

Novel Machine Learning Approach For Weapon Detection

Submitted in partial fulfillment of the requirements for the award of Bachelor of
Engineering degree in Computer Science and Engineering

by

P. Teja Sri Pavan(38110410)

N. Karthik Reddy Naidu (38110240)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade “A” by NAAC | 12B Status by UGC | Approved by
AICTE**

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI – 600 119

MAY – 2022



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with “A” grade by NAAC I 12B Status
byUGC I Approved by AICTE

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119
www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **P. TEJA SRI PAVAN (38110410)** and **N. KARTHIK REDDY NAIDU (Reg. No. 38110240)** who carried out the project entitled “**Novel Machine Learning Approach For Weapon Detection**” under my supervision from November 2021 to March 2022.

Internal Guide

Dr.S.Prayla Shyry M.E., Ph.D.,

Head of the Department

Dr. L. Lakshmanan M.E., Ph.D.,

Submitted for Viva voce Examination held on

**Internal Examiner
Examiner**

External

DECLARATION

I **TEJA SRI PAVAN** hereby declare that the Project Report entitled “**Novel Machine Learning Approach For Weapon Detection**” is done by under the guidance of **Dr. S.Prayla Shyry M.E., Ph.D.**, Department of Computer Science and Engineering at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind of encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph.D.**, Dean School of Computing and **Dr. L. Lakshmanan M.E., Ph.D.**, and **Dr. S. Vigneshwari M.E., Ph.D.**, Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. S. Prayla Shyry M.E., Ph.D.**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

The immense crime rates resulting from using pistols have led governments to seek solutions to deal with such terrorist incidents. These incidents have a negative impact on public security and cause panic among citizens. From this point, facing a pandemic of weapon violence has become an important research topic.

One way to reduce this kind of violence is to prevent it via remote detection and to give an appropriate response in a short time. Video surveillance is the process of monitoring the behavior of people and objects.

Surveillance systems can be employed in security applications as legal evidence. Moreover, it is used widely in suspicious activity detection applications. Intelligent video surveillance systems (IVSSs) are the use of automatic video analytics to enhance the effectiveness of traditional surveillance systems.

With the rapid development in Deep Learning (DL), it is now widely used to address the problems existing in traditional detection techniques. In this article, an approach to detect pistols and guns in video surveillance systems is proposed. The presented approach does not need any invasive tools in the weapon detection process. It uses DL in the classification and the detection processes.

The proposed approach enhances the obtained results by applying Transfer Learning (TL). It employs two different DL techniques: AlexNet and GoogLeNet. Experimental results verify the adaptability of detecting different types of pistols and guns. The experiments were conducted on a benchmark gun database called Internet Movie Firearms Database (IMFDB). The results obtained suggest that the proposed approach is promising and outperforms its counterparts.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
1	INTRODUCTION	1
2	LITERATURE SURVEY	2
	2.1 LITERATURE SURVEY 1	2
	2.2 LITERATURE SURVEY 2	2
	2.3 LITERATURE SURVEY 3	3
	2.4 LITERATURE SURVEY 4	3
3	METHODOLOGY	4
	3.1 EXISITING SYSTEM	4
	3.2 PROPODED SYSTEM	4
	3.3 BLOCK DIAGRAM	5
	3.4 FLOW CHART	6
	3.5 SOFTWARE REQUIRED	6
	3.6 METHODOLOGY	6
	3.7 PREPROCESSING	7
	3.7.1 IMAGE CROPPING	8
	3.7.2 INTENSITY ADJUSTMENT	9
	3.7.3 BRIGHTNESS THRESHOLDING	9
	3.7.4 CLEARING AREAS OF BINARY IMAGE	10
	3.7.5 DETECTING EDGES	10
4	SOFTWARE MODULES	12
	4.1 PYTHON	12
	4.1.1 PYTHON HISTORY	12
	4.2 NUMPY	16
	4.2.1 INSTALL NUMPY	17
	4.2.2 IMPORT NUMPY	17
	4.2.3 USE OF NUMPY	18
	4.2.4 NUMPY ARRAY	18
	4.3 MATPLOTLIB	19
	4.3.1 MATPLOTLIB & PYPLOT	19

4.3.2 MATPLOTLIB & NUMPY	20
4.3.3 MATPLOTLIB & PANDAS	21
4.3.4 MATPLOTLIB & PYPLOT	21
CONCLUSION	25
5.1 CONCLUSION	25
REFERENCE	26
APENDICES	28
A. SAMPLE CODE	28
B. SCREENSHOTS	41

LIST OF ABBREVIATIONS

ABBREVIATION

EXPANSION

CNN	CONVOLUTIONAL NEURAL NETWORK
RCNN	REGION BASED CONVOLUTIONAL NEURAL NETWORK
YOLO V3	YOU ONLY LOOK ONCE VERSION 3

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGE NO.
1	Line plot generated by Matplotlib	22
2	Bar plot generated by Matplotlib	23
3	Compaision of accuracies of different algorithms	40
4	Confusion matrix	40

CHAPTER 1

INTRODUCTION

The immense crime rates resulting from using pistols have led governments to seek solutions to deal with such terrorist incidents. These incidents have a negative impact on public security and cause panic among citizens. From this point, facing a pandemic of weapon violence has become an important research topic.

One way to reduce this kind of violence is to prevent it via remote detection and to give an appropriate response in a short time. Video surveillance is the process of monitoring the behavior of people and objects.

Surveillance systems can be employed in security applications as legal evidence. Moreover, it is used widely in suspicious activity detection applications. Intelligent video surveillance systems (IVSSs) are the use of automatic video analytics to enhance the effectiveness of traditional surveillance systems.

With the rapid development in Deep Learning (DL), it is now widely used to address the problems existing in traditional detection techniques. In this article, an approach to detect pistols and guns in video surveillance systems is proposed. The presented approach does not need any invasive tools in the weapon detection process. It uses DL in the classification and the detection processes.

The proposed approach enhances the obtained results by applying Transfer Learning (TL). It employs two different DL techniques: AlexNet and GoogLeNet. Experimental results verify the adaptability of detecting different types of pistols and guns. The experiments were conducted on a benchmark gun database called Internet Movie Firearms Database (IMFDB). The results obtained suggest that the proposed approach is promising and outperforms its counterparts.

CHAPTER 2

LITERATURE SURVEY

2.1 LITERATURE SURVEY 1

“ Weapon Detection in Real-Time CCTV Videos using Deep Learning”

Author: Muhammad Tahir Bhatti¹ , Muhammad Gufran Khan¹ (Senior Member, IEEE), Masood Aslam² , Muhammad Junaid Fiaz

Content: Security and safety is a big concern for today's modern world. For a country to be economically strong, it must ensure a safe and secure environment for investors and tourists. Having said that, Closed Circuit Television (CCTV) cameras are being used for surveillance and to monitor activities i.e. robberies but these cameras still require human supervision and intervention. We need a system that can automatically detect these illegal activities.

2.2 LITERATURE SURVEY 2

“ Real Time Video Surveillance for Automated Weapon Detection”

Author: Bhagyalakshmi. P | Indhumathi. P | Lakshmi. R | Dr. Bhavadharini

Content:

In this work, live surveillance videos is taken for monitoring and detecting the abnormal events based on real time image processing techniques. Operations of proposed project has three processing modules, first processing module is for object detection using Convolutional Neural Networks(CNN) and second processing module will handle the classification of weapons, monitoring and alarm operations will be carried out by the third processing module.

2.3 LITERATURE SURVEY 3

“ Detection and Classification of Different Weapon Types Using Deep Learning”

Author: Volkan Kaya, Servet Tuncer and Ahmet Baran

Content: Today, with the increasing number of criminal activities, automatic control systems are becoming the primary need for security forces. In this study, a new model is proposed to detect seven different weapon types using the deep learning method. This model offers a new approach to weapon classification based on the VGGNet architecture. The model is taught how to recognize assault rifles, bazookas, grenades, hunting rifles, knives, pistols, and revolvers. The proposed model is developed using the Keras library on the TensorFlow base. A new model is used to determine the method required to train, create layers, implement the training process, save training in the computer environment, determine the success rate of the training, and test the trained model.

2.4 LITERATURE SURVEY 4

“Border Security using IoT”

Author: Pooja S N , Spurthi T M, Rashmi R K, Samreen Unnisa, Deepika J

Content:

Terrorists cross our borders unknowingly. It is not possible for our soldiers to watch the borders at each and every moment. An essential requirement in security is the capability to automatically detect terrorist in borders . In this paper we propose an robot which identifies terrorist using IR sensor and capture image of terrorist using pi camera and sends notification to respective admin. If admin accept to shootout that terrorist then that notification is sent back to robot through server to kill that person, if admin decline to shootout then the process will stop their itself. This development enables security personnel to effectively detect terrorist at low cost.

CHAPTER 3

METHODOLOGY

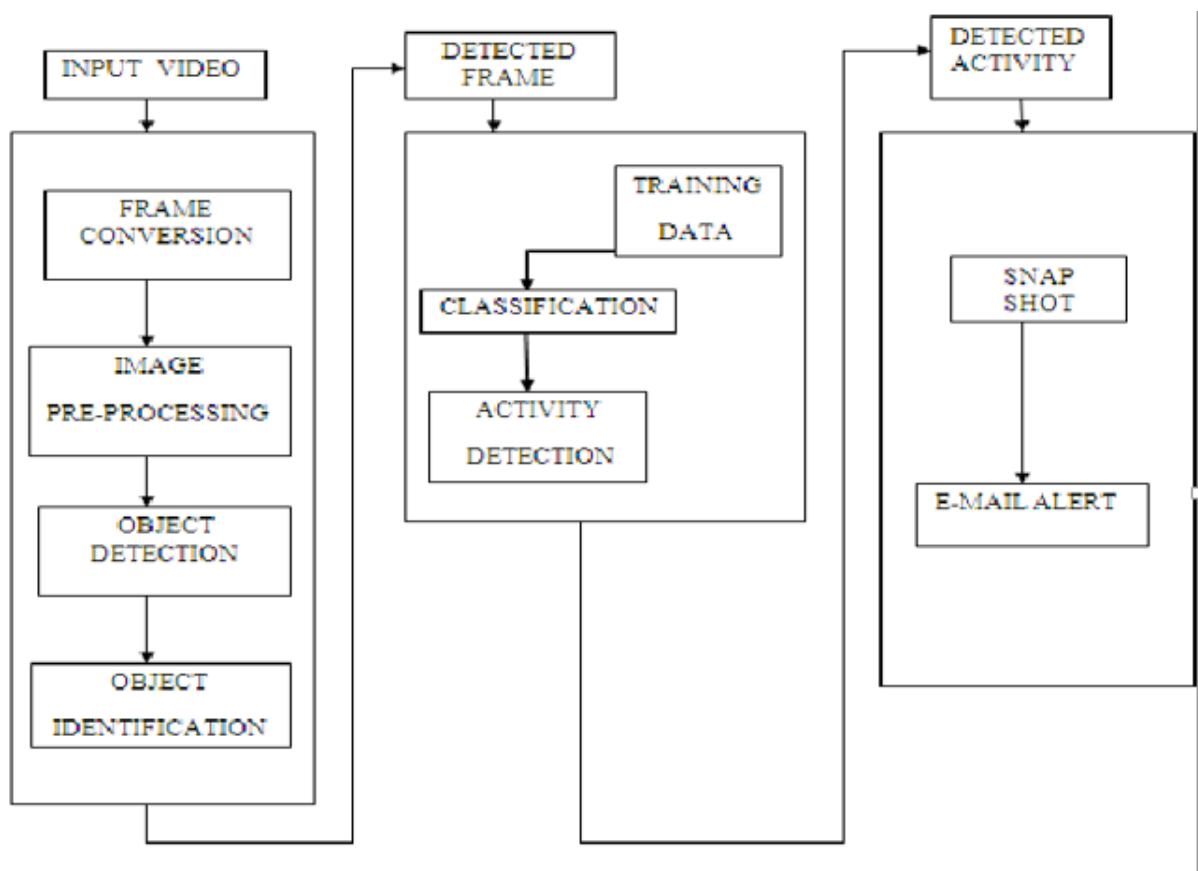
3.1 EXISTING SYSTEM

In previous years, though having surveillance cameras installed, to use them for security purposes was not an easy and dependable method. A human has to be there all the time to monitor screens. CCTV operator has to monitor 20- 25 screens for 10 hours. He has to look, observe, identify, and control the situation that can be harmful to the individuals and the property. As the number of screens increases, the concentration of the person decreases considerably to monitor each screen with time. It is impossible for the person monitoring the screens to keep the same level of attention all the time.

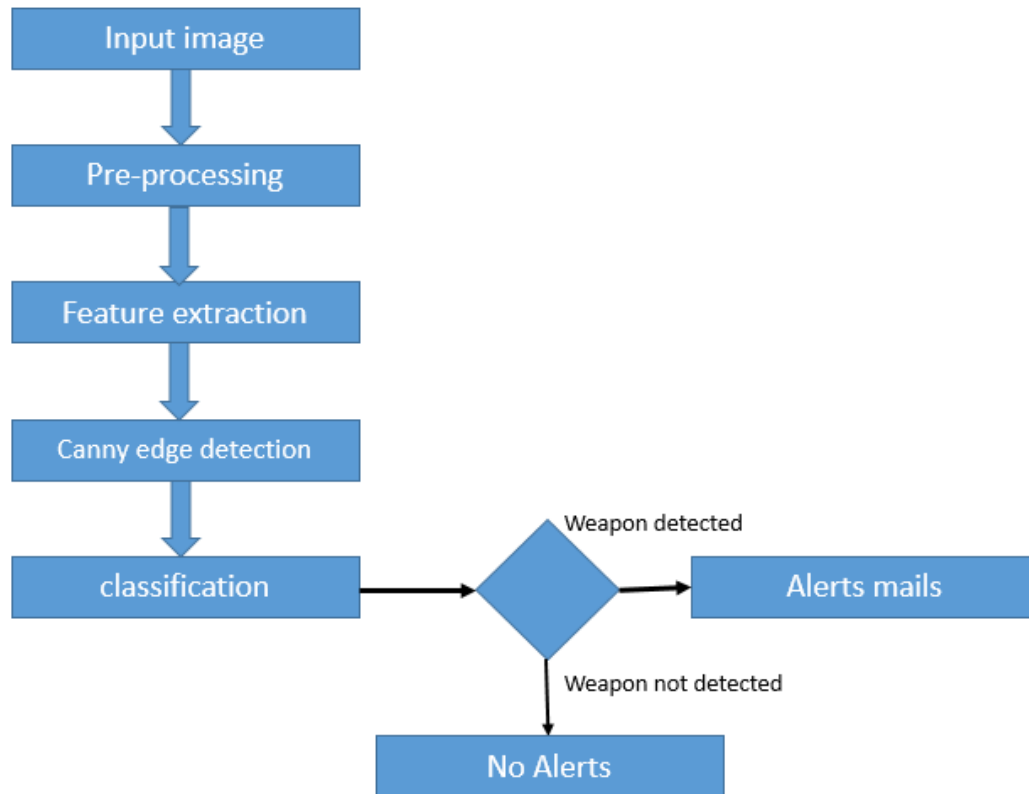
3.2 PROPOSED SYSTEM

In today's conditions, where criminal activities are increasing, it is very important to detect and recognize whether there is a weapon on a person based on images taken from security cameras, without requiring human intervention. Weapon detection and recognition are important to prevent criminal activities before they occur and so that the appropriate parties can take necessary action. Most criminal activities are carried out using handheld or carried weapons. Handheld or carried weapons are the most important elements used for various crimes, such as theft, illegal hunting, and terrorism. It is necessary to determine the types of weapons that may constitute a criminal element in order to predict whether there will be any criminal element in the images taken from security cameras and to take the necessary precautions beforehand.

3.3 BLOCK DIAGRAM



3.4 FLOW CHART



3.5 SOFTWARE REQUIRED

Software tool: Python 3.8

Programming language: python

Libraries: Tensorflow, keras, canny edge detection, CNN algorithms with Yolov3

3.6 METHODOLOGY

Deep learning is a branch of machine learning inspired by the functionality and structure of the human brain also called an artificial neural network. The methodology adopted in this work features the state of art deep learning, especially the convolutional neural networks due to their exceptional performance in this field. The aforementioned techniques are used for both the classification as well as localizing the specific object in a frame so both the object classification and detection algorithms were used and because our object is small with other object in background so after

experimentation we found the best algorithm for our case. Sliding window/classification and region proposal/object detection algorithms were used, and these techniques will be discussed later in this section. We had started by doing the classification using different deep learning models and achieved good precision but for the real-time scenarios, the low frame per seconds of classification models were the real issue in implementation. Oxford VGG, Google Inceptionv3 and InceptionResnetv2 were trained using the aforementioned approach. To achieve high precision, increase number of frame per seconds and improve localization, we moved to the object detection and region proposal methods. The different state of the art deep learning models for object detection were used and the results were compared in terms of precision, speed, and standard metric of F1 score. State of the art deep learning based SSDMobileNetv1, YOLOv3, FasterRCNN-InceptionResnetv2, and YOLOv4 were trained and tested. Different datasets were made keeping in mind the classification and detection problem as both have a separate requirement for performing the tasks to achieve high accuracy, mean average precision as well as frame per second for the realtime implementation. To understand object classification and detection let us first briefly understand object recognition as both the aforementioned types come under the umbrella of this and combined classification and localization make detection possible for any kind of detection problem giving class name as well as the region where our desired object is in the frame.

3.7 PRE-PROCESSING:

Image pre-processing is the term for operations on images at the lowest level of abstraction. These operations do not increase image information content but they decrease it if entropy is an information measure. The aim of pre-processing is an improvement of the image data that suppresses undesired distortions or enhances some image features relevant for further processing and analysis task. Image pre-processing use the redundancy in images. Neighbouring pixels corresponding to one real object have the same or similar brightness value. If a distorted pixel can be 98 picked out from the image, it can be restarted as an average value of neighbouring pixels. Image pre-processing methods can be classified into categories according to the size of the pixel neighbourhood that is used for the calculation of a new pixel brightness. In this paper, it will be presented some pixel brightness transformations and local pre-processing methods realized in python.

3.7.1 IMAGE CROPPING

The first step in image pre-processing is image cropping. Some irrelevant parts of the image can be removed and the image region of interest is focused. This tool provides a user with the size information of the cropped image. Mat Lab function for image cropping realizes this operation interactively waiting for an user to specify the crop rectangle with the mouse and operates on the current axes. The output image is of the same class as the input image. The two-dimensional convolution operation is fundamental to the analysis of images. A new value is ascribed to a given pixel based on the evaluation of a weighted average of pixel values in a $k \times k$ neighbourhood of the central pixel. Convolution kernel or the filter mask is represented with weights supplied in a square matrix. It is applied to each pixel in an image. Discrete form of the 2D convolution operator is defined by the following relationship between the elements $f_i(x, y)$ of the input image, the elements $h(\alpha, \beta)$ of the convolution kernel, and the elements $g(x, y)$ of the output image by the following master formula $g(x, y) = \sum_{\alpha=-\frac{k-1}{2}}^{\frac{k-1}{2}} \sum_{\beta=-\frac{k-1}{2}}^{\frac{k-1}{2}} f_i(x - \alpha, y - \beta) h(\alpha, \beta)$, where x, y, α and β are integers [4]. Coefficients of the kernel H represent a discrete approximation of the analytical form of the response function characterizing the desired filter. In practical cases, the kernel is a square array and $k_x = k_y = k$, where k is odd and much smaller than the linear image dimension. There is the following steps, realized for each pixel P represented by (x, y) :

- Placement of H on P ;
- Multiplication of each pixel in the $k \times k$ neighbourhood by the appropriate filter mask;
- Summation of all products;
- Placement of the normalized sum into position P of the output image.

This tool for pre-processing lets an user explore 2-D Finite Impulse Response filters. By changing the cut-off frequency and filter order, the user can design filter and can see the designed filter's coefficients and frequency response. Median filtering is a non-linear smoothing method that reduces the blurring of edges and significantly eliminates impulse noise [3, 4]. It suppresses image noise without reducing the image sharpness and can be applied iteratively. The brightness value of the current pixel in the image is replaced by the median brightness of either 3-by-3 or 4-by-4 neighbourhood.

3.7.2 INTENSITY ADJUSTEMENT

A gray-scale transformation T of the original brightness p from scale $[p_0, p_k]$ into brightness q from a new scale $[q_0, q_k]$ is given by $q = T(p)$. It does not depend on the position of the pixel in the image. Values below p_0 and above p_k are clipped. Values below p_0 map to q_0 , and those above p_k map to q_k . Alpha argument specifies the shape of the curve describing the relationship between the values in the input image and output image. If alpha is less than 1, the mapping is weighted toward brighter output values. If alpha is greater than 1, the mapping is weighted toward lower darker output values. If the argument is omitted its default value is 1. Graphical controls enable a user to increase and decrease the brightness, contrast and alpha correction. Another offered possibility to enhance the contrast of image, by transforming the values in an intensity image so that the histogram of the output image matches 100 a specified histogram, is histogram equalization technique. Region description is based on its statistical grey-level properties. Histogram provides the frequency of the brightness value in the image. An image with n grey levels is represented with one-dimensional array with n elements. The n -th element of array contains the number of pixels whose grey level is n . Assume that the pixel values are normalized and lie in the range $[0, 1]$. Let $s = T(r)$, for any $r \in [0, 1]$, is transformation function which satisfies the following conditions:

- $T(r)$ is single valued and monotonically increasing in the interval $[0, 1]$;
- $0 \leq T(r) \leq 1$ for any $r \in [0, 1]$.

The original and transformed grey levels can be characterized by their probability density functions. Contrast is the local change in brightness and is defined as the ratio between average brightness of an object and the background brightness. Histogram equalization technique is based on modifying the appearance of an image by controlling the probability density function of its gray levels by the transformation function $T(r)$. This technique enhances the contrast in the image

3.7.3 BRIGHTNESS THRESHOLDING

Brightness thresholding is an indispensable step in extracting pertinent information. A gray-scale image often contains only two level of significant information: the foreground level constituting objects of interest and the background level against which the foreground is discriminated [1]. A complete segmentation of an image R is

a finite set of regions $R_1, R_2, R_m, R = [m \ i=1 \ R_i, R_i \cap R_j = \emptyset \ i \neq j$. If R_b is a background in the image, then $S_{m \ i=1, i \neq b} R_i$ is considered the object and $R_{C \ b} = S_{m \ i=1, i \neq b} R_i$, where $R_{C \ b}$ is the set complement. While there are two principal peaks of the foreground and the background intensities in the image histogram, there are many other gray intensities present. Binarization can be accomplished by the choice of an intensity, between the two histogram peaks, that is the threshold between all background intensities below and all foreground intensities above. The input image I_1 is being transformed to an output binary segmented image I_2 , in the following way $I_2(i, j) = (\ 1; I_1(i, j) \geq T \ 0; I_1(i, j) < T$ where T is the threshold. $I_2(i, j) = 1$ for the object elements and $I_2(i, j) = 0$ for the background elements. There are different approaches for image binarization depending on the type of image [7, 8]. Successful threshold segmentation depends on the threshold selection. A number of conditions like poor image contrast or spatial non uniformities in background intensity can make difficult to resolve foreground from background. These cases require user interaction for specifying the desired object and its distinguishing intensity features.

3.7.4 CLEARING AREAS OF A BINARY IMAGE

If there is a deformation of the expected shape and size of the border and the whole region during the separation of image object from its background, it can be partially overcome. Usually small polygon mask, located next to the region and out of it, is added to clear image area with similar brightness of the region. This mask can reshape image objects and provides a separation image objects from each other and from their image background. This operation is realized interactively, adding vertices to the polygon. Selecting a final vertex of the polygon over a white colour image region, the fill is started and recolored to black. Created fill is a logical mask and the input image is logical matrix. Using logical operator AND under logical arguments, the output image is also obtained as a logical array. If the white regions represents image foreground on the black background, whole objects or their parts can be deleted. Special user requirements about the size and shape of the observed object, can be realized by the same way.

3.7.5 DETECTING EDGES

Edges are pixels where the intensity image function changes abruptly. Edge detectors are collection of local image pre-processing methods used to locate changes in the

brightness function. An image function depends on two variables, co-ordinates in the image plane. Operators describing edges are expressed by partial derivatives. A change of the image function can be described by a gradient that points in the direction of the largest growth of the image function. An edge is a vector variable with two components, magnitude and direction. The edge magnitude is the magnitude of the gradient. The edge direction is rotated with respect to the gradient direction by $-\pi/2$. The gradient direction gives the direction of maximum growth of the function, e.g., from black to white. The boundary and its parts are perpendicular to the direction of the gradient [2, 9]. The gradient magnitude $|\text{grad } g(x, y)| = \sqrt{\left(\frac{\partial g}{\partial x}\right)^2 + \left(\frac{\partial g}{\partial y}\right)^2}$ and gradient direction $\phi = \arctan\left(\frac{\partial g}{\partial y} / \frac{\partial g}{\partial x}\right)$ are continuous image functions where $\arctan(x, y)$ is the angle from x axis to the point (x, y). A digital image is discrete in nature and these equations must be approximated by differences of the image g, in the vertical direction for fixed i and in the horizontal direction for fixed j, by following equations $\Delta_i g(i, j) = g(i, j) - g(i - n, j)$ $\Delta_j g(i, j) = g(i, j) - g(i, j - n)$, where n is a small integer chosen so to provide a good approximation to the derivative and to neglect unimportant changes in the image function. 103 Gradient operators approximating derivatives of the image function using differences use one or several convolution masks. Beside them, this tool uses operators based on the zero-crossings of the image function second derivative. Sobel, Prewitt, and Roberts methods find edges by thresholding the gradient and by them horizontal edges, vertical edges or both can be detected. The Laplacian of Gaussian method thresholds the slope of the zero crossings after filtering the image with a LoG filter. Canny method thresholds the gradient using the derivative of a Gaussian filter. One option in the main menu provides processing of the image region of interest or the whole image. It has to move the pointer over the image on the left and when the cursor changes to a crosshair, it has to click on points in the image to select vertices of the region of interest. Offered operations to perform are UN sharpening, histogram equalization technique, low pass filtering, median filtering, and brightening, darkening, increasing contrast, decreasing contrast and boundary interpolation.

CHAPTER 4

SOFTWARE MODULES

4.1 PYTHON (PROGRAMMING LANGUAGE)

Python is a widely used high-level programming language for general-purpose programming, created by Guido van Rossum and first released in 1991. An interpreted language, Python has a design philosophy which emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly brackets or keywords), and a syntax which allows programmers to express concepts in fewer lines of code than possible in languages such as C++ or Java. The language provides constructs intended to enable writing clear programs on both a small and large scale. Python features a dynamic type system and automatic memory management and supports multiple programming paradigms, including object-oriented, imperative, functional programming, and procedural styles. It has a large and comprehensive standard library. Python interpreters are available for many operating systems, allowing Python code to run on a wide variety of systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation.

4.1.1 PYTHON HISTORY

Python was conceived in the late 1980s, and its implementation began in December 1989 [28] by Guido van Rossum at Centrum Wiskunde & Informatica (CWI) in the Netherlands as a successor to the ABC language (itself inspired by SETL) capable of exception handling and interfacing with the operating system Amoeba. Van Rossum is Python's principal author, and his continuing central role in deciding the direction of Python is reflected in the title given to him by the Python community, benevolent dictator for life (BDFL).

About the origin of Python, Van Rossum wrote in 1996: Over six years ago, in December 1989, I was looking for a "hobby" programming project that would keep me occupied during the week around Christmas. My office ... would be closed, but I had a home computer, and not much else on my hands. I decided to write an interpreter for the new scripting language I had been thinking about lately: a descendant of ABC

that would appeal to Unix/C hackers. I chose Python as a working title for the project, being in a slightly irreverent mood (and a big fan of Monty Python's Flying Circus). Python 2.0 was released on 16 October 2000 and had many major new features, including a cycle-detecting garbage collector and support for Unicode. With this release the development process was changed and became more transparent and community-backed. Python 3.0 (which early in its development was commonly referred to as Python 3000 or py3k), a major, backwards-incompatible release, was released on 3 December 2008 after a long period of testing.

Many of its major features have been backported to the backwards-compatible Python 2.6.x and 2.7.x version series. The End Of Life date (EOL, sunset date) for Python 2.7 was initially set at 2015, then postponed to 2020 out of concern that a large body of existing code cannot easily be forward-ported to Python 3. In January 2017, Google announced work on a Python 2.7 to go transcompiler, which The Register speculated was in response to Python 2.7's planned end-of-life but Google cited performance under concurrent workloads as their only motivation.

Features and philosophy Python is a multi-paradigm programming language: object-oriented programming and structured programming are fully supported, and many language features support functional programming and aspect-oriented programming (including by metaprogramming and metaobjects (magic methods)). Many other 6/13/2017 Python (programming language) paradigms are supported via extensions, including design by contract and logic programming. Python uses dynamic typing and a mix of reference counting and a cycle-detecting garbage collector for memory management. An important feature of Python is dynamic name resolution (late binding), which binds method and variable names during program execution.

The design of Python offers some support for functional programming in the Lisp tradition. The language has `map()`, `reduce()` and `filter()` functions; list comprehensions, dictionaries, and sets; and generator expressions. The standard library has two modules (`itertools` and `functools`) that implement functional tools borrowed from Haskell and Standard ML.

The core philosophy of the language is summarized by the document The Zen of Python (PEP 20), which includes aphorisms such as: Beautiful is better than ugly Explicit is better than implicit Simple is better than complex Complex is better than complicated Readability counts Rather than requiring all desired functionality to be built into the language's core, Python was designed to be highly extensible. Python can also be embedded in existing applications that need a programmable interface.

This design of a small core language with a large standard library and an easily extensible interpreter was intended by Van Rossum from the start because of his frustrations with ABC, which espoused the opposite mindset. While offering choice in coding methodology, the Python philosophy rejects exuberant syntax, such as in Perl, in favor of a sparser, less-cluttered grammar.

As Alex Martelli put it: "To describe something as clever is not considered a compliment in the Python culture." Python's philosophy rejects the Perl "there is more than one way to do it" approach to language design in favor of "there should be one—and preferably only one—obvious way to do it". Python's developers strive to avoid premature optimization, and moreover, reject patches to non-critical parts of C Python that would offer a marginal increase in speed at the cost of clarity. When speed is important, a Python programmer can move time-critical functions to extension modules written in languages such as C, or try using PyPy, a just-in-time compiler. Cython is also available, which translates a Python script into C and makes direct C-level API calls into the Python interpreter. An important goal of Python's developers is making it fun to use. This is reflected in the origin of the name, which comes from Monty Python, and in an occasionally playful approach to tutorials and reference materials, such as using examples that refer to spam and eggs instead of the standard foo and bar. A common neologism in the Python community is pythonic, which can have a wide range of meanings related to program style.

To say that code is pythonic is to say that it uses Python idioms well, that it is natural or shows fluency in the language, that it conforms with Python's minimalist philosophy and emphasis on readability. In contrast, code that is difficult to understand or reads like a rough transcription from another programming language is called unpythonic. Users and admirers of Python, especially those considered knowledgeable or experienced, are often referred to as Pythonists, Pythonistas, and Pythoneers. The third slice parameter, called step or stride, allows elements to be skipped and reversed. Slice indexes may be omitted, for example `a[:]` returns a copy of the entire list. Each element of a slice is a shallow copy. In Python, a distinction between expressions and statements is rigidly enforced, in contrast to languages such as Common Lisp, Scheme, or Ruby. This leads to duplicating some functionality.

For example: List comprehensions vs. for-loops Conditional expressions vs. if blocks The `eval()` vs. `exec()` built-in functions (in Python 2, `exec` is a statement); the former is for expressions, the latter is for statements. 6/13/2017 Python (programming language) - Statements cannot be a part of an expression, so list and other comprehensions or lambda expressions, all being expressions, cannot contain statements.

A particular case of this is that an assignment statement such as `a = 1` cannot form part of the conditional expression of a conditional statement. This has the advantage of avoiding a classic C error of mistaking an assignment operator `=` for an equality operator `==` in conditions: `if (c = 1) { ... }` is syntactically valid (but probably unintended) C code but `if c = 1: ...` causes a syntax error in Python. Methods Methods on objects are functions attached to the object's class; the syntax `instance.method(argument)` is, for normal methods and functions, syntactic sugar for `Class.method(instance, argument)`. Python methods have an explicit `self` parameter to access instance data, in contrast to the implicit `self` (or `this`) in some other object-oriented programming languages (e.g., C++, Java, Objective-C, or Ruby). Typing Python uses duck typing and has typed objects but untyped variable names. Type constraints are not checked at compile time; rather, operations on an object may fail, signifying that the given object is not of a suitable type. Despite being dynamically typed, Python is strongly

typed, forbidding operations that are not well-defined (for example, adding a number to a string) rather than silently attempting to make sense of them. Python allows programmers to define their own types using classes, which are most often used for objectoriented programming.

New instances of classes are constructed by calling the class (for example, `SpamClass()` or `EggsClass()`), and the classes are instances of the metaclass type (itself an instance of itself), allowing meta programming and reflection. Before version 3.0, Python had two kinds of classes: old-style and new-style. The syntax of both styles is the same, the difference being whether the class object is inherited from, directly or indirectly (all new-style classes inherit from `object` and are instances of `type`). In versions of Python 2 from Python 2.2 onwards, both kinds of classes can be used. Old-style classes were eliminated in Python 3.0. The long term plan is to support gradual typing and as of Python 3.5, the syntax of the language allows specifying static types but they are not checked in the default implementation, CPython. An experimental optional static type checker named `mypy` supports compile-time type checking.

4.2 NUMPY

NumPy (**Numerical Python**) is an opensource Python library that's used in almost every field of science and engineering. It's the universal standard for working with numerical data in Python, and it's at the core of the scientific Python and PyData ecosystems. NumPy users include everyone from beginning coders to experienced researchers doing state-of-the-art scientific and industrial research and development. The NumPy API is used extensively in Pandas, SciPy, Matplotlib, scikit-learn, scikit-image and most other data science and scientific Python packages.

The NumPy library contains multidimensional array and matrix data structures (you'll find more information about this in later sections). It provides **`ndarray`**, a homogeneous n-dimensional array object, with methods to efficiently operate on it. NumPy can be used to perform a wide variety of mathematical operations on arrays. It adds powerful data structures to Python that guarantee efficient calculations with

arrays and matrices and it supplies an enormous library of high-level mathematical functions that operate on these arrays and matrices.

4.2.1 INSTALL NUMPY

To install NumPy, we strongly recommend using a scientific Python distribution. If you're looking for the full instructions for installing NumPy on your operating system

If you already have Python, you can install NumPy with:

```
conda install numpy
```

or

```
pip install numpy
```

If you don't have Python yet, you might want to consider using [Anaconda](#). It's the easiest way to get started. The good thing about getting this distribution is the fact that you don't need to worry too much about separately installing NumPy or any of the major packages that you'll be using for your data analyses, like pandas, Scikit-Learn, etc.

4.2.2 IMPORT NUMPY

To access NumPy and its functions import it in your Python code like this:

```
import numpy as np
```

We shorten the imported name to np for better readability of code using NumPy. This is a widely adopted convention that you should follow so that anyone working with your code can easily understand it.

Reading the example code

If you aren't already comfortable with reading tutorials that contain a lot of code, you might not know how to interpret a code block that looks like this:

```
>>> a = np.arange(6)
```

```
>>> a2 = a[np.newaxis, :]  
>>> a2.shape  
(1, 6)
```

If you aren't familiar with this style, it's very easy to understand. If you see `>>>`, you're looking at **input**, or the code that you would enter. Everything that doesn't have `>>>` in front of it is **output**, or the results of running your code. This is the style you see when you run python on the command line, but if you're using IPython, you might see a different style. Note that it is not part of the code and will cause an error if typed or pasted into the Python shell. It can be safely typed or pasted into the IPython shell; the `>>>` is ignored.

4.2.3 USE OF NUMPY

NumPy arrays are faster and more compact than Python lists. An array consumes less memory and is convenient to use. NumPy uses much less memory to store data and it provides a mechanism of specifying the data types. This allows the code to be optimized even further.

4.2.4 NUMPY ARRAY

An array is a central data structure of the NumPy library. An array is a grid of values and it contains information about the raw data, how to locate an element, and how to interpret an element. It has a grid of elements that can be indexed in **various ways**. The elements are all of the same type, referred to as the array dtype.

An array can be indexed by a tuple of nonnegative integers, by booleans, by another array, or by integers. The rank of the array is the number of dimensions. The shape of the array is a tuple of integers giving the size of the array along each dimension.

One way we can initialize NumPy arrays is from Python lists, using nested lists for two- or higher-dimensional data.

For example:

```
>>> a = np.array([1, 2, 3, 4, 5, 6])
```

or:

```
>>> a = np.array([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]])
```

We can access the elements in the array using square brackets. When you're accessing elements, remember that indexing in NumPy starts at 0. That means that if you want to access the first element in your array, you'll be accessing element "0".

```
>>> print(a[0])  
[1 2 3 4]
```

4.3 MATPLOTLIB

Matplotlib is a cross-platform, data visualization and graphical plotting library for Python and its numerical extension NumPy. As such, it offers a viable open source alternative to MATLAB. Developers can also use matplotlib's APIs (Application Programming Interfaces) to embed plots in GUI applications.

A Python matplotlib script is structured so that a few lines of code are all that is required in most instances to generate a visual data plot. The matplotlib scripting layer overlays two APIs:

- The pyplot API is a hierarchy of Python code objects topped by *matplotlib.pyplot*
- An OO (Object-Oriented) API collection of objects that can be assembled with greater flexibility than pyplot. This API provides direct access to Matplotlib's backend layers.

4.3.1 MATPLOTLIB & PYPLOT

The pyplot API has a convenient MATLAB-style stateful interface. In fact, matplotlib was originally written as an open source alternative for MATLAB. The OO API and its interface is more customizable and powerful than pyplot, but considered more difficult to use. As a result, the pyplot interface is more commonly used, and is referred to by default in this article.

Understanding matplotlib's pyplot API is key to understanding how to work with plots:

- **matplotlib.pyplot.figure:** **Figure** is the top-level container. It includes everything visualized in a plot including one or more **Axes**.
- **matplotlib.pyplot.axes:** **Axes** contain most of the elements in a plot: **Axis**, **Tick**, **Line2D**, **Text**, etc., and sets the coordinates. It is the area in which data is plotted. Axes include the X-Axis, Y-Axis, and possibly a Z-Axis, as well.

Installing Matplotlib

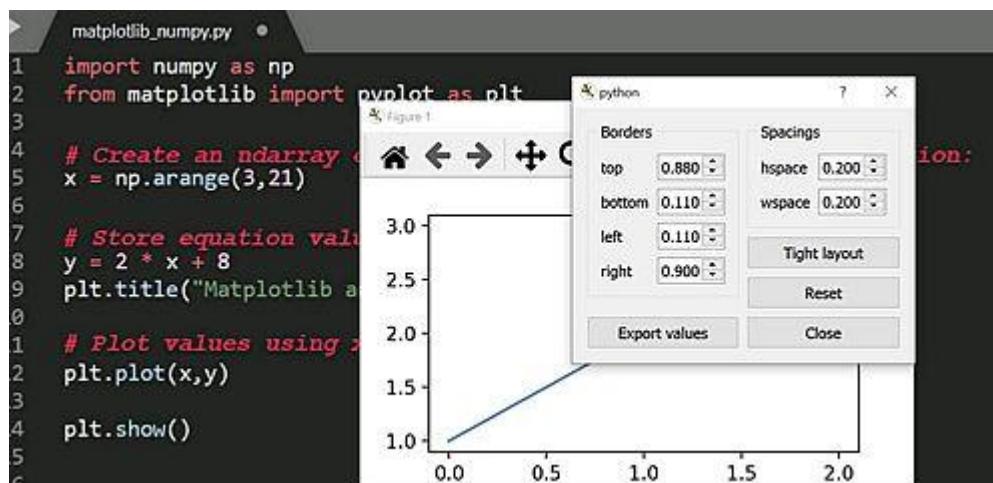
Matplotlib and its dependencies can be downloaded as a binary (pre-compiled) package from the Python Package Index (PyPI), and installed with the following command:

```
python -m pip install matplotlib
```

Matplotlib is also available as uncompiled source files. Compiling from source will require your local system to have the appropriate compiler for your OS, all dependencies, setup scripts, configuration files, and patches available. This can result in a fairly complex installation. Alternatively, consider using the **ActiveState Platform** to automatically build matplotlib from source and package it for your OS.

Matplotlib UI Menu

When matplotlib is used to create a plot, a User Interface (UI) and menu structure are generated. The UI can be used to customize the plot, as well as to pan/zoom and toggle various elements.



4.3.2 MATPLOTLIB & NUMPY

Numpy is a package for scientific computing. Numpy is a required dependency for matplotlib, which uses numpy functions for numerical data and multi-dimensional arrays as shown in the following code snippet

4.3.3 MATPLOTLIB & PANDAS

Pandas is a library used by matplotlib mainly for data manipulation and analysis. Pandas provides an in-memory 2D data table object called a Dataframe. Unlike numpy, pandas is not a required dependency of matplotlib.

Pandas and numpy are often used together, as shown in the following code snippet:

```
matplotlib_pandas.py
1 import numpy as np
2 import pandas as pd
3 import matplotlib.pyplot as plt
4
5 fig, ax = plt.subplots()
6
7 # Hide axes without removing it:
8 fig.patch.set_visible(False)
9 ax.axis('off')
10 ax.axis('tight')
11 # Create NumPy random array in Pandas dataframe, with 10 rows,
12 4 columns:
13 df = pd.DataFrame(np.random.randn(10,4), columns=list('ABCD'))
14
15 ax.table(cellText=df.values, colLabels=df.columns, loc='center')
16
17 fig.tight_layout()
18 plt.show()
```

4.3.4 MATPLOTLIB & PYPLOT

In this example, pyplot is imported as plt, and then used to plot three numbers in a straight line:

```
import matplotlib.pyplot as plt
```

```
# Plot some numbers:
```

```
plt.plot([1, 2, 3])
```

```
plt.title("Line Plot")
```

```
# Display the plot:
```

```
plt.show()
```

Figure 1. Line plot generated by Matplotlib:

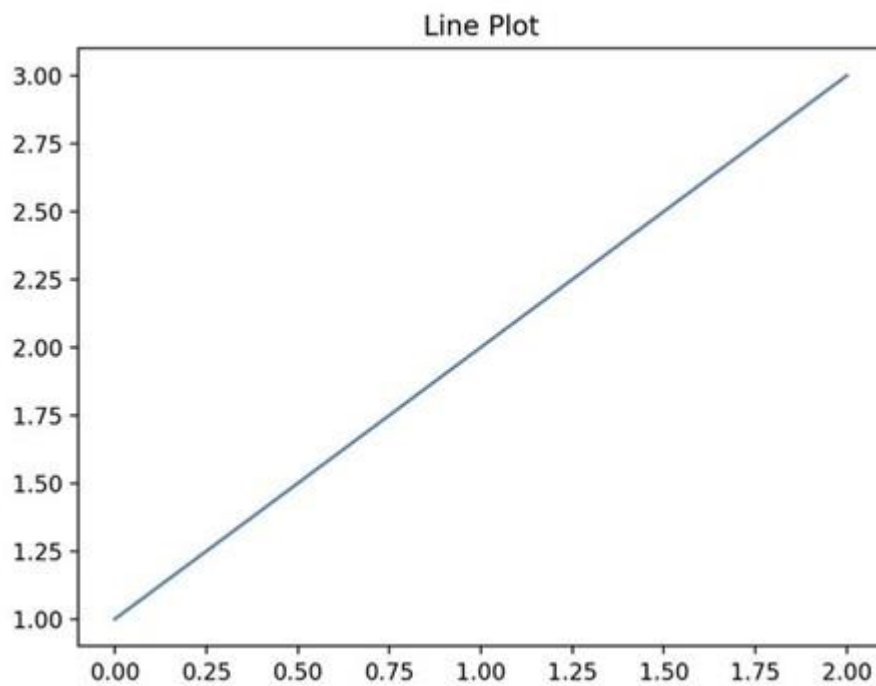
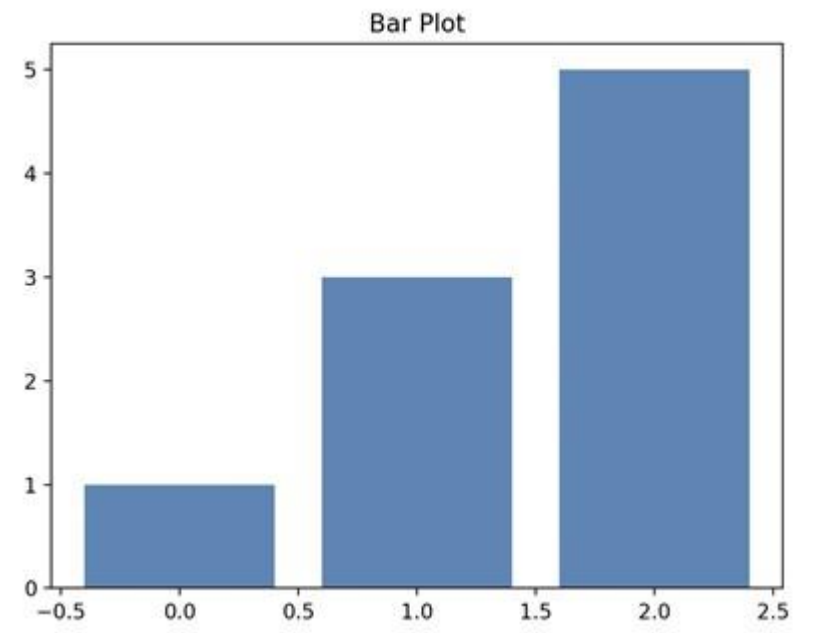


Figure 2. Bar plot generated by Matplotlib:



Matplotlib: Plot a Numpy Array

In this example, pyplot is imported as plt, and then used to plot a range of numbers stored in a numpy array:

```
import numpy as np

from matplotlib import pyplot as plt

# Create an ndarray on x axis using the numpy range() function:

x = np.arange(3,21)

# Store equation values on y axis:

y = 2 * x + 8
```

```
plt.title("NumPy Array Plot")

# Plot values using x,y coordinates:
plt.plot(x,y)
plt.show()
```

CHAPTER 5

CONCLUSION

5.1 CONCLUSION

In this review, the advanced YOLO V3 recognition framework was executed and prepared on informational index to recognize weapons. To decide the wellbeing of the weapon, we give a model that makes the machine or robot apparent, and can tell the commandant when the firearm or weapon is plainly noticeable as an afterthought. Test results show that the planned YOLO V3 has preferable execution over the YOLO V2 and is less expensive in numbers. + Modern observing abilities should be fortified and created to help worker execution. The framework will successfully supplant existing foundation with minimal expense stockpiling, video foundation, and further developed picture handling.

REFERENCE

- [1] (2019). Christchurch Mosque Shootzings. Accessed: Jul. 10, 2019. [Online]. Available: https://en.wikipedia.org/wiki/Christchurch_mosque_shootings
- [2] W. Deisman, "CCTV: Literature review and bibliography," in Research and Evaluation Branch, Community, Contract and Aboriginal Policing Services Directorate. Ottawa, ON, Canada: Royal Canadian Mounted, 2003.
- [3] J. Ratcliffe, "Video surveillance of public places," US Dept. Justice, Office Community Oriented Policing Services, Washington, DC, USA, Tech. Rep. 4, 2006.
- [4] M. Grega, A. Mاتیولاسکی, P. Guzik, and M. Leszczuk, "Automated detection of firearms and knives in a CCTV image," *Sensors*, vol. 16, no. 1, p. 47, Jan. 2016.
- [5] N. Cohen, J. Gattuso, and K. MacLennan-Brown. CCTV Operational Requirements Manual 2009. St Albans, U.K.: Home Office Scientific Development Branch, 2009.
- [6] G. Flitton, T. P. Breckon, and N. Megherbi, "A comparison of 3D interest point descriptors with application to airport baggage object detection in complex CT imagery," *Pattern Recognit.*, vol. 46, no. 9, pp. 2420–2436, Sep. 2013.
- [7] R. Gesick, C. Saritac, and C.-C. Hung, "Automatic image analysis process for the detection of concealed weapons," in *Proc. 5th Annu. Workshop Cyber Secur. Inf. Intell. Res. Cyber Secur. Inf. Intell. Challenges Strategies (CSIIRW)*, 2009, p. 20.
- [8] R. K. Tiwari and G. K. Verma, "A computer vision based framework for visual gun detection using Harris interest point detector," *Procedia Comput. Sci.*, vol. 54, pp. 703–712, Aug. 2015.
- [9] R. K. Tiwari and G. K. Verma, "A computer vision based framework for visual gun detection using SURF," in *Proc. Int. Conf. Electr., Electron., Signals, Commun. Optim. (EESCO)*, Jan. 2015, pp. 1–5.
- [10] Z. Xiao, X. Lu, J. Yan, L. Wu, and L. Ren, "Automatic detection of concealed pistols using passive millimeter wave imaging," in *Proc. IEEE Int. Conf. Imag. Syst. Techn. (IST)*, Sep. 2015, pp. 1–4.

- [11] D. M. Sheen, D. L. McMakin, and T. E. Hall, "Three-dimensional millimeter-wave imaging for concealed weapon detection," *IEEE Trans. Microw. Theory Techn.*, vol. 49, no. 9, pp. 1581–1592, Sep. 2001.
- [12] Z. Xue, R. S. Blum, and Y. Li, "Fusion of visual and IR images for concealed weapon detection," in *Proc. 5th Int. Conf. Inf. Fusion*, vol. 2, Jul. 2002, pp. 1198–1205.
- [13] R. Blum, Z. Xue, Z. Liu, and D. S. Forsyth, "Multisensor concealed weapon detection by using a multiresolution mosaic approach," in *Proc. IEEE 60th Veh. Technol. Conf. (VTC-Fall)*, vol. 7, Sep. 2004, pp. 4597–4601.
- [14] E. M. Upadhyay and N. K. Rana, "Exposure fusion for concealed weapon detection," in *Proc. 2nd Int. Conf. Devices, Circuits Syst. (ICDCS)*, Mar. 2014, pp. 1–6.
- [15] R. Maher, "Modeling and signal processing of acoustic gunshot recordings," in *Proc. IEEE 12th Digit. Signal Process. Workshop 4th IEEE Signal Process. Educ. Workshop*, Sep. 2006, pp. 257–261.
- [16] A. Chacon-Rodriguez, P. Julian, L. Castro, P. Alvarado, and N. Hernandez, "Evaluation of gunshot detection algorithms," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 58, no. 2, pp. 363–373, Feb. 2011.
- [17] (2019). From Edison to Internet: A History of Video Surveillance. Accessed: Jun. 13, 2019. [Online]. Available: <https://www.business2community.com/tech-gadgets/from-edison-to-internet-ahistory-of-video-surveillance-0578308>
- [18] (2019). Infographic: History of Video Surveillance—IFSEC Global | Security and Fire News and Resources. Accessed: Sep. 15, 2019. [Online]. Available: <https://www.ifsecglobal.com/video-surveillance/infographichistory-of-video-surveillance/>

APPENDICES

A. SAMPLE CODE

```
import cv2

import numpy as np


# Load Yolo

# Download weight file(yolov3_training_2000.weights) from this link :-
https://drive.google.com/file/d/10uJEsUpQI3EmD98iwrwzbD4e19Ps-
LHZ/view?usp=sharing

net = cv2.dnn.readNet("yolov3_training_2000.weights", "yolov3_testing.cfg")

classes = ["Weapon"]

# with open("coco.names", "r") as f:

# classes = [line.strip() for line in f.readlines()]


layer_names = net.getLayerNames()

output_layers = [layer_names[i- 1] for i in net.getUnconnectedOutLayers()]

colors = np.random.uniform(0, 255, size=(len(classes), 3))


# Loading image

# img = cv2.imread("room_ser.jpg")

# img = cv2.resize(img, None, fx=0.4, fy=0.4)
```

```
# Enter file name for example "ak47.mp4" or press "Enter" to start webcam
```

```
def value():
```

```
    val = input("Enter file name or press enter to start webcam : \n")
```

```
    if val == "":
```

```
        val = 0
```

```
    return val
```

```
# for video capture
```

```
cap = cv2.VideoCapture(value())
```

```
# val = cv2.VideoCapture()
```

```
while True:
```

```
    _, img = cap.read()
```

```
    height, width, channels = img.shape
```

```
    # width = 512
```

```
    # height = 512
```

```
# Detecting objects
```

```
blob = cv2.dnn.blobFromImage(img, 0.00392, (416, 416), (0, 0, 0), True, crop=False)
```

```
net.setInput(blob)
```

```
outs = net.forward(output_layers)
```

```

# Showing information on the screen

class_ids = []

confidences = []

boxes = []

for out in outs:

    for detection in out:

        scores = detection[5:]

        class_id = np.argmax(scores)

        confidence = scores[class_id]

        if confidence > 0.5:

            # Object detected

            center_x = int(detection[0] * width)

            center_y = int(detection[1] * height)

            w = int(detection[2] * width)

            h = int(detection[3] * height)

            # Rectangle coordinates

            x = int(center_x - w / 2)

            y = int(center_y - h / 2)

            boxes.append([x, y, w, h])

            confidences.append(float(confidence))

            class_ids.append(class_id)

indexes = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.4)

```

```

print(indexes)

if indexes == 0: print("weapon detected in frame")

font = cv2.FONT_HERSHEY_PLAIN

for i in range(len(boxes)):

    if i in indexes:

        x, y, w, h = boxes[i]

        label = str(classes[class_ids[i]])

        color = colors[class_ids[i]]

        cv2.rectangle(img, (x, y), (x + w, y + h), color, 2)

        cv2.putText(img, label, (x, y + 30), font, 3, color, 3)


# frame = cv2.resize(img, (width, height), interpolation=cv2.INTER_AREA)

cv2.imshow("Image", img)

directory = r'C:\Users\dell\Desktop\weapon_detection'

#os.chdir(directory)

filename = 'savedImage.jpg'

#cv2.imwrite(filename, saved_image)

key = cv2.waitKey(1)

if key == 27:

    break

cap.release()

cv2.destroyAllWindows()

```

[net]

Testing

batch=8

subdivisions=1

Training

batch=64

subdivisions=166

width=416

height=416

channels=3

momentum=0.9

decay=0.0005

angle=0

saturation = 1.5

exposure = 1.5

hue=.1

learning_rate=0.001

burn_in=1000

max_batches = 4000

policy=steps

steps=400000,450000

scales=.1,.1

[convolutional]

batch_normalize=1

filters=32

size=3

stride=1

pad=1

activation=leaky

Downsample

[convolutional]

batch_normalize=1

filters=64

size=3

stride=2

pad=1

activation=leaky

[convolutional]

batch_normalize=1

filters=32

size=1

stride=1

pad=1

activation=leaky

[convolutional]

batch_normalize=1

filters=64

size=3

stride=1

pad=1

activation=leaky

[shortcut]

from=-3

activation=linear

Downsample

[convolutional]

batch_normalize=1

filters=128

size=3

stride=2

pad=1

activation=leaky

[convolutional]

batch_normalize=1

filters=64

size=1

stride=1

pad=1

activation=leaky

[convolutional]

batch_normalize=1

filters=128

size=3

stride=1

pad=1

activation=leaky

[shortcut]

from=-3

activation=linear

[convolutional]

batch_normalize=1

filters=64

size=1

stride=1

pad=1

activation=leaky

[convolutional]

batch_normalize=1

filters=128

size=3

stride=1

pad=1

activation=leaky

[shortcut]

from=-3

activation=linear

Downsample

[convolutional]

batch_normalize=1

filters=256

size=3

stride=2

pad=1

activation=leaky

[convolutional]

batch_normalize=1

filters=128

size=1

stride=1

pad=1

activation=leaky

[convolutional]

batch_normalize=1

filters=256

size=3

stride=1

pad=1

activation=leaky

[shortcut]

from=-3

activation=linear

[convolutional]

batch_normalize=1

filters=128

size=1

stride=1

pad=1

activation=leaky

[convolutional]

batch_normalize=1

filters=256

size=3

stride=1

pad=1

activation=leaky

[shortcut]

from=-3

activation=linear

[convolutional]

batch_normalize=1

filters=128

size=1

stride=1

pad=1

activation=leaky

[convolutional]

batch_normalize=1

filters=256

size=3

stride=1

pad=1

activation=leaky

[shortcut]

from=-3

activation=linear

[convolutional]

batch_normalize=1

filters=128

size=1

stride=1

pad=1

activation=leaky

[convolutional]

batch_normalize=1

filters=256

size=3

stride=1

pad=1

activation=leaky

B. SCREENSHOTS

Figure 3. compaision of accuracies of different algorithmhs

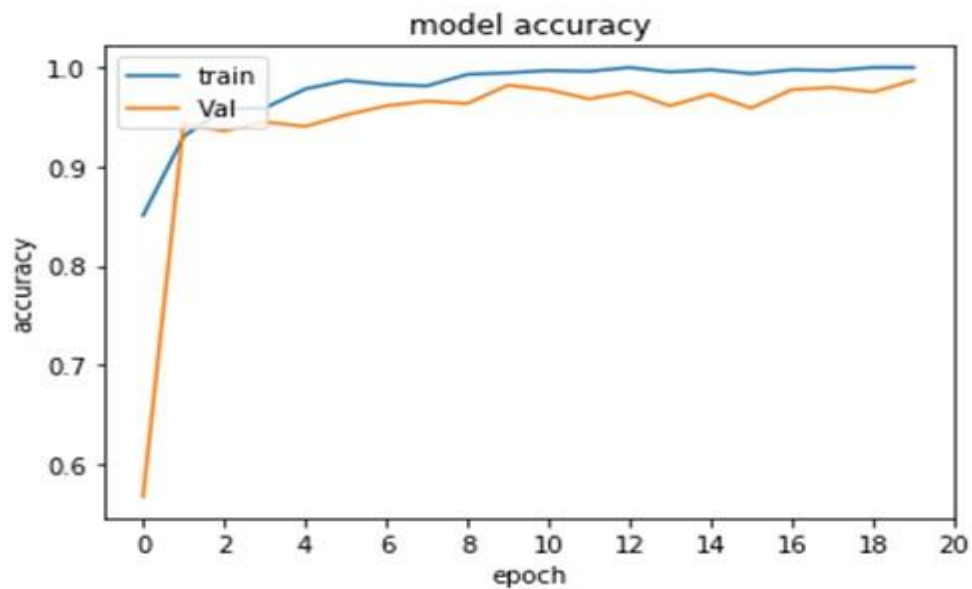
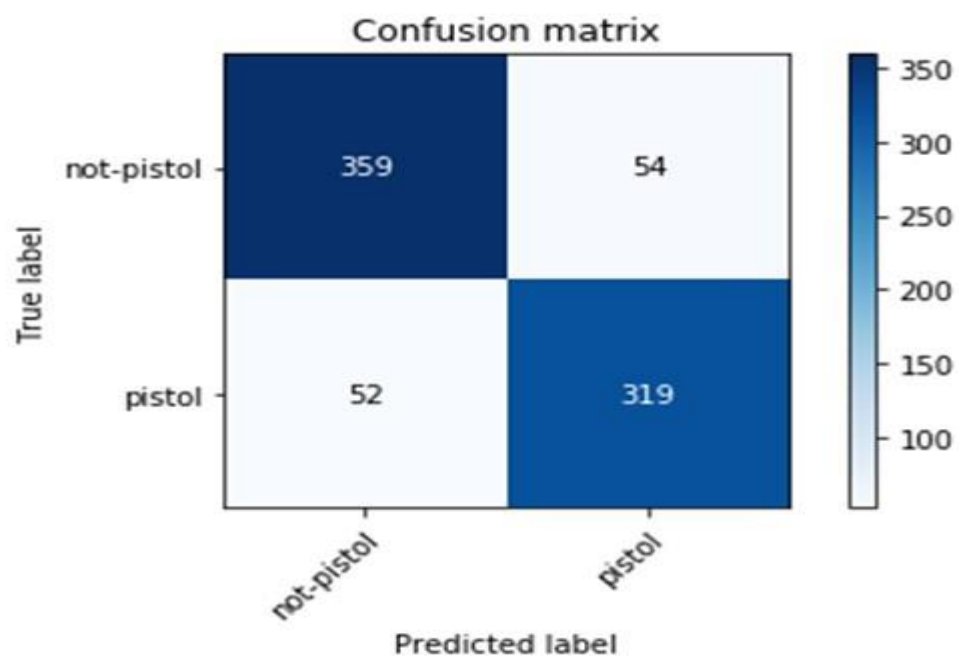
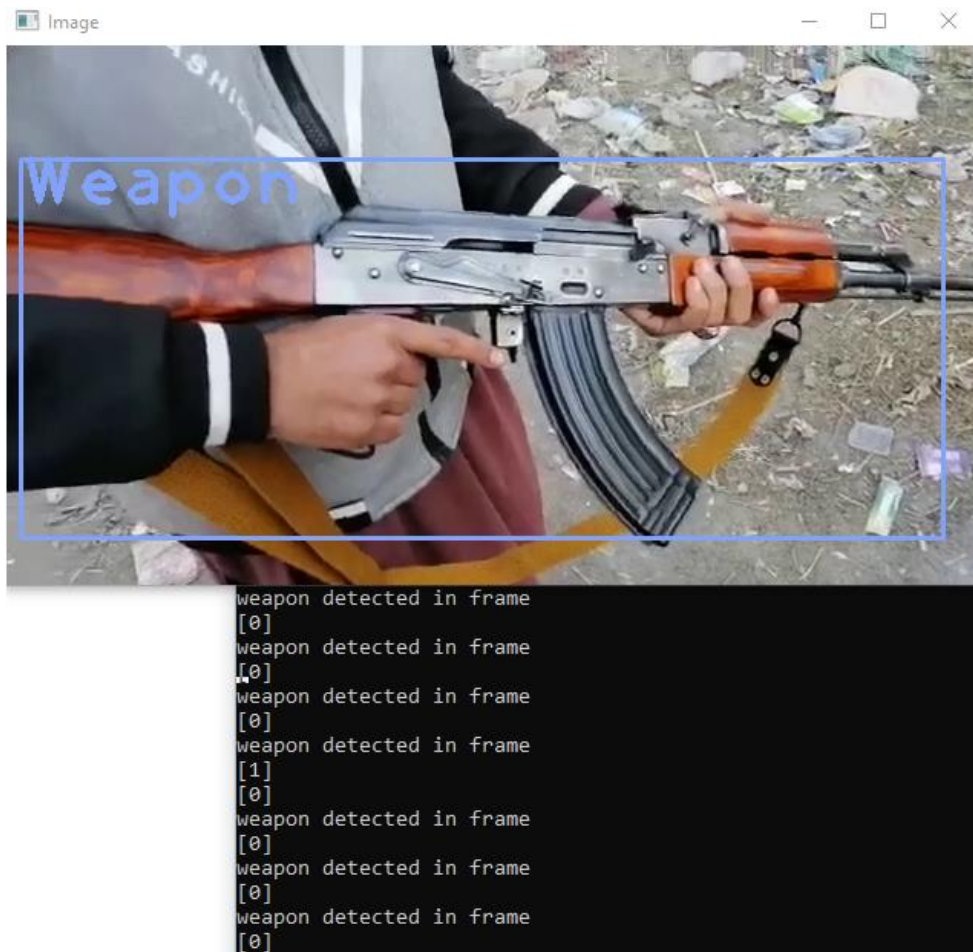


Figure 4. confusion matrix

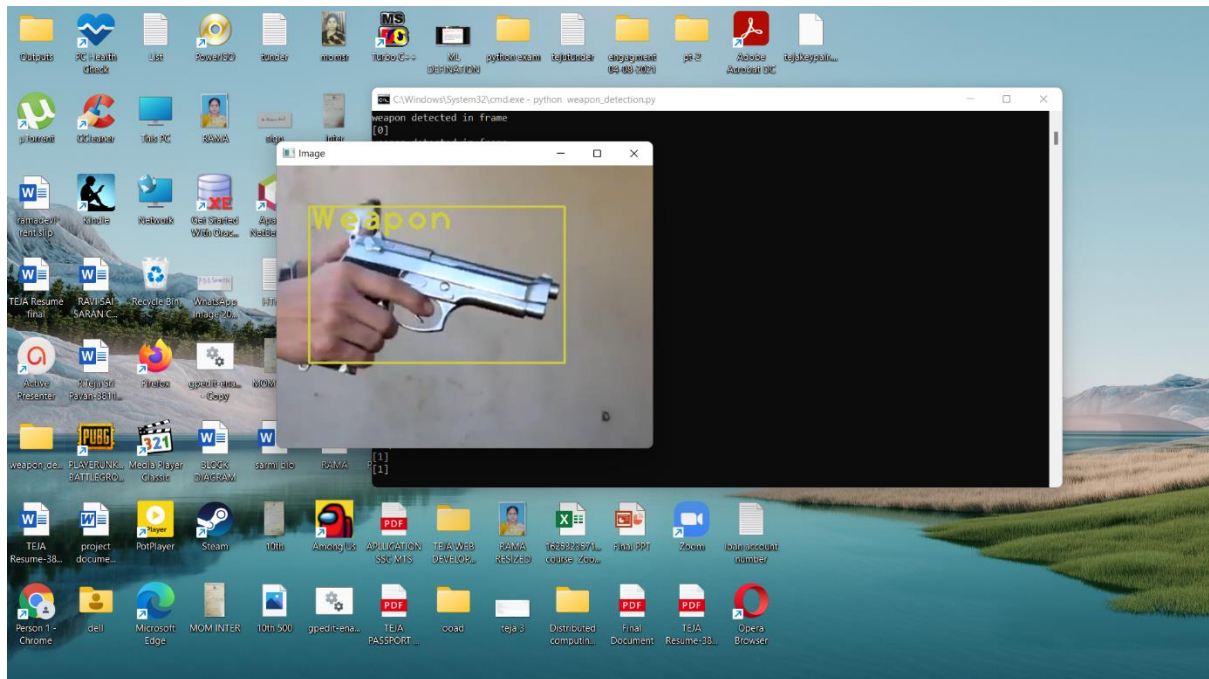


WEAPONS DETECTED IMAGES

A)



B)



NOVEL MACHINE LEARNING APPROACH FOR WEAPON DETECTION

¹Teja Sri Pavan P, ²Karthik, ³S Prayla Shyry, ⁴Y.Bevis
Jinila, ⁵ Ramya G Franklin ^{1,2}

Student, Dept of CSE, Sathyabama Institute of Science and Technology

^{3,4,5} Assoc.Profs, Dept of CSE, Sathyabama Institute of Science and Technology, praylashyry@gmail.com, ysbevish@gmail.com, ramyafranklin.cse@sathyabama.ac.in

ABSTRACT

Various firearm related wrongdoings have provoked state run administrations to search for ways of combatting psychological warfare. These elements unfavorably influence public security and compromise public request. In such manner, the battle against the scourge of firearm viciousness has turned into the subject of exploration. One method for lessening such viciousness is to keep away from distance and settle on the ideal choices temporarily. Picture control is a method for controlling individuals' conduct and things. The checking framework can be utilized as a legitimate security signal. Likewise, it is broadly utilized in dubious exercises. Smart picture global positioning frameworks (IVSSs) are the utilization of mechanized picture examination to work on the presentation of conventional frameworks. Profound learning (DL) is generally used to tackle existing specialized issues because of its quick turn of events. In this article, you want to know how to perceive a gun and a firearm in a video reconnaissance framework. The strategy shown doesn't need the incorporation of weapons in the weapon recognition framework. It utilizes DL to order and comprehend. The proposed strategy doesn't further develop the outcomes acquired utilizing Migration (TL). It utilizes two unique DL strategies, AlexNet and GoogLeNet. The review showed how various kinds of guns can be joined with various sorts of weapons. A review was led on the Internet Film Firearms Database (IMFDB). The outcomes show that the normal technique is more dependable and prevalent than others.

Keywords- *Object Detection, Weapon Detection.*

INTRODUCTION

The quantity of violations connected with the utilization of weapons during the savagery has risen pointedly all over the planet. For a country to create, it should screen laws and guidelines. Regardless of whether we need to draw in financial backers or put resources into the travel industry, we want harmony and security. In many areas of the planet, firearm wrongdoing is wild. This incorporates, specifically, nations approved to have guns. The world is an elite town, and what we say and compose influences individuals. In any case, as indicated by individuals, this is actually what's going on. Individuals are currently extremely discouraged and can't handle their annoyance, and despise discourse can prompt misery. It is feasible to indoctrinate an individual, and mental examination shows that assuming an individual is furnished today, he might black out and perpetrate a wrongdoing.

LITERATURE SURVEY

Weapon Detection in Real-Time CCTV Videos using Deep Learning

**Muhammad Tahir Bhatti¹, Muhammad Gufran Khan¹ (Senior Member, IEEE),
Masood Aslam² , Muhammad JunaidFiaz /2021**

Security and wellbeing is one of the main issues on the planet today. For a country to be financially solid, it should endeavor to guarantee the wellbeing and security of financial backers and sightseers. CCTV cameras are utilized for reconnaissance and burglary; however, these cameras require public examination and interest. A framework is expected to identify these criminal operations right away

Real Time Video Surveillance for Automated Weapon Detection
Bhagyalakshmi. P, Indhumathi. P, Lakshmi. R, Dr. Bhavadharini/2019

This work utilizes facial video reconnaissance by following and perceiving extraordinary articles dependent on certifiable picture handling strategies. The proposed project model has three handling modules, the principal handling module is for understanding anything utilizing Convolutional Neural Networks (CNN), and the subsequent module is for weapons advancement, control, and salvage classes utilizing the third handling technique.

Detection and Classification of Different Weapon Types Using Deep Learning
Volkan Kaya, Servet Tuncer and Ahmet Baran/2021

In this day and age of wrongdoing, selfchecking frameworks are a unique requirement for law authorization. In this review, another strategy for recognizing seven kinds of weapons was created utilizing inside and out research. This model offers another strategy for arranging weapons dependent on the VGGNet storehouse. The model distinguishes long weapons, guns, explosives, shotguns, blades, guns, and fixes. Tests were arranged utilizing the Keras library dependent on Tensor Flow.

Border Security using IoT
Pooja S N , Spurthi T M, Rashmi R K, Samreen Unnisa, Deepika J/2018

Psychological militants enter our lines unconsciously. It is unimaginable so that our fighters might be able to see the boundary consistently. Fundamental security is needed to have the option to promptly recognize fear-based oppressors at the boundary. In this article, we will give a robot that utilizes IR sensors to identify dangers, photo dangers with a PI camera and make an impression on the important specialists.

Automated Detection of Firearms and Knives in a CCTV Image
Grega, Michał, Andrzej Mاتیolański, Piotr Guzik, and Mikołaj Leszczuk

says that Security and wellbeing is one of the main issues on the planet today. For a country to be financially solid, it should endeavor to guarantee the wellbeing and security of financial backers and sightseers. CCTV cameras are utilized for reconnaissance and burglary; however, these cameras require public examination and interest. A framework is expected to identify these criminal operations right away

A Handheld Gun Detection using Faster R-CNN Deep Learning
Verma, Gyanendra K., and Anamika Dhillon

says that in this day and age of wrongdoing, self- checking frameworks are a unique requirement for law authorization. In this review, another strategy for recognizing seven kinds of weapons was created utilizing inside and out research. This model offers another strategy for arranging weapons dependent on the VGGNet storehouse. The model distinguishes long weapons, guns, explosives, shotguns, blades, guns, and fixes. Tests were arranged utilizing the Keras library dependent on Tensor Flow

EXISTING SYSTEM

Observation cameras have been introduced as of late, yet they are difficult to use for security purposes. One should forever be available to control the screen. The CCTV client should screen the screen for 10 to 25 hours. As the quantity of screens builds, the emphasis on the individual to control the screen and the clock diminishes forcefully. It is beyond the realm of possibilities to expect to control the screen to keep similar degree of consideration Consistently.

PROPOSED SYSTEM

In this day and age of wrongdoing, it is essential to know whether or not an individual is equipped, with the assistance of surveillance cameras. Information and ID of weapons is significant for counteraction before a wrongdoing happens, with the goal that the gatherings can make a move. Numerous criminal demonstrations are perpetrated the hard way or outfitted. Robbery or weapons are significant things utilized in different violations like burglary, poaching and terrorizing. It is vital to decide the kind of weapon that could be utilized to carry out a wrongdoing to decide whether there are indications of a wrongdoing in the surveillance camera film and to make the important move.

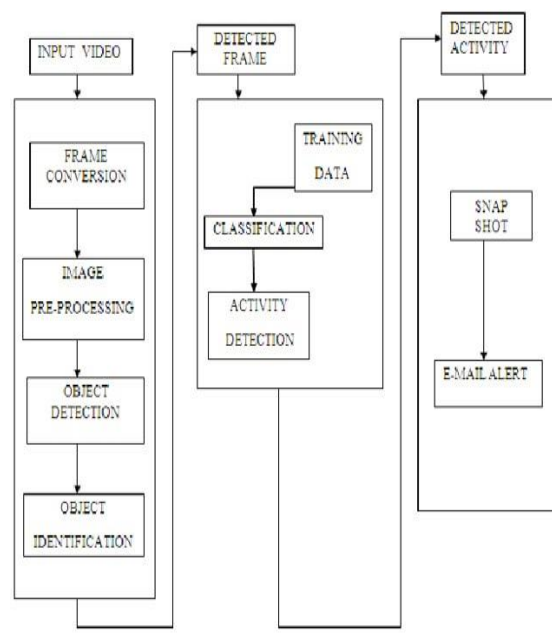
DISADVANTAGES OF EXISTING SYSTEM

In this type of system a person should continuously monitor the surveillance camera and anytime the crime can Happen. And also when the person who is monitoring the camera while he is watching through one camera he may miss the crime related scene. And crime may happen without his presence also. So the presence of the Security guard is must and should.

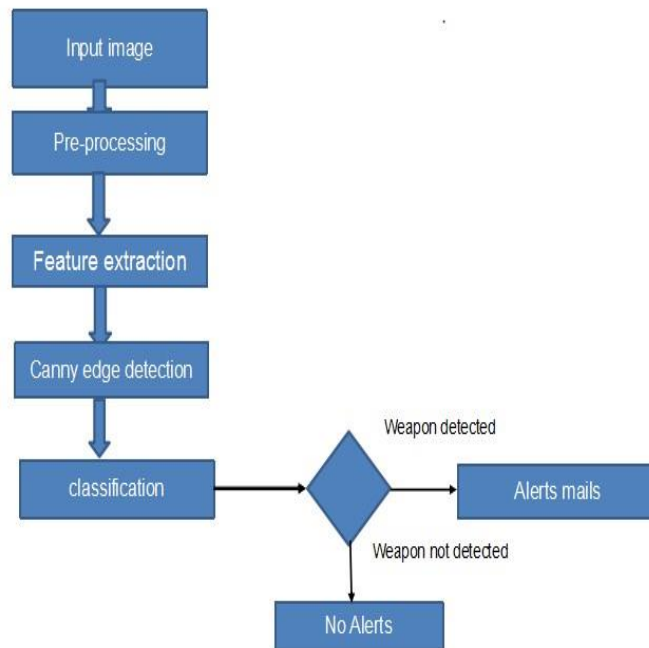
ADVANTAGES OF PROPOSED SYSTEM

In this System the need Security Guard is less when compared to the above mentioned Existing System. So when the Security Guard is busy with some other work and while he is not monitoring the surveillance, This System will send an email Alert so that the Guard can Understand and take the necessary actions. And he can stop the crime

ARCHITECTURE DIAGRAM



FLOW CHART



Pre-processing

The best video altering methods for multi-facet pipelines incorporate catch, change, and drawing with the best gear to work with the start, terminating, and capacity process. This incorporates channels and other mathematical pressure tasks called "prehandling". Picture handling can decrease the quantity of mistakes that can happen during handling, like shading rectification and clamor decrease. It is additionally critical to have the option to see very much delineated pictures that don't contain craftsmanship from the connection or wellspring of the film being changed over to video.

Object detection

PC intellectual innovation This sort of awareness can be utilized to deliberately work out, find, and control anything. PC proficiency is the advancement of PC innovation and pictures connected with PCs that work by distinguishing explicit articles in specific classifications of recordings and recordings (individuals, structures, vehicles, and so on) For MS COCO informational indexes, the most ideal way to realize anything better in 2021 dependent on the moderate proportion (MAP) is YOLOR (MAP 56.1). The calculation is very much constrained by YOLOv4 (MAP 55.4) and EfficientDet (MAP 55.1).

Object localizations

Altering implies knowing where at least one items are in the drawing and drawing a case of a similar size. Join these two capacities, change at least one component into pictures, and comprehend the components to be consolidated. The picture is an inversion of a regular CNN calculation. This calculation predicts a specific number of illustrations. For this situation, the calculation predicts a nonstop 4-digit rundown of x-facilitates, y-organizes, length, width, and so on, and attracts a case close to the object of interest. ID implies that once an item has been recognized, it is put in an appropriate box or, all the more definitively, in a square shape to recognize it from the article. Here are a portion of the ones I viewed as intriguing: The distinction between an article and an item is basic. Basically, a neighborhood object is expected to distinguish the principle (or generally apparent) object in a picture, and the item is an endeavor to characterize all articles and their limits.

Classification

The most common way of recognizing and ordering is to group as per the recently referenced classes or types.

On the off chance that both depend on the idea of the item, the request can be discretionarily drawn limits dependent on autonomous, imperceptible issues. Arranging a picture is to foresee the characterization of a similar thing in the picture. Altering implies knowing where at least one articles are in the drawing and drawing a crate of a similar size. Join these two capacities, change at least one component into pictures, and comprehend the components to be consolidated. Any picture contains valuable and futile data relying upon the interests of the clients. The drawing segment recognizes pictures in the locale, each with its own remarkable limits, and distinguishes regions that can be perceived for better execution, for example, grouping and getting objects.

KEY POINTS

The motivation behind the undertaking is to upgrade the gathering of crude pictures got by satellite, aeronautical cameras, cameras/sensors, or pictures taken in different exercises in day-to-day existence. Arranging is utilized to embed and describe a gathering of pixels or segments in pictures as per similar principles. The standards of requesting can be created utilizing at least one of the accompanying terms or conditions. Edge Identification and Image Processing Find the limits of items in pictures. It works by knowing how to switch out the lights.

Distinguish parts of picture partition and information extraction in regions, for example, picture handling, PC vision, and machine vision.

IMPLEMENTATION PROCESS

Support Vector Machines (SVMs) are a preparation framework that utilizes time to anticipate undeniable level line exercises, prepared via preparing improvement preparing calculations that carry out predisposition dependent on numerical information.

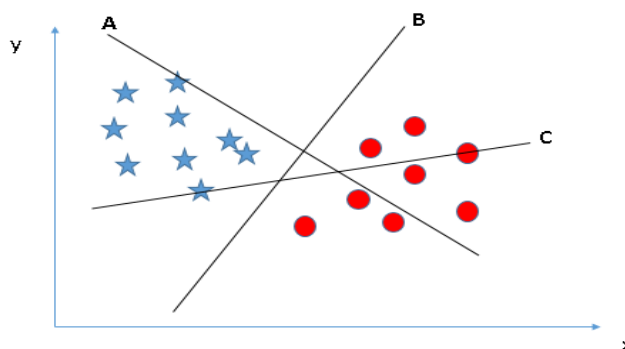
The objective of SVM is to track down the best hyper airplane to lessen the two levels. There might be various planes that partition the two classes, however we center around searching for an airplane like this to see the large contrast between classes. This implies that the hyper plane is chosen to build the separation from the hyper plane to the closest news source.

How does it work?

Above, we know about the most common way of isolating two-stage hyper-airplane. So, the principle question is "How might we decide the right hyperplane?" No concerns, it's not generally so hard as you might suspect!

Identify the right hyper-plane (Scenario-1):

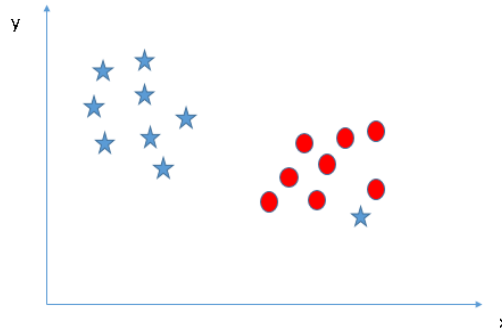
Here we have three planes (A, B, C). Presently look for the right hyper plane to zero in on the stars and wheels.



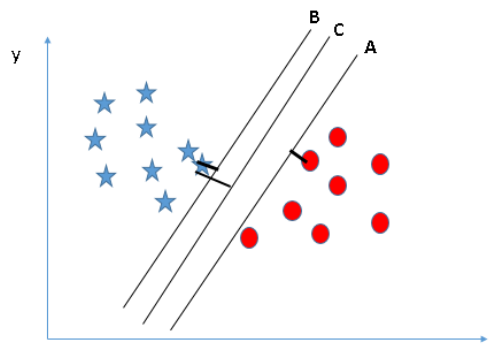
To know the hyper plane of the right, recall the guideline: "Pick a hyper plane that accurately isolates the two classes." Today, the "B" hyper plane functioned admirably.

Identify the right hyper-plane (Scenario-2):

Here we have three hyper planes (A, B, and C) that all separate between the subjects. So how would we characterize the hyperplane on the right?



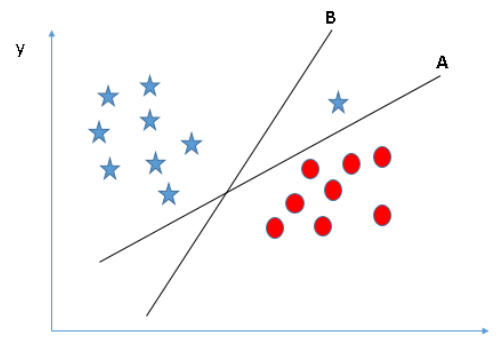
Diminishing the distance between vicinity (both in classification) and ultra-airplane data here will assist us with picking a hyperplane better. This distance is called Margin. How about we take a gander at the image underneath.



From a higher place, you can see that the hyperplane distance C is higher than A and B. So we call the hyper plane to the right C. One more motivation to pick a long reach hyper plane is power. Assuming we pick a hyper plane with a little distance, there is a high likelihood of lining.

Identify the right hyper-plane (Scenario-3):

Tip: Use the principles laid out in the past segment to distinguish the hyperplane on the right.



Some of you might pick hyperplane B since it has a more extended territory than A. In any case, as should be visible, the SVM chooses a hyperplane that groups the arrangement a long time prior to looking at the remove distance. Here, the blunder is arranged in the hyper plane B, and A is put every which way. Accordingly, the right half of the hyper plane is A.

Decision Trees for Classification: A Machine Learning Algorithm:

The decision tree is a sort of control machine where data is continually conveyed by specific standards (this implies information and result are remembered for the activity). A tree can be characterized by two elements: navigation and insidiousness. Leaves are an ultimate choice or arrangement. Choices are made based on data sharing. An illustration of a tree arrangement can be clarified utilizing a double tree above. Assume you need to

anticipate an individual's life as indicated by data like age, dietary patterns, and proactive tasks. Also, the leaves are the "right" or "wrong" arrangement. For this situation, it was a two-section issue

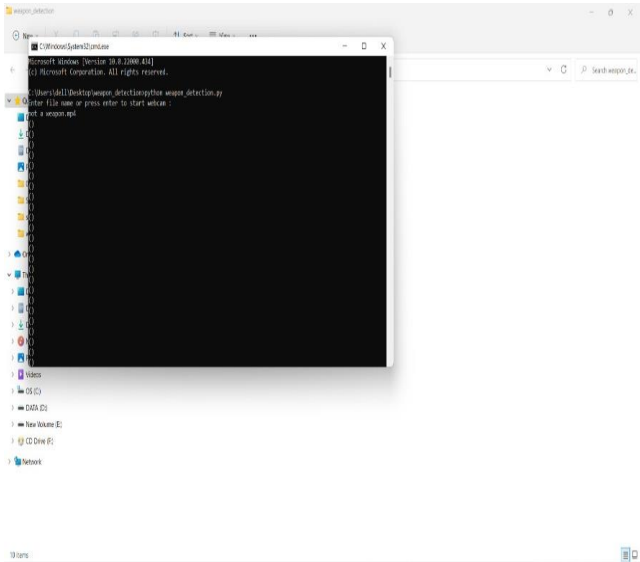
CONCLUSION AND FUTURE WORK

In this review, the advanced YOLO V3 recognition framework was executed and prepared on informational index to recognize weapons. To decide the wellbeing of the weapon, we give a model that makes the machine or robot apparent, and can tell the commandant when the firearm or weapon is plainly noticeable as an afterthought. + e Test results show that the planned YOLO V3 has preferable execution over the YOLO V2 and is less expensive in numbers. + Modern observing abilities should be fortified and created to help worker execution. The framework will successfully supplant existing foundation with minimal expense stockpiling, video foundation, and further developed picture handling. At long last, on the robot, the framework control framework will totally supplant the current control framework with a less expensive PC, video foundation.

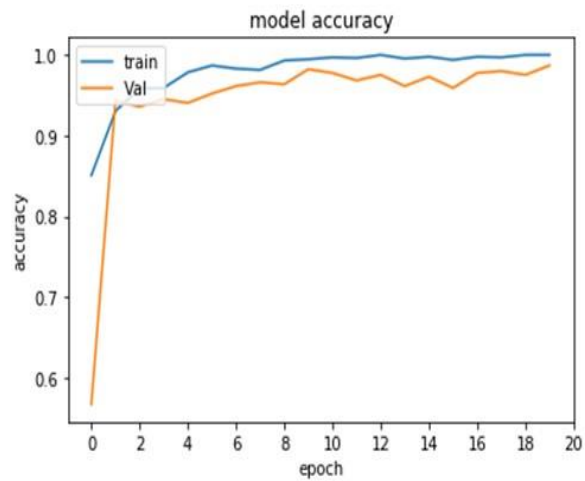
OUTPUT SCREENSHOTS



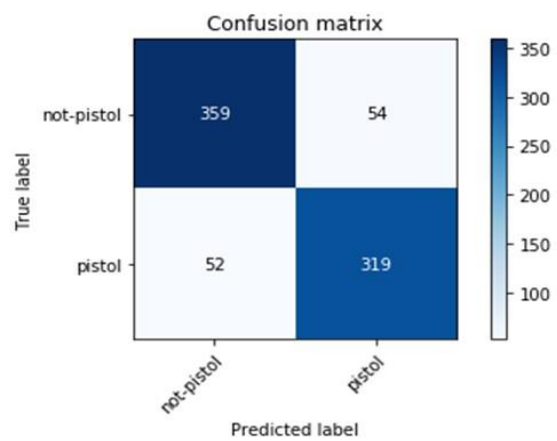
Fig(a)



Fig(b)



Fig(c)



Fig(d)

Figures

Fig(a) shows that weapon is detected in the frame, which is the necessary solution for our Problem Fig(b) shows that weapon is not detected in the frame i.e there is no weapon present in the frame Fig(c) is the model accuracy using SVM and Fig(d) is the confusion matrix for detected weapon and not detected weapon.

REFERENCES:

- 1.Grega, Michał et al. "Automated detection of firearms and knives in a CCTV image." .
- 2.Verma, Gyanendra et al(2017) invented an algorithm."A Handheld Gun Detection using Faster R-CNN Deep Learning." In Proceedings of the 7th International Conference on Computer and Communication Technology, pp. 84-88. ACM.
- 3.Redmon et.al(2018) Invented an algorithm. "Yolov3: An incremental improvement."
- 4.Ren et al (2015) Invented an algorithm "Faster r-cnn: Towards real-time object detection with region proposal networks." In Advances in neural information processing systems.
- 5.Muhammad Tahir Bhatti et al(2018) Invented an algorithm." Weapon Detection in Real-Time CCTV Videos using Deep Learning"
- 6.Pooja S N et al(2018) Invented an algorithm, "Border Security using IoT "
- 7.Bhagyalakshmi. P et al(2019) Invented an algorithm." Real Time Video Surveillance for Automated Weapon Detection"
- 8.Volkan Kaya et al(2021) Invented an algorithm" Detection and Classification of Different Weapon Types Using Deep Learning"

6%

SIMILARITY INDEX

5%

INTERNET SOURCES

4%

PUBLICATIONS

3%

STUDENT PAPERS

PRIMARY SOURCES

NOVEL OBJECT DETECTION METHOD ...

1 www.coursehero.com

Internet Source

1%

2 Internet viso.ai Source

1%

3 Internet www.mdpi.com Source

1%

4 Student Paper Submitted to University of Teesside

1%

5 Internet www.ijert.org Source

1%

6 Internet **isindexing.com** Source

1%



Internet **www.igt Source i-global.co m**

<1%

Exclude quotes

n Exclude bibliography On

Exclude matches

☐ Off