Video Surveillance Threat Detection Using YOLOv8 and Custom Object Detection Models

Team: Adithya Vasudeva (3121889), Siddharth Ramaraj (3121986)

Course: Artificial Intelligence - BST-CS4BD-07A

Instructor: Kristian Rother

1. Project Title:

Automated Threat Detection in Video Surveillance Using YOLOv8 and Custom Object Detection Models for Cybersecurity Applications

2. Objective:

The primary objective of this project is to design and implement an AI-driven video surveillance system capable of automatically identifying and classifying potential threats in real-time video feeds. The system classifies video footage as a threat or not a threat based on the presence of weapons or other suspicious objects. This project aligns with cybersecurity domains involving physical security, anomaly detection, and automated threat assessment, offering a practical solution for monitoring sensitive areas.

3. Tools and Technologies:

- Programming Language: Python 3.10+
- Framework: Ultralytics YOLOv8 (You Only Look Once - Version 8)
- Libraries: PyTorch, Torchvision, NumPy
- Computer Vision: OpenCV for video processing
- Alarm System: winsound for Windows-based audio alert
- Logging: Python logging module for classification results
- Environment: Visual Studio Code on Windows 10
- Hardware: CPU-based training; optional GPU support
- Dataset Tool: Roboflow for dataset preparation and export

4. Dataset:

- Source: Roboflow custom dataset focusing on weapons detection in surveillance scenarios
- Classes Included: gun.
- Data Split: 80% for training, 20% for validation
- Format: YOLOv8 compatible with labeled bounding boxes and class IDs
- Challenges: Ensuring proper labeling accuracy and diversity in object appearances (e.g., different types of guns or knives)

5. Methodology:

5.1 Initial Attempt:

- Deployed pre-trained YOLOv8 models (YOLOv8n, YOLOv8m, YOLOv8l) for object detection.
- Encountered low detection accuracy for weapons; models failed to generalize to surveillance-specific scenarios.
- Conclusion: Pre-trained models lacked domain-specific training data for reliable threat detection.

5.2 Solution - Custom Training:

- Collected and curated a weapons dataset from Roboflow, focused on surveillance imagery.
- Augmented dataset with different lighting, angles, and occlusion to improve model robustness.
- Training executed on YOLOv8m for better balance of speed and accuracy.

5.3 Model Training:

- Training Command:
- yolo task=detect mode=train model=yolov8m.pt data=Data/Datasets/…/Data.yaml epochs=50 imgsz=640 device=cpu workers=2
- Metrics After Epoch 1:
  - Box Loss: 1.435
  - Class Loss: 2.227
  - mAP@0.5: 0.367 (mean Average Precision)
  - mAP@0.5:0.95: 0.162
- Final Trained Model Saved As: best.pt
- Note: Training on CPU was slow (~75 min per epoch), GPU training recommended for future scaling.

5.4 Threat Classification Logic:

- System parses detection results from each frame.
- Threat condition: Presence of any object in [gun] along with human figures.
- Result logged per video with justification for classification.
- Alarm System: On detecting a threat, alarm.wav is played using winsound.

- Logging: Results recorded in results_log.txt with threat justification and timestamp.

6. Implementation:

Detection Script Features

- Handles multiple videos dynamically.
- Detects objects frame-by-frame with confidence filtering.
- Compares detected classes to threat list.
- Plays alarm.wav using winsound on threat detection.
- Logs result with detailed object list and classification result.

Key Features:

- Multi-video support with dynamic input handling
- Frame-by-frame object detection with confidence thresholding
- Intersection of detected classes with predefined threat list
- Sounds an alarm when a Threat is detected.
- Logging of detailed results to results_log.txt

Output Example:

- Video1 has been classified as a THREAT because of: gun.
- Video2 has been classified as NOT A THREAT

7. Results:

- Video1 (Masked men, gun): Classified as THREAT with high confidence.
- Video2 (Woman photographing church): Classified as NOT A THREAT.
- Detection Accuracy:
  - 3 of 4 Videos classified correctly
  - 1 False Positive
  - Overall Accuracy: 75%
- Log File Generated: Classification summary in results_log.txt.
- Alarm Triggered: Successfully for threat detection.

8. Challenges Faced:

- Pre-trained models ineffective in surveillance context due to lack of domain-specific classes.
- Dataset formatting inconsistencies from Roboflow; resolved via manual adjustments and YAML validation.
- Long training time due to CPU-only training; affected real-time application potential.
- Python environment conflicts (e.g., missing modules, path errors) resolved by strict venv usage.

9. Future Work:

- Integrate real-time detection using webcam feeds for live threat assessment.
- Leverage object tracking (e.g., BoxMOT) to maintain continuity across frames and reduce false positives.
- Expand dataset to include more threat classes (explosives, masks, restricted items).
- Deploy model in a lightweight containerized environment (e.g., Docker) for scalable deployment.

10. Conclusion:

This project demonstrates successful AI integration for threat detection using YOLOv8, custom training, and alert systems. It enhances physical security within cybersecurity by enabling automated, real-time surveillance. The system's alarm and logging components augment response efficiency, providing a blueprint for AI-driven security solutions.

Appendix:

- Dataset Source: Roboflow Weapons Dataset
- YOLO Version: Ultralytics YOLOv8m
- Training Logs: runs/detect/train/
- Detection Log File: results_log.txt
- Code Repo: Adithya1798/Video-Surveillance-Threat-Detection-Using-YOLOv8-and-Custom-Object-Detection-Models