

# Data Acquisition and Crime Anomaly Detection



## **Authors**

GC Abdulsamad Burki  
GC M. Farrukh Hussain Jamali  
GC M. Hammad Ajmal  
GC Fahad Imran

## **Supervisor**

Dr. Ata-ur-Rehman

Submitted to the faculty of Department of Electrical Engineering,  
Military College of Signals, National University of Sciences and Technology, in  
partial fulfillment for the requirements of B.E Degree in Electrical Engineering  
(JUNE), 2023

## **CERTIFICATE OF CORRECTIONS & APPROVAL**

Certified that work contained in this thesis titled “**Data Acquisition and Crime Anomaly Detection**”, carried out by GC Abdulsamad Burki, GC Farrukh Jamali, GC Hammad Ajmal and GC Fahad Imran under the supervision of Dr. Ata-ur-Rehman for partial fulfillment of Degree of Bachelor of Electrical Engineering, in Military College of Signals, National University of Sciences and Technology, Islamabad during the academic year 2019-2023 is correct and approved. The material that has been used from other sources it has been properly acknowledged / referred.

**Approved by**

**Supervisor**

Dr Ata ur Rehman

Military College of Signals

Date: 01-05-2023

## **DECLARATION OF ORIGINALITY**

**No portion of work presented in this thesis has been submitted in support of another award or qualification in either this institute or anywhere else.**

## **Acknowledgements**

We are thankful to our Creator ALLAH (SWT) to have guided us throughout this work at every step and for every new thought which, He setup in our minds to improve it. Indeed, we could have done nothing without His priceless help and guidance. Whosoever helped us throughout the course of our thesis, whether our parents or any other individual was His will, so indeed none be worthy of praise but Him.

We would like to express special thanks to our supervisor **Dr. Ata-Ur-Rehman** for his help throughout our thesis and for teaching us the course “Artificial Intelligence”. We can safely say that we have not learned any other engineering subject in such a pragmatic way than the one which he has taught.

## Plagiarism Certificate (Turnitin Report)

This thesis has \_\_\_\_ similarity index. Turnitin report endorsed by Supervisor is attached.

---

Abdul Samad Burki  
NUST Serial no 325595

---

M. Farrukh Hussain Jamali  
NUST Serial no 325031

---

M Hammad Ajmal  
NUST Serial no 325594

---

Fahad Imran  
NUST Serial no 325042

---

Signature of Supervisor

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

## **Abstract**

Data acquisition is an essential component of any data analysis project, including crime anomaly detection. In this thesis, we explore the process of acquiring and preparing data for crime anomaly detection, as well as the challenges and best practices involved in this process.

We begin by discussing the different sources of data that can be used for crime anomaly detection, including crime reports, social media posts. We then examine the process of cleaning and pre-processing this data, which involves removing duplicates, correcting errors, and standardizing formats.

Next, we investigate various techniques for detecting crime anomalies in the data, such as statistical models, clustering algorithms, and machine learning algorithms. We also examine the challenges and limitations of these techniques, including issues related to data quality, computational complexity, and interpretability.

Finally, we present a case study of crime anomaly detection using real-world data from a major city. We describe the data acquisition and preparation process, as well as the anomaly detection techniques used to identify unusual patterns of criminal activity. We also discuss the implications of these findings for law enforcement and public safety.

Overall, this thesis provides a comprehensive overview of data acquisition and crime anomaly detection, highlighting the importance of careful data preparation and the need for advanced analytical techniques to detect and prevent crime.

## Table of Contents

|   |    |
|---|----|
| CERTIFICATE OF CORRECTIONS & APPROVAL .....                                   | 2  |
| Abstract .....  | 7  |
| CHAPTER 1: BACKGROUND .....   | 1  |
| 1.1 Background, Scope and Motivation .....                                    | 2  |
| CHAPTER 2: INTRODUCTION .....   | 4  |
| 2.1 Dataset .....   | 4  |
| 2.2 Dataset Collection/ Acquisition .....                                     | 6  |
| 2.3 Pre-processing .....  | 7  |
| 1. Conversion of all other annotation formats to yolo supported fmt .txt..... | 8  |
| 2. Geometric Transformations. ....  | 8  |
| 3. Tiled images of large resolution images to smaller tiles i.e 640*640.....  | 8  |
| 2.4 Noise Removal .....   | 8  |
| 2.5 Sharpness.....  | 9  |
| 2.6 Removal of Outliers .....   | 10 |
| 2.7 Labelling of dataset .....  | 10 |
| 2.8 What is a Machine Learning Model? .....                                   | 15 |
| CHAPTER 3: DATASET COLLECTION .....   | 16 |
| 3.1 The Process of Dataset Collection .....                                   | 16 |
| CHAPTER 4: TRAINING A PROTOTYPE MODEL.....                                    | 20 |
| 4.1 Data Collection .....   | 20 |
| 4.2 Model training with time comparison .....                                 | 20 |
| 4.3 The accuracy and results .....  | 21 |
| 4.4 The problem Encountered. ....   | 22 |
| CHAPTER 5: TRAINING THE ACTUAL MODEL.....                                     | 23 |
| 5.1 Significance of epochs in training the model consecutively.....           | 23 |



|  |   |                                     |
|--|---|-------------------------------------|
| 5.2  | Steps followed for training the model.....        | 24                                  |
|  | Results of Training.....                          | 28                                  |
|  | Losses and Progress .....                         | 29                                  |
|  | Testing the Fire Class:.....                      | 31                                  |
|  | Testing the Knife Class: .....                    | <b>Error! Bookmark not defined.</b> |
|  | Testing the Rifle Class: .....                    | 32                                  |
|  | Testing the HandGun Class: .....                  | <b>Error! Bookmark not defined.</b> |
| CHAPTER 6: Integrating the Arduino with Application..... |   | 33                                  |
| 6.1  | Establishing the connection .....                 | 33                                  |
| 6.1.1  | Python Code for Establishing the Connection ..... | 34                                  |
| 6.1.1.1  | Python Code Used in the App.....                  | 34                                  |
| 6.2  | Arduino Sketch .....                              | 36                                  |
| 6.3  | Arduino and Buzzer Schematic.....                 | 37                                  |
| CHAPTER 7: DEVELOPING THE USER INTERFACE.....            |   | 38                                  |
| 7.1  | Programming/Markup Languages.....                 | 38                                  |
| 7.2  | Framework .....                                   | 38                                  |
| 7.3  | Libraries .....                                   | 38                                  |
| 7.4  | Tools.....  | 38                                  |
| 7.5  | The Significance of Using Flask .....             | 39                                  |
| 7.6  | Architecture of Flask.....                        | 41                                  |
| 7.7  | Setting Up the Environment.....                   | 41                                  |
| 7.8  | Setting up the Routes .....                       | 43                                  |
| 7.9  | Creating Views.....                               | 45                                  |
| 7.10   | Using the Interface.....                          | 46                                  |

## **CHAPTER 1: BACKGROUND**

Data has exploded in recent years due to the rapid development of digital technologies, opening new potential for a variety of industries, including law enforcement. Law enforcement organizations now have access to a variety of data that may be utilized to spot patterns, trends, and anomalies in crime because to the availability of digital data from numerous sources, including social media, crime reports, and surveillance footage.

The process of gathering, processing, and analyzing data from multiple sources is referred to as data acquisition. When used in relation to law enforcement, data acquisition can entail compiling data from many platforms, such as criminal justice databases, social media sites, and surveillance cameras. Law enforcement authorities can acquire a more complete understanding of criminal activities in their jurisdictions by integrating these many information sources.

The process of discovering odd or unexpected behavior or events that vary from regular patterns is known as anomaly detection, on the other hand. Anomaly detection can assist law enforcement organizations in identifying potential dangers, such as suspicious behavior or unusual activity in specific locations, in the context of preventing crime. Law enforcement organizations can better understand crime patterns and create more specialized prevention tactics by employing data analytics techniques to spot abnormalities in crime data.

Recent years have seen a rise in the use of data collection and anomaly detection in crime prevention, with many law enforcement organizations employing big data analytics techniques to improve their crime prevention efforts. However, using such tools also prompts worries about data security, privacy, and potential biases in the data.

In general, data collection and criminal anomaly detection offer new chances for law enforcement organizations to enhance their crime prevention methods using big data analytics. To make sure that these tools are used responsibly, ethically, and without having any unexpected repercussions, more research and development is necessary.

## **1.1 Background, Scope and Motivation**

### Scope:

This thesis looks into how criminal anomaly detection and data collection could be used to enhance crime prevention tactics. This thesis will specifically examine the application of big data analytics in the context of law enforcement, including the data sources, tools and methodologies for data collecting and analysis, as well as the difficulties and constraints of using big data in crime prevention. The use of anomaly detection in identifying crime patterns and trends, as well as the potential advantages and disadvantages of adopting this strategy in crime prevention measures, will also be covered in this thesis.

### Motivation:

The motivation for this thesis stems from the need for more effective crime prevention strategies in the face of rising crime rates and the increasing availability of digital data. By leveraging the power of big data analytics and anomaly detection, law enforcement agencies can gain valuable insights into crime patterns and trends and develop more targeted and proactive crime prevention strategies. This thesis seeks to contribute to the understanding of how data acquisition and anomaly detection can be used to enhance crime prevention efforts and promote public safety, while also addressing concerns about privacy, data security, and potential biases in the data. Ultimately, the goal of this thesis is to provide

insights and recommendations for law enforcement agencies, policymakers, and researchers to develop and implement effective crime prevention strategies that leverage the power of big data.

## CHAPTER 2: INTRODUCTION

### 2.1 Dataset

Before we dive into the concept dataset we must go through the concepts about data

#### 2.1.1 Data

Machine learning techniques study instances to improve. It's crucial to understand the vocabulary used to describe data as well as the input data.

When thinking of data, rows and columns like those found in an Excel spreadsheet or database table immediately come to mind. This is a typical data format and is used extensively in the machine learning community. Additionally, data can be found in the form :

- Images
- Videos
- Text

For the purpose of this project data of knives was taken in the form of images.

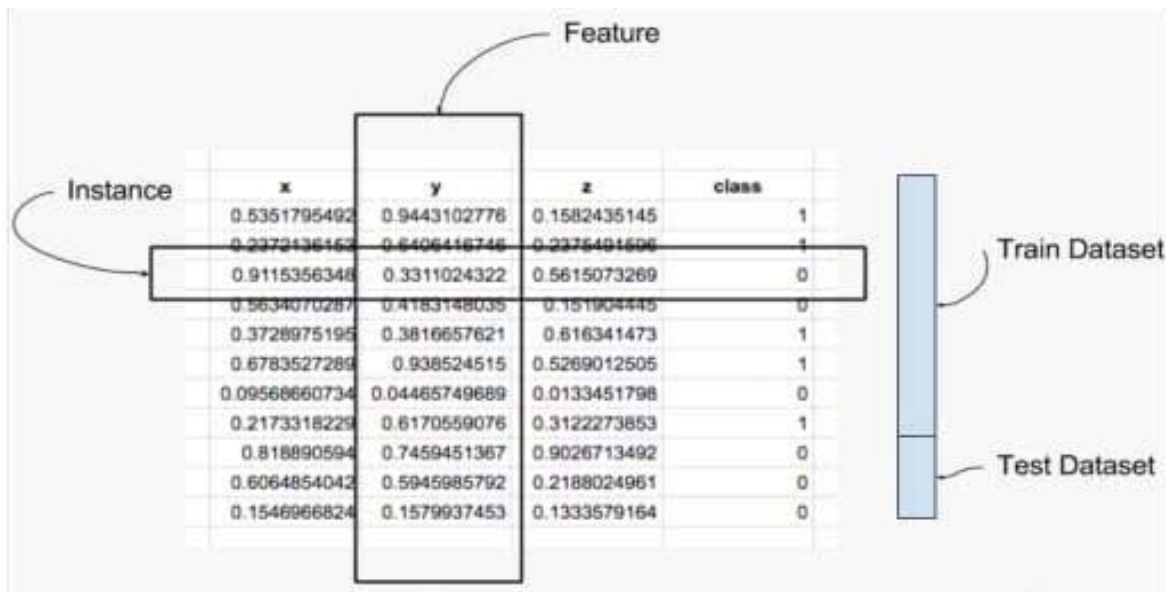


Figure 2.1: the basic demonstration of dataset

## 2.1.2 Datasets

A collection of instances is a dataset and when working with machine learning models we typically need a few datasets at different stages of the process.

### 2.1.2.1 Training Dataset

A dataset that we feed into our machine learning algorithm to train our model.

### 2.1.2.2 Validation Dataset

A sample of data withheld from the training of your model is referred to as a validation dataset and is used to measure the model's skill while adjusting its hyperparameters. It is utilised to provide a fair

assessment of the final tuned model's skill when contrasting or choosing between final models. The training dataset only contains a small piece of this.

### **2.1.2.3 Testing Dataset**

A dataset that we use to validate the accuracy of our model but is not used to train the model. This dataset is held back from the training of the model and is never fed to the model during training.

## **2.2 Dataset Collection/ Acquisition**

Data are bare facts. It serves as an example of both human and automated observations of the world. Data can be used to represent anything at all. Data is all around us when we enter a virtual environment since it is the underlying component of everything we perceive. The data we want to collect is entirely dependent on the issue we are seeking to resolve. The coaching knowledge cannot be surpassed by a machine learning algorithm. As a result, attention must be paid to gathering the best possible image data for the machine learning models. It is possible to train computer vision models using one of the largest picture datasets and deep learning image data thanks to a vast image classification dataset. The most crucial goal is to provide the greatest image data gathering and classification dataset for every computer vision project. The range of services for collecting and annotating image data for use in machine learning and deep learning applications is extensive. The simplest way to gather photographs for the model's training is to use a snipping tool to crop pictures of guns, explosives, and car accidents such that there is too much data around it that could contaminate the model's results during training. The excess data not required for training the model are known as 'Outliers' and are responsible for polluting the training dataset. An example of outliers is shown in figure 2.2 below.



Figure 2.2: Image with outliers



2.3: An image without outliers.

## 2.3 Pre-processing

Data pre-processing, also known as data munging or data cleaning, is an important step for machine learning engineers, and most ML engineers devote a significant amount of time to it before developing the model. Software like Photoshop and other similar programmed can be used for data pre-processing. Outlier detection, missing value treatments, and removing undesired or noisy data are a few examples of data pre-processing.

Like that, "image pre-processing" refers to actions on images that are performed at the most basic level of abstraction. If entropy is a measure of information, then these actions diminish rather than increase the information content of the image. Pre-processing aims to improve the picture data by reducing unwanted distortions or enhancing specific visual properties that are important for subsequent processing and analysis tasks.

There are 4 different types of Image Pre-Processing techniques, and they are listed below.

1. Pixel brightness transformations/ Brightness corrections



2. Geometric Transformations
3. Image Filtering and Segmentation
4. Fourier transform and Image restoration.

In our case we are more focused on:

1. Conversion of all other annotation formats to yolo supported fmt .txt.
2. Geometric Transformations.
3. Tiled images of large resolution images to smaller tiles i.e 640\*640.

## **2.4 Noise Removal**

Digital photographs are susceptible to several kinds of noise. Errors in the picture capture process lead to pixel values that do not accurately reflect the true intensities of the actual scene, which is what is known as noise. Depending on how an image is produced, noise can be added in several different ways. For instance: If a picture is captured from a satellite, it may be blurred as a result of clouds and unfavorable weather.

Noise comes from the film grain if the image was scanned from a film photograph. In addition, the film's condition or the scanner itself may have caused noise.

If the picture is captured, directly in a digital format, the mechanism for gathering the data can introduce noise.

### **2.4.1 Remove Noise by Linear Filtering**

You can use linear filtering to remove certain types of noise. Certain filters, such as averaging or Gaussian filters, are appropriate for this purpose. For example, an averaging filter is useful for removing grain noise from a photograph. Because each pixel gets set to the average of the pixels in its neighborhood, local variations caused by grain are reduced.

## **2.5 Sharpness**

Resolution and acutance are two variables that together make up sharpness. Resolution is uncomplicated and objective. It simply refers to the image file's size in pixels. The more pixels an image has, the better its resolution, and the sharper it can be, all other things being equal. Acutance is slightly more challenging. It is a purely arbitrary measurement of edge contrast. The acutance has no unit.

How clearly defined the details in a picture are, especially the little elements, determines how sharp the image is.

Therefore, sharpening is a method for making a picture appear to be sharper. Acutance must be increased in order to increase perceived sharpness. Edge contrast must be added if you want your image to appear sharper.

There are three main reasons to sharpen your image: to overcome blurring introduced by camera equipment, to draw attention to certain areas and to increase legibility.

## 2.6 Removal of Outliers

Finding outliers is a difficult undertaking. By looking at uncertainty measurements, outliers can be found. However, there are various kinds of outliers. An outlier, for instance, could be a sample that deviates from the norm. For instance, if we want to distinguish between weapons, accidents, and explosives in the dataset but enter the data with people and cars nearby the weapons, accidents, and explosives, we may have "outliers" because the class estimate is unclear.

## 2.7 Labelling of dataset

The model being used in the project is an object detection model from Yolo family (Yolo v5s). It is supervised machine learning. So, data had to be labelled before training the model on the data.

There are several software and tools available for labelling images and drawing bounding boxes around the object of interest in an image.

Some of the famous annotation tools are as follows:

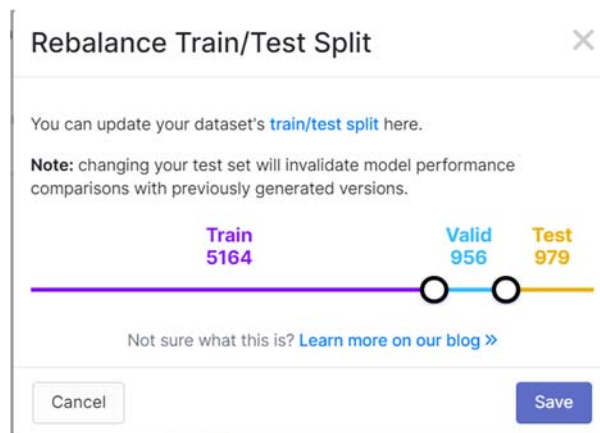
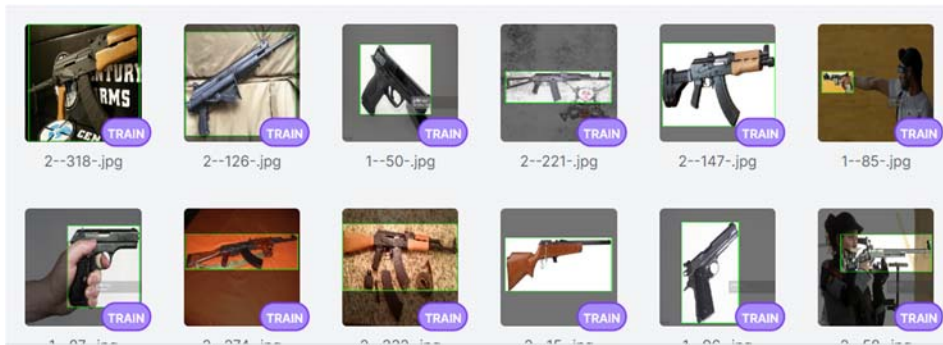
- ★ Roboflow
- ★ VGG Image Annotation Tool
- ★ Supervise.ly
- ★ Labelbox
- ★ Visual Object Tagging Tool (VoTT)
- ★ LabelImg

## 2.7.1 Why Roboflow?

Roboflow was chosen as the annotation software. The main reasons for its selection are as follows:

- ★ Simple interface
- ★ Cloud Storage and Cloud computation
- ★ Labelled data is available in multiple formats (which can be downloaded, or a link is generated to use the prepared dataset in any online platform for use)

Figure 2.4:



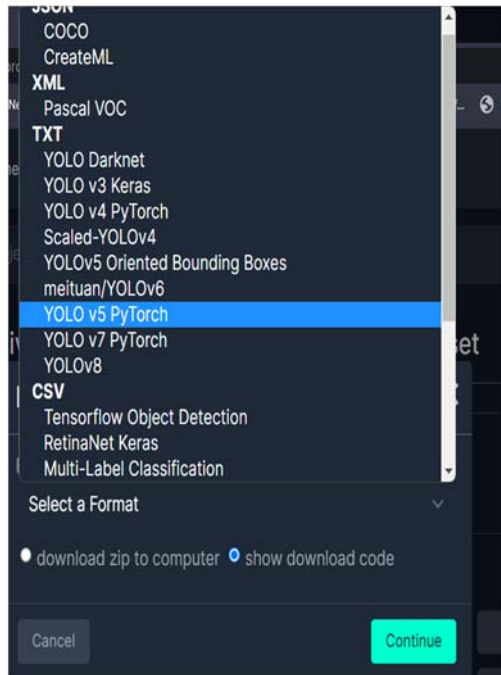


Figure 2.5: Selecting the format of data.



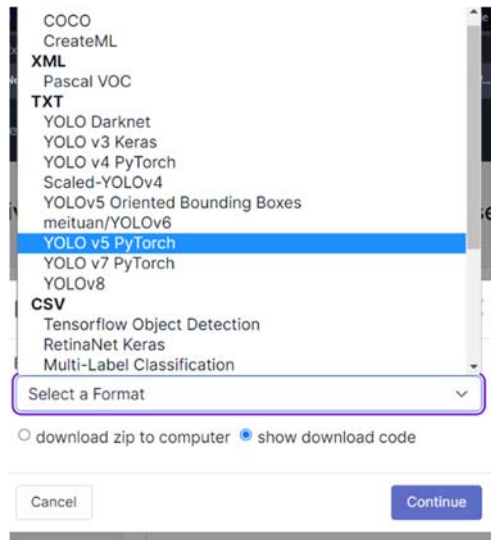


Figure 2.7: Exporting the dataset.

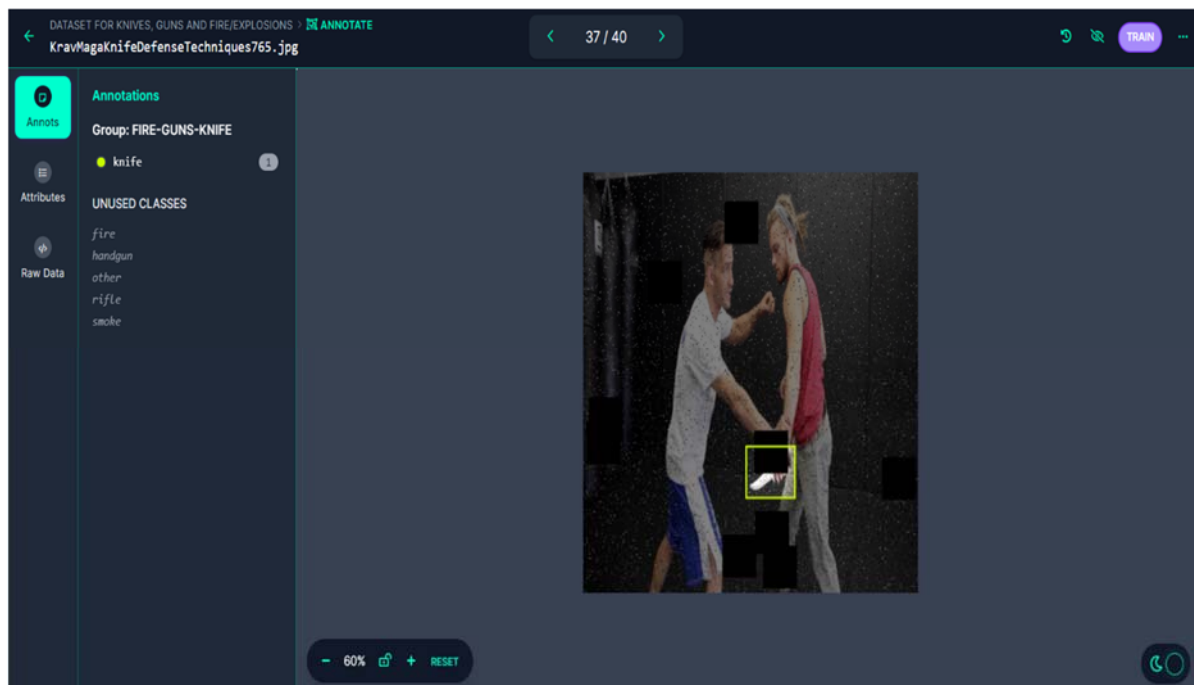


Figure 2.8: The dataset obtained.

## 2.8 What is a Machine Learning Model?

In essence, a machine learning model is a file that contains a machine learning algorithm (in our example, a convolutional neural network) that has been taught to recognize patterns. This machine learning algorithm may analyze and learn from a dataset, which is a specified set of data. The machine learning algorithm can be referred to as a machine learning model once this learning process is finished. It can now be used to analyze, anticipate, and pinpoint well-known tendencies in previously unknown data. For instance, the YOLO V5 model used in this experiment can, after training, identify the objects in unobserved data.



Figure 2.9: Training and Evaluation



## CHAPTER 3: DATASET COLLECTION

One of the most crucial first steps in training a machine learning (ML) algorithm is the collecting of datasets. A dataset can generally be defined as a collection of any data (i.e., photographs, figures, statistics, etc.) that demonstrates a similar pattern and can be collectively classified under a given category, such as items like knives, explosives, guns in a specific location, etc. Datasets can be viewed as the building blocks on which the ML algorithm learns to recognize and recognizes the relevant items, particularly in the field of object identification.

### 3.1 The Process of Dataset Collection

For the purpose of this project, we prepared approx 8000 different images out of which 5164 images were reserved for training purpose 956 image for validating purpose and 979 images for test purpose. We have gathered different data sets from open sources such as Kaggle and Google Images. The steps followed in collection of each set of data are described as follows.

To get a dataset from KAGGLE we followed the following steps:

1. Go to the website([www.kaggle.com](http://www.kaggle.com)) and create an account.
2. Navigate to the “Dataset” section of the website.
3. Browse the available datasets or search for a specific dataset with a keyword.
4. Click on the desired dataset to view more information about it, including its description, format size and any relevant tags.
5. Click the Downloads button to download the dataset.

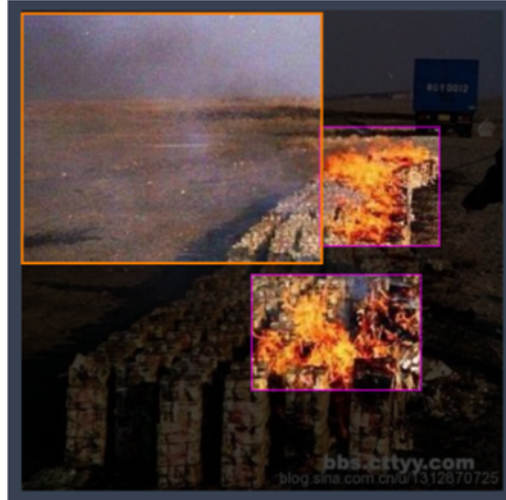


Figure 3.4 Not an orthogonal view



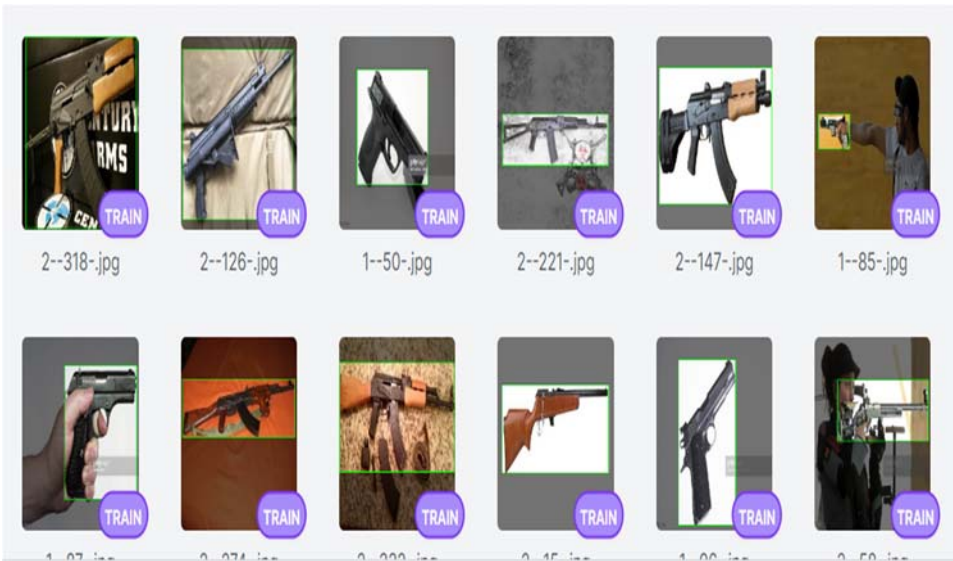
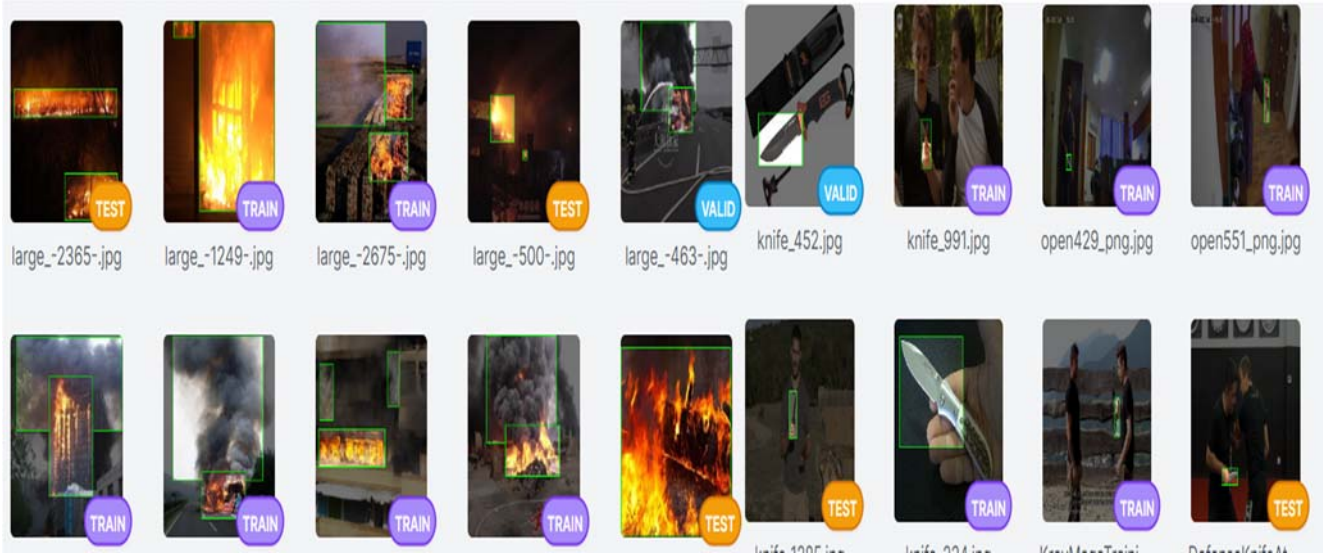
Figure 3.5 An Orthogonal view.

- I. In the above figure the angle at which the fire is observed is of essence i.e., approx. 90 degrees, the altitude however, may not be the same as it will be according to your desire. As soon as you are able to see the object fairly orthogonally, you are ready to crop the image.
- II. For the purpose of cropping, we use the snipping tool. You do not need to crop a big chunk of the image; this would only be increasing the size of your data unnecessarily. Only crop a decent chunk as in figure

3.5 or even smaller, the only requirement is that the whole object should be clearly visible, and no outliers be present. Repeat this step and the relevant previous ones for collecting the number of images you have decided to work with.

III.

If you intend to define classes while labeling the objects for e.g., Rifles, pistols, fire and smoke etc. Then it is recommended that you save the images of these objects in separate folders respectively so that the process of labelling becomes easy.



Some pictures of our dataset

## CHAPTER 4: TRAINING A PROTOTYPE MODEL

To test a concept or procedure, we created an early sample, model of a product. The key advantage of prototyping is that it leads to a quicker and more efficient design cycle.

Because prototypes make it possible for us to test our design in a "real-world" setting, they also make it simpler to see potential issues and avoid making costly mistakes later on, i.e., to see how things operate, become familiar with the procedure, etc.

### 4.1 Data Collection

In order to train the model before deploying our prototype, we collected roughly 5000 photographs. Google photos and Kaggle were used to gather the photos. Annotating the dataset was done once the first phase was finished. Image annotation is the process of classifying or labelling an image using text, annotation tools, or both to display the data aspects you want your model to automatically recognise. It is a key component of machine learning and deep learning. The previous chapter provided a detailed explanation of the annotation idea. Roboflow, a platform that offers the necessary tools for this activity, is where the annotation was completed.

### 4.2 Model training with time comparison

The training was completed using a Roboflow 2.0 Object Detection Model that was available online free of cost. The time required for processing was the biggest issue we ran into when training the model. It took around **14 hours** to train the model using **5164 photos**. The number of epochs was increased for training, which is why it took so long. The goal was to make the model more effective. In the following chapter, the model-training procedure is covered in more detail.

### 4.3 The accuracy and results

After providing the annotated dataset to the Roboflow 2.0 model on **Roboflow**, the time taken for training was **14 hrs with accuracy** of 78%.



Figure 4.1: Confidence level of the prototype.



Figure 4.2: Knife detection

#### **4.4 The problem Encountered.**

We had trouble because we couldn't collect a lot of data because of numerous restrictions. Due to the wide range of weaponry, we were unable to collect a large dataset on them. Despite the fact that we obtained a sizable amount of data on flames and knives. Secondly this model had some restrictions we weren't allowed to deploy it on our any application. We've to buy the premium version in order to train the model for more than 2 times and to deploy it to our machines without any hassle.

The problem is solved by:

- Increasing the size of our data set
- Preprocessing the images with different augmentations.
- Training on multiresolution image

## CHAPTER 5: TRAINING THE ACTUAL MODEL

### 5.1 Significance of epochs in training the model consecutively.

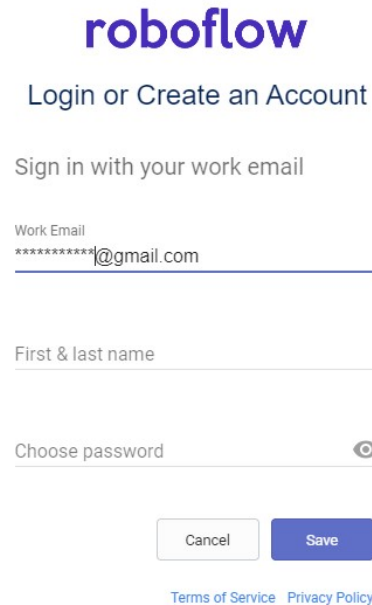
Different techniques can be used to train a model. More than 5500 photos made up the training set, necessitating a quick GPU to train the model. A website uses step-by-step instructions to walk users through the process of training the model. This website uses Google Colab to carry out each step using instructions presented as cells. The training data is first sent to Roboflow, which annotates the photos and divides them into training, validation, and testing sets. Then this version of images is stored in the form of a link that is provided to Google Colab to work further. Once the given steps have been completed, we can train the model using the Google provided GPU to continuously train in the form of batches or epochs. These batches take about 9.6 minutes per batch to complete. Overall, the number of batches was set to 45 to get a good result.

Mostly, the training is performed in sets of batches rather than continuous because it is an arduous task and takes a lot of time. There are some limitations of using Google Colab. **There is a time limitation of 12 hours** after which the files can neither be saved nor retrieved and because of that, we may lose all the time and energy that was put to get the results. Therefore, in order to achieve results in limited time, we train the model using limited sets of epochs and store their result in a model file as “.pt” and later on feed the same file to the training data to continue from that point onwards. Hence, all the files remain accessible and easily available to be reused for future training. In order for the system to continue operating in the background and maintain Internet connectivity, the other option necessitates constant screen time on the used computer. If a total of 45 epochs are to be done, it will take around 7 and a half hours to finish the training method on average because 5 epochs take about 48 minutes to complete.



## 5.2 Steps followed for training the model.

Firstly, create a new account or if it already exists, sign-in to the account.



The image shows the Roboflow login and account creation interface. At the top, the 'roboflow' logo is displayed in a bold, purple font. Below the logo, the text 'Login or Create an Account' is centered. Underneath, the instruction 'Sign in with your work email' is shown. The form consists of three input fields: 'Work Email' with the placeholder '\*\*\*\*\*@gmail.com', 'First & last name', and 'Choose password' which includes a toggle icon for password visibility. At the bottom of the form are two buttons: a light grey 'Cancel' button and a dark blue 'Save' button. Below the buttons, there are two links: 'Terms of Service' and 'Privacy Policy'.

Figure 5.1: Signup Form

Once logged in, a workspace will be available where a sample project is provided for practice. Roboflow is used to annotate and create a dataset to be fed for training the model in yolov5. Here, images can be uploaded and marked for the presence of desired objects. The process of annotation was discussed in detail in chapter 2.

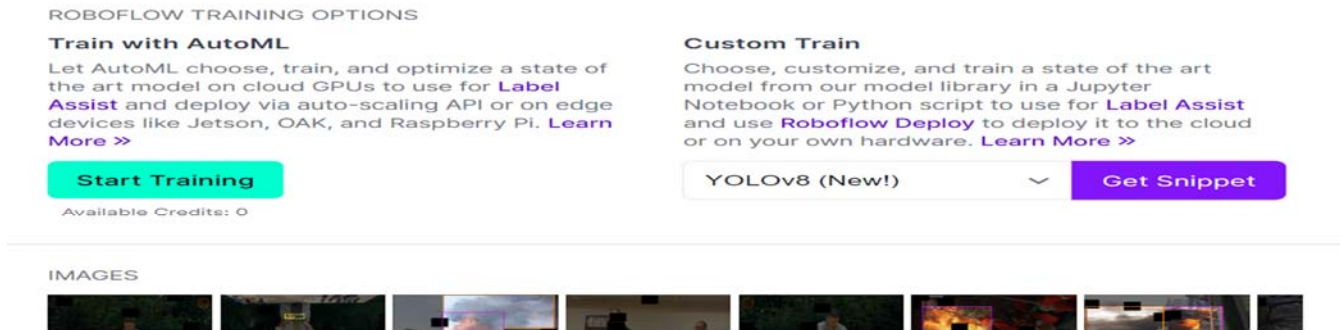


Figure 5.2: Training Option

After annotation, these images can be exported to Google Colab using the highlighted version in the form of a link that is to be inserted in the cells.

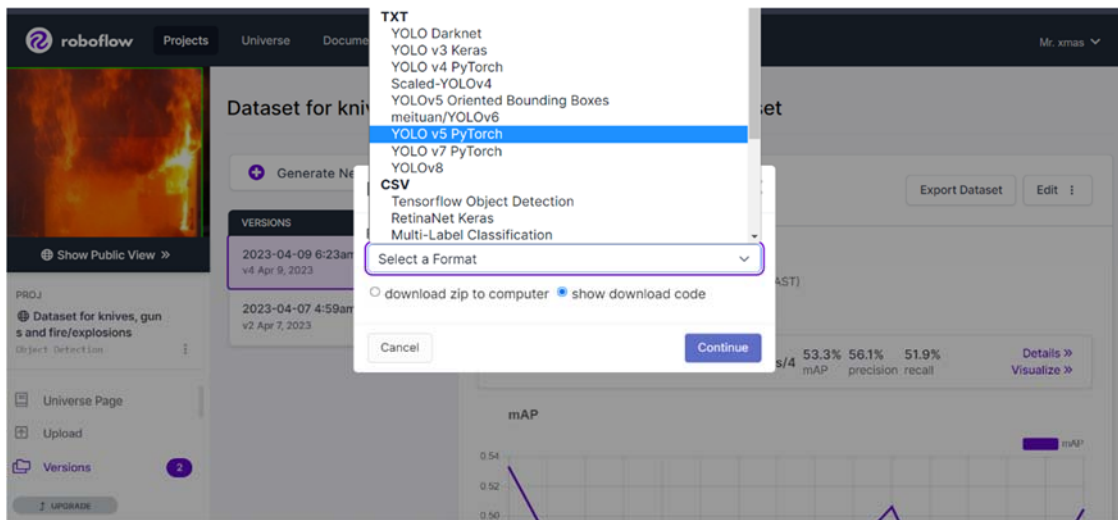


Figure 5.3: Selecting the dataset format



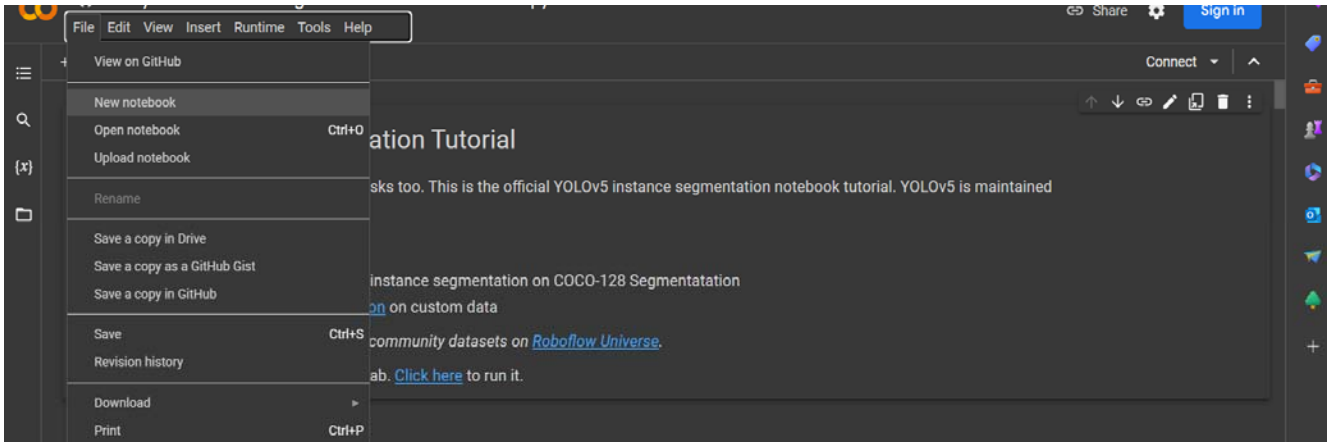


Figure 5.5: Creating a new notebook

II. New notebook is created named as 'Untitled0.ipynb'.

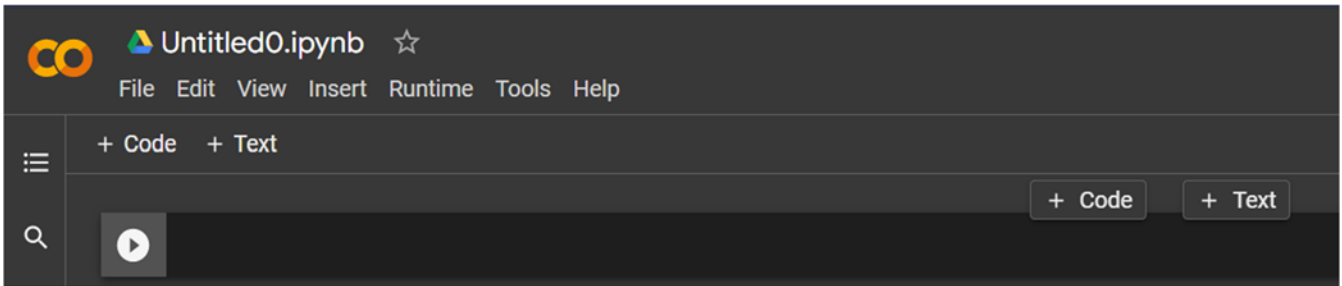


Figure 5.6: Successful creation of the notebook

III. Copy cells from 'Roboflow-Custom-YOLOv5' onto the new notebook.

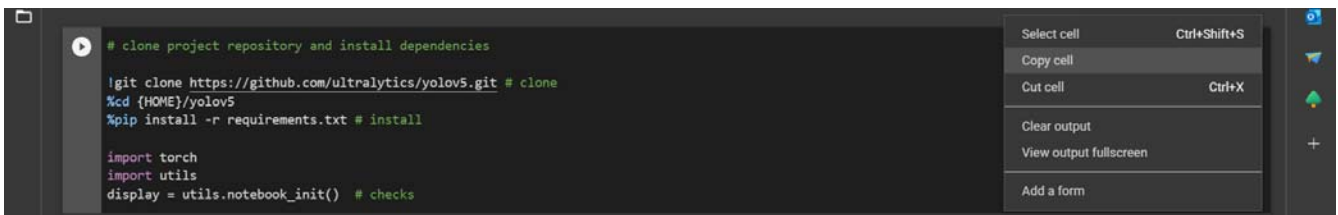
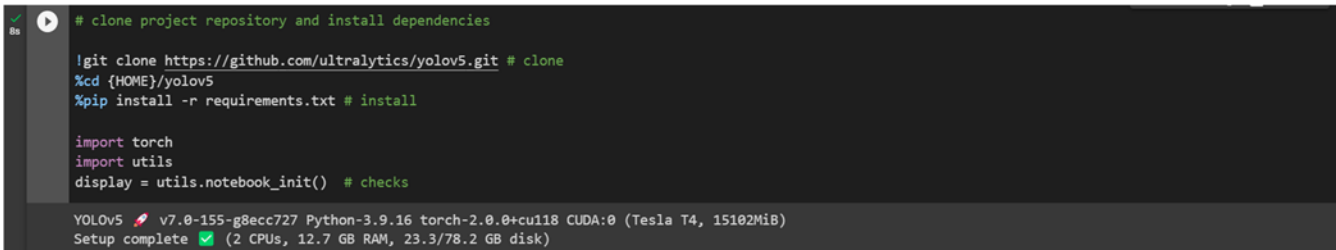


Figure 5.7: Copying Cells

IV. Run each next cell in the new notebook after the last cell has completed its execution.



```
# clone project repository and install dependencies

!git clone https://github.com/ultralytics/yolov5.git # clone
%cd {HOME}/yolov5
!pip install -r requirements.txt # install

import torch
import utils
display = utils.notebook_init() # checks

YOLOv5 v7.0-155-g8ecc727 Python-3.9.16 torch-2.0.0+cu118 CUDA:0 (Tesla T4, 15102MiB)
Setup complete (2 CPUs, 12.7 GB RAM, 23.3/78.2 GB disk)
```

Figure 5.8: Running each cell

After completing the training, we get a “.pt” which is our actual trained model of our dataset. We can implement this file in our WebApps, desktop apps and mobile apks.

After training the model we can also use the TensorFlow to check the health of our dataset and its accuracy with different graphs plotted for it.

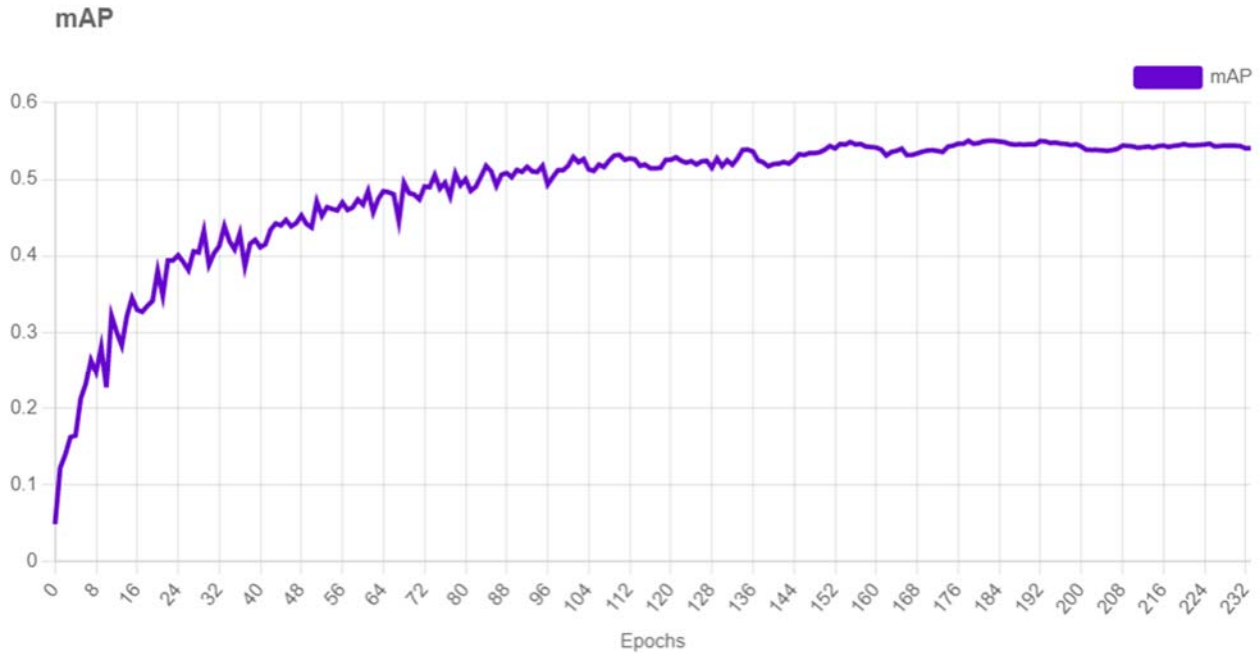
## Results of Training

As it is visible that after training for 232 + Epochs the precision of the model rises to 56%. This is to better detect the specific classes with more and more accuracy.

## Training Results

dataset-for-knives-guns-and-fire-explosions/2    55.0%    56.0%    56.8%  
mAP    precision    recall

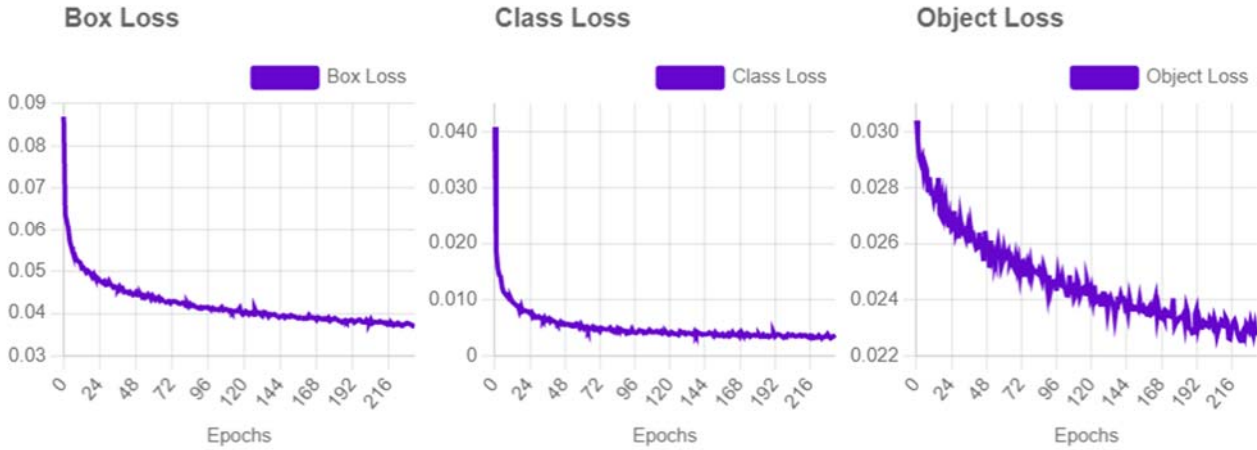
[Details >>](#)  
[Visualize >>](#)



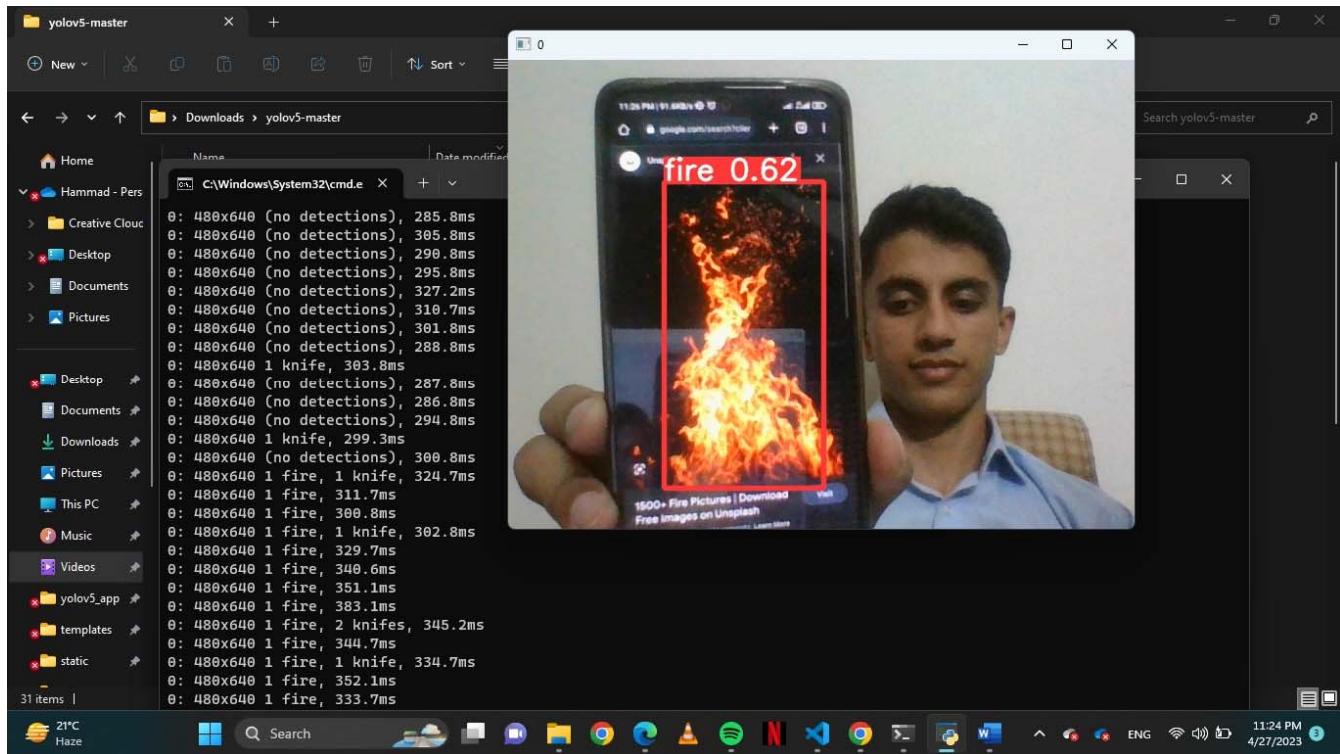
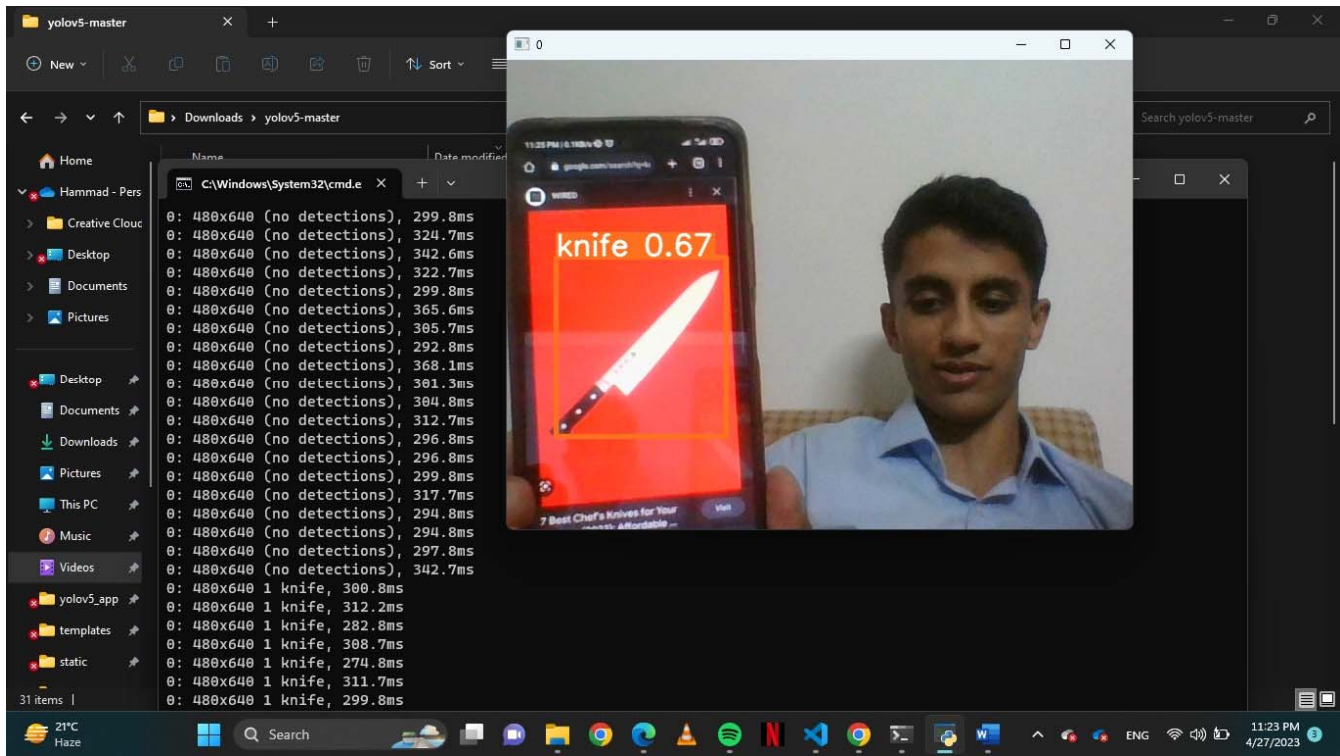
## Losses and Progress

The following graphs show that over the course of training the model, the box loss rate, class loss rate, and object loss rate have all drastically lowered. This results in a model that is more accurate than the previous one and can accurately locate the classes on the screen in addition to accurately detecting

them.

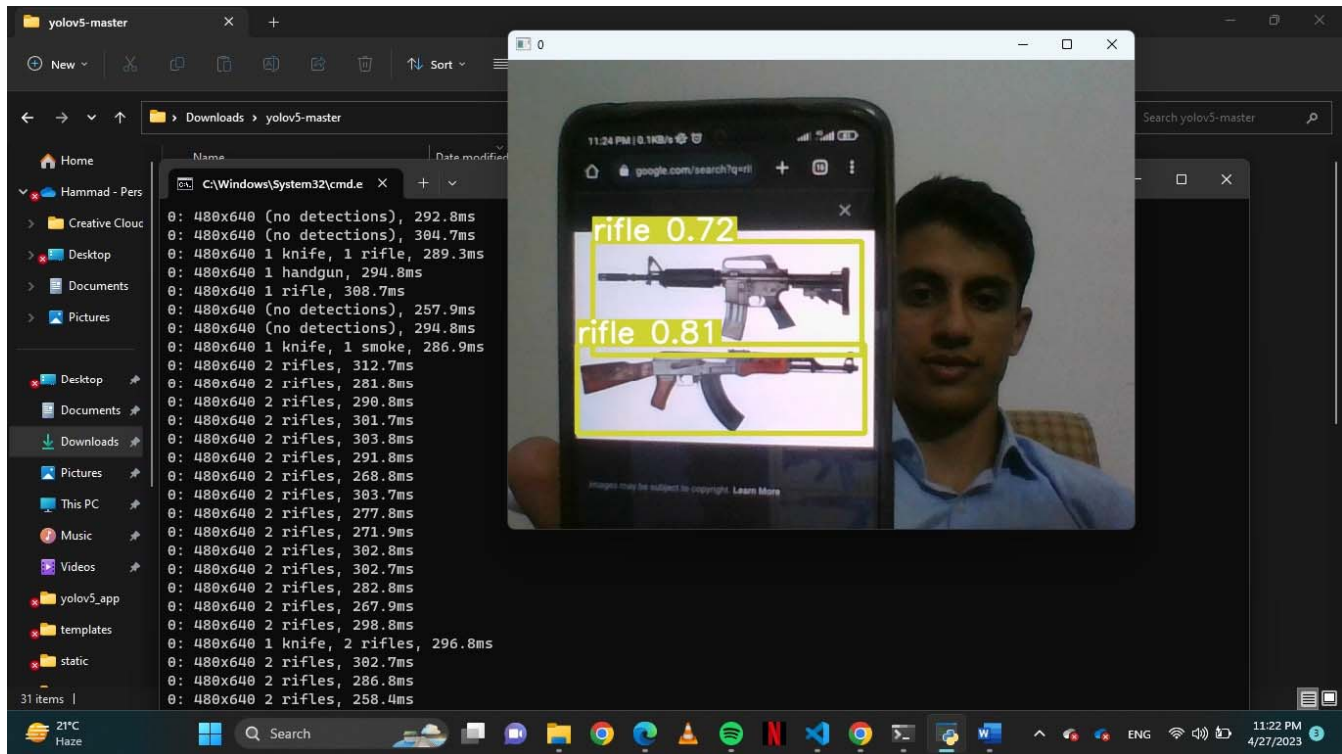
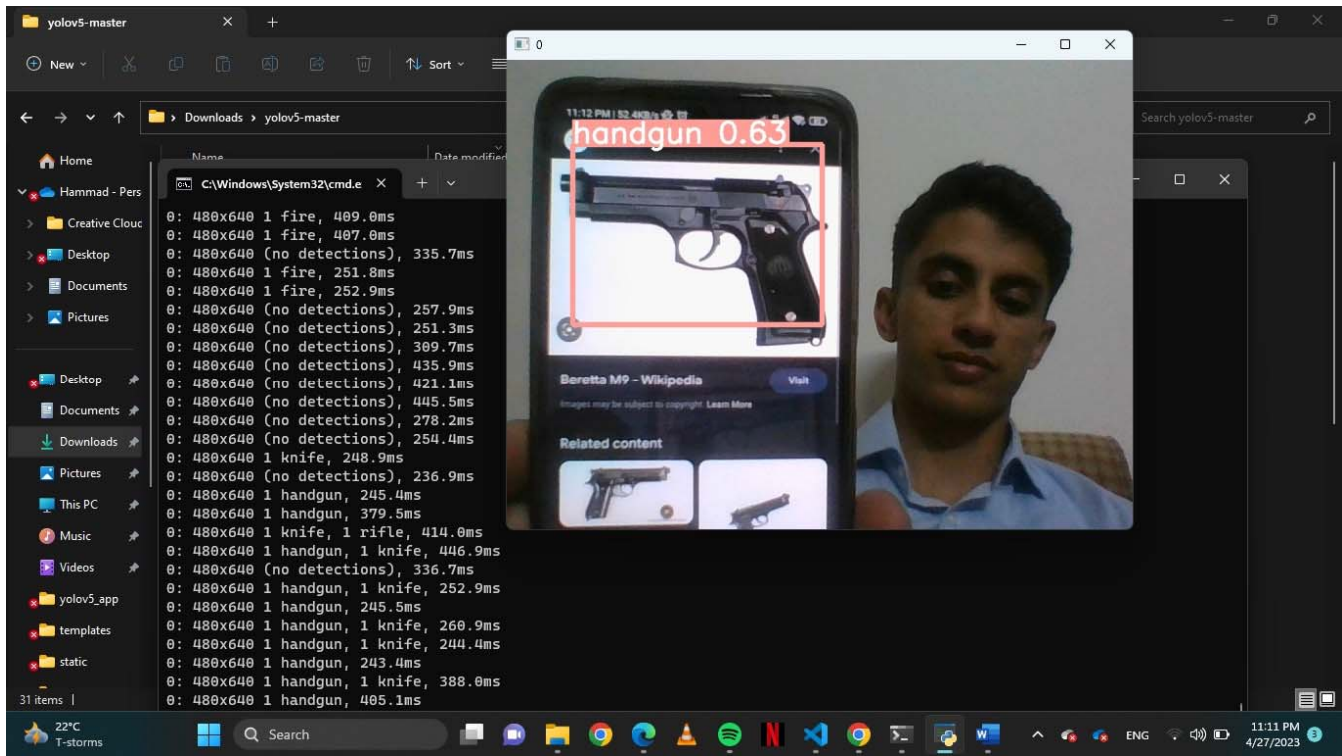


# Testing the Fire and Knife Class:





# Testing the Rifle and Handgun Class:



## CHAPTER 6: Integrating the Arduino with Application

Integration of the of the application was done in two steps:

1. Establishing the connection between Arduino board and Python application.
2. Write an Arduino sketch to receive the commands from the Python application

What was the need of doing these two steps?

We did these two steps to create an alarm system, so that whenever any abnormal pattern is detected it should raise an alarm. It'll help us in alerting the concerned agencies to act before the damage is done beyond the repair. It can help the law enforcement agencies to quickly reach to the sight where abnormal activity is detected and take effective measures to foil any attempt of criminal activity.



Led is turned 'on' when detection occurs

### 6.1 Establishing the connection

Establishing the connection was done by using the python serial library which helps us to send the commands over the designated port.

You can download this library using the following code in cmd or powershell: 'Pip install pyserial'

## 6.1.1 Python Code for Establishing the Connection

```
import serial

# Open the serial port at 9600 baud
ser = serial.Serial('/dev/ttyACMO', 9600)

# Send a command to the Arduino
ser.write(b'led on')
```

You can designate your specific port number in '/dev/ttyACMO' to for example 'COM4'. Now, this code will send the command over Arduino board.

### 6.1.1.1 Python Code Used in the App

We had to modify this code according to our requirements screenshot is given below:

```
import serial
import time

# Change the port and baud rate to match your Arduino board
PORT = 'COM3'
BAUD_RATE = 9600

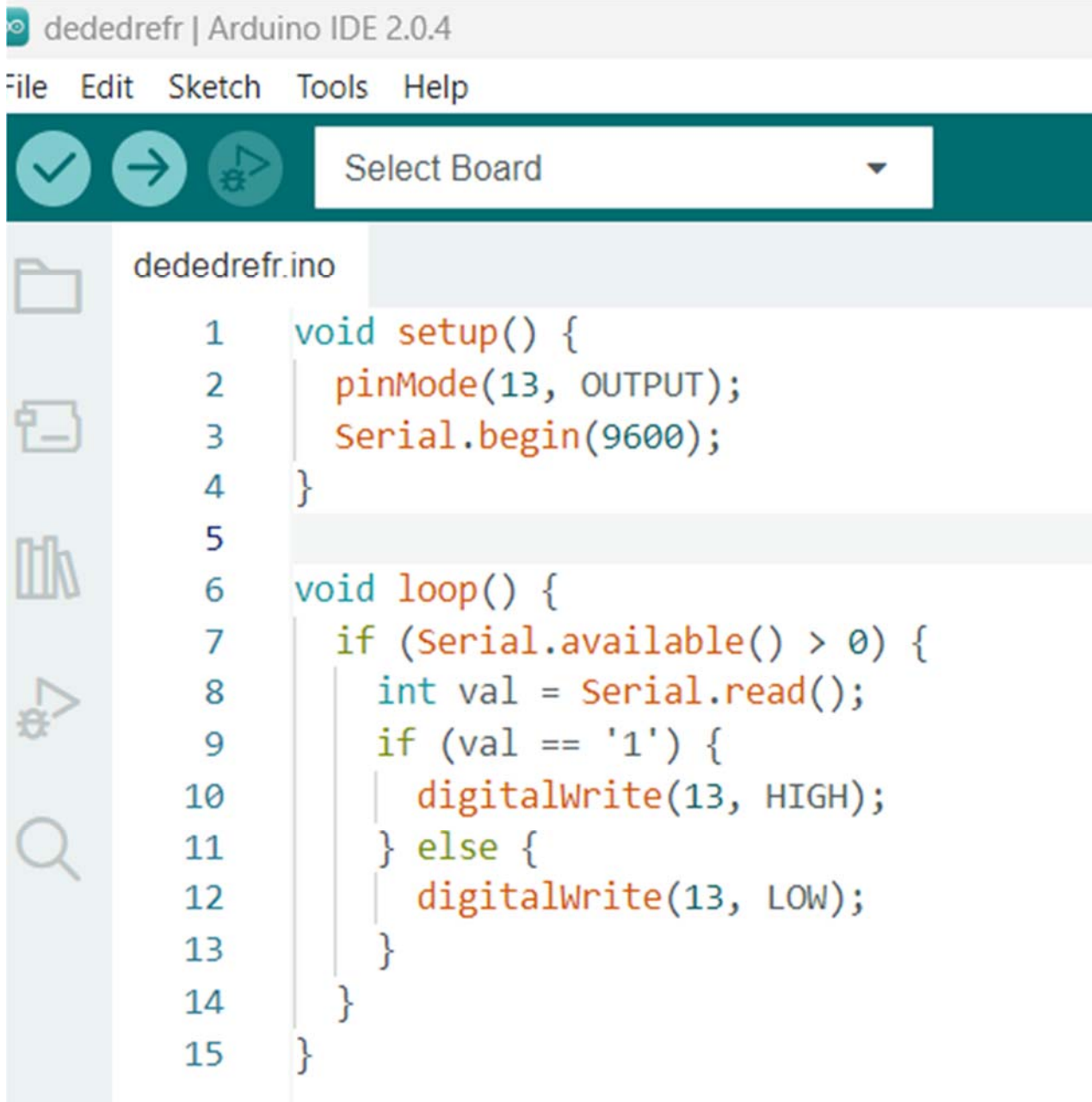
# Open a serial connection to the Arduino board
ser = serial.Serial(PORT, BAUD_RATE)

# Wait for the connection to be established
time.sleep(2)

# Send the command to turn on the LED
ser.write(b'1')

# Close the serial connection
ser.close()
```

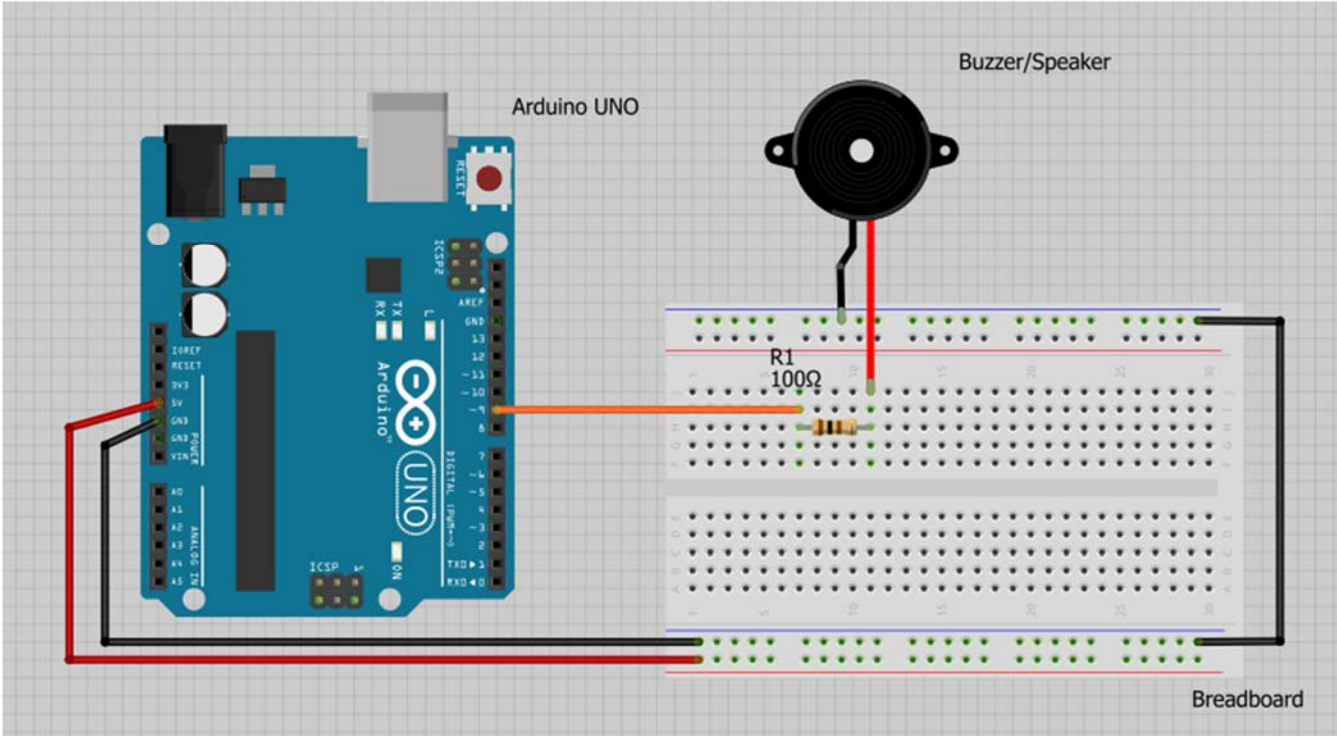
## 6.2 Arduino Sketch



```
dededrefr | Arduino IDE 2.0.4
File Edit Sketch Tools Help
Select Board
dededrefr.ino
1 void setup() {
2   pinMode(13, OUTPUT);
3   Serial.begin(9600);
4 }
5
6 void loop() {
7   if (Serial.available() > 0) {
8     int val = Serial.read();
9     if (val == '1') {
10      digitalWrite(13, HIGH);
11    } else {
12      digitalWrite(13, LOW);
13    }
14  }
15 }
```

Here 'Serial.begin' establishes the connection and 'if (Serial.available() > 0)' means that if a signal is received on that port then turn on the led light.

### 6.3 Arduino and Buzzer Schematic



## **CHAPTER 7: DEVELOPING THE USER INTERFACE**

### **7.1 Programming/Markup Languages**

1. HTML 5
2. JavaScript (ES6)
3. Python (3.8)

### **7.2 Framework**

Flask

### **7.3 Libraries**

Axios Sweet Alert

jQuery (3.2)

### **7.4 Tools**

XAMPP

VS Code

Git



## **7.5 The Significance of Using Flask**

A variety of commercially accessible frameworks and tools are available for the development of web apps. But current developments show that Flask is the greatest option for creating cutting-edge, full-stack web apps.

### **7.5.1 A Progressive Framework**

Popular Python web framework Flask is renowned for its ease of use and adaptability. It makes for a perfect solution for small to medium-sized projects because it enables web application developers to work swiftly and effectively. Flask is created with a minimalistic approach that just offers the essential tools required to get started, making it simple to understand and use.

The progressive nature of Flask is one of its fundamental characteristics. This means that a variety of web applications, from straightforward static web pages to intricate dynamic ones requiring user authentication and database connectivity, may be created using it. Developers can gradually add more sophisticated features and functionality to their apps as they gain experience with Flask.

Fundamentally, Flask offers a straightforward and understandable routing mechanism that converts URLs to Python functions. This makes it simple to develop unique endpoints to manage various requests, including GET, POST, PUT, and DELETE. Before deploying applications to a production environment, it is simple to test and debug them locally using Flask's integrated development server.

### **7.5.2 A Scalable Framework**

Flask supports extensions that can add application features as if they were implemented in Flask itself.



Scalability is one of Flask's strongest suits. According to the demands of your application, Flask is intended to be a lightweight, adaptable framework that can be easily scaled up or down. A modular architecture, which enables developers to add or remove components as needed, is used to achieve this.

Along with its modular structure, Flask offers a variety of in-built scalability-enhancing features. For instance, Flask's built-in support for caching and session management can enhance your application's performance. Asynchronous programming is also supported by Flask, which is helpful when working with big volumes of data or processing-intensive jobs.

The use of extensions is a further means of scaling Flask. Developers from the big and vibrant Flask community have produced a broad variety of extensions that can be utilised to enhance the functionality of your application. You can quickly and simply add additional functionality to your application as needed by installing and configuring these extensions.

### **7.5.3 A Community Framework**

Building a community framework for web applications is possible using Flask. We will examine the major characteristics of Flask, such as its lightweight architecture, straightforward routing system, and potent templating engine, that make it suitable for community-based web development. Additionally, we'll go through some recommended practices for using Flask to develop a community-based online application, such as user authentication, database integration, and security considerations.

Flask can give you the tools and flexibility you need to construct a dynamic and interesting community-based web application, whether you're creating a social network, an online forum, or a platform for

collaborative content. Let's explore Flask's world and see how it can assist you in creating your next online project.

## 7.6 Architecture of Flask

Flask is based on MVC architecture.

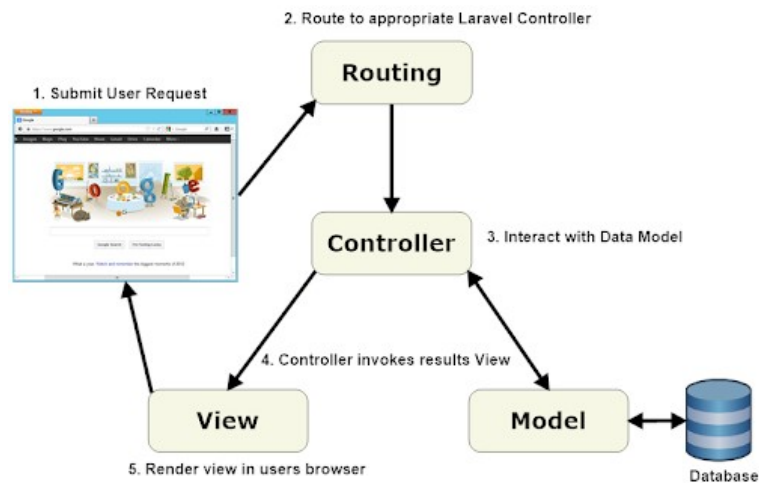


Figure 7.1

## 7.7 Setting Up the Environment

Before getting started with the Flask framework, it is important to set up the environment properly.

Here are the steps to follow:

1. Install Python: Flask is a Python web framework, so you'll need to have Python installed on your computer. You can download Python from the official website (<https://www.python.org/downloads/>). Make sure to install the latest stable version of Python.

2. Install Flask: Once Python is installed, you can install Flask using the following command in the terminal or command prompt:

```
pip install Flask
```

This will install the latest stable version of Flask on your computer.

3. Create a Virtual Environment: It is recommended to create a virtual environment for your Flask projects to avoid conflicts with other Python packages. You can create a virtual environment using the following command:

```
python -m venv myenv
```

This will create a virtual environment named "myenv" in the current directory.

4. Activate the Virtual Environment: Once the virtual environment is created, you need to activate it using the following command:

For Windows:

```
myenv\Scripts\activate
```

For Linux/Mac:

```
source myenv/bin/activate
```

5. Set up a Flask Project: Now that the virtual environment is activated, you can create a new Flask project using the following command:

```
flask new myproject
```

This will create a new Flask project named "myproject" in the current directory.

6. Run the Flask App: Finally, you can run the Flask app using the following command:

```
flask runs.
```

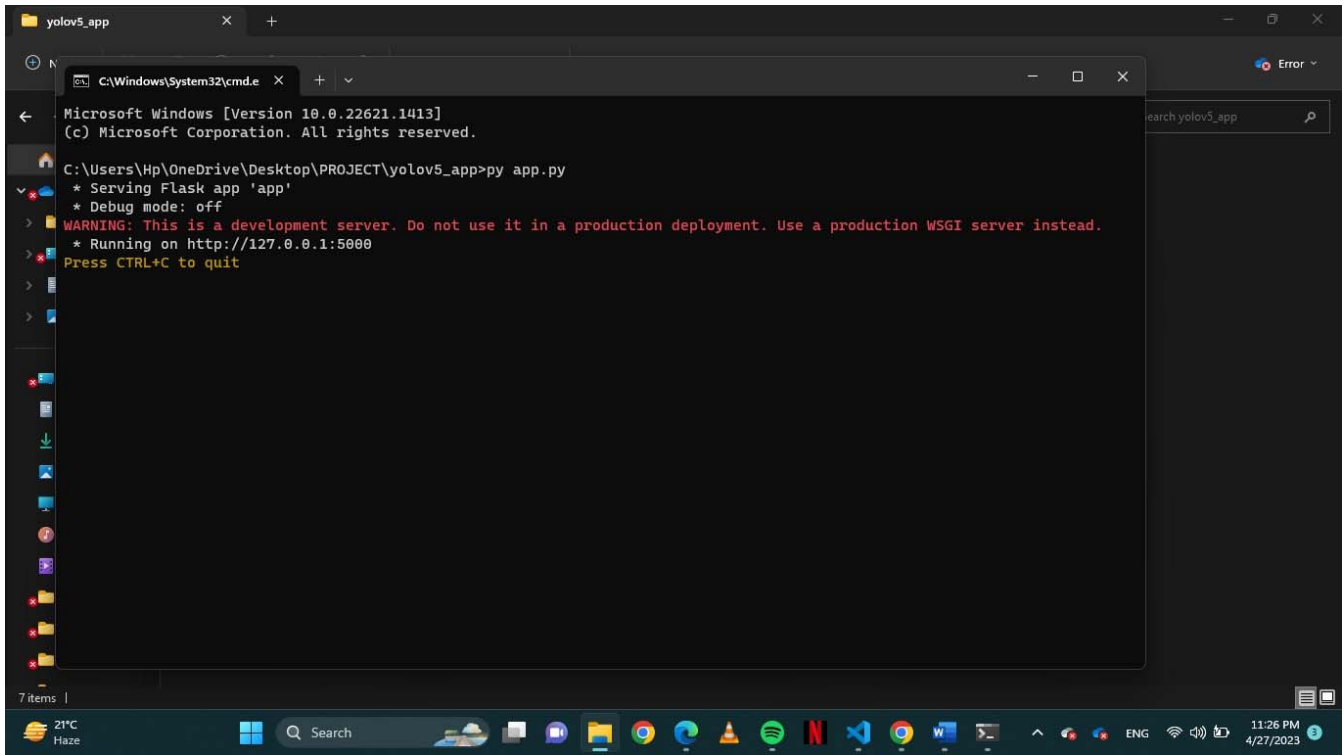
7. This will start the development server on your local machine. You can access the Flask app by visiting <http://localhost:5000/> in your web browser.

## **7.8 Setting up the Routes**

The ability to handle routes, which control how incoming requests are processed by the application, is one of Flask's core capabilities.

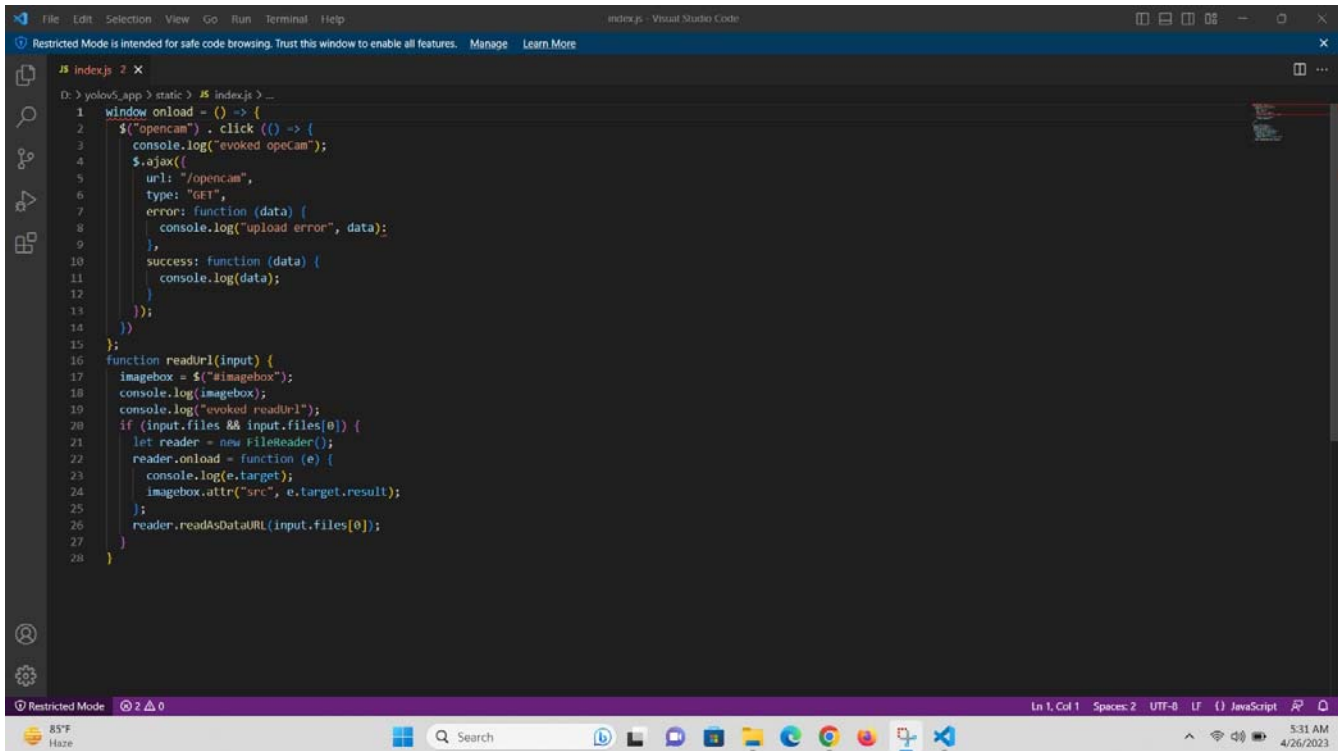
You must first construct a Flask application instance using the Flask class in order to configure routes in Flask. You then use this instance to provide the routes for your application.

The `route()` decorator is used to define routes in Flask. This decorator ties a URL pattern to a view function, which is then called whenever the appropriate URL is requested. It accepts a pattern of URLs as its parameter.



## 7.9 Creating Views

To create a view in Flask, you define a Python function that handles a specific URL. A view function is a Python function that accepts a request object and returns a response object.



```
1 window.onload = () => {
2   $("opencam").click(() => {
3     console.log("evoked opeCam");
4     $.ajax({
5       url: "/opencam",
6       type: "GET",
7       error: function (data) {
8         console.log("upload error", data);
9       },
10      success: function (data) {
11        console.log(data);
12      }
13    });
14  });
15 };
16 function readUrl(input) {
17   imagebox = $("#imagebox");
18   console.log(imagebox);
19   console.log("evoked readUrl");
20   if (input.files && input.files[0]) {
21     let reader = new FileReader();
22     reader.onload = function (e) {
23       console.log(e.target);
24       imagebox.attr("src", e.target.result);
25     };
26     reader.readAsDataURL(input.files[0]);
27   }
28 }
```

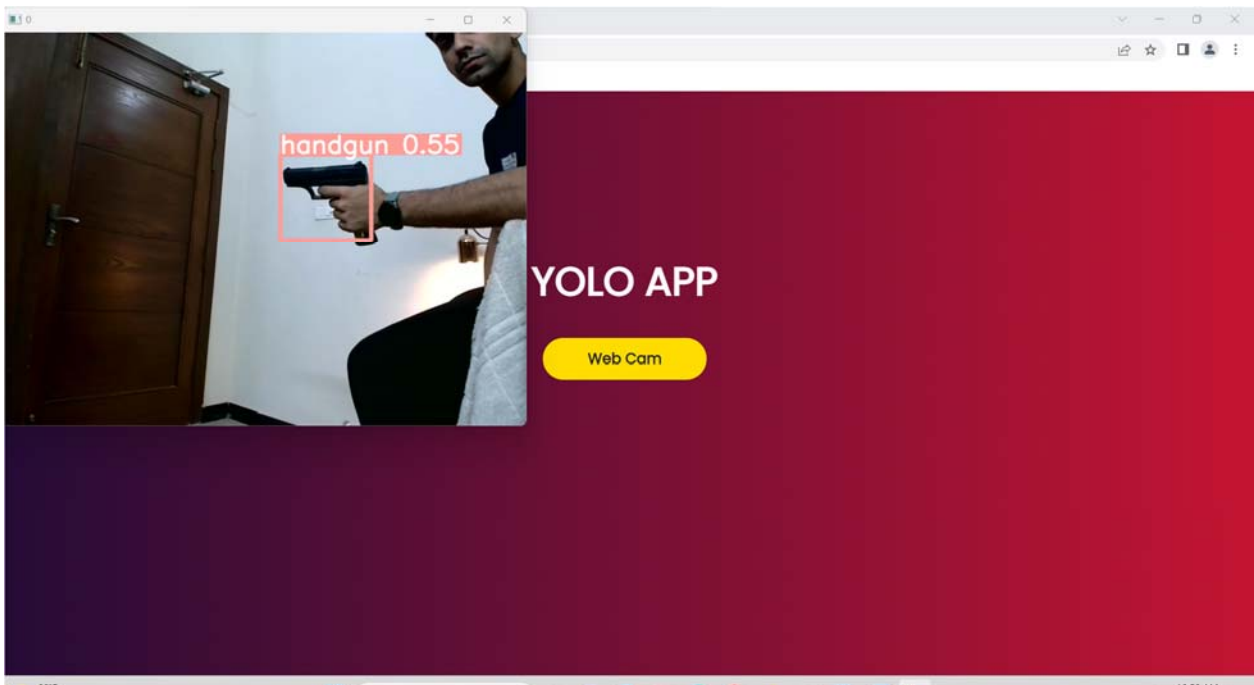
## 7.10 Using the Interface.

I. Home page of the website where user can open the webcam and start using the model



Figure 7.3: Webapp interface

3. Cam portion with detection occurring.



## REFERENCES

- Cano-Rodríguez, A. M., López-Robles, J. R., & García-Vázquez, J. P. (2021). Crime Analysis Using Data Mining Techniques: A Systematic Review. *IEEE Access*, 9, 19724-19742.
- Alkhatib, S., & Zulkernine, M. (2018). Anomaly detection in big data: a survey. *Journal of Big Data*, 5(1), 1-28.
- Ahmadi, H., & Arora, A. (2017). Crime detection in big data using machine learning. *International Journal of Advanced Research in Computer Science*, 8(7), 18-23.
- Amin, M. B., Mekki, K., & Alsmadi, M. K. (2021). Crime prediction in big data using machine learning: A systematic literature review. *Journal of King Saud University-Computer and Information Sciences*, 33(3), 267-275.
- Zhang, Y., Sun, L., & Wang, X. (2019). Crime anomaly detection using a deep belief network based on unsupervised learning. *Neurocomputing*, 357, 242-253.
- Li, J., Li, Y., Chen, G., & Li, K. (2019). Crime hotspot detection using deep learning with social media data. *Future Generation Computer Systems*, 98, 1-8.
- Chen, Y., Du, H., & Qian, Y. (2017). Crime prediction using data mining techniques in big data. *Journal of Ambient Intelligence and Humanized Computing*, 8(2), 289-296.
- Qiao, Z., Fang, Z., & Lu, J. (2021). Crime pattern analysis and prediction based on spatiotemporal data using a convolutional neural network. *IEEE Access*, 9, 25180-25190.
- Koutroumanis, N., & Karakostas, B. (2017). Towards the identification of hotspots in urban crime using big data analytics. In *Proceedings of the 8th Balkan Conference in Informatics (BCI)* (pp. 1-6).
- Zhang, Y., Zhu, X., & Cui, L. (2019). A novel hybrid method for crime prediction using data mining and machine learning. *Knowledge-Based Systems*, 179, 23-34.



- Saha, S., & Mookerjee, S. (2020). Predicting crime hotspots in urban areas using data mining techniques. *International Journal of Intelligent Information Technologies (IJIIT)*, 16(3), 22-44.
- Silva, M. C., & Rosa, N. S. (2020). Crime prediction using machine learning techniques: a systematic literature review. *International Journal of Advanced Intelligence Paradigms*, 16(4), 374-404.
- Wang, X., & Li, C. (2021). A spatiotemporal clustering-based method for crime prediction using big data. *IEEE Access*, 9, 31871-31880.
- Khandagale, P., & Kulkarni, P. (2020). Crime prediction using machine learning: A systematic literature review. *International Journal of Information Technology*, 12(1), 71-84.
- Chen, C., & Lv, Y. (2020). Predicting crime hotspots based on improved FCM clustering algorithm. *Journal of Ambient Intelligence and Humanized Computing*, 11(10), 3837-3849.
- Zhang, Y., Huang, J., & Feng, Y. (2020). Crime hotspot detection based on deep learning with spatiotemporal data. *IEEE Access*, 8, 131707-131716.
- Xu, H., & Wang, S. (2019). A spatiotemporal LSTM network for crime prediction. *IEEE Transactions on Intelligent Transportation Systems*, 21(7), 3078-3087.
- Liu, L., Li, C., & Zhou, X. (2021). A deep learning-based approach for crime prediction with auxiliary data. *Future Generation Computer Systems*, 115, 313-323.
- Wang, D., & Chen, L. (2020). A hybrid approach for crime prediction using spatiotemporal data. *Journal of Ambient Intelligence and Humanized Computing*, 11(12), 5781-5792.
- Elsayed, R., Khorshed, M. S., & Ali, M. A. (2018). Crime prediction in big data using machine learning techniques. In *Proceedings of the 11th International Conference on Computer Science and Education (ICCSE)* (pp. 498-503).

# thesis

---

## ORIGINALITY REPORT

---

**16%**

SIMILARITY INDEX

**10%**

INTERNET SOURCES

**3%**

PUBLICATIONS

**14%**

STUDENT PAPERS

---

## MATCH ALL SOURCES (ONLY SELECTED SOURCE PRINTED)

---

8%

★ Submitted to Higher Education Commission

Pakistan

Student Paper

---

Exclude quotes Off

Exclude matches Off

Exclude bibliography On