SMART SURVEILLANCE SYSTEM

KHWAJA FAREED
UEIT
RAHIM YAR KHAN

Faculty of **Information Technology & Management Sciences**

Submitted By

Umar Zafar

Khubaib Ahmad

2017-2021

Department of Computer Science

Khawaja Fareed University of Engineering & Information Technology

Rahim Yar Khan

2021

Smart Surveillance System

Project

Submitted to

Mr. Shahzad Hussain

Department of Computer Science

In partial fulfilment of the requirements

For the degree of

Bachelor of Science in Computer Science

By

Umar Zafar

CS 172071

Khubaib Ahmad
CS 172074

2017-2021

Khwaja Fareed University of Engineering & Information Technology

Rahim Yar Khan

2021

DECLARATION

We hereby declare that this project report is based on our original work except for citations and quotations which have been duly acknowledged. We also declare that it has not been previously and concurrently submitted for any other degree or award at Khawaja Fareed University of engineering & Information Technology or other institutions.

Reg No	:	Cs 172071	Reg No	:	Cs 172074
Name	:	Umar Zafar	Name	:	Khubaib Ahmad
Signature	:		Signature	:	
Date	:	01/06/2021	Date	:	01/06/2021

APPROVAL FOR SUBMISSION

I certify that this project report entitled **"SMART SURVEILLANCE SYSTEM"** was prepared by **Khubaib Ahmad** and **Umar Zafar** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Science in Computer Science at Khawaja Fareed University of Engineering & Information Technology.

Approved by:

Supervisor : Mr. Shahzad Hussain

Signature : _____

Date : _____

ACKNOWLEDGEMENT

In the Name of ALLAH, the Most Beneficent, the Most Merciful, we would like to thank everyone who had contributed to the successful completion of this project. We would also like to express my gratitude to my Project supervisor, **Mr. Shahzad Hussain** for his invaluable advice, guidance and his enormous patience throughout the development of the research. In addition, We would also like to express our gratitude to our loving parent and friends who had helped and given me encouragement

ABSTRACT

Surveillance and monitoring have been evolved as integral and critical parts of today's era in term of security, scrutiny, tracking or any spatial activity. Relying solely on manual observation by security personnel for long hours is not feasible no efficient and often affected by negligence or any human error. With the evolution of Machine Learning and Deep Learning in collaboration with high speed processing, there is an uphill need of surveillance CCTVs equipped with modern algorithms. It will revolutionize the field of supervision using Deep Neural Network and CNN with Computer Vision and smart image processing. So any type of emergency, violence, suspicious objects or any abnormal activities flows in the range of CCTVs are easily detect and sort by smart AI based algorithms working backend in the system and afterwards informing the local authorities or Law-enforcement with coordinates and location of the observed site.

KEYWORDS: Surveillance, Monitoring, Machine Learning, Deep Learning, CCTV, Deep Neural Network, CNN, Classification, Computer Vision , Situation detection, Law-enforcement.

Table of Contents

List of Figures

LIST OF TABLES

Chapter 1 INTRODUCTION

1.1 Introduction:

Emergency tackling has been a concerning issue from the very beginning of social and technical advancement. While estimating the nature of emergency and further on response according to the situation have been a crucial challenge for all responsible and respective authorities. According to the World Bank's report on accidents, over 1.25 Million People are killed on roads each year due to fatal and non-fatal accidents[1]. Fatal accidents make sense , but what about non-fatal accidents? The report further reveals that about 40% deaths are due to the negligence or poor emergency response time.

While according to the report, the mortality rate is even high in developing countries with 14 deaths out of 1,00000 in Pakistan are due to the car accidents, 21 in India, 28 in Iran , in contrary to just 3 in UK and 6 in Canada[1]. So it should be considered as a burning problem of modern societies with property and lives losses.

Other than just road accidents, Heat-related incidents resulted in nearly 9 million injuries and more than 120,000 deaths worldwide in 2017, according to a new scientific study and 15% of deaths are caused by the delay in emergency response[2]. This is not just co-incidence, a second delay in emergency response can cause a turn to the bout. Moreover sabotage, riots, explosion and use of firearms are causing a major threat to the property and lives. By increasing in Emergency response time, we can save belongings and viabilities. New York times Special survey on US Emergency department divulge that every 3 out of 10 lives are saved by quick response.

But with the collaboration of Human and Artificial intelligence, these fatal issues can be resolve. With the advancement in AI Super Intelligent cameras and high level monitoring system, we can create a clear difference in lives and mortality rate. In past few years, deep learning in general and Convolutional Neural Networks (CNNs) in particular have achieved promising results to all the classical machine learning techniques in image classification, detection and segmentation. Instead

of manually selecting features, deep learning CNNs automatically discover higher level features from data. The system is a concatenation of this Link with the features of desecting:

- ANY EMERGENCY SITUATION INCLUDING FIRE ERUPTION, SHORT-CIRCUIT, SABOTAGE , EXPLOSION OR ACCIDENT.
- RIOT OR ABNORMAL BEHAVIOR IN CROWD.
- SUSPICIOUS OBJECT OR ANY WEAPONRY.

Further operating, if any suspicious or emergency activity is observed by the system cameras, with the evidence of the activity the system an alert call is generated to the local vendor or control room about the nature of the situation to be worst or mild. Now the personnel in the room is to decide about the situation whether to alert the highly authorities or to take local individual actions.

With the evolution of Convolutional Neural Network , it is easily adaptable to check the content of the an image purposed as an input to the CNN model. Upon Convolutional layer to detect the features of the image, Pooling layer and then flattening layer tend the image to classify the situation related to the situation being applied. A proper training of deep CNNs, which can contain thousands of parameters, requires very large datasets, and also very high computing resources accelerated with GPUs. Transfer Learning is becoming very popular and widely accepted alternative to overcome these constraints. It consists of re-using the knowledge learnt from one problem to another related one. Applying transfer learning with deep CNNs depends on the similarities between the original and new problem and also on the size of the new training set.

The layers of the CNN are adjusted to the particularities of the problem and extract high level features. In this project I will be using the pre-trained dataset acquired from many deep learning promoted sites and research institutes. They will be fine-tune it to our own needs using CNNs to detect any emergency or suspicious activity and inform authorities in no time.

1.2 System Overview:

The system will get real time view of the region being observed by the CCTVs or Surveillance cameras and perform operations on the basis of the sensitivity and nature of the situation. The main operations of the system are as detecting and further acting on the situations like:

- Any emergency situation including fire eruption, intense short-circuit and explosion.
- Any building sabotage due to earth-quake or disaster.
- Road accidents.
- Roads sabotage due to flood, or land-sliding.
- Riot or abnormal behavior in crowd.
- Suspicious object including firearms or any weaponry.

So any suspicious or emergency activity is observed by the system cameras, with the evidence of the activity the system an alert call is generated to the local vendor or control room about the nature of the situation to be worst or mild. Now the personnel in the room is to decide about the situation whether to alert the highly authorities or to take local individual actions.

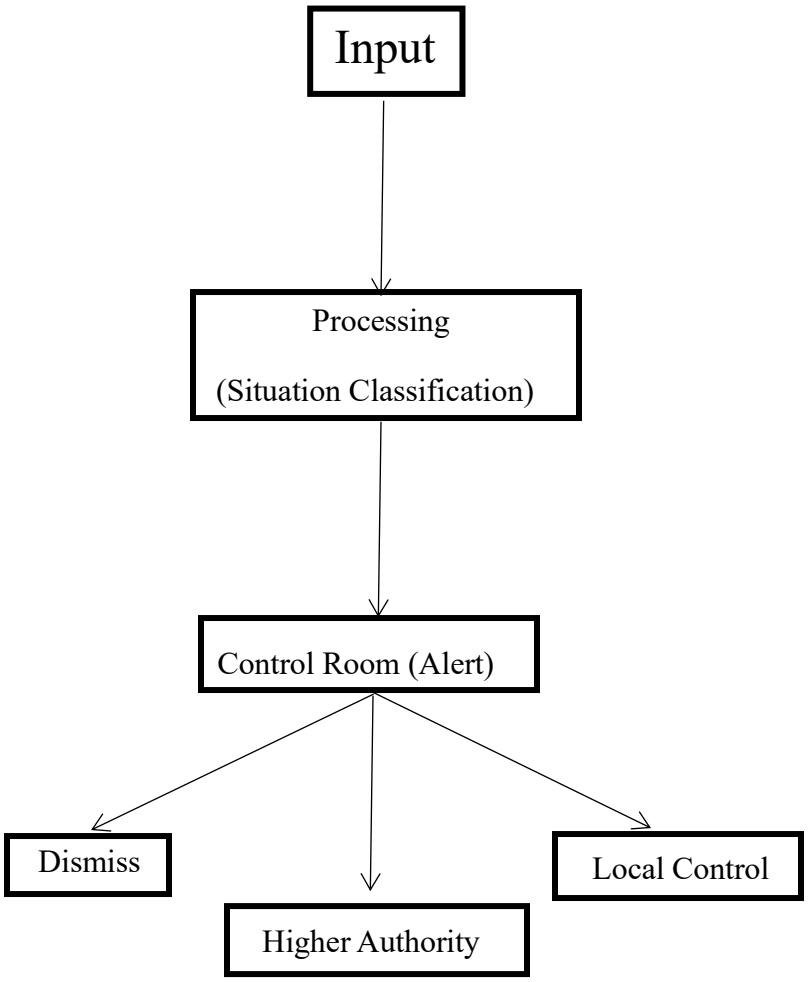
When it terms to generate alert or alarm, there occur some condition which the alert receiver has to follow according to the purposed nature of the problem:

1. Should the problem is such alarming to generate call to law-enforcement?

Or

2. Should the emergency be tackle at individual level?

The person operating the system can select any one of the condition and automatically red alert call with Coordinates of the site are sent to the authority, as in (fig 1.1).



The client may be any high-profile institution where the security and scrutiny are highly sensitive. Moreover banks , financial industries , educational institutes or any classified industries can deploy this system to get real time alert and detection of any abnormal behavior or activity and take quick countermeasures. Smart city can also deploy this system in real time trafficking over the activities of masses.

The system would be consisting of a Convolutional Neural Network instead of the old-fashioned techniques of machine learning for classification, detection and segmentation, instead of selecting the features manually, CNNs are able to discover the high-level features from the data. The

technique of Transfer Learning would be applied on the data in order for detections, also the results from similar scenarios can be applied to the related problems. The process of fine-tuning is done only when the new dataset is large enough otherwise the system would suffer from overfitting.

This network will be getting the input in the form of video which would have been segmented, the network would feed the output to another network and that network would be classifying the scenes as violent on the basis of detection of emergency activities.

1.3 Objectives:

Basically the core intention lies behind the development of this "Smart Surveillance System" is to make smart collaboration of Human and Artificial Intelligence on the slogan "Machines in Service to the Humanity". The system would not only be intelligent enough in early detection of emergency situation but also pointing the exact location of the site so that our main goal "**Reduction of emergency response time as much as possible**". Since according to a study by reducing emergency response time to 25% , we can reduce the casualties and fatalities rate by twice[3]. So in order to overcome the risk involving these sorts of situations to ensure a better civilization the system would be vitally used.

The core objectives behind the fulfillment of this projects are as follow:

- Smart human-machine interaction for effective utilization of resources.
- Vigilance of authorities in tackling emergency response time.
- Minimize loss in emergency and save lives and properties.
- Helpful in effective controlling of street crimes.
- Can be cost effective too, if utilized efficiently.
- Monitoring of mob or any human gathering.

The manual monitoring systems are unable or very less likely to detect the emergency or any critical condition upon time to deliver quick response. So there is a necessary demand of a real-time object detection and emergency informing system that can deliver accurate and authenticated result based real time image it will detect using CCTV or surveillance cameras. There can occur many technical, abnormal or human error which can affect the whole manual monitoring system.

The system can also make sure to reroute the police and other security agencies resources where they are most needed instead of wasting the resources and also to help the police in finding plausible evidence that might also provide them with clues that can help solve the crime or investigation.

The problem which is faced quiet often by the emergency response authorities is that people try to caper them by making scam calls and claiming that they have witnessed an emergency or suspicious activity, which waste useful resources of the involved departments and also instead of waiting for some to call the concerning authorities regarding a crime the system would have to be fully autonomous to detect such kind of violent activity and predict it, so finding plausible evidence that might also provide them with clues that can help to assess the sensitivity of the situation.

1.4 Problem Definition:

Imagine in the mid of night a group of invaders enters into the entrance of a high profile official building equipped with modern weapons and the personnel in the control room tend to do negligence or even in drowsiness condition unable to detect the suspicious activity. Moreover a personnel monitoring hundreds of cameras at a time and ultimately ineffectual to focus on all the cameras.

Furthermore inside a health center premises, there occur a short circuit and turn out a major threat to the critical site which needs to be early detected and tackled before it's too late to create any loss to lives or property.

Current surveillance and manual control systems still require supervision and intervention. The problem faced with the existing system was it still required some whom should be supervising the videos and footage of CCTV cameras and on the basis of that whenever he find some act of emergency than he should inform the respective authorities which might take a reasonable time and in that time the activity which was being informed might have had already occurred, the other problem was scarcity of resources.

So there is a potent need of a smart system to detect any suspicious activity and timely inform the Law-enforcement about the nature and the exact location of event with coordinates system.

1.5 Project Scope:

Any emergency can occur at any time at any place with the existing surveillance system, however most systems rely on constant human supervision which demands vigilance, and can be expensive and ineffective when multiple video streams are present. The only solution and the place where this project finds its scope would be in assisting the emergency response agencies including law-enforcement or Rescue department in detecting emergency or violence so that in places where they are a lot of people like stations and bus stops etc. where it's very difficult to identify the suspects with the traditional method, the system would be automatic and would detect the person on the basis of the presence of weapons in his/her possession.

The other scope would be in detecting abnormal crowed, on the basis of the streaming of the video the system would be able to detect abnormal crowed behavior and on the basis of that abnormal crowed the security agencies can allocate resources accordingly.

1.6 Features:

The main features of the "Smart Surveillance System" are:

- Automatic system with the ability of detecting emergency situation in video stream.
- Firearms detection, and in this regard stopping violent crimes before they can even happen.
- Efficient utilization of resources of the security and respective agencies.

Chapter 2 EXISTING SYSTEM

2.1 Existing Systems

The concept of surveillance with smart cameras has been evolved as a large scale application in China, the United States and many developed countries. With the evolution of high speed processing , many tech giants are focusing on leverage deep learning algorithms for insuring security and smart monitoring along with other deployable disciplines.

The problem of smart monitoring in real-time videos using deep learning is related in part to multiple broad research areas. The first classical system undergo continuous improvement using data as fuel and then check the performance of our models using many assessment methods such as principle component analysis and Confusion Matrix for the event classification using deep CNNs to overcome any risk of not detection.

2.1.1 **Manual Observation:**

The first and the traditional sub-area in smart surveillance focuses on customary way of manual observation. Manual observation has been evolved as an efficient techniques throughout the contemporary world for scanning and security. A person sitting in control room focusing on cameras and dealing with any unpleasing event.

The second sub-area related to monitoring using RGB images using classical machine learning methods. The few existent systems apply methods such as Scale-Invariant Feature Transform (SIFT) or Rotation-Invariant Feature Transform (RIFT). These methods were unable to detect multiple pistols in the same scene. The technique which was used consisted of first, eliminating non related objects from the segmented image using K-mean clustering.

2.1.2 **Detection Systems:**

Emergency detection is the process of recognizing the scenario and finding the location in the input image or video. The existing system have addressed this detection problem by reformulating it into a classification problem, first training the classifier then at time of detection running it to a number of areas in the input image .

ADOBE IOTA:

- Abode has an all new version of its simple home *security system* called Abode Iota. Iota takes the Abode security hub and teaches it some new tricks, thanks to an integrated motion sensor and camera, in addition to a built-in siren. Abode comes at the problem from the security side of things rather than the camera. Abode has built a robust infrastructure over the last few years, and users have plenty of alternatives when it comes to designing a security system for their home around Abode, including a wide range of sensors, the option (but not the requirement) to subscribe to professional monitoring, a cellular backup option, and even the ability to integrate other types of smart home devices with the system.
- The iota system's main unit has a built-in 1080p video camera and siren. It also acts as a hub for controlling other devices, like the door sensor and key fob for arming the system that come with the starter kit. If you've ever seen Canary's Pro or View cameras, the iota hub has roughly the same footprint but is a little more than an inch (2.5 cm) taller. The unit houses the camera, 93 dB siren, motion sensor, microphone, battery backup for handling power outages, and a 4G SIM slot for backup communications if Wi-Fi or Ethernet isn't available[4].
- Is an exhaustive method that consists of large number of candidate windows. It checks the input image at all locations and scales with a window and runs the classifier at each window. The most effective work improves the performance of the detection by building Histogram of Oriented Gradients (HOG) based model, which uses HOG descriptors for feature extraction to predict the object class in each window. The further improvement in HOG is the Deformable Parts Model (DPM) which uses the HOG descriptor to calculate low-level features. This model provides very good accuracies but the detection process is way too slow under the sliding window approach.

FUJITSU'S SMART CITY MONITORING:

- Based on Deep Learning and Recurrent Neural Network model, Fujitsu smart City[5] has developed a region based approach for monitoring and surveillance of cities. Citywide Surveillance V2 uses the following recognition and attribute classification functions based on video or photo images of individuals and vehicles from monitoring cameras :

  1) Vehicles

  Vehicles can be recognized in terms of their type, manufacturer, and body color. Other features include a vehicle registration number analysis, a vehicle counter for vehicles within a specified area or those passing pre-determined lines or sections of lines set up on a screen.

  2) People

  People can be recognized in terms of the type sand colors of their clothing and facial characteristics. Other features include a counter for people within a specified area or those who pass predetermined lines or sections of lines set up on a screen, as well as a feature to automatically estimate seating positions in restaurants, etc.

2.2 Drawbacks of Existing System:

The existing system had following drawbacks:

- Manual observation carried out by control room staff are no longer feasible and often affected by negligence or lack of focus. Moreover detecting hundreds of cameras at a time leads to a fatal mistake on the cost of property and lives loss. An event occur and then it monitored by cameras is a hectic and time consuming process with series of defects. So in the long run, it's not feasible technique.
- Adobe iota has been developed on the region based approach focusing on just specific region analysis and specially designed for home security also. Although it's a siren-based powerful tool but lack the feature of location accuracy and thus less effective.

Furthermore for larger homes, it lacks the feature of integration of many cameras to a specific system.

1. Having a robust and multifeatured Fujitsu smart city application, the feasibility of this system is ineffectual in terms of dealing with multiple situations. Having focus on just single entity scenario, this system has many downsides.

2. Build specifically for masses and vehicle monitoring, its unable to deal with the unseen scenarios like fire eruption or any violent mob movement.

3. Lack the facility of siren(alarm) and mobility.

4. Monitored by a single embedded control system, ineffectual to classify to whom respective authority, the situation should be informed, thus delaying response time which have tragic affects in the long term.

- Some other system are effective , but undeployable in a region like Pakistan or India where traffic or masses patterns are unpredictable. So there are many rooms for economical and robust system in the subcontinent region

2.3 Proposed System:

The proposed system would be and automatic emergency scenario detector that will be able to detect and predict the occurrence of any unpleasing or emergency situation in a streaming video and will be able to inform the authorities about the nature and profound sensitivity of the scenario to reduce quick response time and in the end, preserving lives or belongings. If there is being a violence activity going on somewhere before it can even occur.

The system would consist of a Convolutional Neural Network instead of the old-fashioned techniques of machine learning for classification, detection and segmentation, instead of selecting the features manually, CNNs are able to discover the high-level features from the data. The technique of Transfer Learning would be applied on the data in order for detections, also the results from similar scenarios can be applied to the related problems. The process of fine-tuning is done only when the new dataset is large enough otherwise the system would suffer from overfitting.

The first layers of the network will be used for extracting the edges and colors (low-level features) the remaining layers also known as the high-level feature extracting layers. This network will be

getting the input in the form of video which would have been segmented, the network would feed the output to another network and that network would be classifying the scenes as violent on the basis of detection of crisis. My focus would be on the most used type of intense emergencies like riot, explosion, fire eruption , accident or any violent scenarios . The problem of crisis detection in the videos using deep learning is divided in two parts. The first address to the using of classical methods and the second one addresses the problem of detection of multiple violent situations in the same image.

2.4 Need for Replacing Existing System:

The reason for replacing the system are as follows:

- The current system of the emergency and other security agencies id the traditional old fashioned in which whenever some reports a crisis or emergency scenario only than the necessary resources are allocated, and the action accordingly is taken. The other practice which is being observed in current system is that the cctv feed is being recorded and there must be someone having his/her uninterrupted attention towards the feed so as he/she can identify if there is been a crime and report accordingly also chances of mistakes are fairly higher. The only solution to this problem is an automatic situation detection system.
- Detection is a one way process, the next approach is to inform higher and respective authorities about the situation and complete scenario oneness. Further there are multiple departments operating in dealing with multiple situation, so classifying scenarios and informing to only respective authority is an herculean task for classical systems to deal.
- Since most of the systems are ineffectual in integration of multiple scenarios and then the faulty algorithms of move and quick approach. Previous system for detection either are relying on single scenarios and lack the effectiveness of multi-situation approach making them reckless.

- The previously existent system which has been developed had the drawback that it had depend solely on manual observation and lack the fully digitalized system. Thus the region based, staff accessed approach can fill this gap systematically by give him privilege to deal with the scenarios on their own, by handle locally or inform the respective authority. With this approach even though the accuracies were satisfactory nut it takes too much time. So the proposed system which would be developed on the technique of transfer learning would be quiet faster than the existing system and also this technique would save a lot of resource since i will be re-utilizing the use-case scenarios of similar problems learnt by the cnn and fine-tuning them to my use-case scenario.

2.5 Understanding Proposed System:

2.5.1 **User Involvement:**

The term “User Involvement” in general means subjective psychological state of the individual and defined as the importance and personal relevance that users attach either to a particular system or to IS in general. User involvement and user attitude are psychological aspects that are in the user's mind, while user participation is an "observable behavior of users during the development process of a system”. In present digitalized and changing world, effective way to support market success are innovations originating from the needs of the customers. If users involve into the system development cycle, they can give more information details. The effect of the user participation on successful system development of a system can’t be overlooked.

Keeping in mind the requirements of the user the system would be developed with the involvement of the end-users so that after the system is developed and ready to use the user’s interaction becomes easier, and also the system fulfills the requirements of the user and anything which the user demands from the system must be included, and all those features which are irrelevant to the user must get identified before they even enter the development phase. The user need can be analyzed and gathered in the phase of requirement elicitation.

2.5.2 **Requirement Stimulation:**

The Process of gathering the user's requirement is the most important part in the development of any product, since everything that is being manufactured is directly related to the user's and also it should be sufficient enough to cater to the needs of the user. The process for gathering and realizing the users' needs in case of " Emergency and Crises Detection in Real-Time Streaming Data (ECDRSD)" was divided into two phases by myself:

- **Approach 1**: In this phase generated a questionnaire and distributed it in my entire social circle which consisted of series of questions both related to checking what the user feels about the system while interacting with the system and relating to getting an idea of the demands of the user. Most of the user's opinion in contrast questions to the existing system showed their strong interest in the system. They were eager in seeing such a system especially in a country like Pakistan.
- **Approach 2**: The second approach was that i myself reached out to the and talked to them regarding the system describing them the system and taking a brief idea of what they would actually want to see in a system like this. I make note of what things does most of the users want to see, what things they would not like to see and what type of interaction they would like to have with the system. They also gave me an idea regarding the interface of the system.

Chapter 3 PROPOSED SYSTEM

3.1 Detailed Description of Proposed System:

The proposed system would be and automatic emergency detector that will be able to detect and predict the presence of any crisis or violent situation in a streaming video and will be able to inform the authorities if there is being a critical activity going on somewhere before it can even occur.

The system would be consisting of a Convolutional Neural Network instead of the old-fashioned techniques of machine learning for classification, detection and segmentation, instead of selecting the features manually, CNNs are able to discover the high-level features from the data. The technique of Transfer Learning would be applied on the data in order for detections, also the results from similar scenarios can be applied to the related problems. The process of fine-tuning is done only when the new dataset is large enough otherwise the system would suffer from overfitting.

The first layers of the network will be used for extracting the edges and colors (low-level features) the remaining layers also known as the high-level feature extracting layers. This network will be getting the input in the form of video which would have been segmented, the network would feed the output to another network and that network would be classifying the scenes as violent on the basis of detection of situation. Our focus would be on the most frequent emergency scenarios like aggravation of mob, fire eruption, sabotage , accident or explosion.

The problem of emergency detection in the videos using deep learning is divided in two parts. The first address the scenarios detection using classical methods and the second one addresses the problem of detection of violent or critical scenarios.

The first process in the system would be of the conversion of the video data to images. The frames of the video would be extracted at a certain FPS and stored. The video would first be uploaded and then the extraction process will start. After conversion to images the noise in the image would be removed and then the images would be grouped to make short clips.

The second process would consist of feeding these clips to a pre-trained CNN e.g. GoogleNet, after the CNN classifies the segmented images, transfer values would be assigned. These segmented

videos with transfer values would be fed to the neural network. This network would check the segmented images by the approach of transfer learning and on the basis of these classification techniques, it segmented the nature of event to be occurred .

The last process would be the notify the higher authorities about the situation. Here I have made 3 critical sections ,based on the event the personnel on the control room can judge the situation and then react according to the way. The alert receiver has to follow according to the purposed nature of the problem:

1. Is the problem is such alarming to a generate call to the law-enforcement?
   A) the contact numbers of law enforcement will be one click away, select the desired agency and just click
   B) automatically, an alert call will be generate to the respective authority with the coordinated of the event.
2. Is the emergency be tackle at individual level?
   A) Click the local authority button, and automatically all information will be sent to the local authority.
3. Is the problem be ignorable?
   C) ignore the scenario

The results would be displayed on runtime and accordingly authorities would be alarmed.CNN are a state-of-the-art deep artificial neural network technique for image classification, it clusters similar images and performs object recognition within regions.

The first CNN named LeNet-5 was created by LeCun, Bottou, Bengio, and haffner for handwritten digits recognition. Then CNN research went silent for few years. In 2012, the CNNs arose again on ImageNet Large Scale Visual recognition Competition scaled the structure of AlexNet-5 into a much larger Neural Network which can detect more complex objects using rectified linear units (ReLU) as non-linearity which is applied to the output of each convolutional and fully connected layer, dropout technique to avoid the problem of overfitting and overlapping pooling to avoid the effect in average of average problem.

The structure of the model is divide into two parts when trained, and the model interacts with each layer at different level.

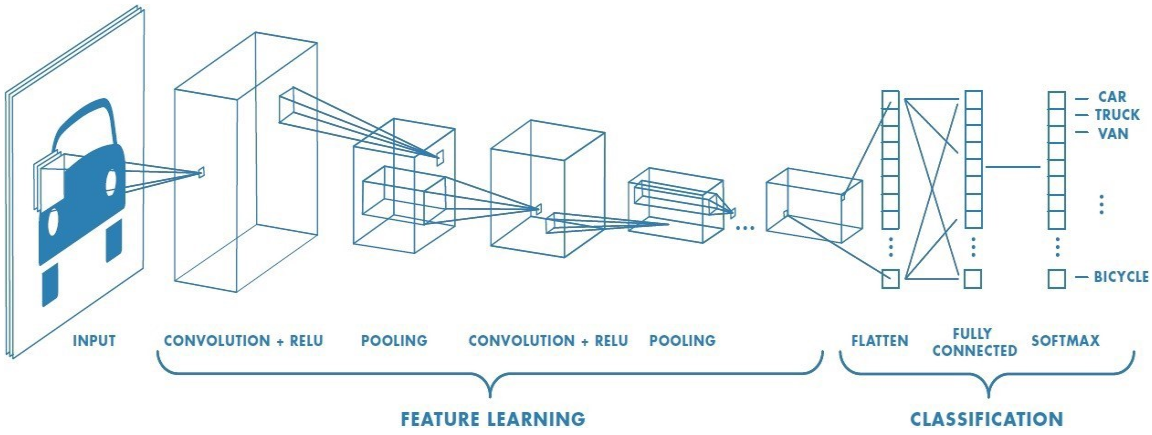


Figure 3-1 3D CNN

Beyond foraging the memoir of CNN, the first model AlexNet[7] achieved 15.3% test error which started a revolution of the CNN on Deep Learning Neural Network with its overwhelming results from others in previous years. ResNet is the next breakthrough of CNN, it was introduced in 2015, a team of Microsoft. They were using the “Deep residual learning framework” in which the main idea was to pass the output of two following convolutional layers and go past the input to the next weight layer. They managed to achieve the error rate of 3.57%. The increasing network depth significantly makes improvements on the model’s error rate. ResNet model with more than 110 layers did not affect the model’s error rate, rather the model error rate stays the same. The reason was that the model was overfitting as the number of the layers was increasing.

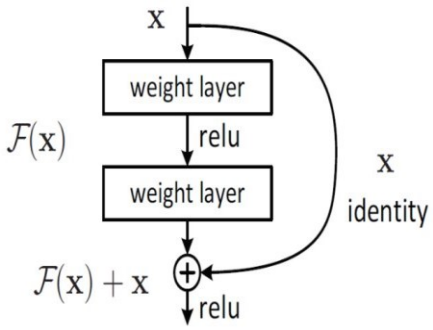


Figure 3-2 Google Deep mind

method			error (%)
Maxout [10]			9.38
NIN [25]			8.81
DSN [24]			8.22
	# layers	# params	
FitNet [35]	19	2.5M	8.39
Highway [42, 43]	19	2.3M	7.54 (7.72±0.16)
Highway [42, 43]	32	1.25M	8.80
ResNet	20	0.27M	8.75
ResNet	32	0.46M	7.51
ResNet	44	0.66M	7.17
ResNet	56	0.85M	6.97
ResNet	110	1.7M	**6.43** (6.61±0.16)
ResNet	1202	19.4M	7.93

Figure 3-3ResNet with different layers

In deep learning, a CNN or ConvNet is a class of deep neural networks, most commonly applied to analyzing visual imagery. They are call shift invariant or space Invariant artificial neural networks, based on their shared- weights architecture and translation invariance characteristics. They have application in image and video recognition, recommender

systems, image classification, medical image analysis, and natural language processing. The basic architecture of CNNs is as:

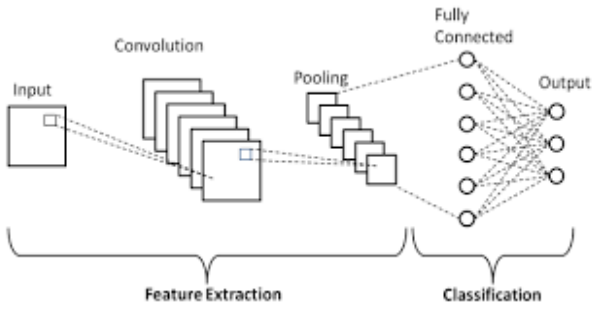


Figure 3-4 Architecture of CNN model

[Input – Convolutional – ReLU – Pool – FullyConnected – Output]

Firstly, the input layer takes image pixel input and passes them to convolutional layer. In CNNs the neurons are constructed in 3D dimensions which the depth means the activation volumes instead of 2D as in simple neural network.

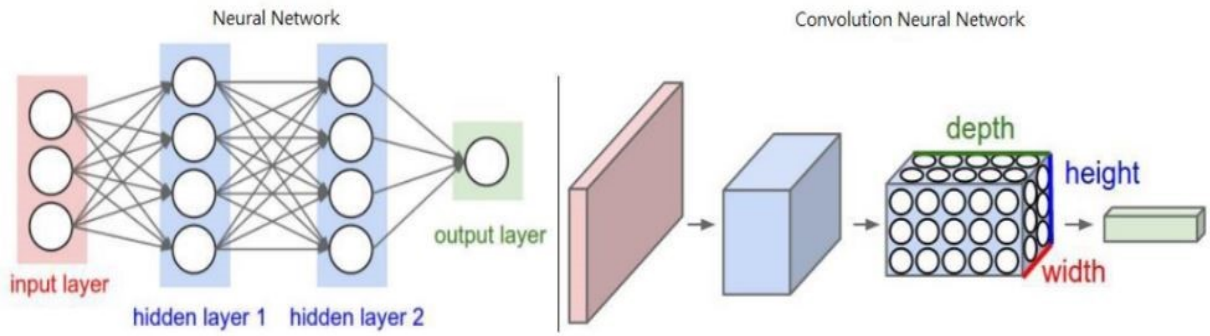


Figure 3-5Neural Network vs Convolutional Neural Network

In convolutional layer, the main purpose of this layer is to extract image features and preserve the spatial connections between pixels from input. It computes the output of neurons which means it transforms the image pixel values into output volumes of final class scores. This convolutional operation is done by striding filter on input image. Every image can be represented as a matrix of pixel values in CNNs and the filter here is called kernel which is a matrix as well.

The operation is striding the filter on feature map each time by one pixel for every position and computing the convolved output by adding up the output of multiplication of the matrices. Each convolutional layer not only computes the final feature map but also there is activation function for taking input volume from previous layers and parameters of neurons such as bias and weights.

In the ReLU layer feature map is passed to this layer to change all negative pixel values to zero. Most of images from real world are non-linear, but CNNs operation is a linear process. To make the CNNs learning the non-linear data, ReLU function is implemented to introduce the non-linearity to CNNs. The difference between applying the ReLU before and after is shown as below:

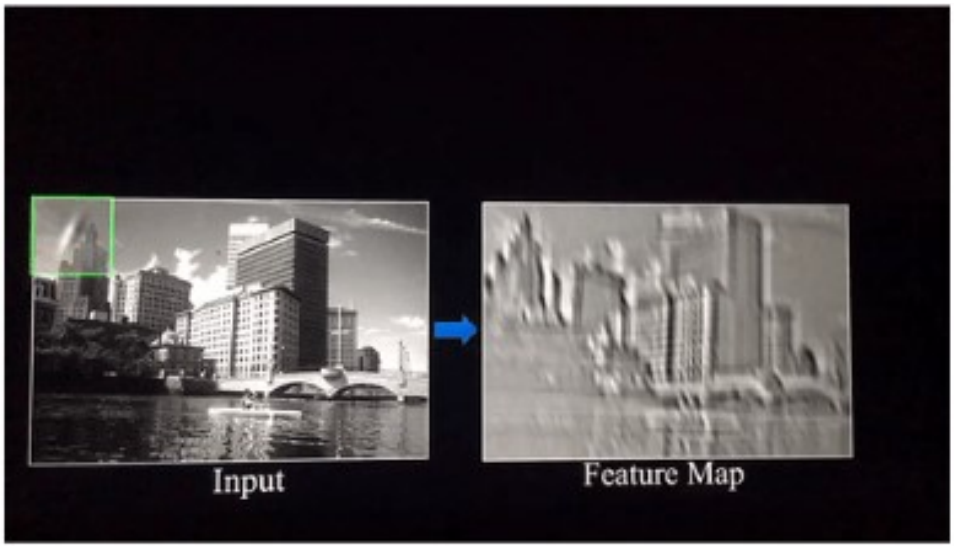


Figure 3-6 ReLU illustration

In the next Pooling layer is inserted between CNNs.The pooling function reduce the size of feature map, but it still preserves the most important spatial details of the feature map. The purpose of pooling function is to reduce parameters and computation in the network, thus it controls overfitting issue. There are many types of pooling functions; max, sum, average etc. The max pooling operation, define a spatial neighborhood first like [2 x 2] and then stride it by 2 to the rectified feature map to select the largest pixel value in each area.

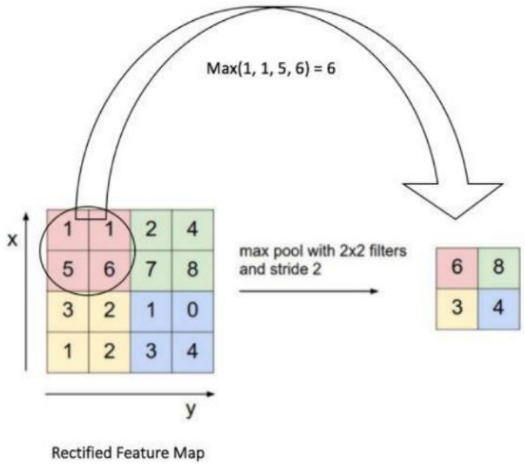


Figure 3-7 Max Pooling

Fully connected layer is the layer where all the neurons are connected to all the other nodes in the next layer, and a common activation function called SoftMax is used in this layer as multinomial logistic regression for multiple classes classification. SoftMax function ensures the output probabilities from fully connected layer is one in the output layer. The complete training phase of a CNNs looks like below:

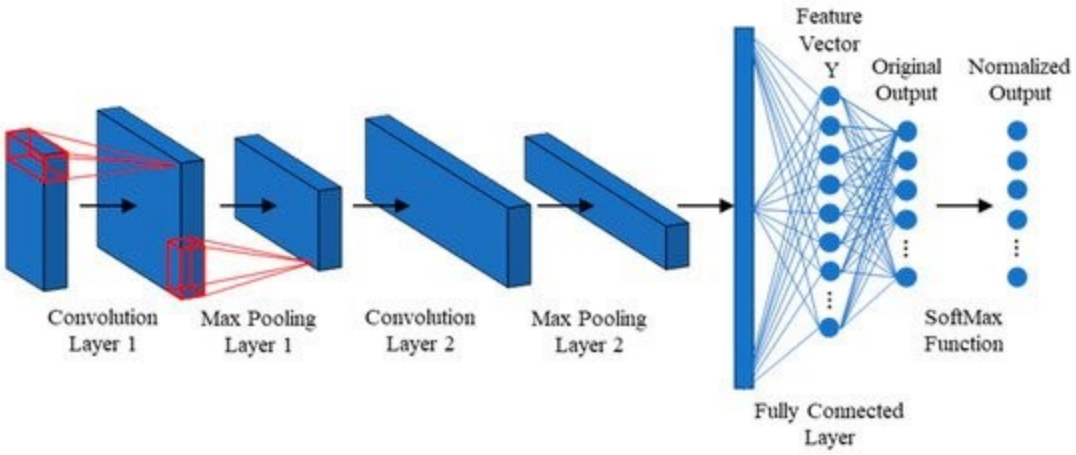


Figure 3-8 Fully connected layer & softmax function

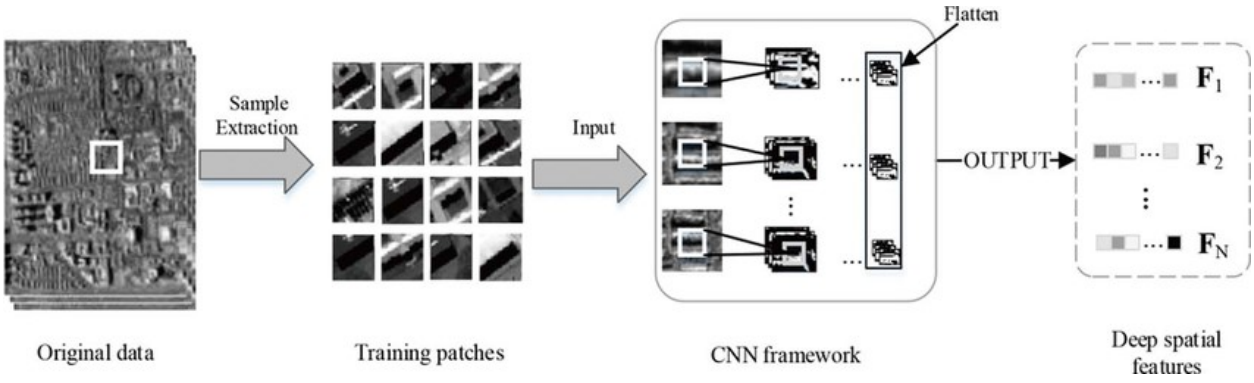


Figure 3-9 Training the CNNs

The main technique used in this project is transfer learning applied on a Faster R-CNN, an object detection network, to be trained and tested with images to perform robust Classification Operation. Moreover in terms of Object detection, it is different to classification where object detection finds different objects in an image and classifies them. Faster R-CNN is the state-of-the-art technique for object detection in machine learning and more precisely deep learning.

The main goal of R-CNNs is to correctly identify where the object is in the image via bounding box (region proposal) and display the identified objects with bounding boxes and correct labels for each object. The proposed method for creating bounding boxes is using selective search to extract region proposals (around 200 times per image), and then it passes the region proposals to the CNN layer to get convolutional features. In the final layer, they use a SVM to classify whether that is an object.

Once the object has been classified, it runs a linear regression model for tightening coordinates of a box.

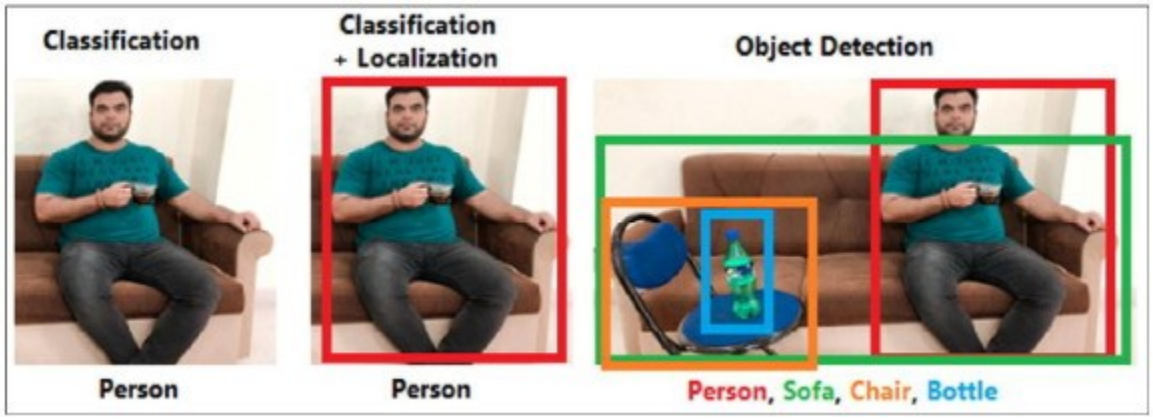


Figure 3-10 R-CNN

Despite it is working well, there are still few drawbacks of R-CNN. The first reason is that it normally requires 2000 forward times for each region proposals in an image, it is computationally expensive to compute CNN features in this number of times. The second reason is that training several modules in R-CNN such as CNN to extract features, SVM to classify objects and linear regression to tighten the bounding boxes, this makes the training harder because of multi-stage pipeline.

The solution this the introduction of a new method called region of interest pooling (Rol pooling) to the Fast R-CNN , Rol pooling layer convert any valid region of interest from an input feature map into smaller feature map by using max pooling, it allows the forward pass of CNN to be shared, This gives the benefits of running CNN in a single time instead of around 2000 times. The second change is that combining train CNN, classifier and linear regression into one model, thus it uses a single network to compute results. These two improvements make it 10 time faster in training time and much faster in testing as well compared to R-CNN.

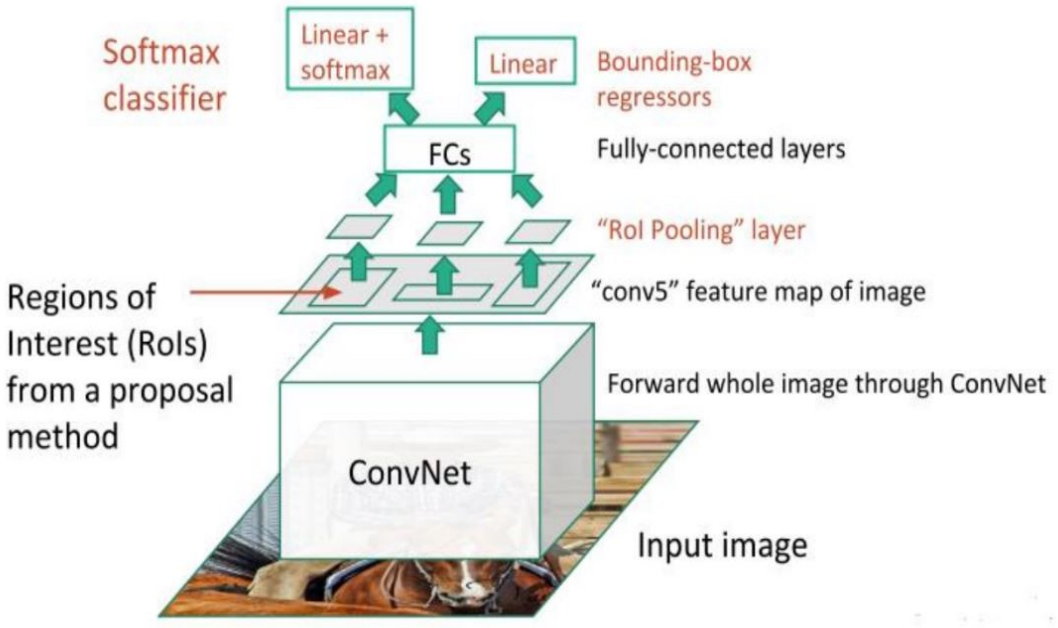


Figure 3-11 Fast R-CNN

However the problem of the Fast R-CNN is that the runtime is dominated by region proposal, so Faster R-CNN solves this problem by using region proposal network (RPN) instead of using selective search to predict region proposals which is introduced by a team from Microsoft. The general idea of Faster R-CNN is reusing the CNN feature map generated to predict region proposals instead of using selective search to find region proposals again. It can avoid the main drawback of Fast R-CNN which the testing time is dominated by region proposals using selective search.

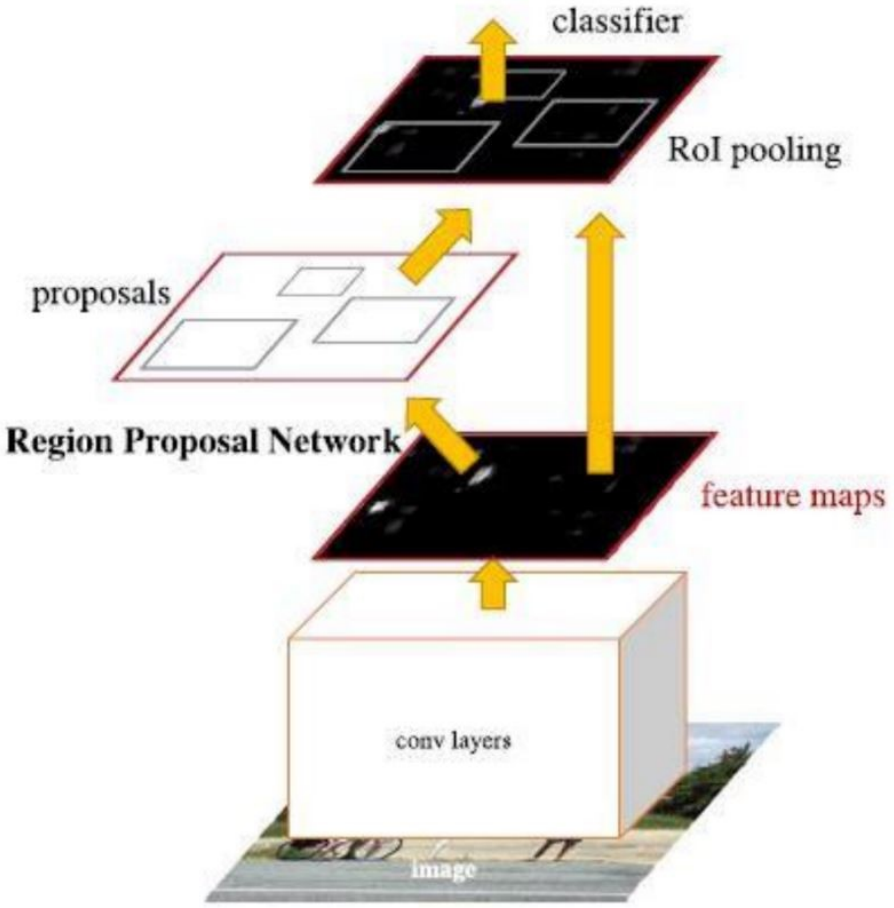


Figure 3-12 Faster R-CNN

3.2 Evaluation of proposed model**:**

Evaluation and performance detection are the crucial phase in terms of assessing the output and result of model. In fact any flaw in detection can lead to catastrophic effects on overall structure of system, so could be handle in an appropriate manner.

There are many matrices and calibers are there to check the performance of classification model. Confusion Matrix is an efficient criteria in terms of dealing with classification problems. Moreover AB-testing and many other matrices can give better results in many scenarios, but surely it depends on the situation and nature of the model you are dealing

with. In terms of CNN for computer vision, the result are generally recognize by viewing the accuracy of classification of image. If accuracy is affected by downfall, hyperparameter tuning are perform and adjusting the weight of parameters.

The fraction of predictions that a classification model got right. In multi-class classification, accuracy is defined as follows:

$$\text{Accuracy} = \frac{\text{Correct Predictions}}{\text{Total Number Of Examples}}$$

After checking manual performance by viewing result, A sophisticated gradient descent algorithm that rescales the gradients of each parameter, effectively giving each parameter an independent learning rate.

3.3 Advantages of Proposed System:

- Early crisis detection has been the core of interest for respective authorities in terms of security, reliable transportation and monitoring of cities with all feasible surveillance tasks. This system tends to help the security agencies as well as crisis management authorities to monitor the activities going on at certain locations using the video feed, and on the basis of the video feed the system would be classifying the activity as safe or criminal and on the basis of these categories generating/displaying relevant message. The advantage of using an “automatic computerized system” would be the efficient use of law enforcement agencies also.
- To reducing the response time every time any situation occur, thus we can rescue and save life as well as property damage on the basis of the analysis of the image(s) and at the same time it would also be helping the authorities to stimulate response time.
- Reducing the cost of patrolling for any scenario to be detected, as well as cost of hiring more cops on patrolling activities. Everything will be monitored via auto integrated system.

3.4 Scope of Proposed System:

Crisis and emergency situation can arise at any place at any with the existing surveillance system, where either the systems primarily rely on constant human supervision which demands vigilance, and can be expensive and ineffective when multiple video streams are present. While other approaches contrary to the proposed approach are no feasible, no efficient.

Detection is a one way process, the next approach is to inform higher and respective authorities about the situation and complete scenario oneness. Further there are multiple departments operating in dealing with multiple situation, so classifying scenarios and informing to only respective authority is an complex task for classical systems to deal.

The only solution and the place where this project will find its scope would be in assisting the local agencies in detecting scenario based classification in the videos so that in places where they are a lot of people like stations and bus stops etc. Where it's very difficult to identify the situation by just staring at the live feed of the CCTV cameras with the traditional method, an event occurred, it catastrophe many lives and then it came under the observation of authorities, so its malignant. The system would be automatic and would detect and classify the situation on the method of transfer learning and detecting single to multiple violent objects and alarming the authorities.

The other scope would be in detecting abnormal crowed or mob, on the basis of the streaming of the video the system would be able to detect abnormal crowed behavior and on the basis of that abnormal crowed the security agencies can allocate resources accordingly.

3.5 Hurdles in Optimizing Current System:

The hurdles in way of optimizing the current system are as follows:

- The most significant challenge would be the collection of data for training the model and for testing it for this purpose my primary sources will be google, kaggle and medium and some official resources of crises management authorities.
- Pre-trained models are used in transfer learning but the challenge of improving the accuracy still remains as it is no use if the accuracy remains the same, for this purpose my

intentions are of checking each model like AlexNet, Inceptionv3 etc. On my dataset. The model that will give the best accuracy would be my choice ultimately.

- Another hurdle would be lack of computing resources for the process of training datasets. As more and more complex cnn algorithms requires high speed computing functionalities.

Chapter 4 SYSTEM DESIGN

4.1 Software Process Model:

The incremental model of software development is a model in which the system is designed, implemented and tested incrementally with a little more is added each increment until the product is finished. In this mode the development and maintenance takes place at each increment. The product is declared as finished when it satisfies all the user requirements. This model combines the technique of the waterfall model with iterative prototyping.

The proposed system would be developed in increments, taking feedback from the user after each increment. The first phase in my system development using the said model involves planning where the components and the brief idea of what features the system should possess. When a clear picture of the features is made than the second increment of user requirement begins and the prototype made by the initial planning is shown to the user and feedback is taken. According to the user requirement system would undergo changes and then those changes would be accommodated in the system.

The next increment comprises of implementing those user needs in the system and accordingly changing the design of the system and another prototype is created whose feedback is taken from the user. At the end of each increment the test of the system takes place where the system is checked for errors and proper functionality and user requirements are checked. When the user accepts the prototype and approves the system, that it caters to the needs the user wants the development process is finalized and the end product is made.

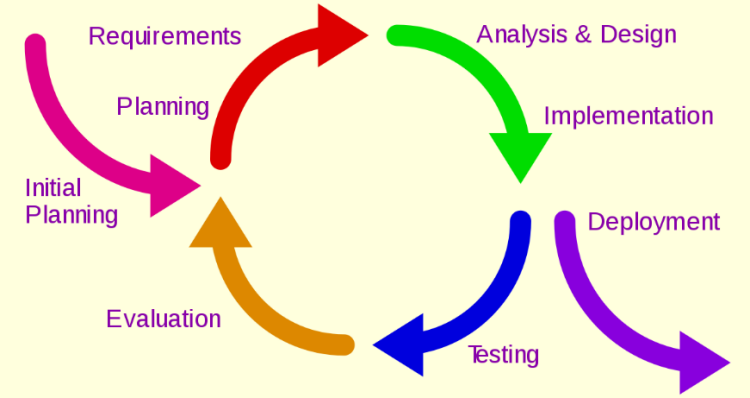


Figure 4-1 Incremental Model

4.2 Software Requirement Analysis:

4.2.1 **Functional Requirements:**

- The system should process the video input from the user only if the video is of a valid format.
- The system should display the error message to the user if the video is not of a valid format.
- The system should convert the video to images.
- The system should detect the image and clarify the scenarios using classification algorithms.
- The system should generate relative output on the basis of detection.
- 64-bit, x86 desktop or laptop with dedicated gpu. The lowest compatible desktop nvidia gpu that is supported geforce gtx640 and laptop gpu is gtx740m and onward.
- Tensorflow with gpu support which is an open-source software library used for machine learning.
- Google object detection api
- Detectron-2 library developed by facebook.
- Labelimg.Exe for labelling images with bounding boxes which is a software used for annotating images and labelling object with bounding boxes in image.

- Python ide 3.5.X onward.
- Jupyter notebook which is an open-source tool developed for interactive programming and supported with many programming languages.

Function	**Full time active cameras**.
Description	Monitoring cameras should be operated actively for input stream.
Input Source	Electric socket or any type of power source e.g batteries.
Output	There shouldn't be any hurdle in camera output.
Destination	It should have continuous power stream.
Action	Action based on detected output.
Requirements	If camera is damage, stream will intercept so replace it immediately.
Pre-condition	Vague Output.
Post-condition	Minimum 480x480 resolution.
Side effect	Alert will not detected if camera is not working.

Table 4-1 Functional Requirements

Function	**Video and image analysis**.
Description	System should have enough processing power to analyze vide no real-time.
Input Source	Camera output.
Output	Alert to the user if detected.
Destination	User (staff room personnel).
Action	CNN Algorithms should be in working condition to detect video stream.
Requirements	Minimum system with 8GB Ram having Internet connection

Pre-condition	Staff should be present to find if any anomaly detected.
Post-condition	It should be able to do inbound alert calls.
Side effect	Can affect efficiency if system is slow.

Table 4-2 Functional Requirements

Function	**Alert Call generation**.
Description	It should alert staff if any anomaly detected and inform authorities through outbound calls.
Input Source	Video Analysis Algorithm.
Output	Authorities will be active to deal with situation.
Destination	None.
Action	Alert generation and calling.
Requirements	Running application with active Internet connection.
Pre-condition	System detect any anomaly.
Post-condition	Alert to authorities without any exception.
Side effect	Authorities will not be informed if any system failure occured.

Table 4-3 Functional Requirements

4.2.2 **Non-Functional Requirements:**

- Performance – the system should be responsive in throughput, utilization and static volumetric.
- The system should have the capacity to deal with real-time event streaming.
- As far as availability concern, the system should have operation 24/7.
- The system should be flexible enough to proceed if any change in scenario occur.

- The system should generate relative output on the basis of detection.
- Data integrity is the most prominent factor while dealing with situation and alert.
- The system should be easy to operate.
- It should deal with unseen scenarios if any occur.

Non-Functional Requirement	Accuracy and Performance
Descriptions	Application must be precise and accurate to detect any anomaly thus preventing from any flaw. Must be enough capable to detect situation accurately and inform to valid authorities
Side Effect	Higher resources consumption of application requires more ram and storage to r un properly. It cost more to use heavy application.

Table 4-4 Non- Functional Requirements

Non-Functional Requirement	Reliability
Descriptions	Rate of failure occurrence is less. System must be consistency. In case of crash, system response time must be quick.
Side Effect	Robot will show misbehave.

Table 4-5 Non- Functional Requirements

Non-Functional Requirement	Portability
Descriptions	Software has a low number of defects and that it reaches the required standards of maintainability, portability.
Side Effect	The system can misbehave.

Table 4-6 Non- Functional Requirements

Non-Functional Requirement	Data integrity
Descriptions	Software should have leak-proof data storage.
Side Effect	Can cast a huge spell to resources if data is misused..

Table 4-7 Non- Functional Requirements

4.3 Design:

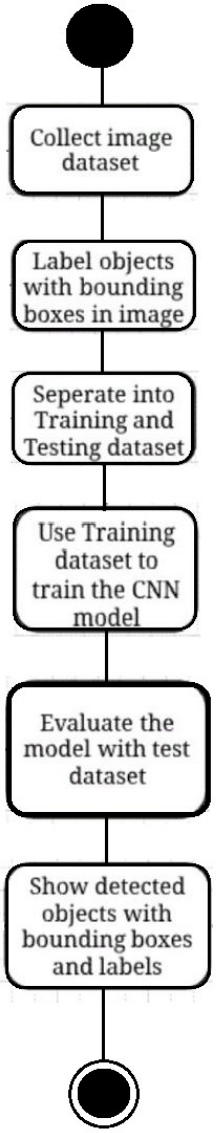


Figure 4-2 Flow Chart

4.4 Benefits of Selected Model:

- Performance – the system should be responsive in throughput, utilization and static volumetric.
- The system should have the capacity to deal with real-time event streaming.

- As far as availability concern, the system should have operation 24/7.
- The system should be flexible enough to proceed if any change in scenario occur.
- The system should generate relative output on the basis of detection.
- Data integrity is the most prominent factor while dealing with situation and alert.
- The system should be easy to operate.
- It should deal with unseen scenarios if any occur.

4.5 Limitations of Selected Model:

There are some limitation to theselected mode:

- Each new build must be integrated with previous builds and any existing systems. If there are few builds and each build degenerates this turns into build-and-fix model, however if there are too many builds then there is little added utility from each build.
- Each phase of an iteration is rigid and do not overlap each other.
- Problems may arise pertaining to system architecture because not all requirements are gathered up front for the entire software life cycle.

4.6 Use Case Diagram:

4.6.1 **Complete Use Case:**

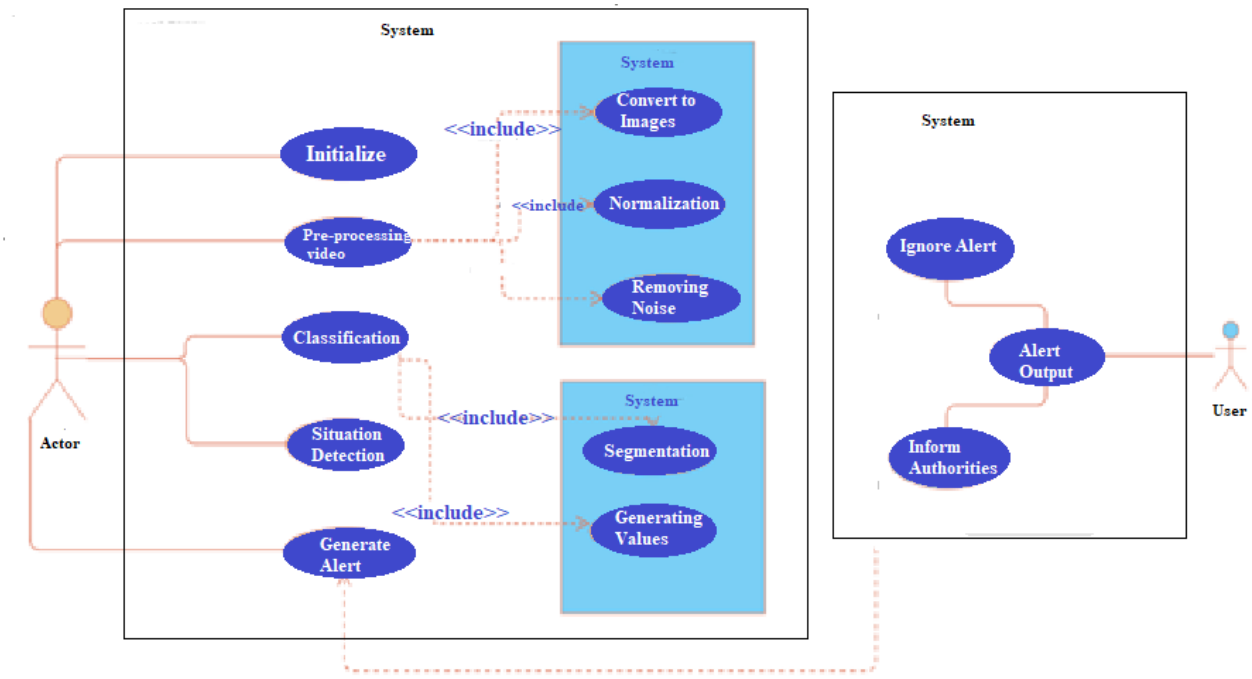


Figure 4-3 Complete Use-Case

4.6.2 **Use Case Diagram User:**

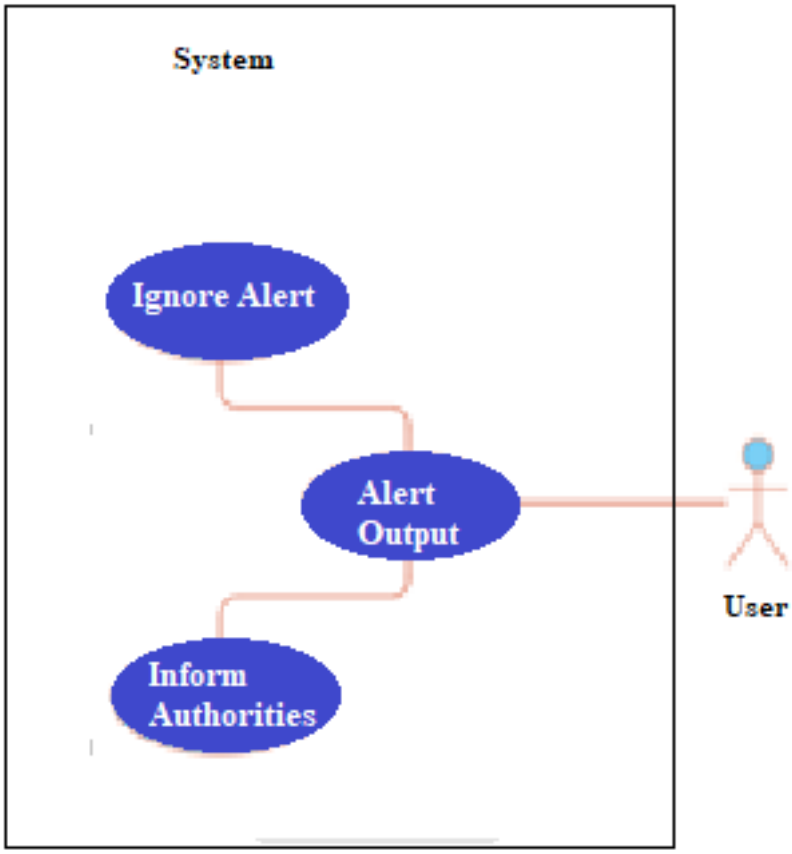


Figure 4-4 Use-Case for User

User Module:

Use Case	**Description**
Actor	**Control-room staff**
Pre-condition	Security camera real-time input should be available.
Main Scenario	Control room personnel takes account of the alert.
Extension Scenario	If the input recording video format is incompatible. Error Message Displayed.
Post-Condition	Situation detection and alarm generation.

4.6.3 **Use Case Diagram for System:**

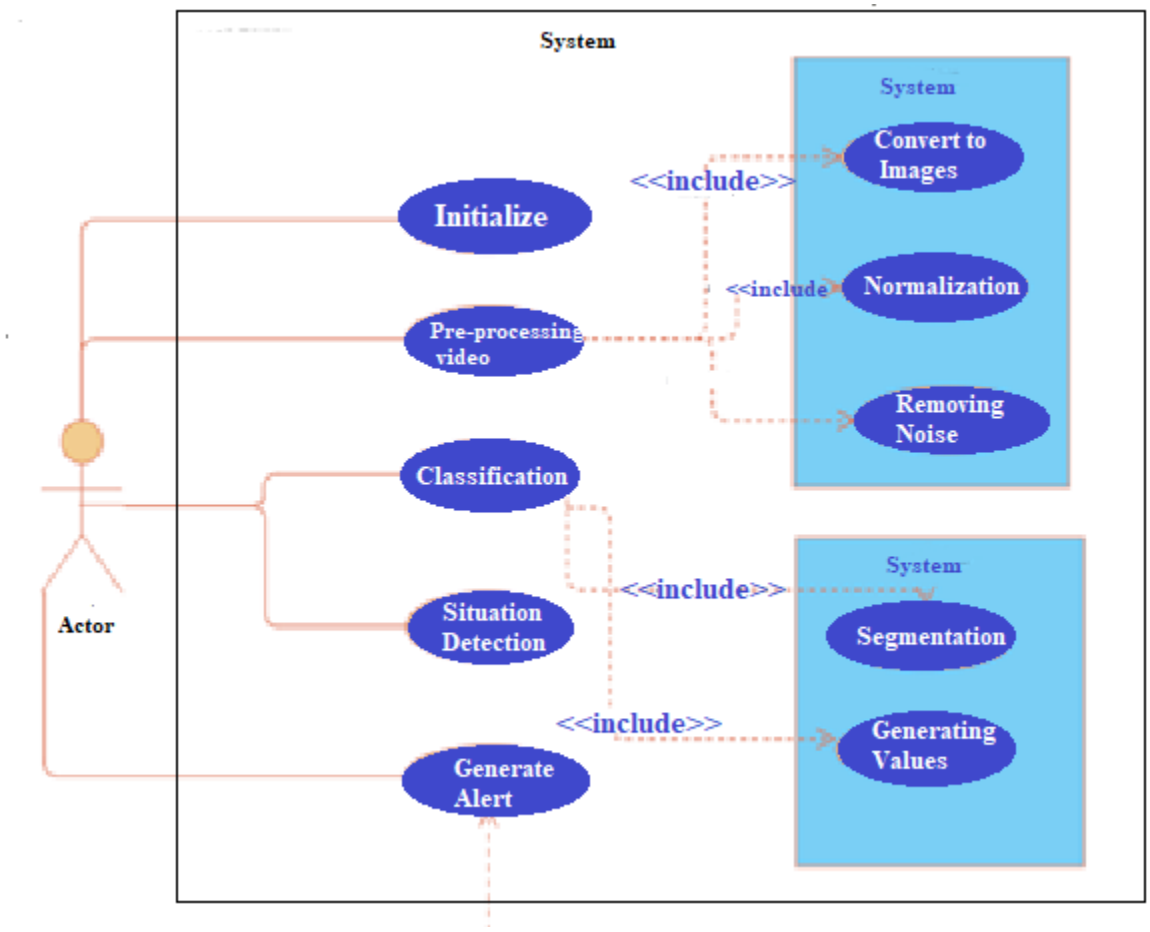


Figure 4-5 System Use-Case

Pre-processing Module:

Table 4-8 System Pre-processing Module

Use Case	**Description**
Actor	**System**

Pre-condition	input video.
Main Scenario	Pre-processing is carried out by converting the video to images.
Post-Condition	Classifying image before detection.

Classification Module:

Use Case	**Description**
Actor	**System**
Pre-condition	Pre-processed video should be available.
Main Scenario	Classification of images. Allocation of transfer values and segmentation into groups.
Post-Condition	Situation detection(if any) and Alarm generation.

4.7 Sequence Diagram:

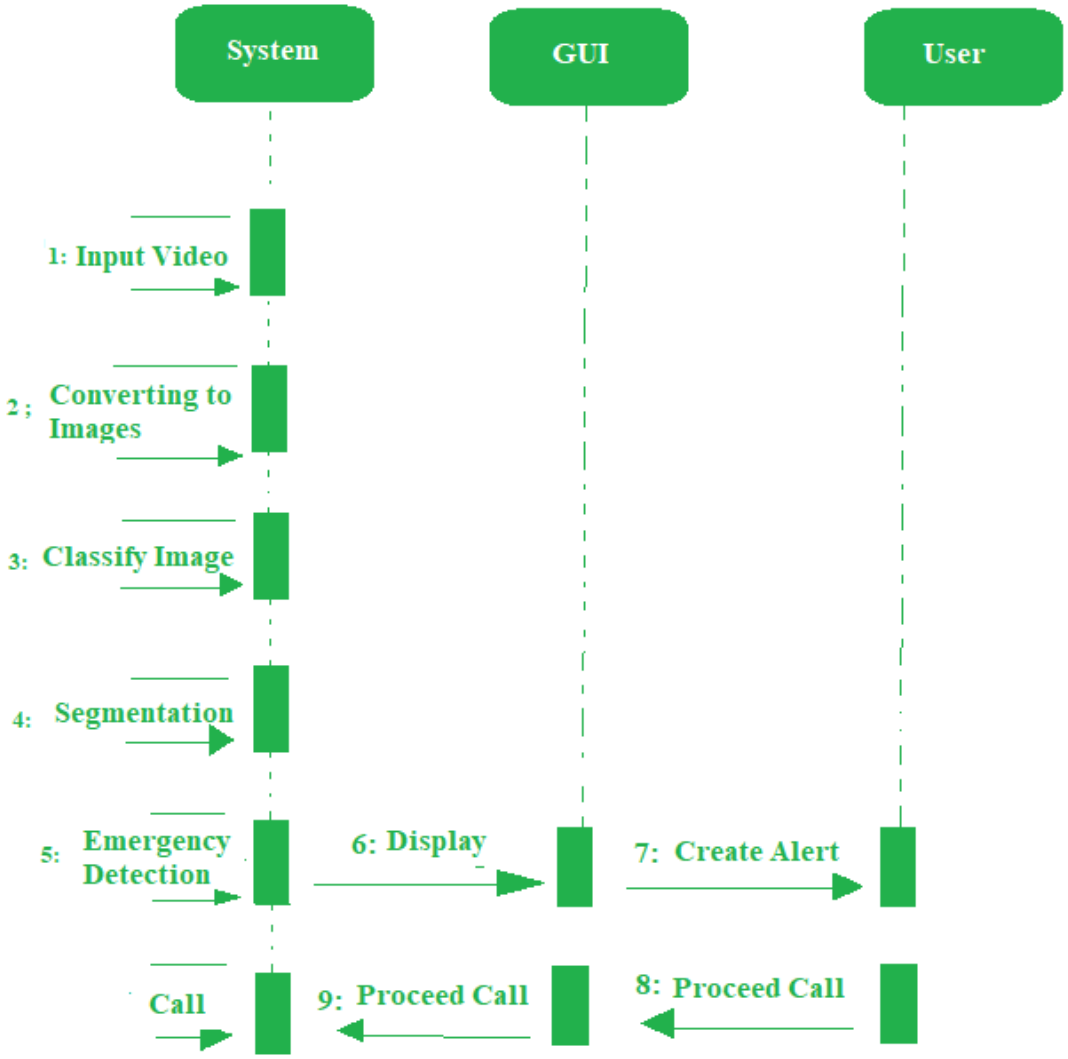


Figure 4-6 Sequence Diagram

4.8 Activity Diagram:

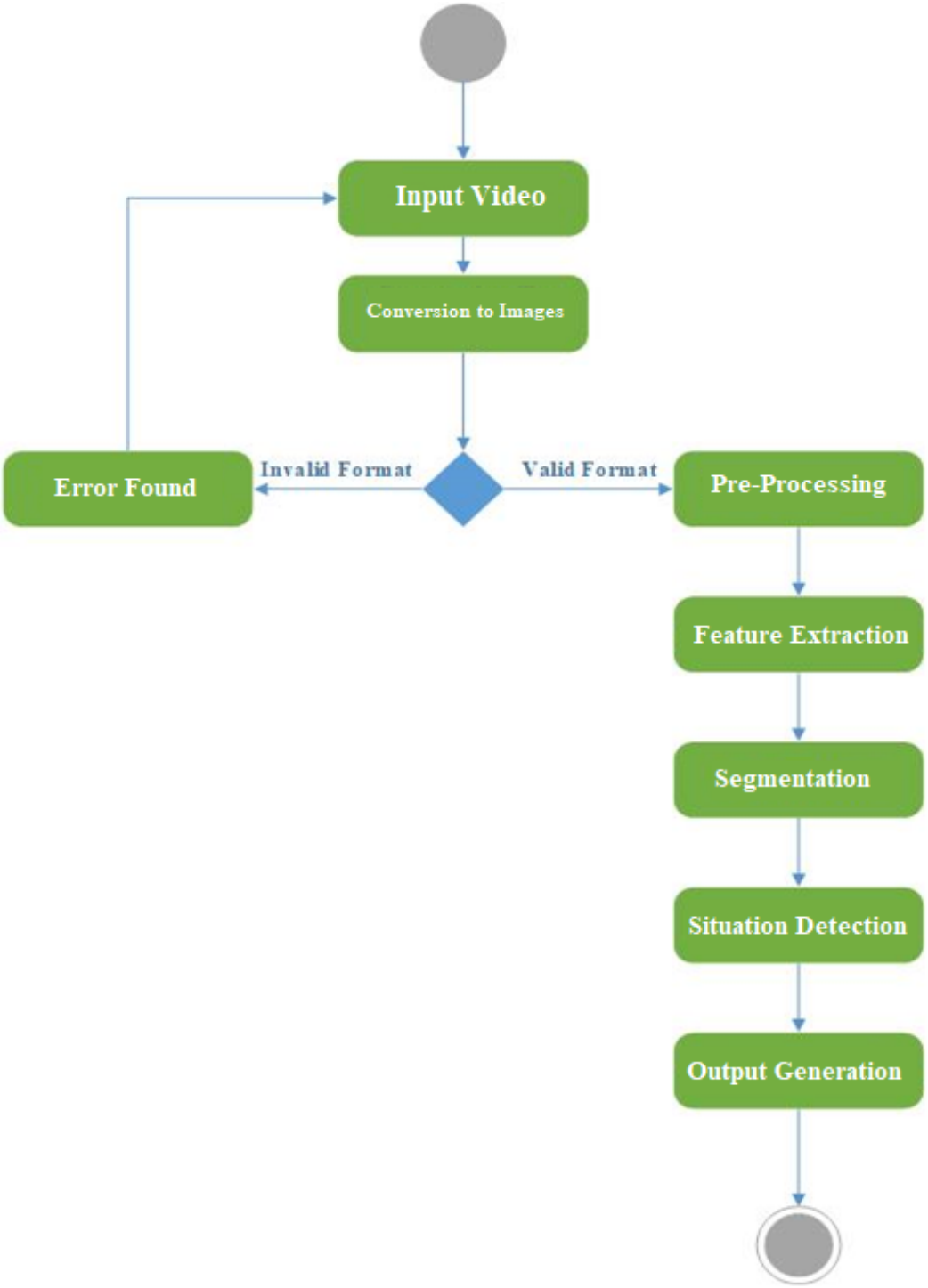


Figure 4-7 Activity Diagram

Chapter 5 DATASET & PREPROCESSING

5.1 Implementation:

An image with minimum acceptable resolution of 640 x 480 pixels is considered in this project for the sake of training, as the main aim of this project is to any emergent activity in videos and real-time surveillance videos. Implementation of this project is done in python and test images and videos are taken from usde, kaggle, and medium , videos were made on mobile phone camera with setting of 1280 x 720 pixels. Compared to the size of total image, size of the detection classes like fire , explosion and accident could be very small and therefore is considered to be present in the window of size not more than 100 x 100 pixels.

The proposed system consists of various steps which are discussed below:

5.1.1 **Fire Detection:**

- **Segmentation of fire or explosion :** for the purpose of segmentation 100 x 100 pixels sliding windows was used. For detecting fire in the image of resolution 640 x 480 pixels image this sliding window was started from top leftmost pixel and each segmented image was tested for the presence of flame or blaze. After classification of the segment the sliding window was moved horizontally by 20 pixels. Then the sliding window was shifted 20 pixels in the vertical direction and scanning was started from the left most pixel. This process was continued till end of the image was reached.
- **Feature extraction for fire image**: After extraction of 100 x 100 pixel segments through sliding window mechanism one after another, corresponding hog features was calculated for each segment. Horizontal and vertical derivatives for the segmented images was calculated. Then the 100 x 100 pixels segments were divided into cells of 8 x 8 pixels containing 64 pixels in each cell. Unsigned orientation with 9 histogram bins was used. Therefore for each cell 9 bin histogram representing orientations ranging from 0 to 180 degree was computed. Rectangular blocks were formed using 4 cells each with 50% overlapping and cells in the block were normalized. For each 100 x 100 pixel image a total

of 121 overlapping blocks were formed. Therefore a feature vector of length 4356 bins was obtained.

- **Neural network for detection of fire class:** the classification of a single flame object was done after taking in account the training resu8lts of several different classifiers and neural networks was designed that had the least error rate and the highest accuracy. The parameters of each classifier are as:

1 : Classifier A

- First hidden layer: 32 neurons.
- Second hidden layer: 32 neurons.
- Positive training examples: 1000.
- Negative training examples: 1000.

2: Classifier B:

- first hidden layer: 32 neurons.
- second hidden layer: 64 neurons.
- positive training examples: 990.
- negative training examples: 990.

3: Classifier C:

- first hidden layer: 32 neurons.
- second hidden layer: 32 neurons.
- positive training examples: 500.
- negative training examples: 500.

4: Classifier D:

- first hidden layer: 32 neurons.
- second hidden layer: 64 neurons.
- positive training examples: 500 images with plain background.
- negative training examples: 500.

With the increase in number of neurons in each layer, complexity and computation time for testing and training the network increases but the performance of the network does not changes after a certain level. If the number of neurons used were very low neural network would not be capable to classify images properly. Therefore, an optimum number is selected by experimentation.

Different combinations of neural network with different number of neurons in the hidden layer were used to train the neural network. All the neural networks were trained with the learning rate of 0.01 and bipolar sigmoid function was used as an activation function.

The classifier B has 96 neurons in the hidden layer as compared to the classifier A with 64 neuron in the hidden layers. Due to greater amount of neurons in the hidden layers the classifier B has correctly classified greater number of images as compared to the number of images correctly classified by classifier A. If the number of neuron in the hidden layer are further increased, computation time increases while the performance of neural network would not increase. In classifier C, number of training samples are reduced. By reducing the size of training set , false positive rate increased. In classifier D object with only plane background images were considered by which performance level decreased drastically due to low training and the neural network could not classify images with background.

5.1.1 Detection of accident In Image

In this project the neural network was trained such that if the output of the activation function of output neuron was greater than threshold , output of the function was marked as collision and a bounding box was drawn surrounding deformed or collided vehicles. If the value was less than the threshold then the considered segment was marked as safe and discarded. Threshold was chosen as 0.8 because neural network was trained to output a value close to 1 when accident was present and close to 0 when the output was negative sample.

Figure 5-1 Accident Detection

5.1.2 Accident Detection In Recorded Video

Implementation of the proposed algorithm was done on recorded video. Video was recorded by city cctv cameras and also by me with the help of mobile camera with 640 x 480 pixels resolution to test the performance of the algorithm on videos. As background subtraction method was found to be faster among the three methods this method was used in detection of knife in the video. Two videos with different background were created and tested. Videos were recorded with 25 FPS and both videos were approximately 30 seconds in length. Testing was done by extracting the frames from the videos and subtracting the frame under consideration with the reference background. After detecting the knife in each frame it was observed that some of the false positive appeared at different positions in each frame. Hence to reduce the number of false positives 4 consecutive frames were compared with each other and the detected segment with common coordinates were marked with each other and the detected segments with common coordinates were marked as positives. By this method I was able to reduce the number of false positives but I was not able to discard all false positives.

5.1.2 **Abnormal Activity Detection:**

Segmentation For Abnormal event: Abnormal event included stealing, fighting, vandalism and rapid irregular moment of people caused by any unpleasant event. For the purpose of segmentation 100 x 100 pixels sliding windows was used. For detecting event in the video of resolution 640 x 480 pixels image this sliding window was started from top leftmost pixel and each segmented image was tested for the presence of any activity. After classification of the segment the sliding window was moved horizontally by 20 pixels. Then the sliding window was shifted 20 pixels in the vertical direction and scanning was started from the left most pixel. This process was continued till end of the image was reached.

Neural Network for abnormality Detection: The classification of a objects was done after taking in account the training resu8lts of several different classifiers and neural networks was designed that had the least error rate and the highest accuracy. With the increase in number of neurons in each layer, complexity and computation time for testing and training the network increases but the performance of the network does not changes after a certain level. If the number of neurons used were very low neural network would not be capable to classify images properly. Therefore, an optimum number is selected by experimentation. Different combinations of neural network with different number of neurons in the hidden layer were used to train the neural network. All the neural networks were trained with the learning rate of 0.01 and bipolar sigmoid function was used as an activation function.The classifier has 80 neurons in the hidden layer as compared to the classifier with 50 neuron in the hidden layers. Due to greater amount of neurons in the hidden layers the classifier has correctly classified greater number of images as compared to the number of images correctly classified by classifier. If the number of neuron in the hidden layer are further increased, computation time increases while the performance of neural network would not increase. In classifier, number of training samples are reduced. By reducing the size of training set, false positive rate increased. In classifier object with only plane background images were considered by which performance level decreased drastically due to low training and the neural network could not classify images with background.

Abnormal event Detection In Video: Implementation of the proposed algorithm was done on recorded video. Video was recorded by surveillance cctv cameras with 640 x 480 or brtter pixels

resolution to test the performance of the algorithm on videos. As background subtraction method was found to be faster among the three methods this method was used in detection of gun in the video. Two videos with different background were created and tested. Videos were recorded with 25 FPS and both videos were approximately 30 seconds in length. Testing was done by extracting the frames from the videos and subtracting the frame under consideration with the reference background. After detecting the any activity in any frame it was observed that some of the false positive appeared at different positions in each frame. Hence to reduce the number of false positives, 4 consecutive frames were compared with each other and the detected segment with common coordinates were marked with each other and the detected segments with common coordinates were marked as positives. By this method I was able to reduce the number of false positives but I was not able to discard all false positives.

5.1.3 **Detection of Fire & Accidents:**

In this approach, to design a multiclass classifier to detect both the objects namely fire and collided vehicles were the target. For this purpose a multiclass network is to be designed that classifies the desired semantic objects, the model is similar to that of OCR. To detect both the objects with a single network a four layer multi class classifier pre-trained on the DCSASS dataset was chosen and Faster R-CNN Inception-v2 was selected. The model was applied the technique of Transfer Learning to fine tune and the weights were adjusted to our needs and the classes variables were configured. The model was then given the TF-records which are the weights on which the model starts its training. On these weights the model starts training and the training was continued for more than 7 hours till the point the losses were constantly reduced to less than 0.05. the training was stopped and the inference graph that were generated were stored locally and the model was then ready to make detections on the basis of the training.

Figure 5-2 Accident Without Detection

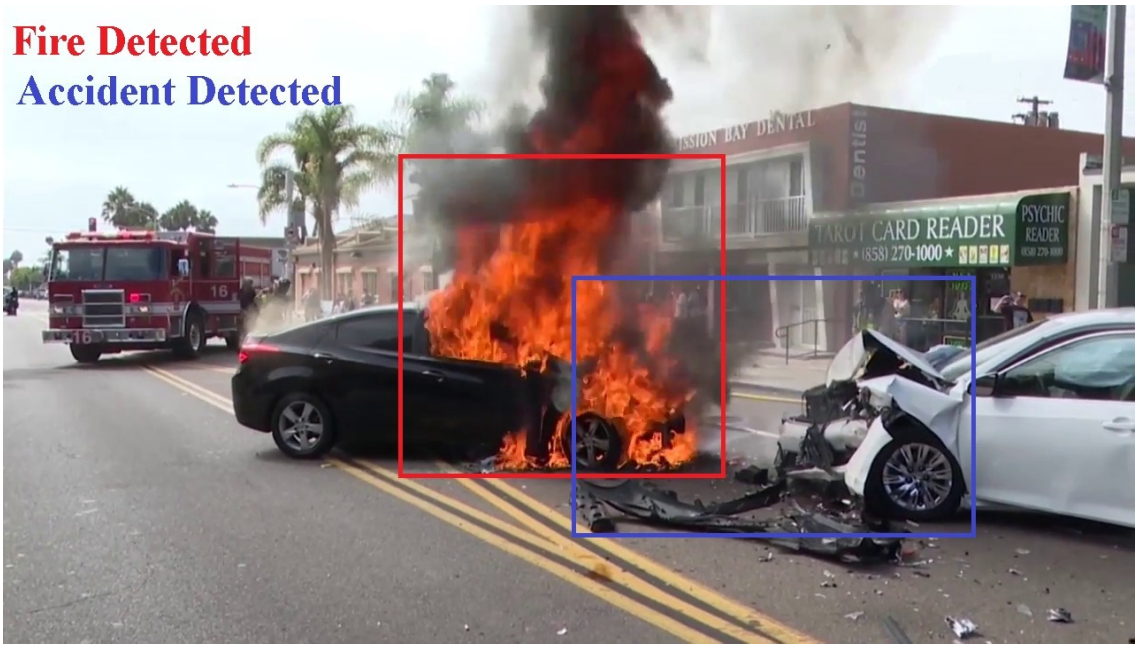


Figure 5-3 Accident With Detection

5.2 Algorithm:

Traditionally situation classification was implemented using simple template matching techniques. In these algorithms, target objects used to be cropped and using specific descriptors like HOG and SIFT, features for the same used to be generated. The approach subsequently used a sliding-window on the image and compared each location with the database of object feature vectors. Enhanced algorithms have used classifiers, like trained SVM classifiers to replace the use of these databases. Since objects will be of different sizes, different window sizes and image sizes. These complex pipelines managed partly solved the object-detection problem, but had many drawbacks. The pipelines were computationally time consuming, and the hand engineered features using algorithms like HOG and SIFT were not highly accurate.With the advent of the use of deep learning in machine-vision and the staggering results of its algorithms deep-learning solutions to solve object detection problems were considered.

One of the first important algorithms to solve object detection using deep learning was R-CNN. Object detection is still a relatively unsolved problem in machine-vision with a lot of improvements expected to come over the next few years. While image classification accuracy rates are touching an 'top 5 error rate' of 2.25% object-detection algorithms are still in early stages of development, with state of the art object detection algorithms achieving only 40.8 mAP (100 is the max) on the COCO data-set. In these situations, understanding where situation-detection algorithms fail and carefully curating data-sets for the particular use case is a a well-followed approach to achieve optimal results.This project will discuss how the deep learning algorithms work and which algorithm I selected to perform the task of object detection, whats is basic architecture and how it classifies the objects into different classes and how it can be changed according to our use case. The Project used the faster R-CNN network, but before explaining its architecture it is essential to see the difference and similarities between its other variants like R-CNN, Fast R-CNN.

5.2.1 **R-CNN:**

To build an R-CNN object-detection pipeline, the following approach can be used:

- The approach starts with a standard network like VGG or ResNet pre-trained on Image-net this network will act like a feature extractor for the image. The approach removes the class specific classification layer and uses the bottleneck layer to extract features from the image. In R-CNN the approach has used VGG network and we get a 4096-dim vector for each image proposal.
- Make sure that the features are extracted accurately, they train the network using these warped images. While training VOC data-set, they place 20+1 (n_class + background) layer at the end and train the network. Each batch size contains 32 positive windows and 96 background windows. A Region proposal is said to be positive if it has an IoU ≥ 0.5 with the ground truth box. Usually obtain these 128 (32+96) proposals from 2 images (batch size). Since there will be ~2k proposals from each image, we sample the positive images and negative images(background) images separately. We use selective search to generate these proposals.

- After Fine-tuning the network, we will send the proposals through the network and obtain a 4096 dim vector. The authors of this paper performed a grid search on IOU to select the positive samples, IOU of 0.3 worked well for them.
- For each object class, train a SVM (one versus other) classifier. You can use hard negative mining to improve the classification accuracy.
- We also train a bounding box regressors to improve upon the localization errors. This is applied on the proposal once the class specific SVM classifier classify the object.

5.2.2 **Testing the algorithm:**

- For testing, R-CNN generates around 2000 category-independent region proposal from the input images and videos, extracts a fixed-length feature vector for each proposal using a CNN (VGG Net), and then classifies each region with category specific linear SVM's. This gives class specific 'classification' score for each proposal, which are then sent a non-maxim suppression algorithm.
- Test time inference takes 13 sec per image on GPU and 53s/image on CPU.
- Sending ~2000 proposals to the Neural network, thus bringing test time to 13 sec on GPU.
- Complex pipeline for training and inference. No end to end training pipeline. Neural network is trained separately, SVM classifiers are trained individually.

5.2.3 **3D-CNN:**

3D-CNN network works in the following way

- The network processes the whole image to produce a convolutional feature map. Then for each object proposal ROI pooling layer extracts fixed-length feature vector, which is finally passed to subsequent FC layers.
- The FC layers branch into two sibling output layers, one that estimates softmax probability over K+1 object classes, and another layer producing the refined bounding box positions.

- 2000 proposals of particular image are not passed through the network as in R-CNN, Instead, The image is passed only once and the computed features are shared across ~2000 proposals like the same way it is done in SPP Net.
- Also, the ROI pooling layer does max pooling in each sub-window of approximate size h/H X w/W. H and W are hyper-parameters. It is a special case of SPP layer with one pyramid level.
- The two sibling output layers' outputs are used to calculate a multi-task loss on each labeled ROI to jointly train for classification and bounding-box regression.
- They have used L1 loss for bounding box regression as opposed to L2 loss in R-CNN and SPP-Net which is more sensitive to outliers.

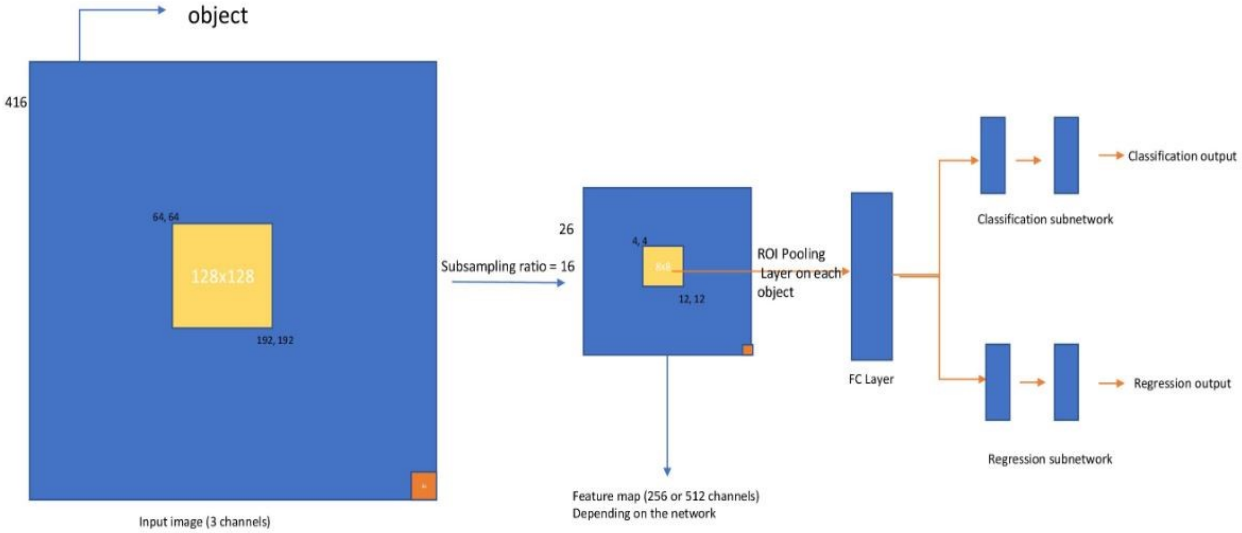


Figure 5-4 RoI Pooling

RoIPooling:

Region of Interest pooling, It is important to understand as it is used in both Faster R-CNN and Mask R-CNN. Since objects are of different sizes, the pooling layer needs to be applied on different sizes of feature maps. Suppose in the above diagram the object location [64, 64, 192, 192] on image is [4, 4, 12, 12] on feature map. object location [128, 128, 384, 400] on image is [8, 8, 24, 25] on feature map. Now both pooling layer

should be applied on both [4, 4, 12, 12] and [4, 4, 16, 18] and extract a fixed length feature vector.

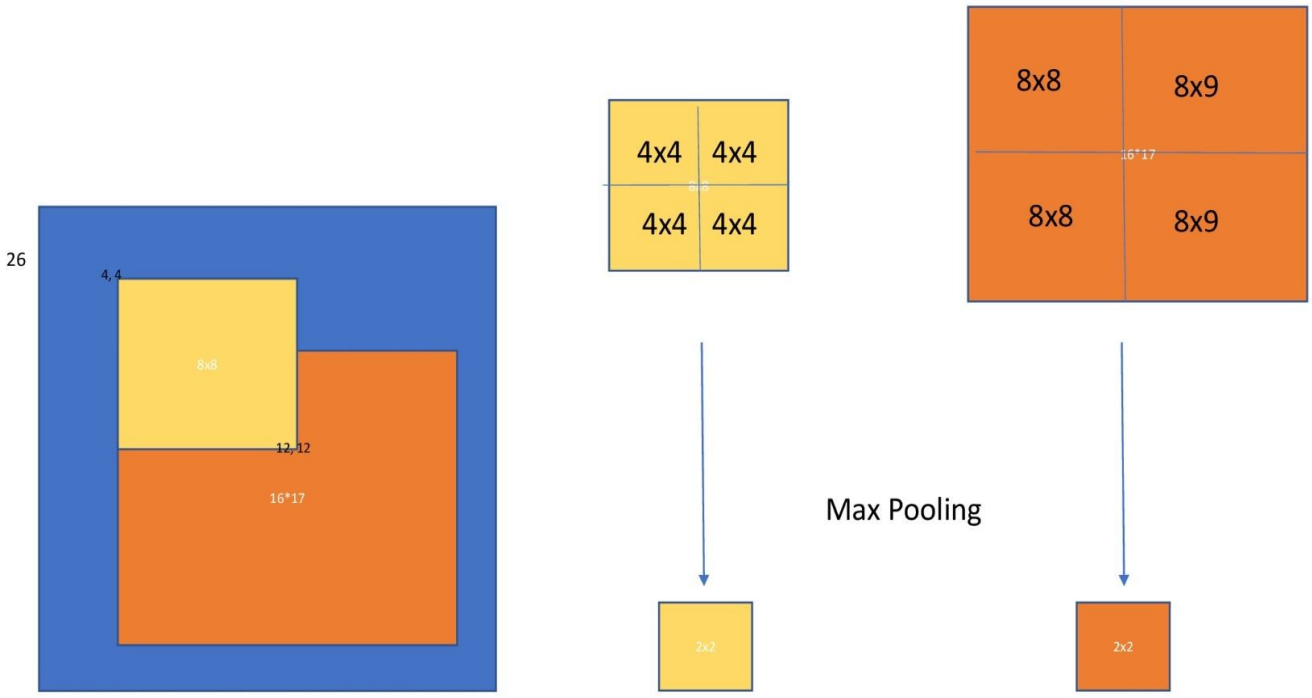


Figure 5-5 Fast R-CNN

Loss Functions:

3D-CNN network has two sibling networks as discussed above, the classification layer outputs discrete probability scores and the regression layer outputs offsets of x, y, w, h values of the object. For classification we can use cross entropy loss. for regression they use smooth L1 loss.

Training:

While training 3D-CNN uses batch size of 2. Since for a batch size of 2 we have ~2000 x 2 = 4000 boxes and majority of them won't contain the object, there will be extreme

class imbalance while training this network. For this sake, we randomly sample 64 +ve boxes and 64 -ve boxes from the ~4000 boxes and compute losses only for these boxes. Incase if we have less +ve boxes, we will pad with negative boxes. In total we will have 128 boxes for each iteration. The same approach is used in Faster R-CNN also.

Encoder:

A Proposal is said to be foreground if it has iou> 0.5 with any of the bounding box, we will assign class to the proposal accordingly. proposals which has [0.1, 0.5) are termed to be negative. Remaining all proposals are ignored.

Results:

- Training is 9x faster than 3D-CNN. While inference Fast R-CNN is 213x faster than R-CNN. Similarly using VGG16, Fast R-CNN is 3x faster while training and 10x faster while inference compared to SPP Net.
- mAP on VOC-2007–12 dataset achieves 71.8%. On COCO Pascal-style mAP is 35.9. The new coco-style mAP, which averages over IoU thresholds, is 19.7% on DCSASS dataset.

5.3 Slow-Fast Model

This approach is still the best choice for most of the researchers and has achieved incredible results. The algorithm has surpassed all the previous results in-terms of both accuracy and speed. Slow-Fast model also has come up with new techniques which have become gold standards for all upcoming frameworks. Lets have deep look into these methods.

5.3.1 **Changes to the Slow-Fast Model:**

These are the two major changes brought by Faster R-CNN. Anchor boxes became very common from here for all the frameworks. RPN network can work as single object detector or generate proposals for Slow-Fast network. One thing is for sure, we have removed the traditional computer vision techniques completely and made a full fledged Deep neural network which gets trained end to end.

Removed Selective search and added a Deep Neural network for generating proposals

5.3.2 **Workflow of RPN:**

Take an input image of 800x800 and send it to the network. For a VGG Network, after subsampling ratio of 16, the output will be [512, 50, 50]. An RPN network is applied on this feature map, which generates (50*50*9) boxes regression and classification scores. So regression output is 50*50*9*4 (x,y, w,h) and classification output is 50*50*9*2. Here 9 implies the number of anchor boxes at each location. Below is the procedure on how anchor boxes are generated on the image.

- RPN produces object proposals wrt to each anchor box along with their objectness score. An anchor box is assigned +ve if it has max_iou with the ground truth object or iou greater than 0.7. An anchor box is assigned negative if it has iou< 0.4. All the anchor boxes with iou [0.4, 0.7] are ignored. Anchor boxes which fall outside the image are also ignored.
- Again, since vast majority of them will have negative samples, we will use the same Fast R-CNN strategy to sample 128+ve samples and 128-ve samples (total 256) for a batch size of 2 for training.
- Smooth L1 loss and cross entropy loss can be used for regression and classification.
- Once RPN outputs are generated, we need to process them before sending to the RoI pooling layer (aka fast R-CNN network) . The Slow-Fast model says, RPN proposals highly overlap with each other. To reduced redundancy, we adopt non-maximum supression (NMS) on the proposal regions based on their cls scores. We fix the IoU threshold for NMS at 0.7, which leaves us about 2000 proposal regions per image. After an ablation study, the authors show that NMS does not harm the ultimate detection accuracy, but substantially reduces the number of proposals. After NMS, we use the top-N ranked proposal regions for detection. In the following we training Slow-Fast model using 2000 RPN proposals. During testing they evaluate only 300 proposals, they have tested this with various numbers and obtained this.

5.3.3 **Results:**

- Inference at 5 FPS on a GPU.
- Using VGG backend, with pretraining on ImageNet and COCO, On VOC Faster Slow-Fast has achieved 73.2% mAP.
- On DCSASS dataset, Slow-Fast model has achieved 21.9 mAP.

5.3.4 **Slow-Fast model for RPN:**

In Slow-Fast model, a 3x3 conv layer slides over the feature map and extract features which further are evaluated by two sibling networks, one for regression (1x1 conv) and the other for classification (1x1 conv). Now while implementing FPN, we build the same network, but here we move the network on each and every feature map.

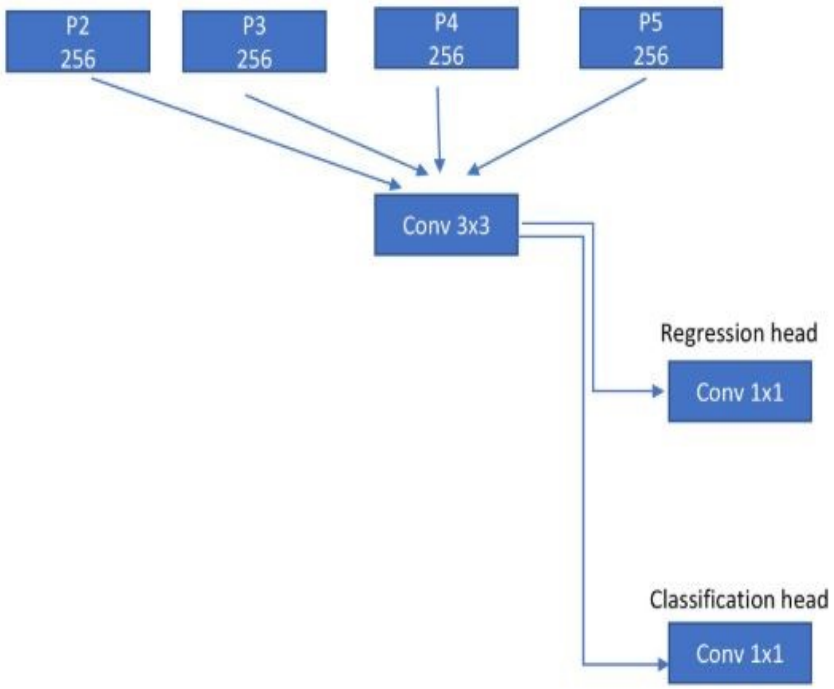


Figure 5-6 Region Proposal Network

5.4 Code Details**:**

Before discussing the code details I would like to state how to setup the project use case with some sample dataset.

Use case: Train some sample images of Shirts, T-Shirts and Jeans, so that out model should recognize the Shirt, T-Shirt and Jeans from the given image.

Dataset Preparation: We downloaded some sample images from google and assigned labels for those images. Labelling can be done with the tool called labeling.

Things to setup before start modelling:

Installation:

Step 1:

Downloading and installing Anaconda with Python 3.6/3.7 here and create a virtual environment by issuing the following commands respectively.

1. C:\>conda create -n tensorflow1 pip python=3.6
2. C:\> activate tensorflow1

Step 2:

1. System with NVIDIA Graphics: Install TensorFlow-GPU, CUDA and cuDNN by following the given steps.
2. System with normal Intel Graphics: Install regular TensorFlow and you do not need to install CUDA and cuDNN.

TensorFlow installation can be done by using "(tensorflow1) C:\> pip install — ignore-installed — upgrade tensorflow-gpu" for GPU versions and "(tensorflow1) C:\> pip install — ignore-installed — upgrade tensorflow" for regular versions.

Step 3:

Install the other necessary packages by issuing the following commands:

(tensorflow1) C:\>conda install -c anaconda protobuf

(tensorflow1) C:\> pip install pillow

(tensorflow1) C:\> pip install lxml

(tensorflow1) C:\> pip install Cython

(tensorflow1) C:\> pip install twilio

(tensorflow1) C:\> pip install jupyter

(tensorflow1) C:\> pip install matplotlib

(tensorflow1) C:\> pip install pandas

(tensorflow1) C:\> pip install opencv-python

A few libraries to understand which are necessary for the purpose of model training.

- Protocol Buffers (protobuf): It's a method of translating image files into a format that can be stored in a file, useful in developing programs to communicate with each other over a wire or for storing data. It will emphasize simplicity and performance.
- Pillow: Python Imaging Library (abbreviated as PIL) (in newer versions known as Pillow) is a free library for the Python programming language that adds support for opening, manipulating, and saving many different image file formats.
- LXML: Helps in processing XML and HTML in the Python language.
- Cython: Gives C-like performance with code that is written mostly in Python with optional additional C-inspired syntax.
- Jupiter: Interactive python web application used to create, share, view and compile code.
- Twilio: Twilio's APIs (Application Programming Interfaces) power its platform for communications. Behind these APIs is a software layer connecting and optimizing communications networks around the world to allow your users to call and message anyone, globally.
- Matplot: Plotting library for python helps in providing plots.
- Pandas: Python Data Analysis Library.
- Open-CV: It's a library of Python bindings designed to solve computer vision problems.

Step 4:

Getting tensorflow classification API and creating a working directory in C: and name it "tensorflow1", it will contain the full TensorFlow object detection framework, as well as training images, training data, trained classifier, configuration files, and everything else needed for the object detection classifier.

Step 5:

Getting a pre-trained object detection models on DCSASS dataset from the tensorflow model zoo. Thefaster_rcnn_inceptiont_v2_DCSASS architecture of the model is unique which allows for faster/slower detections and more/less accuracy etc.

Some models have an architecture that allows for faster detection but with less accuracy, while some models (such as the Faster-RCNN model) give slower detection but with more accuracy.

Step 6:

Go to folder named research and then remove all the folders except Object Detection, Slim folders and set.py file.

To train object detector:

- All files in \Situation_Detection\images\train and \ Situation _detection\images\test
- The “test_labels.csv” and “train_labels.csv” files in \object_detection\videos
- All files in \ Situation _detection\training
- All files in \object_detection\inference_graph

Paste our dataset in train and test folders respectively and label the images using Labelimg tool as shown below.

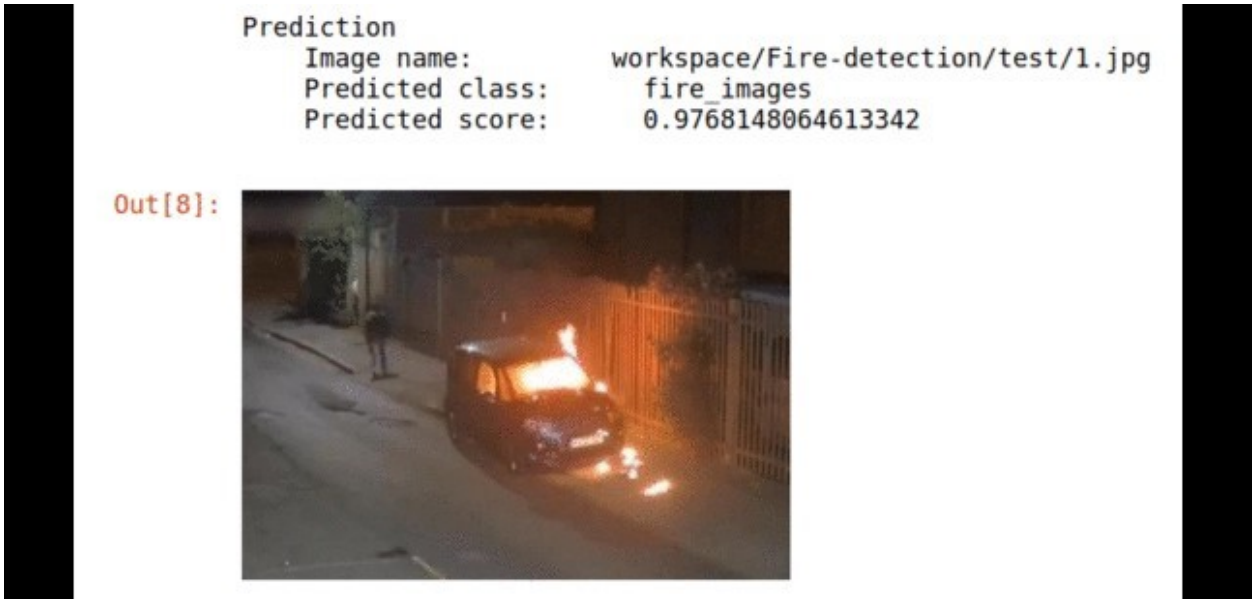


Figure 5-7 labeling Interface

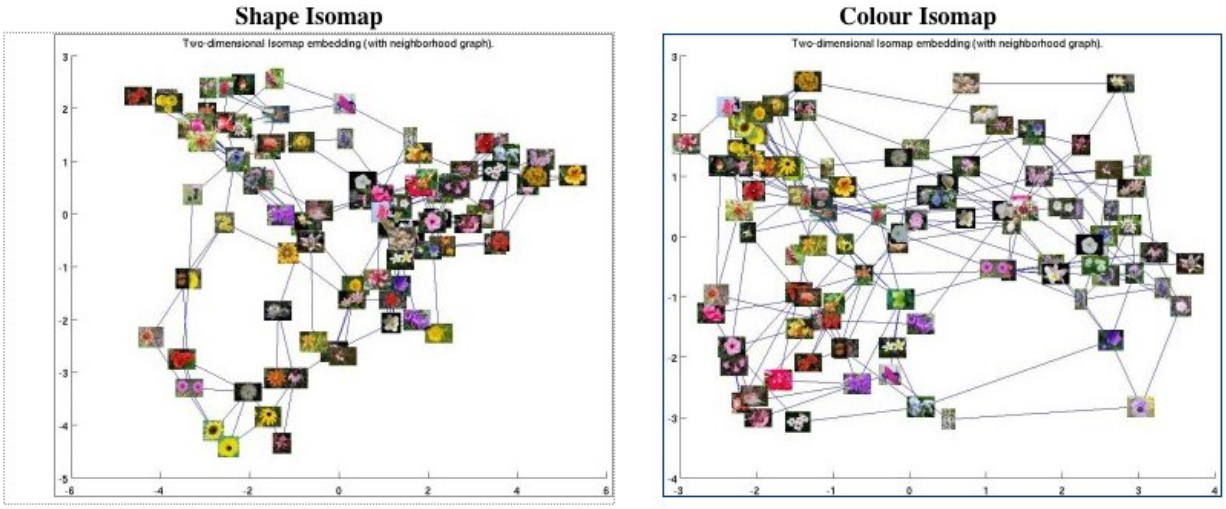


Figure 5-8 Labeling

An xml file contains the label data will be generated for each image and stored in train and test folders respectively. These xml files are converted to csv files using the following python script:

```
import os
import glob
import pandas as pd
import xml.etree.ElementTree as ET
def xml_to_csv(path):
xml_list = []
    for xml_file in glob.glob(path + '/*.xml'):
        tree = ET.parse(xml_file)
        root = tree.getroot()
        for member in root.findall('object'):
            value = (root.find('filename').text,
                     int(root.find('size')[0].text),
                     int(root.find('size')[1].text),
                     member[0].text,
                     int(member[4][0].text),
                     int(member[4][1].text),
                     int(member[4][2].text),
                     int(member[4][3].text)
```

```
                    )
            xml_list.append(value)
            column_name = ['filename', 'width', 'height', 'class', 'xmin', 'ymin', 'xmax',
            'ymax']
            xml_df = pd.DataFrame(xml_list, columns=column_name)
               return xml_df
            def main():
               for folder in ['train','test']:
            image_path = os.path.join(os.getcwd(), ('images/' + folder))
            xml_df = xml_to_csv(image_path)
            xml_df.to_csv(('images/' + folder + '_labels.csv'), index=None)
                  print('Successfully converted xml to csv.')
            main()
```

Figure 5-9 csv to xml

Step 7:

Compiling the Protobuf files, which are used by TensorFlow to configure model and training parameters. Every.proto file in the \object_detection\protos directory must be called out individually by the command. Before hiting the command you should be in this path \research and issue the following command:

"(tensorflow1)C:\>tensorflow1\models\research>protoc —python_out=. .\object_detection\protos\anchor_generator.proto .\object_detection\protos\argmax_matcher.proto .\object_detection\protos\bipartite_matcher.proto .\object_detection\protos\box_coder.proto .\object_detection\protos\box_predictor.proto .\object_detection\protos\eval.proto .\object_detection\protos\faster_rcnn.proto .\object_detection\protos\faster_rcnn_box_coder.proto .\object_detection\protos\grid_anchor_generator.proto .\object_detection\protos\hyperparams.proto .\object_detection\protos\image_resizer.proto .\object_detection\protos\input_reader.proto .\object_detection\protos\losses.proto.\object_detection\p

rotos\matcher.proto .\object_detection\protos\mean_stddev_box_coder.proto .\object_detection\protos\model.proto .\object_detection\protos\optimizer.proto .\object_detection\protos\pipeline.proto .\object_detection\protos\post_processing.proto .\object_detection\protos\preprocessor.proto .\object_detection\protos\region_similarity_calculator.proto .\object_detection\protos\square_box_coder.proto .\object_detection\protos\ssd.proto .\object_detection\protos\ssd_anchor_generator.proto .\object_detection\protos\string_int_label_map.proto .\object_detection\protos\train.proto .\object_detection\protos\keypoint_box_coder.proto .\object_detection\protos\multiscale_anchor_generator.proto .\object_detection\protos\graph_rewriter.proto"

This creates a name_pb2.py file from every name.proto file in the \object_detection\protos folder.

Step 8:

Then, generate the TFRecord files by issuing these commands from the \object_detection folder:

1. Pythongenerate_tfrecord.py — csv_input=images\train_labels.csv — image_dir=images\train — output_path=train.record
2. python generate_tfrecord.py — csv_input=images\test_labels.csv — image_dir=images\test — output_path=test.record.

The label map will tell the trainer what each object is by defining a mapping of class names to class ID numbers. Use a text editor to create a new file and save it as labelmap.pbtxt in C:\tensorflow1\models\research\situation_detection\training folder the file type should be .pbtxt

The label map ID numbers should be the same as what is defined in the generate_tfrecord.py file, the labelmap.pbtxt file will look like below:

```
def class_text_to_int(row_label):
    if row_label == 'gun':
        return 1
elifrow_label == 'knife':
        return 2
    else:
        None
```

Step 9:

The situation detection training pipeline should be configured. It defines which model and what parameters will be used for training. The slow-fast model inception .config file in a text editor. Make some necessary changes to the .config file, mainly changing the number of classes and examples, and adding the file paths to the training data.

Make the following changes to the faster_rcnn_inception_v2_pets.config file.Line 9. Change num_classes to the number of different objects you want the classifier to detect. For the above shirt, t-shirt and jeans detector, it would be num_classes: 2.Line 106. Change fine_tune_checkpoint to:

- fine_tune_checkpoint:
  "C:/tensorflow1/models/research/object_detection/ faster_rcnn_inception_v2_coco_2018_01_28/model.ckpt".
  Lines 123 and 125. In the train_input_reader section, change input_path and **label_map_path:**
  input_path: "C:/tensorflow1/models/research/object_detection/train.record"
  label_map_path:"C:/tensorflow1/models/research/object_detection/training/labelmap.pbtxt"
  Line 130. Change num_examples to the number of images you have in the \images\test directory.Lines 135 and 137. In the eval_input_reader section, change input_path and **label_map_path:**
  input_path: "C:/tensorflow1/models/research/object_detection/test.record"
  label_map_path:"C:/tensorflow1/models/research/object_detection/training/labelmap.pbtxt"
  The training job is all configured and ready to run.

Chapter 6 DEVELOPMENT

6.1 Development of Computer Program:

Computer programming is the process of designing and building an executable computer program for accomplishing a specific computing task. Programming involves tasks such as analysis, generating algorithms, profiling algorithms accuracy and resource consumption, and the implementation of algorithms in a chosen programming language. The source code of a program is written in only one languages for the development.

6.2 Hardware:

The project uses no as such hardware but for the future work a low cost Raspberry Pi Microcomputer and USB or IP-webcam for image processing applications on which the trained model is run for the sake of detection and the model for this purpose should be a faster model like ssd-mobilnet. A Linux based operating system (NOOBS) burned in a memory card with python scripts helps in accessing the GPIO pins for controlling external hardware. A laptop webcam in this project is use for video streaming.

The Raspberry Pi 3B has an embedded Broadcom system, with a BCM2837 SoC chipset. It has a ARM Cortex-A53 instruction set. It has a quad-core 64-bit processor with a processing speed of 1.2 GHz. For loading Linux operating system, it makes use of a SD RAM of 1 GB already present in it. For data storage, it uses an external micro SD card as it does not include a built-in hard drive. It also includes a 10/100 Mbps Ethernet. Another reason for using this model is that it is the most powerful image processing tool incorporating heavy tasks which cannot be carried out using any other high level micro controller. Its Wi-Fi module, USB and image processing cameras, gets itself an edge against other controllers which lack these features.

6.3 Software:

6.3.1 **Why Python?**

It has never as easy as it is now-a-days to take a picture. All it usually needs is a mobile phone. These are the bare essentials to shoot and to view an image. Taking a photograph is

free, if we don't take the costs for the mobile phone into considerations. Just a generation ago, hobby artists and real artists needed special and often expensive and the costs per picture were far from being free.

I took pictures to preserve great moments in time. Pickled memories ready to be "opened" in the future at will.Similar to pickling things, we have to pay attention to the right preservatives. Of course, mobile phone also provide us with a range of image processing software, but as soon as we need to manipulate a huge quantity of photographs we need other tools. This is when programming and Python comes into play. Python and its modules like Numpy,OpenCV,Scipy,lxml, Pandas, sklearn, pillow, cython, Matplotlib and other special modules provide the optimal functionality to be able to cope with the flood of pictures.

- **OpenCV**: Image processing library mainly focused on real-time computer vision with application in wide-range of areas like 2D and 3D feature toolkits, facial & gesture recognition, Human-computer interaction, Mobile robotics, Object identification and others.
- **Numpy and Scipy libraries:** For image manipuation and processing.
- **Scikit** − Provides lots of alogrithms for image processing.
- **Python Imaging Library (PIL) :**To perform basic operations on images like create thumnails, resize, rotation, convert between different file formats etc.
- **Pillow**: Python Imaging Library (abbreviated as PIL) (in newer versions known as Pillow) is a free library for the Python programming language that adds support for opening, manipulating, and saving many different image file formats.
- **Lxml**: Helps in processing XML and HTML in the Python language.
- **Cython**: Gives C-like performance with code that is written mostly in Python with optional additional C-inspired syntax.

6.3.2 **Libraries Used:**

The library that are used in my project areOpenCv.Tensorflow,Keras, pandas, Numpy, Sklearn, Scipy, Cython.

- **OpenCV (Open source computer vision):** is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage then Itseez (which was later acquired by Intel. The library is cross-platform and free for use under the open-sourceBSD license. OpenCV supports the deep learning frameworks TensorFlow, Torch/PyTorch and Caffe.

- **Twilio(Telecommunication Networking):** T**wilio's APIs** (Application Programming Interfaces) power its platform for communications. Behind these **APIs** is a software layer connecting and optimizing communications networks around the world to allow your users to call and message anyone, globally. **Twilio** has a whole host of **APIs**, from **SMS** to Voice to Wireless

- **Numpy:** is the fundamental package for scientific computing with Python. It contains among other things:
    - A powerful N-dimensional array object
    - Sophisticated (broadcasting) functions
    - Tools for integrating C/C++ and Fortran code
    - Useful linear algebra, Fourier transform, and random number capabilities
    - Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

- **lxml:**lxml is a Pythonic, mature binding for the libxml2 and libxslt libraries. It provides safe and convenient access to these libraries using the ElementTree API.It extends the ElementTree API significantly to offer support for XPath, RelaxNG, XML Schema, XSLT, C14N and much more.

6.4 Platform Selected

6.4.1 **User Interface Component**

User interface design or UI design generally refers to the visual layout of the elements that a user might interact with in a website, or technological product. This could be the control buttons of a radio, or the visual layout of a webpage. User interface designs must not only be attractive to potential users, but must also be functional and created with users in mind.

User interface design can dramatically affect the usability and user experience of an application. If a user interface design is too complex or not adapted to targeted users, the user may not be able to find the information or service they are looking for. In website design, this can affect conversion rates. The layout of a user interface design should also be clearly set out for users so that elements can be found in a logical position by the user.

User interface designs should be optimized so that the user can operate an application as quickly and easily as possible. Many experts believe that UI design should be simple and intuitive, often using metaphors from non-computer systems. With a more intuitive user interface design, users will be able to navigate around a software easily, finding the product or service they want quickly. One way to check the intuitiveness of a user interface design is through usability testing.

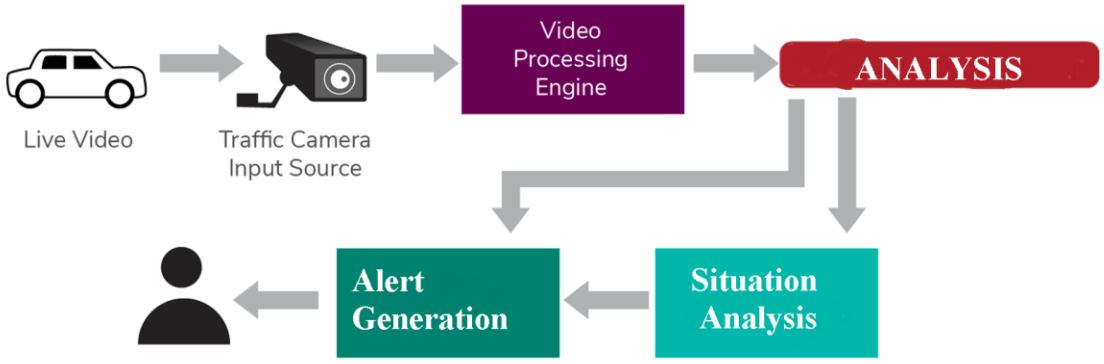


Figure 6-1 Architecture

6.4.2 **Design Concept & Process:**

The prime focus of this project is to introduce a system capable of situation detection and alert generation. My model is programmed with the help of Python programming language and it uses computer vision and machine learning libraries that provide functionality of detecting objects in the images and videos and live streaming videos. The initial step of the segmentation process detects the regions that contain unknown objects. In the segmentation process, regions which contain unknown objects are to be detected initially.The extraction of an object occurs in the next step. It is done after extracting certain features from the regions previously segmented.

After detection, the next process would be the notify the higher authorities about the situation. Here I have made 3 critical sections ,based on the event the personnel on the control room can judge the situation and then react according to the way. The alert receiver has to follow according to the purposed nature of the problem:

1. Is the problem is such alarming to a generate call to the Law-Enforcement?

2 The contact numbers of Law enforcement will be one click away, select the desired agency and just click
3 Automatically, an alert call will be generate to the respective authority with the coordinated of the event.

4 Is the emergency be tackle at individual level?
5 Click the local authority button, and automatically all information will be sent to the local authority.
6 Is the problem be ignorable?
7 Ignore the scenario

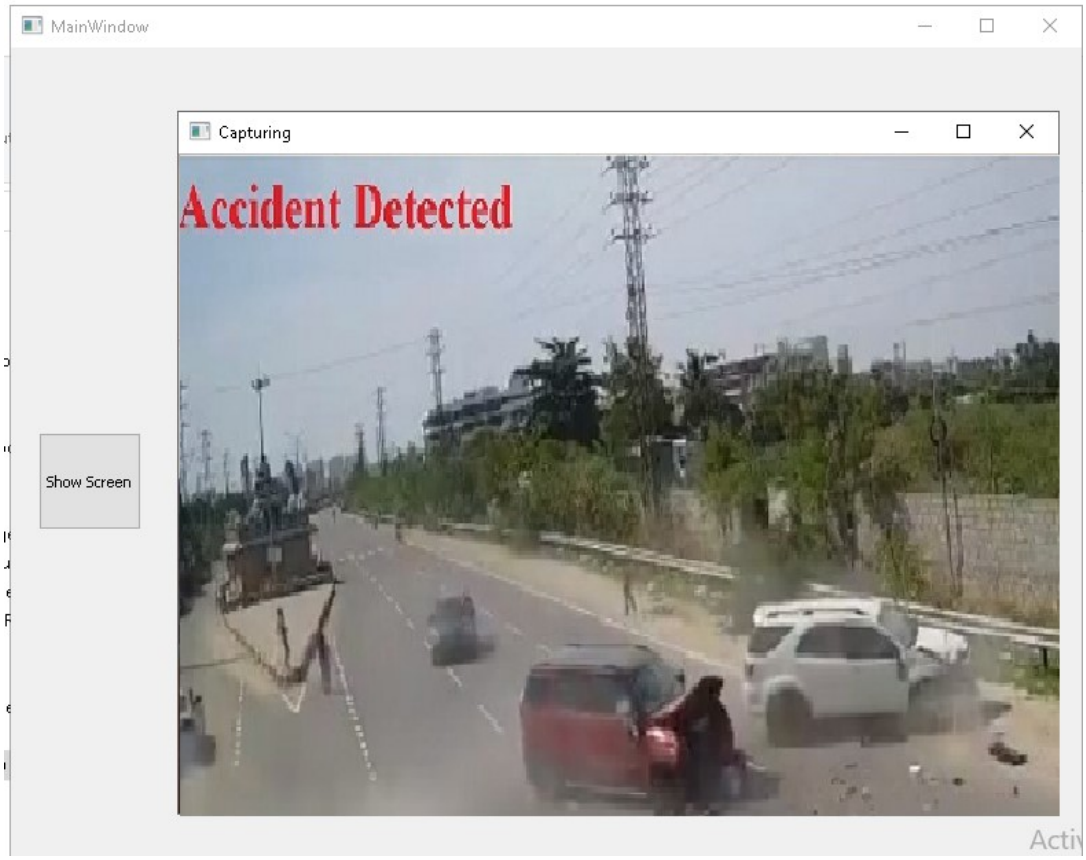


Figure 6-2 Real-time accident detection

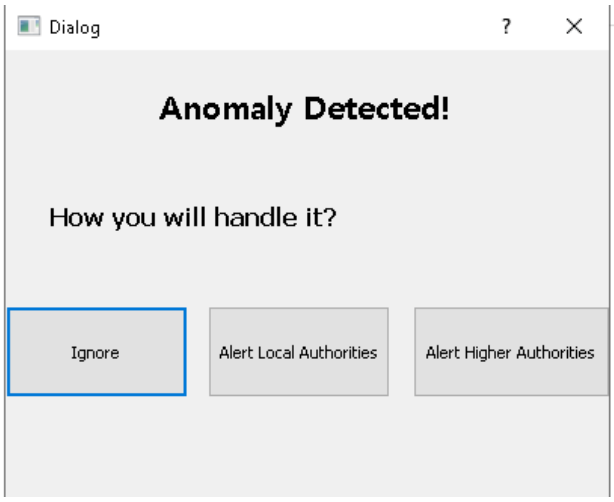


Figure 6-3 Anomaly Alert

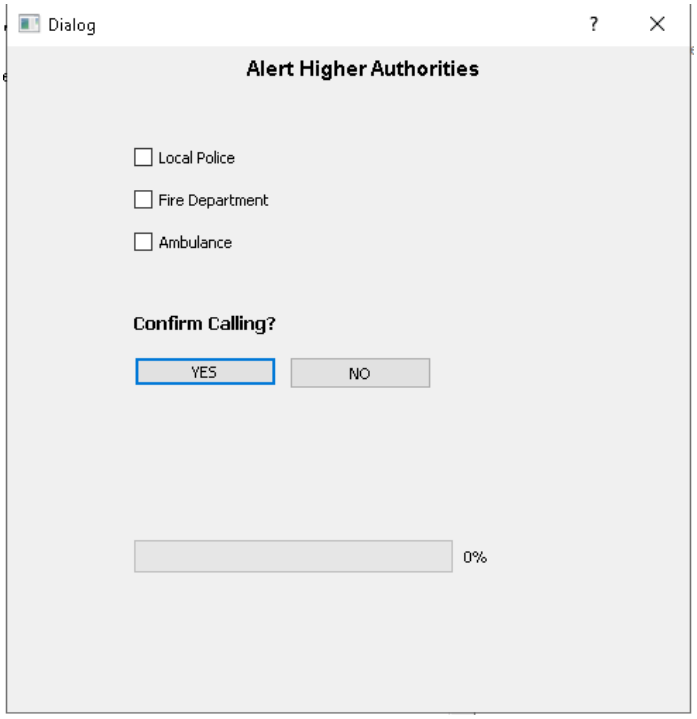


Figure 6-4 Call generation

6.4.3 **Business Logic Layer:**

Business logic is the programming that manages communication between an end user interface and a database. The main components of business logic are business rules and workflows. A business rule describes a specific procedure; a workflow consists of the tasks, procedural steps, required input and output information, and tools needed for each step of that procedure. Business logic describes the sequence of operations associated with the system to carry out the business rule. In our system business rule, we made a system for the solution of the traffic congestion that is the major traffic problem. The image sequences from a camera are analyzed using various edge detection and object counting methods to obtain the most efficient technique. Subsequently, the number of vehicles at the intersection is evaluated and traffic is efficiently managed. The paper also proposes to implement a real-time emergency vehicle detection system. In case an emergency vehicle is detected, the lane is given priority over all the others.

6.5 Managed Complexity

I managed complexity by using different libraries and research on it. I watched different videos, read blogs and research articles nearly related to our problem take help from different person related to that field. Explore research papers related to our project. Read articles on existing system of entry system. Got support from our supervisor who plays an important role for the help of the project. He helped me a lot at every moment. There are four important steps to manage complexity of large projects.

- Diagnose Project Complexity
- Assign Competent Leader
- Select Project Approach
- Manage Complexity Dimensions

6.6 System Development

It is a conceptual model used in project management that describes the stages involved in an information system development project. There are several Software Development Life

Cycle Models, each having its strengths and weaknesses and suitable in different situations and project types. A typical SDLC consist of the following stages including

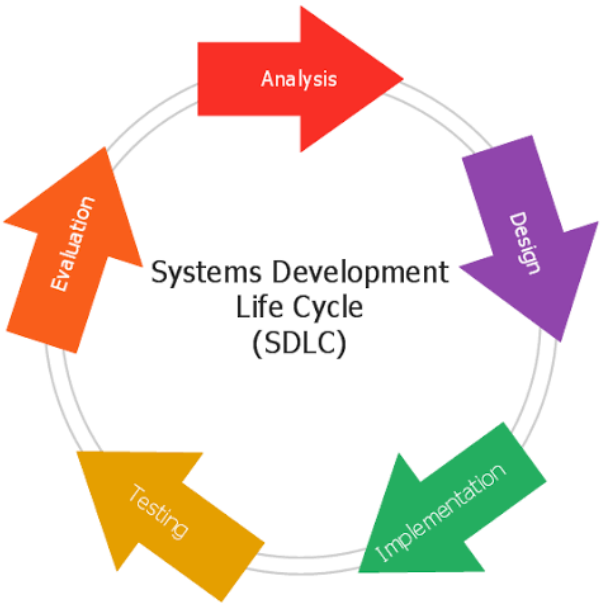


Figure 6-5 Development Lifecycle

The Stages of System Development Life Cycle includes:

1. **Requirement Analysis and Gathering**

   During this phase, all the relevant information is collected from the customer to develop a product as per their expectation. Any ambiguities must be resolved in this phase only.

   Business analyst and Project Manager set up a meeting with the customer to gather all the information like what the customer wants to build, who will be the end-user, what is the purpose of the product. Before building a product a core understanding or knowledge of the product is very important.

2. **Design**

In this phase, the requirement gathered in the SRS document is used as an input and software architecture that is used for implementing system development is derived.

3. Coding or Implementation

Implementation/Coding starts once the developer gets the Design document. The Software design is translated into source code. All the components of the software are implemented in this phase.

4. Testing

Testing starts once the coding is complete and the modules are released for testing. In this phase, the developed software is tested thoroughly and any defects found are assigned to developers to get them fixed.

Retesting, regression testing is done until the point at which the software is as per the customer's expectation. Testers refer SRS document to make sure that the software is as per the customer's standard.

5. Deployment

Once the product is tested, it is deployed in the production environment or first UAT (User Acceptance testing) is done depending on the customer expectation.In the case of UAT, a replica of the production environment is created and the customer along with the developers does the testing. If the customer finds the application as expected, then sign off is provided by the customer to go live

6. Maintenance

After the deployment of a product on the production environment, maintenance of the product i.e. if any issue comes up and needs to be fixed or any enhancement is to be done is taken care by the developers.

6.7 Presentation Layer

Presentation layer implements the functionality required to allow user to interact with the application/system. Operating system in which our project run is as follows:

- Microsoft Windows
- Python installed in a virtual environment

My system requirements of the raspberry pie includes:

- At least 8 GB of RAM
- Minimum of 4 GB of hard disk space
- TensorFlow-GPU
- Nvidia GTX1070 with 8GB of VRAM.

6.8 Program Coding:

6.8.1 **Dataset Collection:**

In the collection step of system design images collected from different sources to train the model for sufficient and strong predictions for testing and training purpose.

In this project, images were collected from Kaggle, DCSASS, USDE and from Github. In this project, gun and knife has its own datasets and they are divided into two different classes. The process of labeling bounding boxes on the dataset of guns and knife was based on indoor settings. Different objects other than the objects of interest are present in the images so that the training process can execute in a proper manner.

No. Images	**Fire**	**Accident**
Training	1000	1000
Testing	300	300
No. Videos	**Anomalies**	**Accidents & Fire**
Training	500	500

Testing	150	150

6.8.2 **Image Labeling:**

In the next step image labeling, a software called LabelImg was used to draw bounding boxes on the images around the objects of interest and giving them a separate class namely fire, accident and aomaly. This tool saves the coordinates of the object in the image in the form of an xml file. Then all the xml files are converted into a CSV file. Once all the images are processed and CSV files are created then next step is to convert training and testing images dataset into TFrecord which is a supported format for TensorFlow for storing image information in a sequence of binary strings.

Figure 6-6 labelImg

6.8.3 **Separation of Dataset:**

There is no fixed rule to divide image dataset into training and testing, normally the common ratios are 80/20 , 90/10 or 70/30 for training and testing dataset. A common rule that is being

widely used when it comes to splitting the dataset is the 70/30 ratio. In this project, the dataset used consists of one thousand images in total and out of them 1000 are fire images and 1000 are knife images.

6.8.4 **Model Training :**

Once the dataset is well prepared, the dataset is then fed into the model to start training. TensorFlow, Google Object Detection API and anaconda virtual environment are chosen to implement this project. The TensorFlow GPU version is used to train the dataset on the local machine. The training goes continued after setting up the environment and installing necessary libraries and pre-requisites. After installing all the necessary components, the training was started and continued for about 9 hours when the losses were constantly below 0.05

The pre-trained model chosen to implement this project was Google Inception-v2, which produces highest accuracy. The accuracy which it achieved at the highest was 87.87%. To use this pre-trained model on DCSASS dataset a Faster R-CNN architecture was used.

The concept of transfer learning was applied, which is to reuse and transfer the knowledge which was previously learnt by the model on a different use case. In order to run the model a configuration file .config is required, this file contains the information about the model such as model type, feature extractor type, adjustable parameter for training and testing, post-processing score converter such as SoftMax etc. Initially configuration file usually contains optimal configuration for previous training on different dataset.

Another necessary file is PBTXT file. This file basically contains an identification number for each class to train in the model. Once all the necessary documents and images are prepared, next step is running the training from the python script and wait for the Global Losses to drop below 0.05.

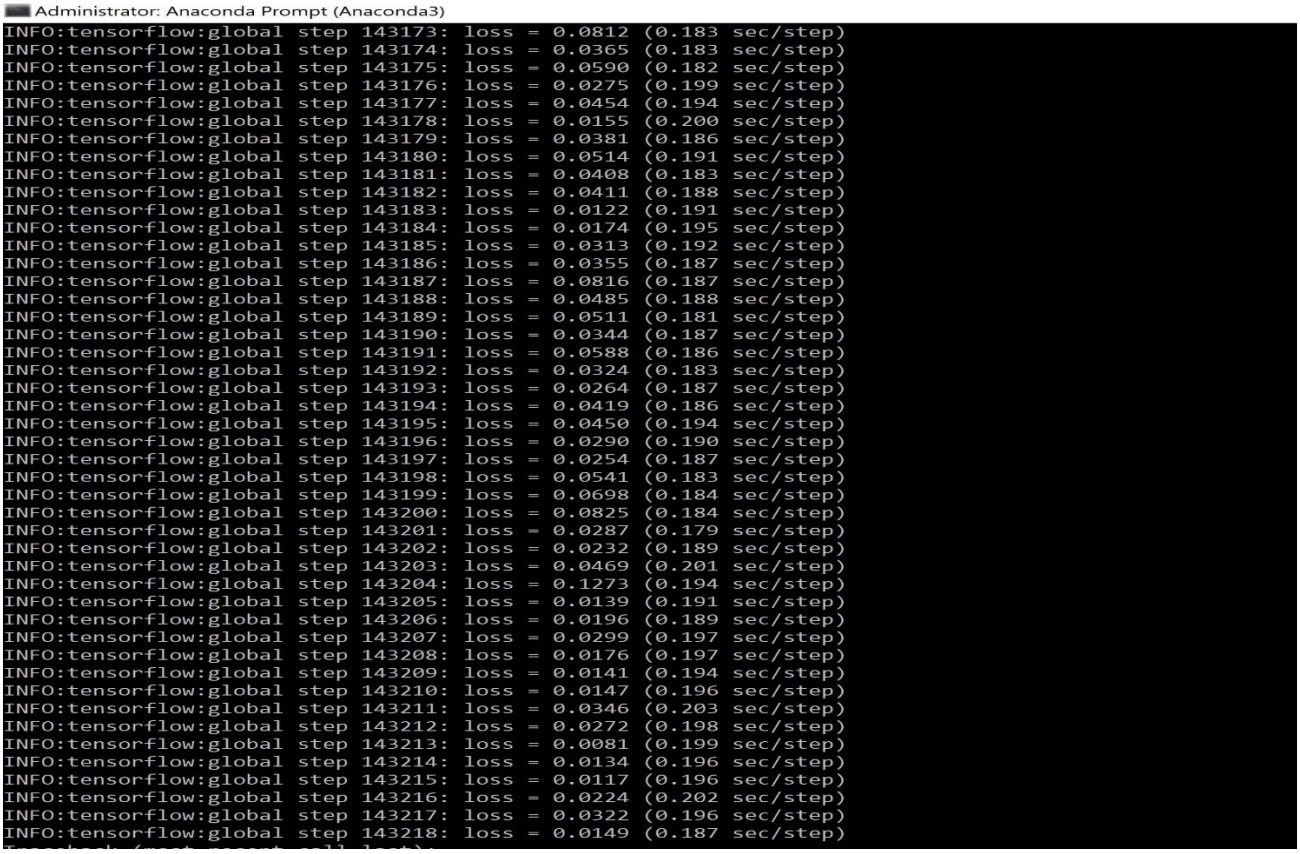


Figure 6-7 Training

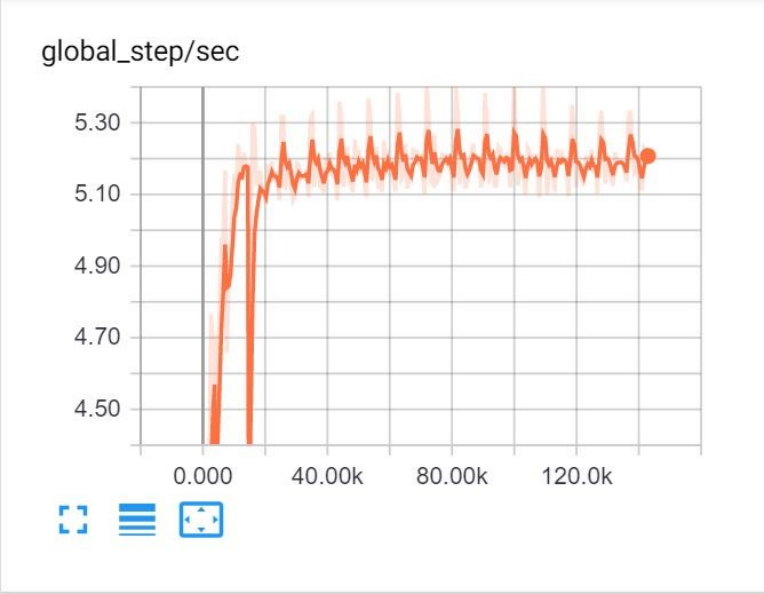


Figure 6-8 Global Training accuracy

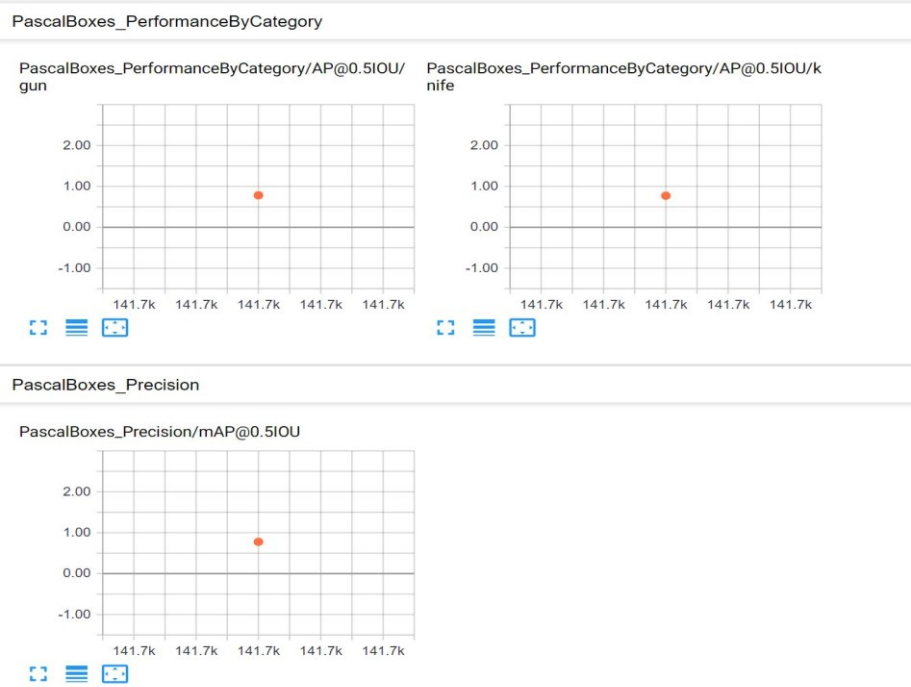


Figure 6-9 Global static Viewpoint

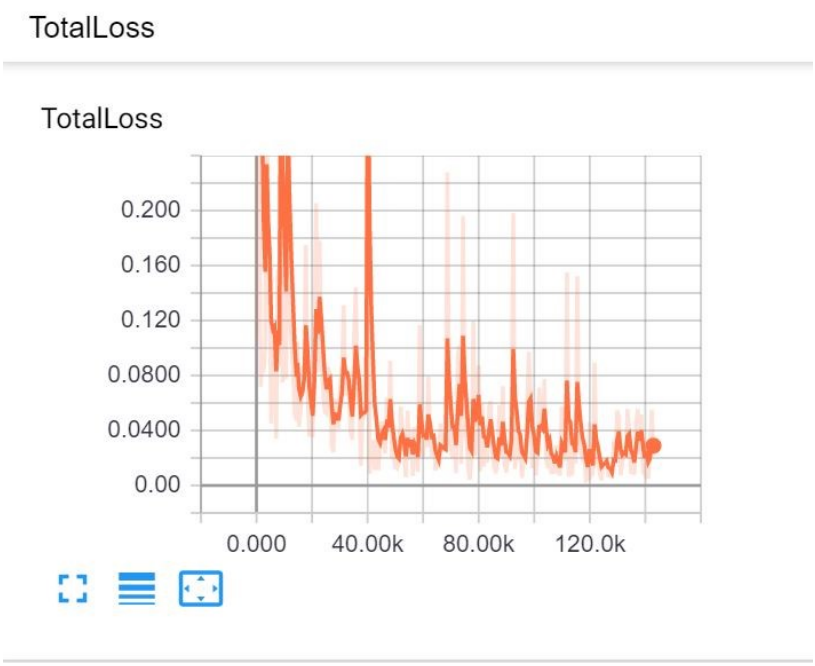


Figure 6-10 Global Loss

Chapter 7 TESTING

7.1 Testing:

The software testing is a critical element of software quality assurance represents the ultimate review of the specification, design, and coding. The main objective of the system testing is to find the errors or bugs in the system and to see whether the system fulfils expectations of the user. Testing is a procedure of executing a program with the intent of finding errors. But Testing cannot show the absence of defects it can show only that software errors are present.

7.2 Test Plan

7.2.1 **Type of Testing Involved:**

Acceptance Testing:

Acceptance testing is a process of software testing in which the module is tested for the acceptability. The purpose of this testing is to evaluate the system's compatibility with the requirements and the business value, and it is decided that whether it is acceptable for delivery or some improvements are still needed.

Unit Testing:

Unit testing is a process for testing the software components or small units. Its main purpose is to validate each form or unit over it will be tested with test data, output will be compared with the expected one.

Integration Testing:

Integration and system testing is a type of software testing, this makes sure that tests such as the system and integration are done before releasing the product. Software testing has very strict set of rules and guidelines that it follows to make sure each individual part of the software is thoroughly checked before it is given the OK, this makes sure that there are no errors and that the software runs how it's supposed to.

7.3 TestCases:

S.No	Test ID	Title	Description	Expected Result	Actual Result
1	001	Run Python Script	Check whether the name of the script is invalid	There should be an error due to invalid name	As it was expected
2	002	Run Python Script	Check whether script is valid	It should be successfully running	As Expected
3	003	Importing data correctly	Check whether all the libraries are imported correctly	It should be successfully running if all works correct	As expected
4	004	Labeling Images	Checkout the image labels	The images should be labeled correctly	As Expected

5	005	xml files generation	Checkout all the xml files for the dataset are generated correctly	All the images should have respective xml files	As expected
6	006	Generating Tfrecords	Check if the TFrecords are generated correctly	There should be TFrecords for both test and train sets	As expected
7	007	Check Training Process	The model is properly trained	The model should be trained properly	As expected
8	008	Exporting Inference Graph	Check the model is exported correctly	Model in exported properly	As expected
9	009	Testing Model	Checking if the model predicts accurately	The prediction of the model are tested	As expected

10	010	Displaying Results	Checking the output on the UI	Checking if the models predictions are displayed properly	As expected

Table 7-1 Test Cases

7.4 Model Testing:

During the process of training the total loss for each global step was recorded and saved it in event files which can be loaded and demonstrated by using the TensorBoard. The training was stopped somewhere around 144000 steps when the total loss was lower than 0.05. The training time taken for this model was around 8 hours in total.

Model Name	Training Time (Hours)	Global Steps	Total Loss
Slow-Fast model _v2_DCSASS	10 Hours	144000	0.05

Table 7-2 Model Testing

7.5 Training and Testing Accuracy:

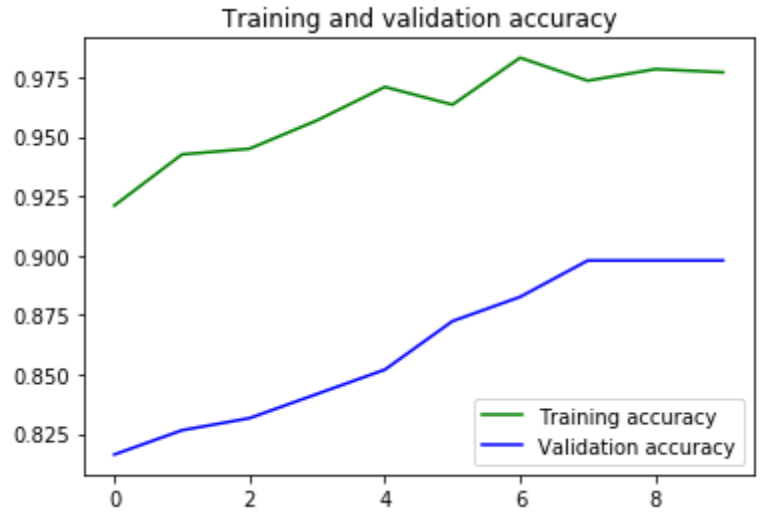


Figure 7-1 Training and Testing Accuracy

7.6 Training and Testing Loss:

Figure 7-2 Training and Testing Loss

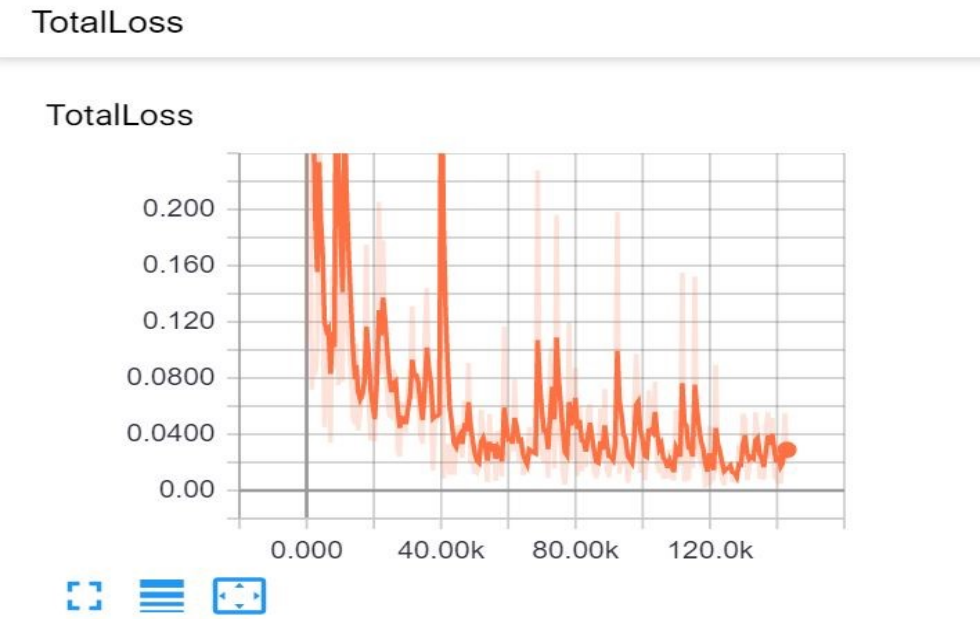


Figure 7-3 Overall Loss

7.7 Testing Results:

The output of the testing results was obtained after the testing process finished processing the test images of all classes. The testing depends on the pecification of the hardware and number of images in the testing set. Since it was running locally on the GPU version of tensorflow instead of Google Colab, the running time was slower but much faster then the tensorflow CPU version. The time taken for the testing process was quiet little as compared to the training time.

7.7.1 **Inferencing Results:**

In this step, the trained Inception v-2 model will be used to perform inferencing on the input video from the webcam frame by frame. The speed is not fast on a local laptop with running it on TensorFlow CPU, but the fps are fairly fine when run on GPU version of the TensorFlow.

The inference results from object detection taken from the webcam are shown below: The inception model correctly detected the fire with accuracy.

Chapter 8 RESULTS AND EVALUATION

8.1 Model Testing:

Model testing is a process of evaluation on the trained model to output final results. The output from the testing process helps to understand how well the model has been trained. The process of testing is similar to the training process but it is using a different python script to perform the test which is provided. The python script for evaluation of outputs for the model show the total score of the trained model on the Tensor Board in the form of graphs.

8.2 Exporting Model:

Lastly, the trained model will be tested on a new input dataset that is not the part of the training dataset to check whether the model correctly detects the object. Before deploying the model on real-time streaming data it is required to export the model first. A python script to perform this task is designed. Once the model is converted, it should get a new directory, inside the inference folder of the project, it should have a new checkpoint data, a saved model directory and the most important file frozen_inference_graph.pb. The optimized code is as follow:

```
from __future__ import division
from __future__ import print_function
from __future__ import absolute_import

import os
import io
import pandas as pd
import tensorflow as tf

from PIL import Image
from object_detection.utils import dataset_util
from collections import namedtuple, OrderedDict

flags = tf.app.flags
flags.DEFINE_string('csv_input', '', 'Path to the CSV input')
flags.DEFINE_string('image_dir', '', 'Path to the image directory')
flags.DEFINE_string('output_path', '', 'Path to output TFRecord')
FLAGS = flags.FLAGS


# TO-DO replace this with label map
def class_text_to_int(row_label):
    if row_label == 'gun':
        return 1
    elif row_label == 'knife':
        return 2
    else:
        None


def split(df, group):
    data = namedtuple('data', ['filename', 'object'])
    gb = df.groupby(group)
    return [data(filename, gb.get_group(x)) for filename, x in zip(gb.groups.keys(), gb.groups)]


def create_tf_example(group, path):
    with tf.gfile.GFile(os.path.join(path, '{}'.format(group.filename)), 'rb') as fid:
        encoded_jpg = fid.read()
    encoded_jpg_io = io.BytesIO(encoded_jpg)
    image = Image.open(encoded_jpg_io)
    width, height = image.size

    filename = group.filename.encode('utf8')
    image_format = b'jpg'
    xmins = []
    xmaxs = []
    ymins = []
    ymaxs = []
    classes_text = []
    classes = []

    for index, row in group.object.iterrows():
        xmins.append(row['xmin'] / width)
        xmaxs.append(row['xmax'] / width)
        ymins.append(row['ymin'] / height)
        ymaxs.append(row['ymax'] / height)
        classes_text.append(row['class'].encode('utf8'))
        classes.append(class_text_to_int(row['class']))

    tf_example = tf.train.Example(features=tf.train.Features(feature={
        'image/height': dataset_util.int64_feature(height),
        'image/width': dataset_util.int64_feature(width),
        'image/filename': dataset_util.bytes_feature(filename),
        'image/source_id': dataset_util.bytes_feature(filename),
        'image/encoded': dataset_util.bytes_feature(encoded_jpg),
        'image/format': dataset_util.bytes_feature(image_format),
        'image/object/bbox/xmin': dataset_util.float_list_feature(xmins),
        'image/object/bbox/xmax': dataset_util.float_list_feature(xmaxs),
        'image/object/bbox/ymin': dataset_util.float_list_feature(ymins),
        'image/object/bbox/ymax': dataset_util.float_list_feature(ymaxs),
        'image/object/class/text': dataset_util.bytes_list_feature(classes_text),
        'image/object/class/label': dataset_util.int64_list_feature(classes),
    }))
    return tf_example
```

```
    for index, row in group.object.iterrows():
        xmins.append(row['xmin'] / width)
        xmaxs.append(row['xmax'] / width)
        ymins.append(row['ymin'] / height)
        ymaxs.append(row['ymax'] / height)
        classes_text.append(row['class'].encode('utf8'))
        classes.append(class_text_to_int(row['class']))

    tf_example = tf.train.Example(features=tf.train.Features(feature={
        'image/height': dataset_util.int64_feature(height),
        'image/width': dataset_util.int64_feature(width),
        'image/filename': dataset_util.bytes_feature(filename),
        'image/source_id': dataset_util.bytes_feature(filename),
        'image/encoded': dataset_util.bytes_feature(encoded_jpg),
        'image/format': dataset_util.bytes_feature(image_format),
        'image/object/bbox/xmin': dataset_util.float_list_feature(xmins),
        'image/object/bbox/xmax': dataset_util.float_list_feature(xmaxs),
        'image/object/bbox/ymin': dataset_util.float_list_feature(ymins),
        'image/object/bbox/ymax': dataset_util.float_list_feature(ymaxs),
        'image/object/class/text': dataset_util.bytes_list_feature(classes_text),
        'image/object/class/label': dataset_util.int64_list_feature(classes),
    }))
    return tf_example


def main(_):
    writer = tf.python_io.TFRecordWriter(FLAGS.output_path)
    path = os.path.join(os.getcwd(), FLAGS.image_dir)
    examples = pd.read_csv(FLAGS.csv_input)
    grouped = split(examples, 'filename')
    for group in grouped:
        tf_example = create_tf_example(group, path)
        writer.write(tf_example.SerializeToString())

    writer.close()
    output_path = os.path.join(os.getcwd(), FLAGS.output_path)
    print('Successfully created the TFRecords: {}'.format(output_path))


if __name__ == '__main__':
    tf.app.run()
```

Figure 8-1 Exporting Model

8.3 Results:

During the process of training the total loss for each global step was recorded and saved it in event files which can be loaded and demonstrated by using the TensorBoard. The training was stopped somewhere around 144000 steps when the total loss was lower then 0.05. The training time taken for this model was around 8 hours in total.

Model Name	Training Time (Hours)	Global Steps	Total Loss
faster_rcnn_inception_v2	8 Hours	144000	0.05

Table 8-1 Testing Details

The Loss is as follow:

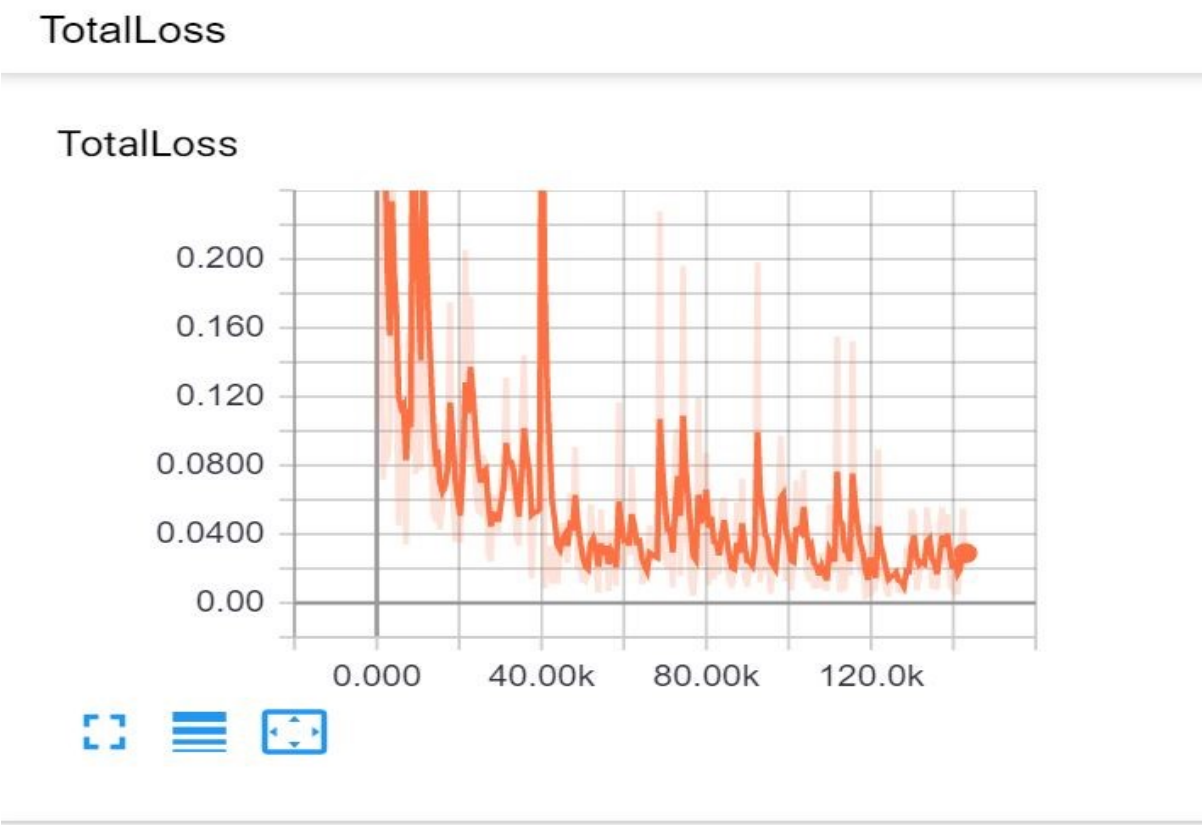


Figure 8-2 Testing Loss

8.4 Testing Results:

The output of the testing results was obtained after the testing process finished processing the test images of all classes. The testing depends on thspecification of the hardware and number of images in the testing set. Since it was running locally on the GPU version of tensorflow

instead of Google Colab, the running time was slower but much faster then the tensorflow CPU version. The time taken for the testing process was quiet little as compared to the training time.

8.4.1 **Testing Accuracy:**

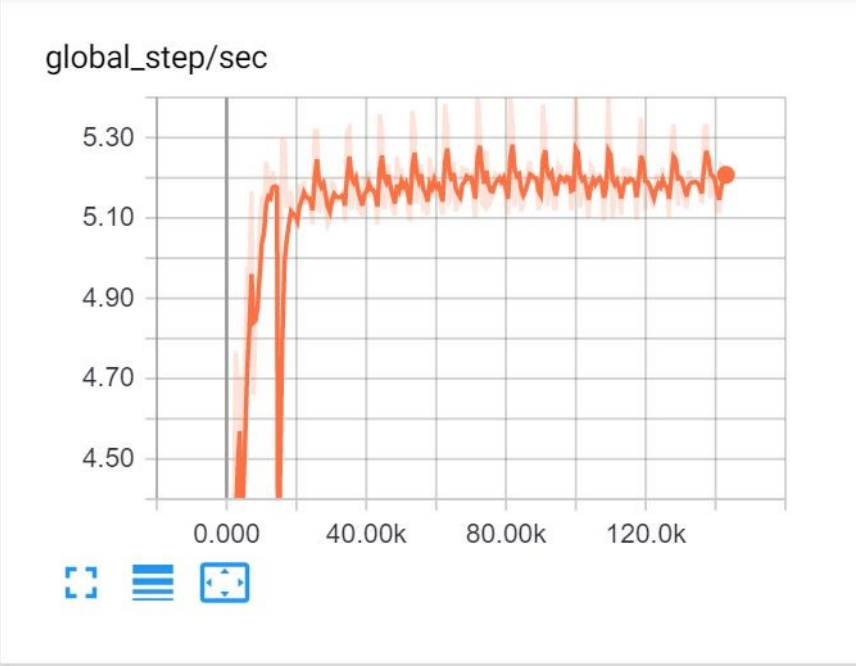


Figure 8-3 Testing Accuracy

8.4.2 **Real-Time Testing:**

The trained Inception v-2 model will be used to perform inferencing on the input video from the webcam frame by frame. The speed is not fast on a local laptop with running it on TensorFlow CPU, but the fps are fairly fine when run on GPU version of the TensorFlow.

The inception model correctly detected the fire with accuracy. The inference results from situation detection taken from the webcam are shown below:

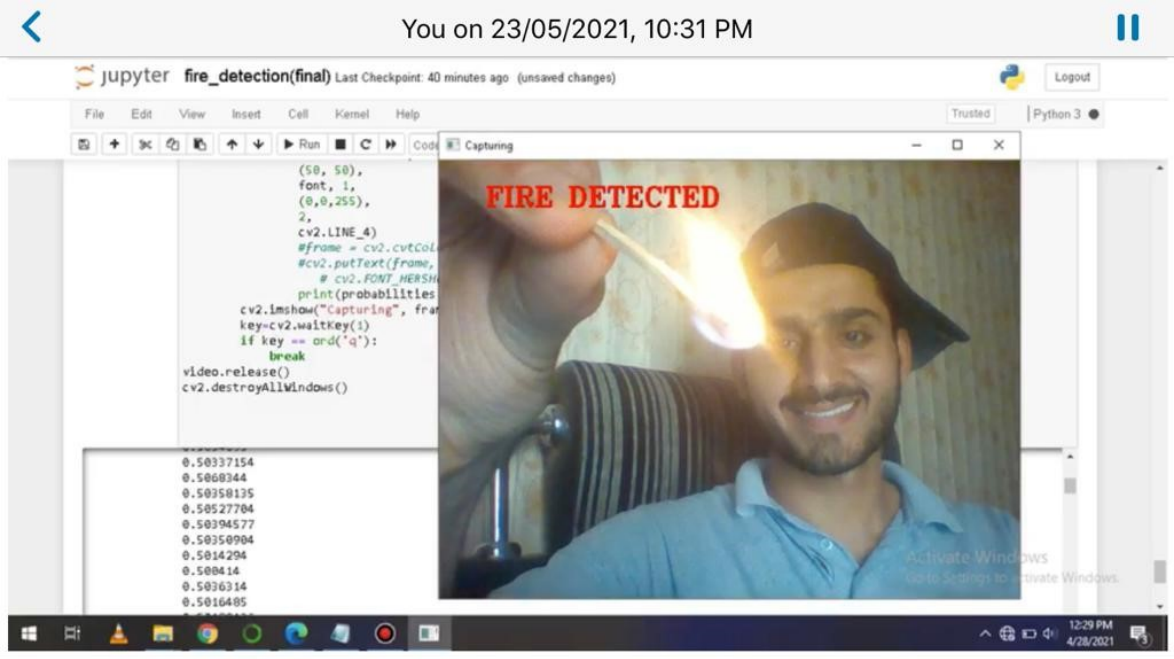


Figure 8-4 Live Webcam Detection

8.5 Discussions:

In the training stage, the details of the training process are recorded and saved in the event file. Firstly, the training process is actually quite fast in terms of time taken, it only takes 8 hours to get the total loss to an extremely low level of 0.05. As a recommendation, it can be stopped earlier as the total loss has been around 0.2- 0.1, for a while, this could save more time and energy if running on a local machine and still manage to get good accuracy and a good trained model.

In the testing stage, the results are the main contribution for evaluating on how effective the model is. Firstly, looking at the AP and mAP for each class from the output of the testing process. The AP for detecting in different backgrounds was not soo good but it was acceptable.

The possible reason could be the training data is not large enough, the sampling of the class is not comprehensive to teach the network to learn properly. Another reason could be overfitting problem, the image dataset is limited to only 1000 images.

8.6 Recommendations & Future Work:

With the experience and inspiration of this project, it is recommended to collect image dataset from various sources to give a comprehensive dataset for training CNNs. And for each class of object, it is recommended to obtain a consistent object dataset, this means in the case of this project, only assigning monitors to the monitor object class instead of using different kind of normal objects to a single one class. Possible future works can be expanding number of images to the dataset to avoid overfitting issues and separating the non-weapon class objects into their own categories are a possible way to increase the accuracy.

8.7 Conclusions:

Anomaly detection using neural networks is one of the future solutions to our security and Monitoring. For the goal of autonomous situation-classification, accuracy is important. This project has shown that with normal class objects, it does not improve the accuracy of the anomaly detection. However, with more and categorized images in each class, a higher accuracy could be achieved. Lastly, from the exploration of this project, a great knowledge has been learned on the tool of machine learning and neural networks.

REFERENCES

[1] D. (. Sarkar, "https://towardsdatascience.com," [Online].

[2] P.SHARMA, "https://www.analyticsvidhya.com/blog/2018/11/implementation-faster-r-cnn-python-object-detection/?utm_source=blog&utm_medium=a-step-by-step-introduction-to-the-basic-object-detection-algorithms-part-1," [Online].

[3] J. Xu, "https://towardsdatascience.com/deep-learning-for-classification-a-comprehensive-review-73930816d8d9," [Online].

[4] G. Yadam, "https://medium.com/object-detection-using-tensorflow-and-coco-pre/object-detection-using-tensorflow-and-coco-pre-trained-models-5d8386019a8," [Online].

[5] M.-V. R. Group, "https://medium.com/@fractaldle/brief-overview-on-object-detection-algorithms-ec516929be93," [Online].

[6] E. T. Liu, "Fire Detections Using Neural Networks In Surveillance".

[7] K. H. P. Fox, "https://edition.cnn.com/2017/10/03/americas/us-gun-statistics/index.html," [Online].

[8] ujjwalkarn, "https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/," [Online].

[9] S. T. a. F. H. Roberto Olmos, "Realtime Accident Detection Alarm in Videos Using Deep Learning," 2017.

[10] S. M. Justin Lai, "Developing A Real-Time Gun Detection Classifier".

[11] R. M. P. R. Y. Rohith Vajhala, "Weapon Detection In Surveillence Camera Images".

[12] S. A. L. L. H. Youssef Elmir, "Deep Learning for Automatic Detection of Fire in Video Sequences," *CEUR-WS,* vol. 2351.

[13] A. D. Gyanendra K.Verma, "A Anomaly Detection Using FasterR-CNN Deep Learning," in *International Conference Computer & Communication Technologies*, 2017.

[14] Wikipedia, "https://en.wikipedia.org/wiki/Convolutional_neural_network," [Online].