# FACE RECOGNITION AT VARYING ANGLES

**Abstract:**

Nowadays, there has been a rise in the amount of disruptive and offensive activities that have been happening. Due to this, security has been given principal significance. Public places like shopping centres, avenues, banks, etc are increasingly being equipped with CCTVs to guarantee the security of individuals. Subsequently, this inconvenience is making a need to computerize this system with high accuracy. Since constant observation of these surveillance cameras by humans is a near-impossible task. It requires workforces and their constant attention to judge if the captured activities are anomalous or suspicious.Hence, this drawback is creating a need to automate this process with high accuracy. Moreover, there is a need to display which frame and which parts of the recording contain the uncommon activity which helps the quicker judgment of that un ordinary action being unusual or suspicious. Therefore, to reduce the wastage of time and labour, we are utilizing deep learning algorithms for Automating Threat Recognition System. Its goal is to automatically identify signs of aggression and violence in real-time, which filters out irregularities from normal patterns. We intend to utilize different Deep Learning models (CNN and RNN) to identify and classify levels of high movement in the frame. From there, we can raise a detection alert for the situation of a threat, indicating the suspicious activities at an instance of time

# CHAPTER 1

## Introduction

“This recognition problem is made difficult by the great variability in head rotation and tilt, lighting intensity and angle, facial expression, aging, etc. Some other attempts at facial recognition by machine have allowed for little or no variability in these quantities.
Yet the method of correlation (or pattern matching) of unprocessed optical data, which is often used by some researchers, is certain to fail in cases where the variability is great.
pictures of the same person with two different head rotations”. In this work, videos are categorized into segments. From there, a detection alert is raised in the case of a threat, indicating the suspicious activities at an instance of time.
In this work, the videos are classified into two categories: Threat (anomalous activities) and Safe (normal activities).
Further, we recognize each of the 12 anomalous activities - Abuse, Burglar, Explosion, Shooting, Fighting, Shoplifting, Road Accidents, Arson, Robbery, Stealing, Assault, and Vandalism. These anomalies would provide better security to the individual

# CHAPTER 2

## Literature survey

### 2.1 Deep Surveillance with Deep Learning – Intelligent Video Surveillance

Surveillance security is a very tedious and time-consuming job. In this tutorial, we will build a system to automate the task of analyzing video surveillance. We will analyze the video feed in real-time and identify any abnormal activities like violence or theft.

There is a lot of research going on in the industry about video surveillance among them; the role of CCTV videos has overgrown. CCTV cameras are placed all over the places for surveillance and security.

In the last decade, there have been advancements in deep learning algorithms for deep surveillance. These advancements have shown an essential trend in deep surveillance and promise a drastic efficiency gain. The typical applications of deep surveillance are theft identification, violence detection, and detection of the chances of explosion.

We have generally seen deep neural networks for computer vision, image classification, and object detection tasks. In this project, we have to extend deep neural networks to 3-dimensional for learning spatio-temporal features of the video feed.

For this video surveillance project, we will introduce a spatio temporal autoencoder, which is based on a 3D convolution network. The encoder part extracts the spatial and temporal information, and then the decoder reconstructs the frames. The abnormal events are identified by computing the reconstruction loss using Euclidean distance between original and reconstructed batch.

**Intelligent Video Surveillance with Deep Learning**

e will use spatial temporal encoders to identify abnormal activities.

**The dataset for abnormal event detection in video surveillance:**

Following are the comprehensive datasets that are used to train models for anomaly detection tasks.

**CUHK Avenue Dataset:**

This dataset contains 16 training and 21 testing video clips. The video contains 30652 frames in total.

The training videos contain video with normal situations. The testing videos contain videos with both standard and abnormal events.

**UCSD pedestrian Dataset:**

This dataset contains videos with pedestrians. It includes groups of people walking towards, away, and parallel to the camera. The abnormal event includes:

- Non-pedestrian entities
- Anomalous pedestrian motion patterns

**Summary:**

In this deep learning project, we train an autoencoder for abnormal event detection. We train the autoencoder on normal videos. We identify the abnormal events based on the euclidean distance of the custom video feed and the frames predicted by the autoencoder.

We set a threshold value for abnormal events. In this project, it is 0.0068; you can vary this threshold to experiment getting better results.

## **2.2**Smart Video Surveillance System Based on Edge Computing

Abstract

New processing methods based on artificial intelligence (AI) and deep learning are replacing traditional computer vision algorithms. The more advanced systems can process huge amounts of data in large computing facilities. In contrast, this paper presents a smart video surveillance system executing AI algorithms in low power consumption embedded devices. The computer vision algorithm, typical for surveillance applications, aims to detect, count and track people's movements in the area. This application requires a distributed smart camera system. The proposed AI application allows detecting people in the surveillance area using a MobileNet-SSD architecture. In addition, using a robust Kalman filter bank, the algorithm can keep

track of people in the video also providing people counting information. The detection results are excellent considering the constraints imposed on the process. The selected architecture for the edge node is based on a UpSquared2 device that includes a vision processor unit (VPU) capable of accelerating the AI CNN inference. The results section provides information about the image processing time when multiple video cameras are connected to the same edge node, people detection precision and recall curves, and the energy consumption of the system. The discussion of results shows the usefulness of deploying this smart camera node throughout a distributed surveillance system.

## 1. Introduction

Nowadays, deep learning has shown great advantages in several research fields, for example finance [1], medicine [2], automatic modulation classification in cognitive radios [3], and many others. In particular, computer vision was the first research field in which deep learning took place [4]. New processing methods based on deep learning are replacing traditional computer vision algorithms relying on physical representations, models, functions with some level of meaning deep learning vs. traditional computer vision [5]. The more advanced systems are able to process huge amounts of data in large computing facilities. Current challenges are related to training the machine learning system with enough information which requires the labelling of those data. On the other hand, in this paper, we focused our attention on smart camera nodes distributed along a surveillance area which are far from that approach.

The modernisation of technologies at both the software and hardware level has allowed the design of smart video systems capable not only of managing the video feeds from a closed camera circuit but also analysing and extracting information in real time from the video streams.

These embedded systems are installed in both public and private locations, being able to control the number of people in an area, crowds movement, detecting anomalous behaviours, etc. Most of these systems are centralised, executing the computer vision algorithms in one single location. This central processing system receives and processes the information from all the camera network. Old systems did not process video information and required the operator's analysis, who was duty bound to carefully monitor any camera and whose analysis efficiency may decrease due to fatigue and boredom [6,7].

The modern video processing centralised systems [8,9] store and process the video information retrieved from the camera network. Only some alerts or video-clips are shown to the central console/operator without requiring a high level of attention into the management of the surveillance system.

On the other hand, the emergence of the Internet of Things (IoT) and the computing on the edge nodes [10], has led to the appearance of many research works that propose distributed video-surveillance systems based on this concept [11]. Hence, the intelligence of the system is distributed in multiple nodes, where each one can include a camera and a processing system that performs simple tasks before sending the information to the operator, facilitating its work.

For more complex tasks, current detection algorithms for computer vision include deep neural networks (DNNs). To execute DNNs, a high-end hardware system with

high computing power, such as graphical process units (GPUs), is usually required. Apart from being expensive, these hardware modules have a high energy consumption, and it can be difficult to embed them in the distributed smart camera nodes.

In this context, our paper presents a smart video-surveillance system for detecting, counting and tracking people in real time, in a embedded hardware system using new vision processing units (VPUs) hardware modules. The paper describes both the hardware architecture used in the embedded system and the developed computer vision algorithm for detecting, tracking and counting people in real time. This system can be easily installed and configured to work as a smart camera edge node in a distributed video-surveillance system.

The proposed system is based on a low-cost embedded platform UpSquared2 [12], that includes a VPU, the Myriad-X [13]. This allows the parallelisation of the developed algorithms to allow them to work in real time, with a reduced power consumption.

A MobileNet-SSD architecture [14] has been selected for the task of tracking and detecting people. Furthermore, a bank of Kalman filters allows people tracking and counting. We evaluated the performance of the system, making a comparison with other algorithms and extracting the values of the edge node related to performance, computer power and consumption, making a pipelined architecture capable of processing up to 12 video streams simultaneously.

## 3. Embedded Processing System: Hardware and Software Components

Since one of the requirements of the proposed systems is its portability and flexibility for deployment, the selection of the hardware embedded platform was one of the main tasks addressed in the research work. Below, the hardware components characteristics and the software framework used later are briefly explained.

The embedded platform selected for the smart node is the UpSquared2 system [12], a specialised hardware characterised by: an Intel Atom x7-E3950 microprocessor, an 8 Gigabyte (GB) RAM memory, a 64GB embedded MultiMediaCard (eMMC) ROM. This also includes the deep learning module of Intel Movidius Myriad X VPU [13], a System-On-Chip (SoC) that can be used for accelerating AI inference with a low-power footprint.

The objective of the Myriad-X device is to process DNNs' inference at high speed and with the lowest possible power consumption. According to the description given by the manufacturer, the architecture of the device allows it to perform more than 4 trillion operations per second (TOPS). This amount of FLOPS is achieved thanks to the combination of the neural compute engine and the 16 128-bit VLIW SHAVE (streaming hybrid architecture vector engine) processors that make up the device. In addition, the system is composed of two LEON4 Cores CPUs (RISC; SPARC v8). As previously mentioned, this device enables focusing on achieving high processing

speeds in the inference with low power consumption, resulting as perfect for developing an inference in the limit and in a battery-powered embedded system.

To execute AI algorithms in the VPU, the OpenVino [51,52] framework has been used, which eases the optimising and deployment of CNNs. The OpenVino framework includes two different tools, as shown in Figure 1: the model optimizer and the inference engine. These tools allow optimising the model and executing it in different hardware platforms such as Intel CPUs, VPUs or GPUs, reducing the execution time. One of the advantages of this framework is that it can be installed in any device that meets the minimum requirements. This also allows the installation of these two modules separately, being able to have the model optimizer in the PC used to train the network and the inference engine in the embedded system.

The Model Optimizer is an application that runs on command line and that allows to adjust and optimise the neuronal models to achieve an acceleration in the inference of the system. The OpenVino model optimizer can work with different libraries such as Tensorflow, Pytorch or Caffe. The model optimizer provides the intermediate representation (IR) files. These files are one with an extension .xml which defines the layers, sizes and connections of the architecture and a file .bin, which defines the weights of each parameter of the architecture.

Regarding the inference engine, as mentioned above, this OpenVino module can be installed on any device, independently of the model optimizer. This module loads the IR files and runs the inference on the hardware selected by the plugging. It can be run on CPU, VPU or GPU. In addition, the inference engine is in charge of balancing the inference load so as not to overload any device. Thus, the load is distributed between the CPU cores or between a set of VPUs.

Go to:

## 4. AI Image Processing to Detect and Track People at the Edge Node

One of our research goals was to design an AI application which could detect, track and count people in an embedded system. The proposed architecture (Figure 2) was based on the parallelisation of several processes in order to use the hardware modules available in the most efficient way. The analysis was divided into two processes, which communicate through the use of independent buffers.

In the first process, the so-called data analytics process, the preprocessing of the image and the post-processing of the information returned by the AI inference engine were carried out. Before the preprocessing of the image, different algorithms such as noise reduction, image edge detector [53] or any image enhance low-level pixel processing could be applied. As this low-level preprocessing is dependent on the real scenario, and due to the extra computational cost this preprocessing would add, the preprocessing has been reduced to a minimum: ROI selection, image resize to 300 × 300 pixels and (−1.0, +1.0) range normalization of pixel values, in order to adapt the data to the requirements for the input of the first MobileNet-SSD layer.

The post-processing consists of the reorganisation of the data generated by the MobileNet-SSD network, which contain the bounding boxes of where it has predicted that people are located and then that information is analysed by a Kalman filter bank that predicts movements and future overlaps, returning a more optimal and reliable result. This first process was entirely executed on the CPU of the system. The second

process that confers the system is the one that performs the network inference. The inference is designed to be executed at the edge using a VPU. However, there is also the option of running it on a CPU. Once the MobileNet-SSD performs the inference, it returns and stores the bounding boxes in a common buffer.

Figure 2 shows a schematic of the flows that compose the main system. By using elements such as the VPU, more than one inference could be implemented at the same time, being able to process more than one video stream. In the case of using a CPU, the capacity to perform more than one inference is determined by the CPU and the number of cores it contains.

Despite the fact that these processes are pipelined for one single video feed, also several video streams might be considered in the processes pipelining. In the case of the MobileNet-SSD inference, when executed by the VPU, it allows the inference of up to four video streams in an efficient way. This process is explained in more detail in later sections. Depending on the hardware limitations, a higher or lower level of parallelisation can be achieved, which is able to interleave the different processes that make up the system to enable the processing of multiple video streams at the same time.

In the following sections, we discuss the people detection with a MobileNet-SSD and the use of tracking and counting through a Kalman filters bank.

### 4.1. People Detection Using a MobileNet-SSD

This section explains how it works and why this network has been chosen for this system. The chosen network is a MobileNet-SSD, a MobileNet architecture [14], which uses the SSD [45] method for object detection. The architecture that forms this network is very similar to the one used in the VGG-16 [54]. The main difference is the replacement of the VGG-16 module by the MobileNet module. The reasons for using this architecture are two-fold: firstly because a fast architecture was needed, which could be implemented with selected algorithms such as SSD; and secondly, the resource consumption was low, because the devices used in this project are portable and its hardware is not as powerful as a high-performance PC. For this reason, MobileNet architecture was chosen, because its main feature is the speed of computing and the use of a type of convolutional layers that allow the use of fewer resources.

The architecture used is shown in Figure 3. It can be seen that at the beginning of the network there is the MobileNet module, composed of 35 convolutional layers, which are responsible for the feature extraction. A set of five layers of these 35 are in charge of carrying out the classification of objects applying an SSD.

his architecture reduces the intensity in data processing due to the use of separable depthwise and pointwise convolutional layers.

Thus, we compared the computational cost of a standard convolution used in any type of architecture such as the VGG-16 with respect to the operations required in the MobileNet convolutions.

To measure the computational cost of a standard convolution, first it is necessary to know the size of the kernels that compose it. Standard convolution kernels can be expressed graphically as shown in Figure 4, where the number of channels in the

image is *M*, the number of kernels needed is *N*, which corresponds to the number of output channels, and the size of the kernels is Dk · Dk.

The sequence of work performed for the Kalman filter bank is as follows:

Tracker creation: when an object is first detected in the scene for N (this number is configurable) consecutive frames, the system must assign a Kalman filter to that new object;

Object association: the system must be able to correctly identify the detections made by the detection system with the already assigned Kalman filters. This means that for each object detected, its Kalman filter must be associated;

Kalman filter iteration: once the measurements of each object have been delivered to each Kalman filter, the iteration of each one of the filters must be done to fulfil the prediction and correction stages. If a previously identified object does not appear in the image due to an occlusion or a failure of the detector, the Kalman filter can use the estimated values until the object appears again in the scene;

This tracking module will receive information directly from the MobileNet-SSD in the form of the bounding boxes of the detected objects. For each detected object with the MobileNet-SSD and exceeding a detection threshold, a Kalman filter will be assigned.

### 4.3. Pipeline Operation

The pipeline operation shown in Figure 2 was designed to parallelise each of the elements that form the system. The main elements that form the architecture are the preprocessing, the Kalman filter and the MobileNet-SSD inference. This parallelisation allows the processing of more than one video stream at the same time, since the inference is separated from the CPU processing. This inference is processed through the use of VPUs. These devices permit the processing of more than one inference at a time and also in a more optimal way than through CPUs. Figure 8 shows the structure of the processes that make the system up, a structure which can be parallelised as much as the hardware allows. This permits not only the processing of more video streams but also a more efficient use of the hardware systems that make the device up.

## 5. Results

The main objective of the contribution presented in the paper is to propose a portable and fast embedded system for detection, tracking and counting people.

When talking about portability, we mean two goals: firstly, that it can be easily installed anywhere, as shown in  and secondly, that it can be maintained and correctly operated without a connection to mains power, i.e., it can be powered by portable batteries.

The results section is divided into two parts. The first part describes the experimental setup, the datasets used and the edge hardware modules. The second part shows the comparison of obtained results with other similar research works. As mentioned below, the target features a portable system with robust detection and capable of running in real time. To evaluate these requirements, the following characteristics are analysed:

### 5.1. Experimental Setup

For the system evaluation, the EPFL [59] dataset was used. The characteristics of this dataset fit a possible real scenario in which there are overlaps, large numbers of people and changes in the illumination. This dataset consists of two environments: the first one is a laboratory (Figure 10b), which is a large space in which there is a set of four people overlapping each other and changing their position in the image, moving away from and towards the camera. The second scenario is a university corridor (Figure 10a), in which there are up to eight people in a small space, at different distances from the camera, with changes in lighting between the different videos and with a high number of overlapping.

## 6. Conclusions

The paper proposes a portable video surveillance system with AI CNN processing at the edge, which can detect and track people in a robust and reliable way. The novel computer technology used at the edge is VPU hardware modules, which allow performing the CNN inference faster and more efficiently than a CPU in low-end devices. The designed system permits implementing a computer vision application with low power consumption and high computational performance. To achieve this, an embedded device UpSquared2 was used, which integrates a Myriad-X VPU where the chosen CNN, a MobileNet-SSD, is executed. One of the requirements was the real-time execution of the system. To achieve real-time, we used the OpenVino 2020.2 framework, which allows an optimisation of the CNN, streamlining its inference and facilitating the execution on the VPU. In addition, this framework allows the parallel inference of the CNN on the VPU SHAVES. The inference was executed both on VPU and CPU. An improvement in the processing speed of 64.59% was obtained when the processing was done with the VPU instead the CPU.

The paper also presents a software system based on deep learning (MobileNet-SSD) and Kalman filter banks. The system was compared with other state-of-the-art machine learning (ACF, PCL-MUNARO) and deep learning (DPOM, YOLOv3, YOLO-depth) methods for people detection. The achieved results are in line with the current state of the art, having a precision of 81.43% and a recall of 80.6% in the

EPFL-corridor dataset, while in the EPFL-laboratory, both precision and recall were above 87%.

Another important point of the system is the processing of multiple video streams in real time (+30 fps), supporting up to 12 streams with the hardware provided by the UpSquared2 using the integrated VPU. In addition to this, the system has been designed to be portable and easy to manipulate, both in the adjustment of internal software parameters such as adjusting the region of interest and the detection threshold, as well as the power consumption of the system. Having an average power consumption of 12 W, with peaks of up to 15 W, when running AI inference on the VPU. Being able to power the system with portable batteries or renewable systems provides great flexibility in distributed surveillance camera systems

## Acknowledgments

The authors would like to thank the GEINTRA research group (geintra-uah.org) for their support and background work. In addition, the authors would like to thank the anonymous referees for their constructive comments and suggestions that led to the improvement of the description of the results and the overall quality of the paper

.

### 2.3Performing AI on Embedded Devices

The main contribution of the paper is to build a smart camera system capable of processing videos locally, at the edge nodes, using embedded devices. For this reason, we need to review the design and capabilities of constrained devices to analyse if they can execute real-time artificial intelligence processing. Shi [15] defines edge computing as “enabling technologies that allow computing to be performed at the edge of the network, on downstream data on behalf of cloud services and on upstream data on behalf of IoT services”. This allows using the full potential offered by artificial intelligence, without losing computing capacity in the processor.

There is a variety of devices on the market that allows bringing artificial intelligence to the edge. The best known acceleration coprocessor is the GPU. Most GPUs are designed for being used as part of a desktop computer, but in recent years, different embedded development kits have emerged, including a GPU, such as the Jetson AGX Xavier [16], or the Jetson Nano [17], mainly oriented towards autonomous systems. Another type of technology for hardware acceleration is the Coral [18] family provided by Google. It is a set of devices that can be connected via USB and generate an acceleration system for tensor-oriented architectures.

The latest technology for the development of video-surveillance systems is the VPU, developed by Intel, and focused on the parallel processing of neural networks. VPUs are oriented towards high-speed inference processing with very low power consumption, being able to implement this type of device in embedded systems, drones, or systems powered by external power supplies. Intel has already released three versions of the device, with the Myriad-1 and Myriad-2 being discontinued and now only Myriad-X being on the market. The VPU can be connected via USB (NCS2) or PCIe.

This type of technology is becoming increasingly valuable in the development of smart systems, as evidenced by Kristianin et al. in [19] by designing an object classification system, or Adnan et al. in [20] with the optimisation of a detection system on a Rasberry. Moreover, in the field of astronomy, Surabhi Agarwal et al. [21] describe the design of a star tracker using an NCS2.

### 2.42.2. People Detection

Object detection is defined as the ability to recognise and locate a particular type of "object" in an image. In this context, the literature related to people detection in video-surveillance applications is very extensive [22,23]. Among the different proposed methods, two large groups can be distinguished: those that are based on classical methods, which make use of mathematical techniques and people feature modelisation, and modern approaches based on deep learning.

Belonging to the image pattern recognition classical methods, one of the most popular is the one proposed by Dalal and Triggs [24]. This algorithm obtains image features using a histogram of gradient (HOG) and then applies a support vector machine (SVM) classifier (widely used in other applications such as natural language [25] or speech recognition [26]). The authors in [27] also proposed the use of HOG features for people's head and shoulders detection, combined this time with a classifier based on principal components analysis (PCA).

There are other video-surveillance proposals that address both people detection and tracking using different techniques. In this context, the proposal in [28] presents a method that combines a Kalman filter and HOG features, whereas [29] describes an approach based on image segmentation and a Kalman filter.

In recent years, the use of RGBD cameras [30] that provide both RGB images and depth information (distance from each 3D point to the camera) using different technologies, such as stereo vision, or time of flight, has impressively increased. Due to this, numerous works have appeared that use this information (instead of only RGB images) for people detection. Some of these works again use HOG features obtained from RGBD data [31,32], whereas other approaches are based on PCA [33]. Furthermore, there are also works that use only depth information to detect people in a robust way while preserving their privacy [34,35].

The improvements in the processing systems' capacity has led to a change of paradigm in computer vision. Particularly, image classification and object recognition have embraced the use of neural networks. One of the first networks used was the so-called AlexNet in 2012 [36]. In addition, this type of network can be used for the detection of specific elements, such as pedestrians, as shown in the paper [37], which uses a network based on the detection of pedestrians, which consists of extensive part detectors.

This type of network uses sets of convolutional layers that extract the data of an image, which are used by a dense neural network to classify the image. To train these networks, datasets labelled [38,39,40] are used, allowing to obtain classifiers with a high capacity to generalise and precision values far beyond those given by classical methods. These datasets must be of large size for correct training, and in the case that it is not large enough, there are methods such as [41], which allow to improve the

training with a smaller number of images reducing the problem of overfitting. In spite of the good results both in general and in precision, the high computational cost must be taken into account when working with this type of network, making it expensive and difficult to work in real time.

To achieve real-time execution and speed up the calculations, several mathematical models have been proposed. These are commonly named one-shot algorithms, due to their ability to locate the region with the more relevant information within the image. Among these algorithms, the most known are R-CNN [42], fast R-CNN [43], faster R-CNN [44], SSD (single shot detection) [45] and YOLO [46]. The last three can process images in real-time in constrained devices. We selected SSD for our people detection algorithm.

These algorithms are combined with CNNs to develop systems with high accuracy and that can run in real time. The resulting architectures, when compared with the classical methods, are slower in training but have better results both in time and in accuracy in the inference. It is possible to find systems as Masuzawa et al. [47] which adopted the YOLO object detection method through a Darknet framework to generate the bounding box of detected people.

Concerning the detection of people by artificial intelligences in embedded systems, there is a great variety of articles related to this topic. This is due to the great variety of existing AIs, the number of embedded systems on the market, as well as the way to implement one in another. In the literature, there are a lot of systems implemented in Rasberry Pi, which is due to its price and its computing capacity. For this device, we find people detection systems using classical AI such as the use of HOG+SVM with thermal images in the case of the paper [48] or by using deep learning models as in the case of [49] which uses it for pedestrian detection in cars. Apart from the rasberri pi, there are other hardware systems such as the NVIDIA Jetson Nano, which contains an internal GPU that allows the inference of networks with ease as in the case .

## CHAPTER 3

**problem statement**

CCTV cameras are typically fitted at varying angles, usually at the top of a pole, buildings etc. to protect it against vandalism and thefts. This makes it quite difficult to capture a clear picture of a person’s face directly from the front.

This is quite unlike ideal situations when camera is conveniently placed closer to and right in the front (at the eye level) of a person, for example, getting a passport-sized photo clicked at a photo studio. The quality of image captured in both of the above situations is drastically different.

Whereas there are several solutions based on facial recognition technology to search for a person in a video feed, the accuracy of results in case footage is captured under non-ideal conditions is generally poor. This is precisely the problem that the solution should overcome.

Build a face detection-based solution which can identify suspects in CCTV feeds even when their faces are partially visible due to the varying angles at which CCTV cameras are typically placed.

The solution should be able to handle people of different age, skin colour, gender and facial structure.

Solution should also maximize accuracy of object classification, object detection and object tracking.

The solution should be designed in a highly scalable manner such that it is able to look for o.ne or multiple persons across multiple videos concurrently. Solution should support both recorded videos as well as live CCTV feeds.

Existing system:

- The first neural network is convolutional, which has been utilized to obtain the high-level feature maps of the images.
- This will reduce the intricacy of the input for the second neural net. We are utilizing a pre-trained model called inceptionV3 created by Google. This model applies transfer learning [20] as a widely used object identification models.
- This has several parameters and can require a large period of time to train completely. Henceforth, Transfer learning utilizes a previously learned model that simplifies loads of this work.
- The model has learned for various classes like ImageNet which is then re-trained for the weights of new classes.
- The second neural network is utilized as a recurrent neural net to extract meaning from the chain of the actions portrayed in a fixed time duration. This model will be used to classify the segments of videos as a threat and safe.

Disadvantage:

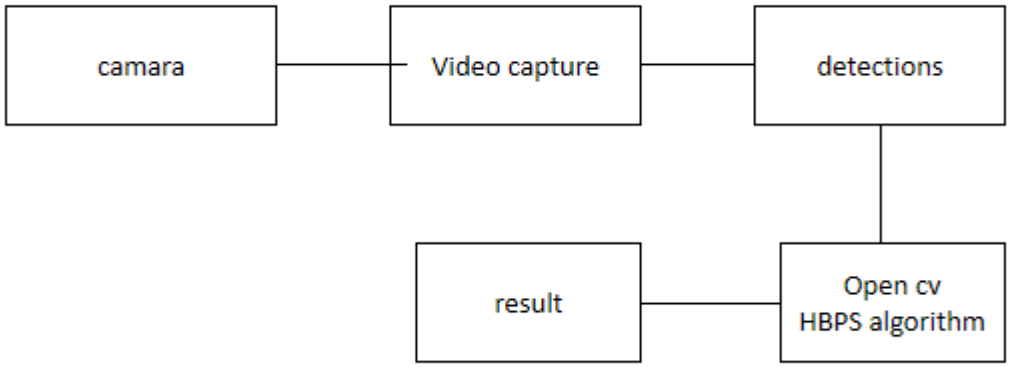
- algoprithm is a image processing technique it not used for monitoring.
- Less efficiency.
- Slow process.
- image processing method is not perform well.
- Result accuracy is 70% only.

Proposed system:

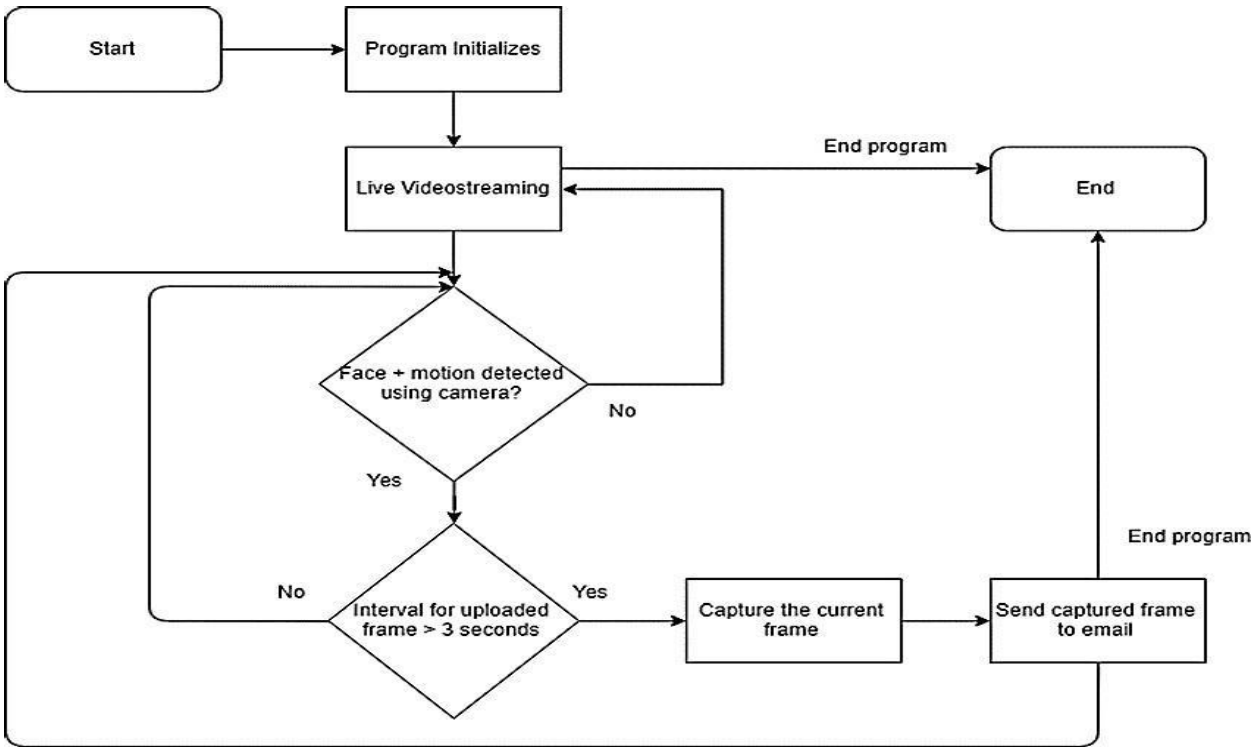
- Safety and security are major concerns in day era.

- This project aim at providing one such idea to ensure safety and security of once own property.
- In this project we proposer to provide a smart cctv surveillance system with intrusion detection.
- Camera are installing for live streaming and monitoring purpose.
- This system performs face recognition as an authentication procedure and alert the owner when an unknown face is detected.

BLOCK DIAGRAM:




FLOW DIAGRAM:




CHAPTER 4

## Artificial Intelligence

The Artificial Intelligence tutorial provides an introduction to AI which will help you to understand the concepts behind Artificial Intelligence
. In this tutorial, we have also discussed various popular topics such as History of AI, applications of AI, deep learning, machine learning, natural language processing, Reinforcement learning, Q-learning, Intelligent agents, Various search algorithms, etc. Our AI tutorial is prepared from an elementary level so you can easily understand the complete tutorial from basic concepts to the high-level concepts.

### What is Artificial Intelligence?

In today's world, technology is growing very fast, and we are getting in touch with different new technologies day by day.
Here, one of the booming technologies of computer science is Artificial Intelligence which is ready to create a new revolution in the world by making intelligent machines.The Artificial Intelligence is now all around us. It is currently working with a variety of subfields, ranging from general to specific, such as self-driving cars, playing chess, proving theorems, playing music, Painting, etc.

AI is one of the fascinating and universal fields of Computer science which has a great scope in future. AI holds a tendency to cause a machine to work as a human.
Artificial Intelligence is composed of two words **Artificial** and **Intelligence**, where Artificial defines "man-made," and intelligence defines "thinking power", hence AI means "a man-made thinking power."
Artificial Intelligence exists when a machine can have human based skills such as learning, reasoning, and solving problems
With Artificial Intelligence you do not need to preprogram a machine to do some work, despite that you can create a machine with programmed algorithms which can work with own intelligence, and that is the awesomeness of AI.
It is believed that AI is not a new technology, and some people says that as per Greek myth, there were Mechanical men in early days which can work and behave like humans.

## Why Artificial Intelligence?

Before Learning about Artificial Intelligence, we should know that what is the importance of AI and why should we learn it. Following are some main reasons to learn about AI:

- With the help of AI, you can create such software or devices which can solve real-world problems very easily and with accuracy such as health issues, marketing, traffic issues, etc.
- With the help of AI, you can create your personal virtual Assistant, such as Cortana, Google Assistant, Siri, etc.
- With the help of AI, you can build such Robots which can work in an environment where survival of humans can be at risk.
- AI opens a path for other new technologies, new devices, and new Opportunities.

## Goals of Artificial Intelligence

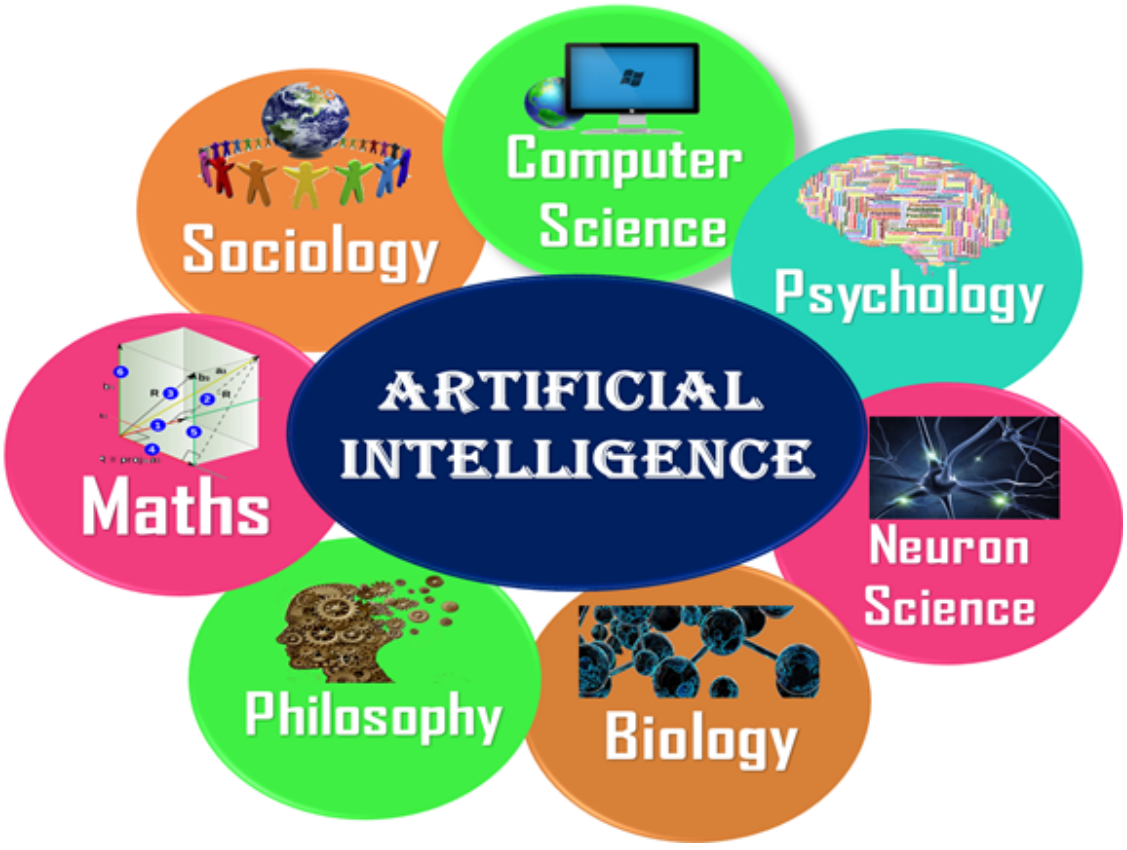
Following are the main goals of Artificial Intelligence:

1. Replicate human intelligence
2. Solve Knowledge-intensive tasks
3. An intelligent connection of perception and action
4. Building a machine which can perform tasks that requires human intelligence such as:
    - Proving a theorem
    - Playing chess
    - Plan some surgical operation
    - Driving a car in traffic
5. Creating some system which can exhibit intelligent behavior, learn new things by itself, demonstrate, explain, and can advise to its user.

## What Comprises to Artificial Intelligence?

Artificial Intelligence is not just a part of computer science even it's so vast and requires lots of other factors which can contribute to it. To create the AI first we should know that how intelligence is composed, so the Intelligence is an intangible part of our brain which is a combination of **Reasoning, learning, problem-solving perception, language understanding, etc**.
To achieve the above factors for a machine or software Artificial Intelligence requires the following discipline:

- Mathematics
- Biology
- Psychology
- Sociology
- Computer Science
- Neurons Study
- Statistics




## Advantages of Artificial Intelligence

Following are some main advantages of Artificial Intelligence:

- **High Accuracy with less errors:** AI machines or systems are prone to less errors and high accuracy as it takes decisions as per pre-experience or information.
- **High-Speed:** AI systems can be of very high-speed and fast-decision making, because of that AI systems can beat a chess champion in the Chess game.

- **High reliability:** AI machines are highly reliable and can perform the same action multiple times with high accuracy.
- **Useful for risky areas:** AI machines can be helpful in situations such as defusing a bomb, exploring the ocean floor, where to employ a human can be risky.
- **Digital Assistant:** AI can be very useful to provide digital assistant to the users such as AI technology is currently used by various E-commerce websites to show the products as per customer requirement.
- **Useful as a public utility:** AI can be very useful for public utilities such as a self-driving car which can make our journey safer and hassle-free, facial recognition for security purpose, Natural language processing to communicate with the human in human-language, etc.

## Disadvantages of Artificial Intelligence

Every technology has some disadvantages, and thesame goes for Artificial intelligence. Being so advantageous technology still, it has some disadvantages which we need to keep in our mind while creating an AI system. Following are the disadvantages of AI:

- **High Cost:** The hardware and software requirement of AI is very costly as it requires lots of maintenance to meet current world requirements.
- **Can't think out of the box:** Even we are making smarter machines with AI, but still they cannot work out of the box, as the robot will only do that work for which they are trained, or programmed.
- **No feelings and emotions:** AI machines can be an outstanding performer, but still it does not have the feeling so it cannot make any kind of emotional attachment with human, and may sometime be harmful for users if the proper care is not taken.
- **Increase dependency on machines:** With the increment of technology, people are getting more dependent on devices and hence they are losing their mental capabilities.
- **No Original Creativity:** As humans are so creative and can imagine some new ideas but still AI machines cannot beat this power of human intelligence and cannot be creative and imaginative.

## Prerequisite

Before learning about Artificial Intelligence, you must have the fundamental knowledge of following so that you can understand the concepts easily:

- Any computer language such as C, C++, Java, Python, etc.(knowledge of Python will be an advantage)
- Knowledge of essential Mathematics such as derivatives, probability theory, etc.

## Audience

Our AI tutorial is designed specifically for beginners and also included some high-level concepts for professionals.

## Problems

We assure you that you will not find any difficulty while learning our AI tutorial. But if there any mistake, kindly post the problem in the contact form.

CHAPTER 5

**Identify the Family Member feature**

- This feature is very useful feature of our minor project, It is used to find if the person
- the frame is known or not. It do this in two steps :

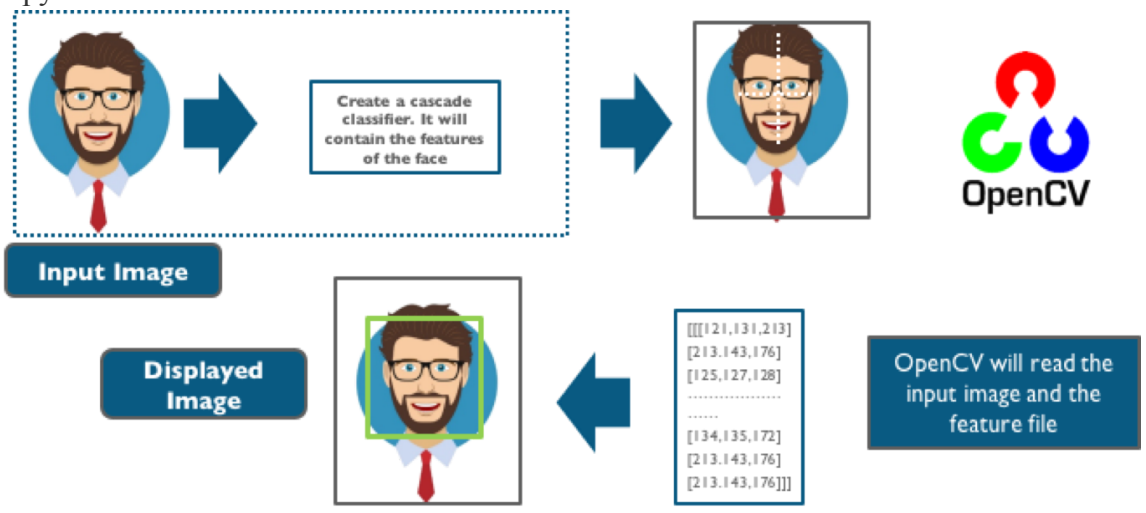
1 – Find the faces in the frames

2 – Use LBPH face recognizer algorithm to predict the person from already trained model.

- Luckily , thanks to skimage package in python we dont have to replicate all this mathematical calculation in python since skimage has pre build feature that d o all of these tasks for us with just calling its in-built function.
- We just have to feed in two images/frames which we have captured earlier, so we just feed them in and its gives us out the masked image with score.

**1 – Detecting faces in the frames**

This is done via Haarcascade classifiers which are again in-built in openCV module of python



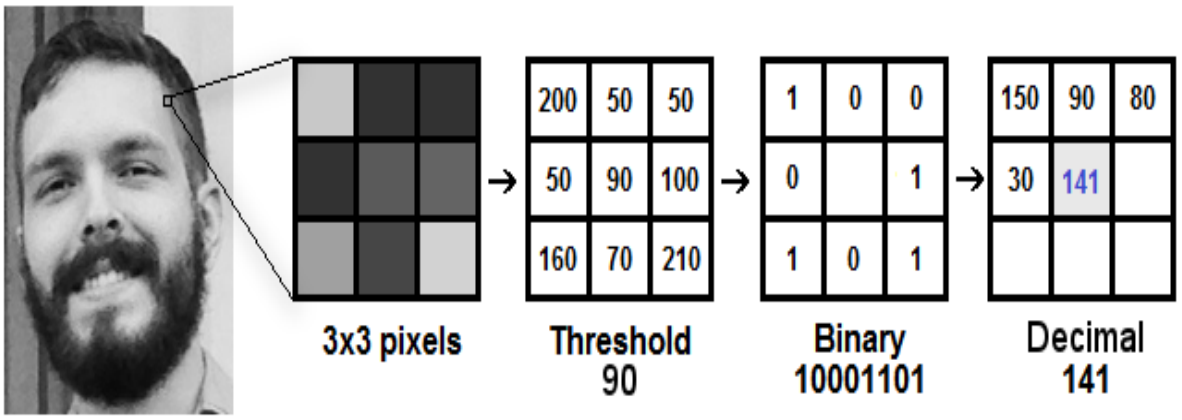

- Cascade classifier, or namely cascade of boosted classifiers working with haar-like features, is a special case of ensemble learning, called boosting. It typically relies on Adaboost classifiers (and other models such as Real Adaboost, Gentle Adaboost or Logitboost).
- Cascade classifiers are trained on a few hundred sample images of image that contain the object we want to detect, and other images that do not contain those images.

**There are some common features that we find on most common human faces :**

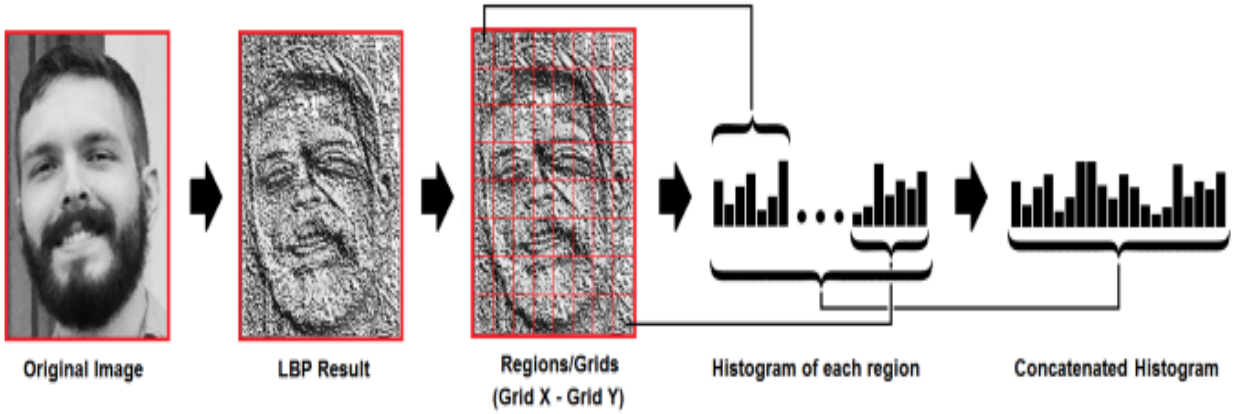
- a dark eye region compared to upper-cheeks
- a bright nose bridge region compared to the eyes
- some specific location of eyes, mouth, nose

**Using LBPH for face recognition**

- now we have detected for faces in the frame and this is the time to identify it and check if it is in the dataset which we've used to train our lbph model
- The LBPH uses 4 parameters:
- Radius: the radius is used to build the circular local binary pattern and represents the radius around the central pixel. It is usually set to 1.
- Neighbors: the number of sample points to build the circular local binary pattern. Keep in mind: the more sample points you include, the higher the computational cost. It is usually set to 8.
- Grid X: the number of cells in the horizontal direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.
- Grid Y: the number of cells in the vertical direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector.




- The first computational step of the LBPH is to create an intermediate image that describes the original image in a better way, by highlighting the facial characteristics. To do so, the algorithm uses a concept of a sliding window, based on the parameters radius and neighbors. Which is shown perfectly via the above image.
- Extracting the Histograms: Now, using the image generated in the last step, we can use the Grid X and Grid Y parameters to divide the image into multiple gird.




**Detect for Noises in the frame**

- This feature is used to find the noises in the frames well this is something you would find in most of the cctv's but in this module we'll see how it works.
- Talking in simple way all the frames are continously analyzed and checked for noises. Noise in checked in the consecutive frames. Simply we do the absolute difference between two frames and in this way the difference of two images are analyzed and Contours( boundaries of the motion are detected ) and if there are no boundries then no motion and if there is any there is motion.

frame1

| | | | |
|---|---|---|---|
| 10 | 90 | 16 | 16 |
| 0 | 11 | 11 | 11 |
| 18 | 30 | 33 | 33 |
| 18 | 18 | 18 | 18 |

frame2

| | | | |
|---|---|---|---|
| 10 | 90 | 16 | 16 |
| 0 | 13 | 17 | 11 |
| 18 | 34 | 31 | 33 |
| 18 | 17 | 19 | 18 |

frame2 - frame1

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 2 | 6 | 0 |
| 0 | 4 | -2 | 0 |
| 0 | -1 | 1 | 0 |

abs (frame2 - frame1)

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 2 | 6 | 0 |
| 0 | 4 | 2 | 0 |
| 0 | 1 | 1 | 0 |

- As you would know all images are just integer/ float values of pixels which tells the brightness of pixel and similarly every pixel has that valules of brigthness.
- So we just do simply absolute difference because negative will make no sense at all.

**Visitors in room detection**

This is the feature which can detect if someone has entered in the room or gone out. So it works using following steps:

1 – It first detect for noises in the frame.

2 – Then if any motion happen it find from which side does that happen either left or right.

3 – Last if checks if motion from left ended to right then its will detect it as entered and capture the frame.

Or vise-versa

- So there is not complex mathematics going on around in this specific feature.
- So basically to know from which side does the motion happened we first detect for motion and later on we draw rectangle over noise and last step is we check the co-ordinates if those points lie on left side then it is classified as left motion

ALGORITHM USED:
HBPS algorithm:

- LBPH (Local Binary Pattern Histogram) is a Face-Recognition algorithm it is used to recognize the face of a person. It is known for its performance and how it is able to recognize the face of a person from both front face and side face.
- In this article, we will explore the Local Binary Patterns Histogram algorithm (LBPH) for face recognition. It is based on local binary operator and is one of the best performing texture descriptor. The need for facial recognition systems is increasing day by day.

Diffrence:

- Face Detection: it has the objective of finding the faces (location and size) in an image and probably extract them to be used by the face recognition algorithm.
- Face Recognition: with the facial images already extracted, cropped, resized and usually converted to grayscale, the face recognition algorithm is responsible for finding characteristics which best describe the image.

Open cv:

- OpenCV-Python is a library of Python bindings designed to solve computer vision problems. ... OpenCV-Python makes use of Numpy, which is a highly optimized library for numerical operations with a MATLAB-style syntax. All the OpenCV array structures are converted to and from Numpy arrays.
- OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products.
- OpenCV is a great tool for image processing and performing computer vision tasks. It is an open-source library that can be used to perform tasks like face detection, objection tracking, landmark detection, and much more. ... Some of these functions are really common and are used in almost every computer vision task.
- OpenCV is a great computer vision library, all the algorithms, processing techniques are available . You can even accelerate opencv logic with cuda support. The documentation is really good with lots of examples available in Python, C/C++, android and ios as well.

## CHAPTER 6

Result:
This work suggests an approach to spot variation from the norm in real-world CCTV recordings. The normal data alone may not be effective to distinguish abnormalities in these recordings. Therefore, to handle the complexity of these realistic anomalies, both normal and anomalous videos have been considered and hence, maximised the

accuracy of the model. Furthermore, to prevent the efforts-requiring temporal annotations of abnormal sections in training recordings, a general model of anomaly detection has been learned utilizing two distinct neural networks with a poorly labelled dataset. A rarely processed large-scale anomaly dataset consisting of 12 real-world anomalies has been utilized for learning with the aim of validating the suggested approach. The experimental results obtained during the work conclude that our suggested anomaly detection approach performs significantly better than the previously used methods

## 6. Conclusion

This work suggests an approach to spot variation from the norm in real-world CCTV recordings. The normal data alone may not be effective to distinguish abnormalities in these recordings. Therefore, to handle the complexity of these realistic anomalies, both normal and anomalous videos have been considered and hence, maximised the accuracy of the model. Furthermore, to prevent the efforts requiring temporal annotations of abnormal sections in training recordings, a general model of anomaly detection has been learned utilizing two distinct neural networks with a poorly labelled dataset. A rarely processed large-scale anomaly dataset consisting of 12 real-world anomalies has been utilized for learning with the aim of validating the suggested approach. The experimental results obtained during the work conclude that our suggested anomaly detection approach performs significantly better than the previously used methods.

As specified in Table 3., this Threat Recognition Model classifies the anomalies into thirteen categories: Abuse, Burglar, Explosion, Shooting, Fighting, Shoplifting, Road Accidents, Arson, Robbery, Stealing, Assault, Vandalism, and Normal. The model concatenates 8 frames to form a chunk. The optimizer used in this model is Adam and the error function is categorical_crossentropy. The model uses three different types of activation function; Relu, Sigmoid, and Softmax. Moreover, to further increase the testing accuracy of the model the dataset has been doubled by flipping the videos horizontally. Hence, the overall accuracy of the model is 97.23% with reduced overfitting. Eventually, to implement this model in real-time, it is necessary to consider all the hardware constraints explained in this work. Hence, a proper implementation plan will reduce the computation power, optimise the use of resources and eventually reduce the overall cost of the system.

## Reference:

J. Kooij, M. Liem, J. Krijnders, T. Andringa, and D. Gavrila. Multi-modal human aggression detection. Computer Vision and Image Understanding, 2016.

[2] S. Mohammadi, A. Perina, H. Kiani, and M. Vittorio. Angry crowds: Detecting violent events in videos. In ECCV, 2016.

[3] Convolutional Neural Network (CNN) in Keras, https://towardsdatascience.com/building-a-convolutional-neural-network-cnn-in-keras 329fbbadc5f5

[4] Recurrent Neural Networks (RNN) in Keras, https://towardsdatascience.com/understanding-lstm-and-its-quick-implementation-in-keras-forsentiment-analysis-af410fd85b47
[5] W. Li, V. Mahadevan, and N. Vasconcelos. Anomaly detection and localization in crowded scenes. TPAMI, 2014.
[6] X. Cui, Q. Liu, M. Gao, and D. N. Metaxas. Abnormal detection using interaction energy potentials. In CVPR, 2011