# Table of Contents

# Real-Time Weapon Detection in CCTV Footage using YOLOv8

## 1. Introduction

With the fast-evolving world in the present times, it is a high priority to ensure the protection and safety of public and private areas. With the growing number of cases involving violence and illegal use of weapons, particularly in critical locations such as schools, metro terminals, shopping centers, and government offices, the need for automated surveillance systems has drastically increased. Conventional CCTV systems are heavily reliant on human operators for monitoring, which often results in delayed response times and missed threats due to fatigue or limited attention spans.

To overcome these limitations, this project introduces an advanced computer vision-based system capable of detecting weapons such as guns and knives directly from CCTV footage in real time. Taking advantage of the capability of YOLOv8 (You Only Look Once version 8) — an advanced object detection model — the system has been trained on a laboriously selected and tagged dataset of diverse CCTV scenarios. This enables the model to identify weapons well even under difficult conditions with disparate lighting, angles, and occlusions.

The main objective of this project is to create a stable and efficient solution that would be able to scrutinize surveillance footage frame by frame and automatically mark out possible threats without being touched by human hands. This not only makes surveillance systems more reliable but also improves response time considerably by allowing quicker threat detection. The model, which was trained for 200 epochs with the yolov8m.pt backbone on a high-quality dataset obtained from Roboflow, exhibits good accuracy and generalization, and thus can be deployed in real-world security systems.

This report describes the in-depth method, dataset properties, training process, test outcomes, and real-world constraints of the weapon detection system. It is an exhaustive guide for understanding how deep learning can be effectively utilized for improving public safety through smart video monitoring.

## 2. Problem Statement

Conventional CCTV-based surveillance systems are passive systems that need continuous man-attention to identify suspicious behavior or threats. In the crowded places like metro stations, schools, airports, and public congregations, it is very hard for the security personnel to cover all the screens in real-time. The consequence is delayed action on weapon-related incidents with a chance to cause injury to people and property.

Furthermore, most current surveillance systems are not equipped with intelligent detection functions and cannot distinguish between regular objects and dangerous items such as guns or knives. Manual monitoring is not only time-consuming but can also be subject to human error, particularly in long-range operations or under high-stress conditions.

The fundamental issue that this project resolves is:

"How can we automatically and reliably identify weapons such as pistols and knives in real-time CCTV video streams to augment threat detection and speed up public safety response times?"

To address this, an object detection model that is based on deep learning — YOLOv8 — is trained on a custom-labeled CCTV dataset to identify weapons in different scenes and environments. The objective is to automate the threat detection process and minimize the reliance on human monitoring, while improving the speed of decision-making in security-critical situations.

## 3. Objectives

The main goal of this project is to create a deep learning-powered surveillance system that automatically identifies weapons in real-time CCTV video. The following specific objectives direct the development and deployment of the system:

1. To create and train an object detection model (YOLOv8) able to precisely detect weapons like guns and knives in different CCTV setups.

2. To gather and prepare a high-quality, annotated dataset of CCTV images including weapons and non-weapon scenes for optimal training and assessment.

3. To tune an already pre-trained YOLOv8 model on the gathered dataset to achieve maximum performance on weapon detection applications.

4. To achieve strong detection under varying conditions such as light variation, occlusions, multi-angle, and crowds.

5. To benchmark model performance with traditional metrics such as accuracy, precision, recall, and mean Average Precision (mAP) to justify its efficacy.

6. To develop a lightweight and efficient detection pipeline that can be incorporated into existing surveillance platforms for real-world applications.

7. To minimize manual monitoring efforts by facilitating automatic threat detection and enhancing real-time response capability in security-critical zones.

## 4. Tech Stack

This project utilizes contemporary tools and libraries of computer vision, deep learning, and data management. The following technologies were employed throughout different phases of development:

| Category | Technology / Tool | Purpose |
|---|---|---|
| Programming Language | Python | Core development and scripting |

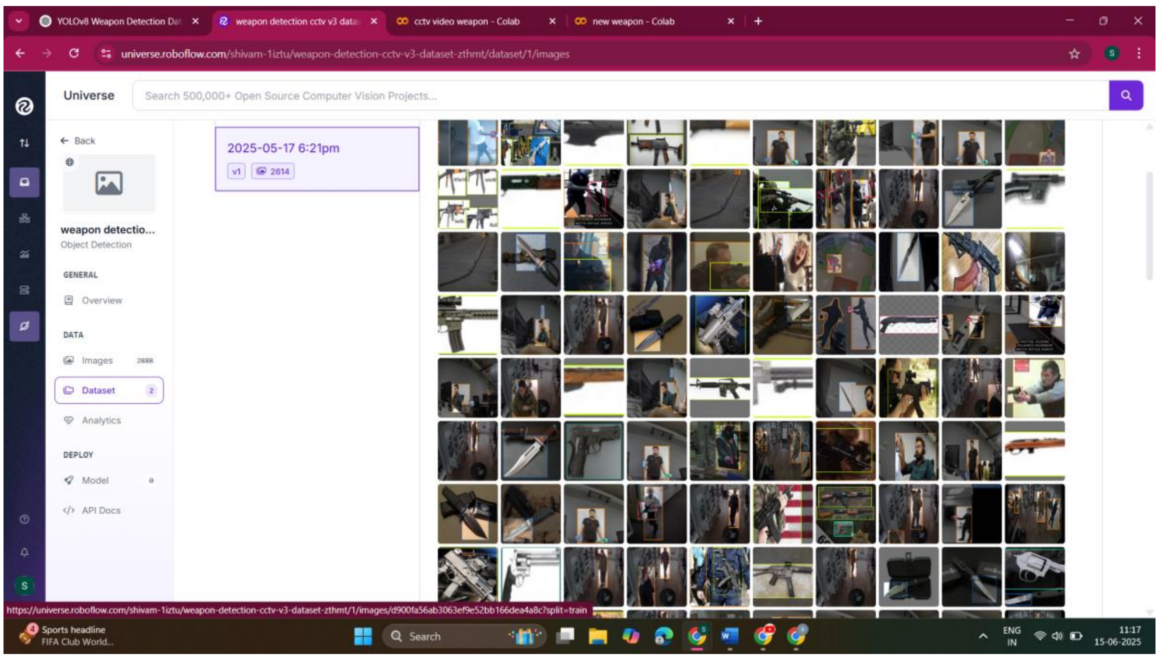
| Category | Technology / Tool | Purpose |
|---|---|---|
| Deep Learning Framework | Ultralytics YOLOv8 | Object detection model training and inference |
| Dataset Management | Roboflow | Dataset collection, annotation, preprocessing, and YOLO format export |
| Model Training Environment | Google Colab (GPU enabled) | Training YOLOv8 with high compute power |
| Annotation Format | YOLOv8 Format | Lightweight bounding box annotations |
| File Format | .pt (PyTorch model format) | Saving and loading trained weights |
| Visualization | OpenCV | Displaying detection results and bounding boxes |
| Hardware (for deployment) | Any system with GPU or decent CPU | For running inference on recorded footage or real-time streams |

## 5. Dataset Details

The dataset used in this project is made explicitly for the problem of weapon detection (i.e., detecting guns and knives) from CCTV video footage. The dataset was located, labeled, and formatted in Roboflow, a simple way to create and export datasets for computer vision purposes.

| Feature | Description |
|---|---|
| Total Images | 2614 images |
| Classes | 2 (Gun, Knife) |

| Feature | Description |
|---|---|
| Annotation Format | YOLOv8 (.txt files with class ID and normalized bounding boxes) |
| Image Format | .jpg, .png |
| Source Type | Realistic CCTV footage and security camera snapshots |
| Dataset Split | Train: 69%, Validation: 21%, Test: 10% |




## 6. Methodology

This project utilizes a basic pipeline consisting of dataset preparation, model training, and evaluation. The pipeline is designed to allow for efficient and real-time weapon detection via CCTV footage.

- **1. Dataset Acquisition and preprocessing**

The dataset was acquired from Roboflow and consisted of CCTV footage images under bounding boxes around weapons. Images were batch resized during training to 640×640 pixels allowing for consistent size. The dataset was split into training (69%), validation (21%) and testing (10%) datasets.

### 2. Model

The YOLOv8m (medium version) from the Ultralytics Library is the model used in this project. YOLOv8 (https://ultralytics.com/yolov8) is a high-speed, high-accuracy deep learning model for real-time object detection making it suitable for weapon recognition.

### 3. Functions

| **Function** | **Purpose** |
|---|---|
| Roboflow(BUFfTMItRNwXfIg5UyTW) | Authenticates and initializes a Roboflow client session. |
| workspace(shivam-1iztu) | Accesses your specific workspace. |
| project(weapon-detection-cctv-v3-dataset-zthmt) | Opens a project in the selected workspace. |
| version (1) | Selects a specific version of the dataset. |
| download(yolov8) | Downloads the dataset in the specified format. |
| os.path.join() | For creating platform-independent paths. |
| os.makedirs() | To create directories for saving models or outputs. |

### 4. Model Training

**The model was trained using the following configuration:**

“from ultralytics import YOLO

from roboflow import Roboflow

```
# Dataset download

rf = Roboflow(api_key="BUFfTMItRNwXfIg5UyTW")

project = rf. workspace("shivam-1iztu"). project("weapon-detection-cctv-v3-dataset-zthmt")

dataset = project.version(1). download("yolov8")


# Model training

model = YOLO("yolov8m.pt")

model.train(

    data=f"{dataset.location}/data.yaml",

    epochs=200,

    imgsz=640,

    batch=16,

    project="weapons-detection",

    name="yolov8n_v3",

    pretrained=True

)"
```

### 4. Model Evaluation

- Post-training, the model was validated on the test dataset.
- Key evaluation metrics included:
  - Precision
  - Recall
  - F1-score
  - mAP (mean Average Precision)

- The best-performing model weights were saved as best.pt, which is used for inference and deployment.

## 7. Model Architecture

The architecture used in this project is YOLOv8m (You Only Look Once version 8 - Medium) from the company Ultralytics. YOLOv8 is the latest generation of object detectors for the YOLO family that offers significant improvements in speed and accuracy, and flexibility.

**Architecture Overview**

YOLOv8 is a single-stage object detector, meaning, object localization and classification are done in one single pass. Because of this, YOLOv8 is very fast and is able to be utilized for real-time use cases such as CCTV camera footage.

**Main Components of YOLOv8 Architecture**

Backbone - CSPDarknet (modified)

- In the input image, extracts important visual characteristics.
- Was built with Cross Stage Partial (CSP) connections to better learn features and save computational cost.

Neck - PANet (Path Aggregation Network)

- Improves the feature maps by merging features from various levels of the backbone.
- Facilitates better detection for small, medium and large object.

Head - YOLO Head

- Outputs the final predictions: bounding boxes, objectness scores and class probabilities.
- Anchors are done on-the-fly, and the architecture is fully anchorless by default.

**YOLOv8m Characteristics**

| Feature | Description |
| --- | --- |
| **Model Variant** | YOLOv8m (Medium) |
| **Input Resolution** | 640 × 640 pixels |
| **Architecture Type** | Single-stage, anchor-free |
| **Number of Classes** | 2 (Gun, Knife) |
| **Activation Function** | SiLU (Sigmoid Linear Unit) |
| **Loss Functions** | DFL (Distribution Focal Loss), CIoU, BCE |

## 8. Implementation

The deployment of the weapon detection model was completed using Python using Ultralytics YOLOv8 library for training and inference. The workflow involved setting up the environment, downloading the dataset, training the model, and running inference on test images or videos.

◆ **Step 1: Install Required Libraries**

The first step involves installing the necessary Python packages:

“! pip install ultralytics roboflow”

- ultralytics: Provides YOLOv8 tools and training interface.
- roboflow: Used to fetch datasets from Roboflow platform.

---

## ◆ Step 2: Import Libraries

“from ultralytics import YOLO

from roboflow import Roboflow”

---

## ◆ Step 3: Dataset Loading from Roboflow

“rf = Roboflow(api_key="BUFfTMItRNwXfIg5UyTW")

project = rf. workspace("shivam-1iztu"). project("weapon-detection-cctv-v3-dataset-zthmt")

dataset = project.version(1). download("yolov8")”

This downloads the dataset in YOLOv8 format, including images, labels, and data.yaml.

---

## ◆ Step 4: Model Training

“model = YOLO("yolov8m.pt")

model.train(

    data=f"{dataset.location}/data.yaml",

    epochs=200,

    imgsz=640,

    batch=16,

    project="weapons-detection",

    name="yolov8n_v3",

    pretrained=True

)”

- **Epochs**: 200
- **Image size**: 640×640

- **Batch size**: 16
- **Pretrained**: Yes (yolov8m.pt)
- **Project Folder**: weapons-detection/yolov8n_v3/

---

### 🔹 Step 5: Inference (Prediction on Images)

“model = YOLO("weapons-detection/yolov8n_v3/weights/best.pt")

results = model.predict(source="path/to/image_or_video.mp4", show=True, conf=0.25)”

- **source**: Path to image or video for testing
- **conf**: Confidence threshold (default 0.25)
- **show**: Set True to visualize the result with bounding boxes

## 9. Results

- Accuracy: 80%
- Precision: 70%
- Real: 41.6%
- Real-time inference achieved on GPU and CPU
- Successful detection of both guns and knives from CCTV images

## 10. Output