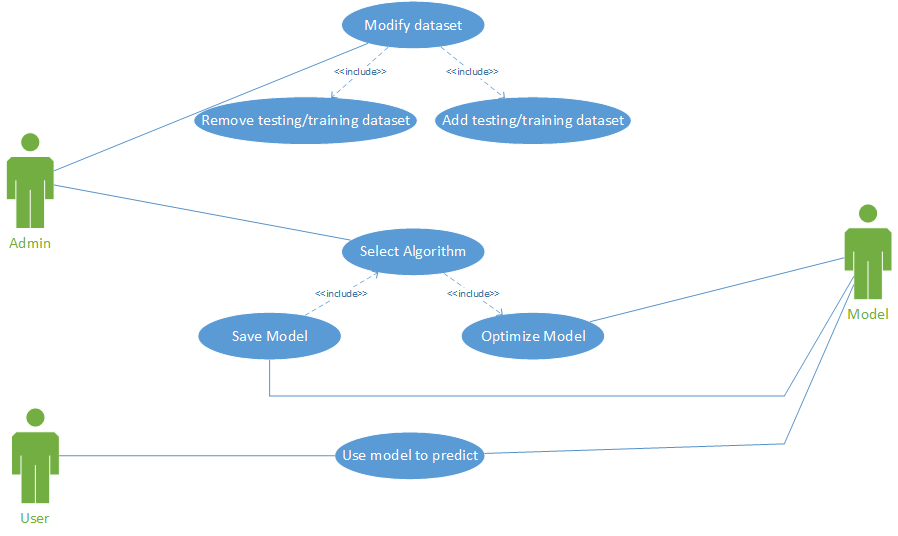
The fifth section discusses the reason why functional or non-functional that are not derived from the use-cases.

The sixth section supporting information includes a table of contents, index, and appendices. It also includes use-case storyboard, user-interface prototypes, and CRC cards.

# Overall Description

This section will give an overview of the whole system. The focus of this project is the implementation of an algorithm that distinguishes unusual activity in a public scenario. The detection of this abnormal events would be determined using deep learning in surveillance images and videos monitored by public cameras. The method proposed will send a notification to the corresponding authorities which could help to avoid dangerous public disturbances. At the end of this section, the assumptions and dependencies for the system will be presented.

## Use-Case Model Survey

* + 1. Use Case Diagram  
         
       
    2. **Written Use Case**  
       1. **Use Case 1**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use case name:**  Use model to get abnormality | | **ID:**  ABN-001 | | **Priority:**  HIGH |
| **Primary actor:**  User, Model | **Source:**  A user starts the program with his/her input | | **Use case type:**  Technical requirement | |
| **Brief description:**  Get the result of abnormality from the given video input and output the results. | | | | |
| **Precondition:**  A trained model should already be saved. The input should be of same context as trained model | | | | |
| **Typical flow of events:**  1. User starts the system.  2. User enter the name of video he/she wants to analyze on the console.  3. The system breaks the video into frames and compare each frame to the train model.  4. The system output the result of abnormality by using the model with input data to predict.  5. The system asks the user if the result is accurate.  6. If the user enters no, then the system will ask the user to input the time in HH:MM:SS format of the video which has abnormal behavior. The system will take the frames during that second and save that result as label data for future training purpose.  7. The system will exit. | | | | |

* + - 1. **Use Case 2**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use case name:**  Add training dataset | | **ID:**  ABN-002 | | **Priority:**  HIGH |
| **Primary actor:**  User, Model | **Source:**  A user starts the program with his/her input | | **Use case type:**  Technical requirement | |
| **Brief description:**  Add or remove the training dataset. | | | | |
| **Precondition:**  There should be some data in training dataset. | | | | |
| **Typical flow of events:**  1. User opens the excel file which has trained data.  2. User enters the new data. Data should have file location, file format, and label associated with it. File format can include JPEG for images, MPEG for videos, XML/XLSX for disk file format.  3. User runs the system to train the model.  4. The system runs the testing after training the model.  5. The system outputs the testing result.  6. The system prompts the user if he/she wants to save the model.  7. If the user enters yes, then system will select this model for predicting inputs and save the previous model in past models folder with name in ALGORITHM-MM-DD-YY-AccuracyOfTestingData format. If the user enters no, then the system will discard this model.  8. The system will exit. | | | | |
|  |  |  |  |  |

* + - 1. **Use Case 3**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use case name:**  Add or remove testing dataset | | **ID:**  ABN-003 | | **Priority:**  HIGH |
| **Primary actor:**  User, Model | **Source:**  A user starts the program with his/her input | | **Use case type:**  Technical requirement | |
| **Brief description:**  Add or remove the testing dataset | | | | |
| **Precondition:**  There should be some data in testing dataset | | | | |
| **Typical flow of events:**  1. User opens the excel file which has testing data.  2. User enters the new data. Data should have file location and file format. File format can include JPEG for images, MPEG for videos, XML/.XLSX for disk file format.  3. User runs the system to test the model.  4. The system outputs the testing result.  5. The system will exit. | | | | |
|  |  |  |  |  |

* + - 1. **Use Case 4**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use case name:**  Select algorithm | | **ID:**  ABN-004 | | **Priority:**  HIGH |
| **Primary actor:**  User, Model | **Source:**  A user starts the program with his/her input | | **Use case type:**  Technical requirement | |
| **Brief description:**  Choose an algorithm to get the model | | | | |
| **Precondition:**  There should be some data in training and testing dataset | | | | |
| **Typical flow of events:**  1. User runs the system with select algorithm flag.  2. The system will prompt the user to input number from the given list. Each number will correspond to the algorithm. The user will enter the number depending upon the algorithm he/she wants.  3. The system runs the training using the algorithm specified by user and using hyperparameters from the range of values defined by the system.  4. The system runs the testing from the model.  5. The system outputs the testing result.  6. The system prompts the user if he/she wants to save it.  7. If the user enters yes, then system will select this model for predicting inputs and save the previous model in past models folder with name in ALGORITHM-MM-DD-YY-AccuracyOfTestingData format. If the user enters no, then the system will discard this model.  8. The system will exit. | | | | |

## 

## Assumptions and Dependencies

An assumption about the system is that there will be open source labelled and unlabelled data available based on surveillance camera recorded videos. Our training system should have enough processing speed to train the model on time.

In case that the project is delayed, there are some requirements that can be transferred to the next version of the system.

# Specific Requirements

This section contains all of the functional and quality requirements of the system. It gives a detailed description of the system and all its features.

## External Interface Requirements

This section provides a brief description on what different interfaces that will be involved in the system. The different interfaces include User, Hardware, Software and communications.

* + 1. **User Interfaces**

The only user interface that will be used during this project is the command line, end-users will not be interacting with the system. The system is meant to predict anomalies based on data given, only admin access will be permitted to modify the data through the command line.

* + 1. **Hardware Interfaces**

Since our deep learning model will not have any designed hardware, it does not have any direct hardware interfaces. In the subsequent release however, we will use HD cameras to detect anomalies using the predictive model.

* + 1. **Software Interfaces**

The two main software framework will be Keras and Tensorflow. Keras is a neural network api written in python, which will work with Tensorflow to compute the model. Our main integrated development environment will be Pycharm, in which we will implement our code.

* + 1. **Communications Interface**

The only communication is between the admin/user and the system. System will be accessed through PyCharm IDE.

## UML Class Diagram

## Analysis Classes

|  |  |
| --- | --- |
| **Class name:** Anomaly Detection | |
| **Brief description:** Determines if an anomaly is present within the data | |
| **Attributes (fields)** | **Attribute Description** |
| Data entry | 10 consecutive frames that are extracted from the video input we are detecting from. |
| **Methods (operations)** | **Method Description** |
| Detect Anomaly | Return whether an anomaly was found or not by running the input frames through our trained model. |
| Results | Prints out the result of the Anomaly Detection function. |

|  |  |
| --- | --- |
| **Class name:** Algorithm Selection | |
| **Brief description:** Allows user to use a different algorithm for detection | |
| **Attributes (fields)** | **Attribute Description** |
| Current Algorithm | Stores the current Algorithm that is chosen  to use in training and detecting. |
| Algorithms list | Stores a list of available algorithms depending on the user preference. |
| **Methods (operations)** | **Method Description** |
| Test Algorithm | Tests the algorithm using test data to make sure the user got their intended results. |
| Change Algorithm | Changes the current algorithm to the new algorithm |
| Save Model | Saves the latest model for the specific algorithm and overwrites a previous trained model if it exists. |

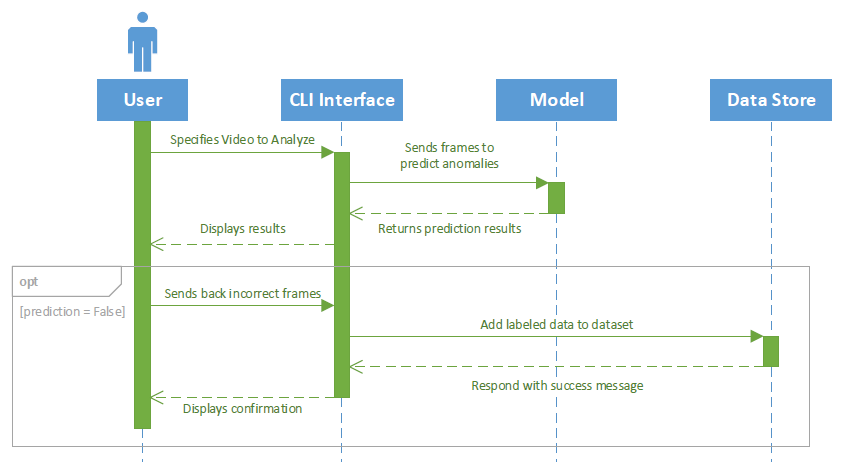
|  |  |
| --- | --- |
| **Class name:** Dataset | |
| **Brief description:** Allows modification to training and testing datasets | |
| **Attributes (fields)** | **Attribute Description** |
| Current Dataset | Contains the current dataset |
| Conditional Variable | Stores the decisional value for modifying the dataset |
| Training Set | Stores the portion of data that is used for training. |
| Testing Set | Stores the portion of the data that is used for testing. |
| Validation Set | Stores the portion of the data that is used for validation. |
| **Methods (operations)** | **Method Description** |
| Add to Data | Adds additional data to the current dataset |
| Remove Data | Removes data from the dataset |
| Divide data | By default data entries extracted from data set should be randomized and divided into 70% training set, 15% testing set, and 15% validation set. Percentages can be changed if indicated in the method parameters. |

|  |  |
| --- | --- |
| **Class name:** Main | |
| **Brief description:** Combines all the classes for a smoother interaction | |
| **Attributes (fields)** | **Attribute Description** |
| Algorithm | Algorithm object |
| Data | Dataset object |
| Detection | Anomaly Detection object |
| Conditionals | Can use multiple of this type. Used for determining what the user wants in each section, choosing if they want to modify the algorithm or modify the dataset. |

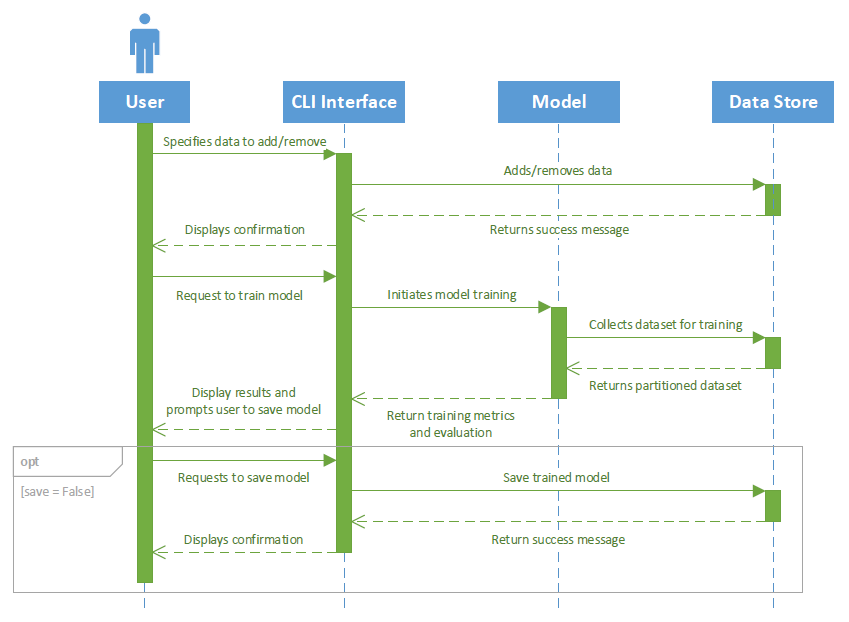
## Sequence Diagrams

The following section includes a number of sequence diagrams, one for each Use Case, outlining the flow of all interactions within the system.

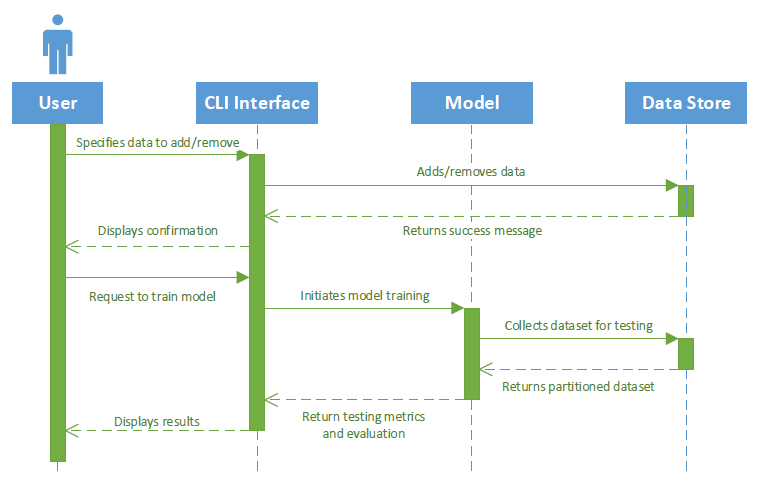
* + 1. **Sequence Diagram for Use Case 1**

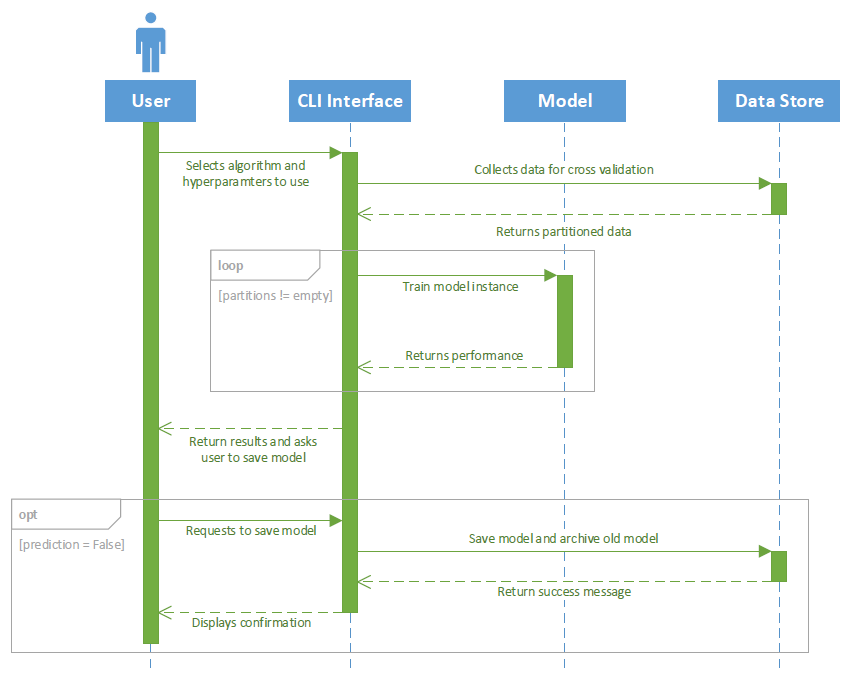
****

* + 1. **Sequence Diagram for Use Case 2**

****

* + 1. **Sequence Diagram for Use Case 3**

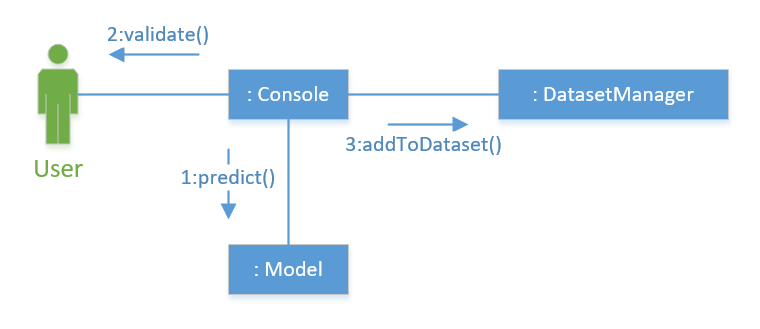
****

* + 1. **Sequence Diagram for Use Case 4**

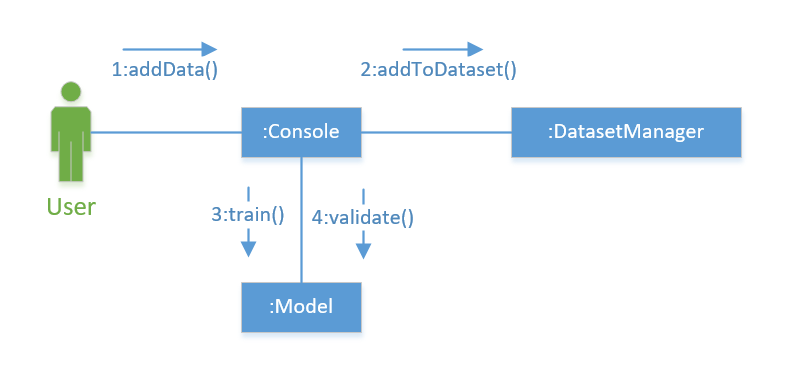
## Object Collaboration Diagrams

The following section includes two object collaboration diagrams that help to further describe some of the use cases.

* + 1. **Object Collaboration Diagram for Use Case 1**

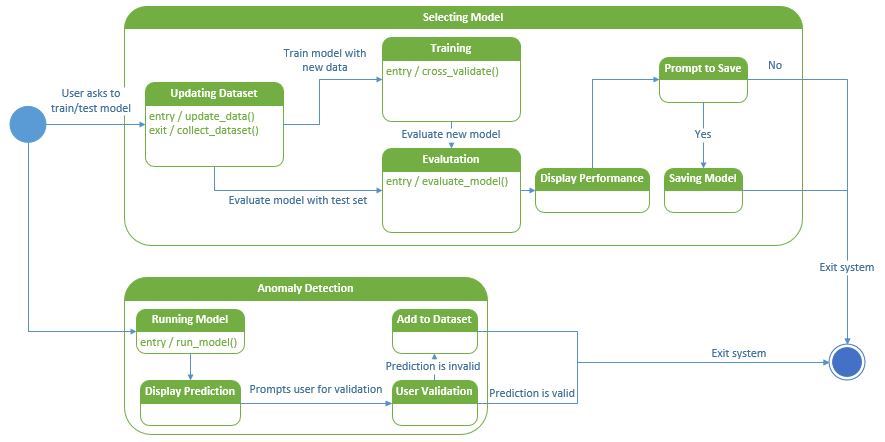
****

* + 1. **Object Collaboration Diagram for Use Case 2 and Use Case 3**

****

## Object Behavior Diagrams

The following section includes a large UML state diagram that outlines the state changes throughout the life of the system. There are two main states of the system: Anomaly Detection (used for predicting the presence of anomalies in video) and Selecting Model (used for training/testing a model).

****

## Performance Requirements

# 

|  |  |
| --- | --- |
| Requirement | Description |
| Data | Need at least 1GB of labeled frames data to be able to get accurate results. |
| Accuracy | Our trained model should at least achieve 70% accuracy. |
| Error percentage | Our trained model should have an error percentage of less than 20%. |
| Performance | Our trained model should be fast enough to be able to predict anomalies while a video is streaming. |

# Other Requirements

* Data should be labeled for training.
* Train model to detect specific types of anomalies.
* Have an interface where the user can upload a data entry so we can run the trained model over it.
* CPU/GPU power to train the model in a sufficient amount of time.
* Use Linux Operating System as the default platform for the system
* The system will be implemented using Python 3.6, so Python 3.6 interpreter is require.

# Supporting Information

## Appendix I - Learning Sources

Research papers related to Anomaly Detection can be found here:

“An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos.” 9 Jan 2018.

<https://drive.google.com/file/d/12nr7H5Ev39a6KDHmyLg4E6dPFx04EEbL/view?usp=sharing>

“Deep-Cascade: Cascading 3D Deep Neural Networks for Fast Anomaly Detection and Localization in Crowded Scenes.” April 2017.

<https://drive.google.com/file/d/1b_ictTnyNmkaGF0hljhzumIcvmWEGpwu/view?usp=sharing>

“Detecting abnormal events in video using Narrowed Motion Clusters.” 12 Jan 2018.

<https://drive.google.com/file/d/16bil5GRNwyCKfLEEoTQgLsbRdoTNQVcq/view?usp=sharing>

“Joint Detection and Recounting of Abnormal Events by Learning Deep Generic Knowledge.” 2017.

<https://drive.google.com/open?id=1wkD0bVxREeKJFiOnxJ_OU3YBCTKvcWvp>

“Modeling Video-based Anomaly Detection using Deep Architectures: Challenges and Possibilities.” 2015.

<https://drive.google.com/open?id=10p-c58_8M_X464rDdQIGZE8haNxQ4YI3>

“Real-world Anomaly Detection in Surveillance Videos.” 12 Jan 2018.

<https://drive.google.com/open?id=169WEm5Ce2SdwnYQiqw5kvhvoXIfc5uy9>

“Unsupervised Sequential Outlier Detection With Deep Architectures.” 9 Sept 2017.

<https://drive.google.com/open?id=1Opig31otfz_BuXBi8Jv9AGlYCz8m7Cwa>

“Video anomaly detection using deep incremental slow feature analysis network.” 2016.

<https://drive.google.com/open?id=1hepZfQFjf7UhFZgo4hhbmMuYb4D9av12>

## Appendix II - Helpful Resources

Here is a collection of helpful slides and lectures over deep learning and anomalous detection:

<https://drive.google.com/open?id=18tAWj16IWAM_GPhi32Ep33w7rvNX6Z8Z>