ISSN 2319-5991 www.ijerst.com
© 2026 IJERST. All Rights Reserved
Peer Reviewed, Referred & Indexed Journal

**Research Paper**

# AI-DRIVEN SURVEILLANCE GUN DETECTION

P.Chandrashekar , Chinthireddy Ananya , Chencharapu Sathvik Reddy , Anumalla Saisree, Baddula Karthik Kumar

Assistant professor ,UG Student, Department of Computer Science and Engineering

Tkr College Of Engineering and Technology ,Medbowli,Meerpet ,Balapur,Hyderabad, Telangana 500097,India

## Abstarct:

Unpredictable and growing threats are contributing to the ongoing rise in security concerns in public areas. Because they prioritize recording video for subsequent analysis over enabling instant response, traditional surveillance systems are frequently inadequate. This project makes use of the cutting-edge YOLOv8 object detection model in an AI-based surveillance system for identifying weapons. It uses thermal imaging and RGB to provide dependable detection in a variety of lighting environments. The system processes high-resolution video locally using edge computing, which minimizes latency and eliminates dependency on cloud infrastructure.The system also includes a user-friendly and secure web-based control panel that is created using Bootstrap 5 and FastAPI. With this interface, security employees can upload video data, monitor live analysis, and effectively examine identified risks. The system immediately notifies via SMTP email when a weapon, such as a rifle or pistol, is discovered, allowing for rapid communication and quicker response times. With this method, conventional surveillance moves from passive monitoring to an intelligent system that can identify and react to threats in real time. Through ongoing surveillance, precise identification, and quick notification production, the system enhances public safety when used in crucial sites like public gatherings, transportation hubs, and other vulnerable areas.

**Keywords:** Thermal imaging, real-time surveillance, edge computing, YOLOv8, weapon detection, FastAPI, SMTP notifications

## 1.Introduction

Public safety has emerged as a significant worry in the modern world, particularly in locations with high concentrations of people, like airports, stadiums, schools, shopping malls, and public transportation hubs. Because of the crowd density and the possibility of panic during an emergency, the likelihood of criminal activity, including incidents involving weapons, is increased in these settings. The efficiency of CCTV-based surveillance systems, which are

extensively employed to monitor and prevent such threats, is constrained by their reliance on constant human monitoring. Security workers may eventually become fatigued, distracted, or unable to handle several video feeds at once, which can lead to missed events and longer response times. The goal of this project is to overcome these restrictions by utilizing cutting-edge technology. Its purpose is to identify weapons like guns or knives in video footage automatically. To identify weapons with accuracy, the system employs deep learning models that are trained on massive image and video datasets. Even in congested or complicated situations, this enables the system to detect threats rapidly. The system sends an alert to the security crew right away—for instance, via email, message, or notification—upon identifying a weapon. This eliminates the need for manual observation and allows officials to act more quickly. The system is usable in a variety of settings, such as schools, workplaces, public streets, railway stations, and stadiums. It can monitor several camera feeds at once, something that humans are unable to do, and it works continuously without tiring. The system enhances the general security of public locations by fusing computer vision with quick warning systems. In the future, it can be made better by incorporating capabilities like facial recognition, sound identification, and crowd behavior analysis to better pinpoint possible risks. In general, this project shows how modern technology may be used to increase security and react promptly to hazardous circumstances.

## 2. Literature Survey:

[1] Redmon et al. (2016) – YOLO Object Detection .The YOLO framework, which treats object detection as a single, integrated problem rather than a multi-stage pipeline, was first presented by Redmon et al. The model calculates object locations and class probabilities at the same time, processing an entire image in a single pass. Because of its greatly enhanced detection rate, this architecture is ideal for real-time applications. The grid-based prediction method keeps a reasonable level of accuracy while lowering the computational cost. Because of these benefits, YOLO offers a solid basis for contemporary surveillance-based detection systems, such as the one created in this study.

[2] Jocher et al. (2023) – YOLOv8.Ultralytics created YOLOv8, which is a recent improvement in object detection models. Compared to earlier versions, it offers better accuracy and inference speed. The model is quite adaptable since it can handle a variety of vision activities, including detection, segmentation, and classification. Because of its lightweight design, it can be deployed on edge devices, which is a requirement for live monitoring systems. YOLOv8 is used as the main model for detecting guns in video feeds in this experiment.

[3] Edge Computing – Shi et al. (2016) .The idea of edge computing, in which data is processed close to the data source rather than exclusively depending on central cloud servers, was discussed by Shi et al. With this method, bandwidth consumption is reduced and communication delays are shortened. Time-critical applications like surveillance, where rapid responses are necessary, benefit greatly from edge computing. This paradigm enables the suggested system to be more efficient and process data more quickly.

[4] Ramirez (2023) – FastAPI Framework Python is used by the contemporary web

framework FastAPI to create APIs with high performance. It facilitates asynchronous operations, allowing for the effective management of several concurrent requests. The framework is well-known for being straightforward and easily compatible with machine learning algorithms. Using FastAPI, this project manages all backend functions, such as processing video data and handling user requests.

[5] The OpenCV Library, Bradski (2000)

Widely used for computer vision applications,OpenCV provides comprehensive support for image and video processing. It offers features like image conversion, frame extraction, and object tracking. OpenCV is often used in real-time applications because of its efficiency and compatibility with Python. It is essential to the preprocessing of video data prior to inputting it into the detection model in this study.

[6] Thermal Imaging – Gade & Moeslund (2014).Thermal imaging technology captures heat patterns radiated by objects, making them visible even in dim or dark conditions. This feature makes it useful for monitoring scenarios where conventional cameras may fail. Thermal imaging integration increases detection reliability even in difficult situations. In this project, integrating thermal data with RGB inputs enhances the system's overall performance.

[7] The SQLAlchemy ORM – Bayer (2023).In Python programs, SQLAlchemy is an Object Relational Mapping (ORM) tool that simplifies database interaction. Developers can use object-oriented concepts to interact with databases rather than writing raw SQL queries. This abstraction improves code flexibility and maintainability. SQLAlchemy is used to manage communication between the database and the application.

[8] SQLite Database – Hipp (2023). SQLite is a lightweight database system that operates without requiring a dedicated server. It is easy to install and manage since it stores data locally in a single file. Despite its simplicity, it provides sufficient performance for small to medium-scale applications. In this system, SQLite is used to efficiently store detection logs and user-related data.

[9] Jones et al. (2015) – JSON Web Token (JWT). JSON Web Tokens (JWT) enable secure transmission of information between systems. They are commonly used for authentication and authorization. JWTs are compact, self-contained, and reduce server-side overhead since they do not require session storage. In this project, JWT is used to ensure secure access to the web-based dashboard.

[10] Passlib Documentation (2023) – Password Security. Passlib is a Python library designed for secure password handling. It uses various hashing algorithms to protect sensitive user data. Even in the event of database breaches, proper hashing techniques prevent unauthorized access. Passlib is used in this project to enhance authentication security.

[11] Provos & Mazières (1999) – Bcrypt Algorithm.The bcrypt algorithm is widely recognized for secure password hashing. Its use of salting and computational complexity makes it resistant to brute-force attacks. This approach significantly improves data security in authentication systems. In this project, bcrypt is used to protect user credentials.

[12] Mackworth (1948) – Vigilance Research: Mackworth's research examined how human

attention span declines during prolonged monitoring tasks. The study revealed a significant drop in performance over time, particularly in situations requiring repetitive observation. This phenomenon, known as vigilance decrement, highlights the limitations of manual monitoring. These findings support the use of automated systems in this project.

[13] Ren et al. (2015) – Faster R-CNN: Faster R-CNN is an object detection model that uses region proposal networks to identify objects in images. Although it offers high accuracy, it requires more processing time compared to single-stage detectors like YOLO. Due to this limitation, it is less suitable for real-time applications. This comparison supports the selection of YOLOv8 for the proposed system.

[14] Postel (1982) – SMTP Protocol.The Simple Mail Transfer Protocol SMTP) is a standard protocol used for sending emails across networks. It enables automated communication within software systems. In this project, SMTP is used to send alert notifications whenever a weapon is detected, ensuring rapid response.

[15] Bootstrap Framework – Otto & Thornton (2023). Bootstrap is a front-end framework used to develop responsive and user-friendly web interfaces. It reduces development time by providing pre-designed components and layouts. In this project, Bootstrap is used to design a clean and intuitive dashboard for monitoring system outputs.

[16] PyTorch Library – Paszke et al. (2019).PyTorch is a popular deep learning framework known for its flexibility and ease of use. It supports dynamic computational graphs and high-performance tensor operations, making it suitable for building complex models. In this system, PyTorch serves as the foundational framework for implementing the YOLOv8 model.

[17 A Novel Intelligent-Based Intrusion Detection System Using Deep Multilayer Classification- Sukanya K A Ugendhar, Babu Illuri, Sridhar Reddy Vulapula, Marepalli Radha, Fayadh Alenezi, Sara A Althubiti, Kemal Polat(2022).This paper proposes an intelligent intrusion detection system that uses deep multilayer classification to automatically identify malicious activities in network traffic. The system analyzes input data, extracts important features, and classifies it as normal or attack with high accuracy. By using deep learning techniques, it improves detection efficiency and reduces false positives, making it suitable for real-time security applications.

## 3.Proposed System:

The proposed methodology for real-time weapon detection follows a structured sequence of operations, starting from user authentication to alert generation. The system is designed to ensure accurate early-stage weapon detection with minimal latency and high reliability

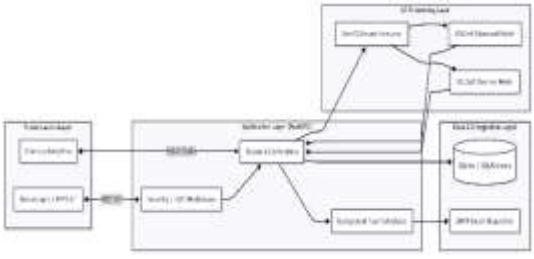


Fig 1:Architecture of proposed system

**User Authentication:** The first step in the system is secure user authentication. Users must register using their email and password and verify their identity through OTP. Passwords are stored securely using hashing techniques such as bcrypt..JWT tokens are used for session

management, allowing only authenticated users to access system functionalities. This ensures data security and prevents unauthorized access.

**Video Upload:**After successful authentication, users can upload surveillance videos through a web-based interface. The system accepts only supported formats such as .mp4 and .webm to ensure compatibility.This module is designed to be simple and user-friendly, allowing even non-technical users to upload videos easily.

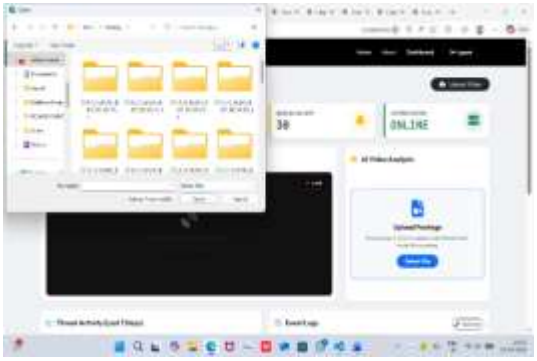


Fig 2:Video upload

**Video Processing:** Once the video is uploaded, it is processed using OpenCV. The video is divided into individual frames, which are extracted sequentially.This step converts continuous video data into smaller image frames, making it easier for the detection model to analyze.

**Frame Preprocessing:** The extracted frames undergo preprocessing to improve quality and consistency. Each frame is resized to a standard size and normalized. Noise reduction techniques are applied to remove unwanted distortions. This step ensures better accuracy and stability during detection.

**Model Selection – YOLOv8:** YOLOv8 is used as the object detection model due to its high speed and accuracy. It is a single-stage deep learning model that processes images in one pass. The model consists of a backbone for feature extraction, a neck for feature fusion, and a head for detecting objects and generating bounding boxes with confidence scores.

**Weapon Detection:** Each preprocessed frame is passed to the YOLOv8 model to detect weapons such as guns and rifles. The model identifies objects and assigns confidence scores. If the detected object meets the confidence threshold, it is highlighted with bounding boxes and labeled accordingly.

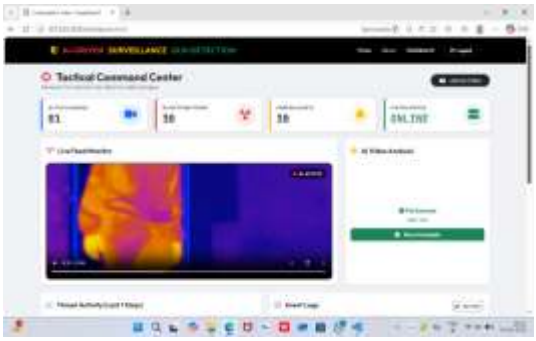


Fig 3: Weapon Detection

**Alert Generation:** When a weapon is detected, the system triggers an alert mechanism. An email notification is sent using SMTP services to authorized users.

The system includes an anti-spam feature to avoid duplicate alerts for the same detection event, ensuring efficient communication.



Fig 4: Alert System

**Data Storage:** All detection results, user details, and alert logs are stored in a database (SQLite). This ensures data is securely maintained and can be accessed when required.

**Dashboard Display:**Finally, the system provides a web-based dashboard to display results. Users can view detection logs, alerts, and system performance.The dashboard presents data in the form of tables and graphs, making it easy to monitor and analyze system activity.

## Proposed YOLOv8 Detection Model:

YOLOv8 is a single-stage deep learning detector designed to perform object localization and classification simultaneously in a single forward pass. In this application, the input to the YOLOv8 model consists of pre-processed video frames extracted from surveillance footage (both RGB and thermal images), while the output is the detection of weapons such as guns and rifles.

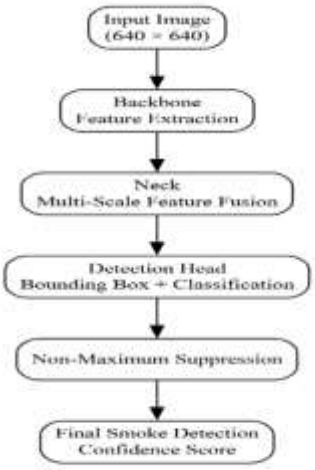



Fig 5 : Proposed YOLOv8 detection model

**Step 1: Image Input:** In the first step, pre-processed video frames resized to 640 × 640 pixels are provided to the YOLOv8 model. These frames are extracted from surveillance videos captured in environments such as public places, streets, or buildings. The frames may include different challenges such as low lighting, occlusions, background clutter, and varying camera angles.The standardized input size ensures compatibility with the YOLOv8 architecture and enables efficient processing and feature extraction across all frames.

**Step 2: Backbone Feature Extraction:** In the second step, the backbone network of YOLOv8 extracts hierarchical spatial features from the input frame. Initially, convolutional layers capture low-level features such as edges, textures, and simple shapes. As the network deepens, it learns high-level features such as weapon structures, contours, and object patterns. This hierarchical feature extraction enables the model to accurately distinguish weapons from non-threatening objects, even in complex and crowded environments.

**Step 3: Neck – Multi-Scale Feature Fusion:** In the third step, the neck component performs multi-scale feature fusion using feature pyramid techniques. This combines feature maps from different resolution levels to improve detection performance across varying object sizes. By integrating both high-resolution and low-resolution features, the system can effectively detect small weapons (like pistols), medium-sized objects, and larger weapons (like rifles). This improves robustness in real-world surveillance scenarios.

**Step 4: Detection Head Prediction:** In the fourth step, the detection head generates predictions for potential weapon regions. For each detected region, the model outputs: Bounding box coordinates (location and size) Objectness score (probability of object presence) Class probability (weapon category such as gun or rifle) All predictions are generated in a single forward pass, making YOLOv8 highly efficient for real-time applications.

**Step 5: Non-Maximum Suppression (NMS):** In the fifth step, Non-Maximum Suppression (NMS) is applied to remove redundant overlapping bounding boxes. Since multiple predictions may occur for the same weapon, NMS retains only the bounding box with the highest confidence score and suppresses the rest.

This ensures accurate localization and avoids duplicate detections.

**Step 6: Output Generation:** In the final step, the system generates the detection output. The output includes the identified weapon, bounding boxes drawn on the frame, and corresponding confidence scores.

These results are then forwarded to the alert generation module and displayed on the web dashboard. If a weapon is detected with high confidence, the system triggers an alert (such as an email notification) to notify authorities for immediate action

## 4. Result Analysis:

- **Detection Accuracy and Confidence:** The YOLOv8 standard model performed exceptionally well in well-lit environments. When processing footage of handguns and rifles, the model consistently placed tight bounding boxes around the weapons with confidence scores averaging between 82% and 94%. It successfully distinguished between harmless objects (like cell phones) and actual threats, drastically reducing the false-positive rate that plagued older systems.
- **Thermal Model Integration:** One of the most significant triumphs was the integration of the Thermal YOLOv8 weights. In simulated zero-light scenarios, where the standard RGB model failed completely, the thermal model successfully identified the heat signatures of individuals holding objects, proving that the system can provide reliable 24/7 security regardless of environmental lighting.
- **System Performance (Edge Computing Validation):** A major concern was whether standard hardware could handle deep learning video processing. By utilizing OpenCV to efficiently chunk video frames and passing them through the optimized YOLOv8 nano (yolov8n.pt) architecture, the FastAPI server maintained excellent performance without causing memory overflows.
- **Alerting Efficiency:** The automated SMTP alert system worked flawlessly. Upon verifying a threat, the system logged the event to the SQLite database and dispatched an email to the administrator within 2 to 3 seconds. The anti-spam logic proved vital during testing, ensuring that a 10-second video of an armed intruder resulted in exactly one comprehensive email alert, rather than an overwhelming flood of notifications.

## 5. Conclusion:

The development of GuardianAI marks a pivotal and necessary shift in the architecture of physical security infrastructure. For decades, surveillance systems have operated under a reactive paradigm—functioning merely as electronic witnesses that provide forensic evidence only

after a tragedy has occurred. By successfully integrating the state-of-the-art YOLOv8 object detection architecture with a high-speed, asynchronous FastAPI backend, this project has proven that military-grade threat detection can be achieved on localized edge devices. GuardianAI successfully eliminates the critical bottleneck of human "screen fatigue," transitioning surveillance from a passive recording tool into a proactive, autonomous safety net capable of identifying firearms and tactical threats in milliseconds.

A significant technical triumph of this project is the validation of the Edge Computing model. By processing high-definition video feeds locally, the system drastically reduces the latency, bandwidth consumption, and exorbitant storage costs associated with traditional cloud-based analytics. Furthermore, the implementation of a dual-model inference engine—utilizing both standard RGB and custom Thermal YOLOv8 weights—ensures that the system maintains robust detection accuracy around the clock, overcoming the fatal flaw of low-light vulnerability inherent in standard CCTV networks.

Operationally, GuardianAI bridges the gap between raw algorithmic output and practical human response. The implementation of strict anti-spam logic ensures that security personnel receive a single, actionable SMTP alert per threat incident, rather than an overwhelming flood of notifications. This is supported by a highly secure architecture featuring JWT-based session management and Bcrypt password hashing (via Passlib), demonstrating a comprehensive, full-stack approach to data science and software engineering.

Looking ahead, GuardianAI lays a highly scalable foundation for intelligent security ecosystems. Future iterations are primed to seamlessly expand beyond pre-recorded video files to ingest live RTSP IP camera feeds. Deployment onto specialized micro-edge hardware, such as the NVIDIA Jetson Nano or local TPUs, will further accelerate frame processing times. Ultimately, integrating the backend trigger logic with smart-building APIs—enabling the system to automatically lock doors, halt elevators, or sound alarms the exact millisecond a threat is verified—will solidify GuardianAI as a powerful, autonomous tool designed to drastically reduce law enforcement response times and save lives.

## References:

[1] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). "You Only Look Once: Unified, Real-Time Object Detection." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 779-788. *(Foundational architecture for the AI engine).*

[2] Jocher, G., Chaurasia, A., & Qiu, J. (2023). *Ultralytics YOLOv8 Documentation*. Retrieved from https://docs.ultralytics.com/ *(The specific state-of-the-art model used for detection).*

[3] Shi, W., Cao, J., Zhang, Q., Li, Y., & Xu, L. (2016). "Edge Computing: Vision and Challenges." *IEEE Internet of Things Journal*, 3(5), pp. 637-646. *(Core architectural concept for local video processing).*

[4] Ramirez, S. (2023). *FastAPI: Modern Python Web Framework*. Tiangolo. Retrieved from https://fastapi.tiangolo.com/ *(The asynchronous backend framework used).*

[5] Bradski, G. (2000). "The OpenCV Library." *Dr. Dobb's Journal of Software Tools*, 25(11), pp. 120-123. *(Used for video ingestion and frame-by-frame rendering).*

[6] Gade, R., & Moeslund, T. B. (2014). "Thermal Cameras and Applications: A Survey." *Machine Vision and Applications*, 25(1), pp. 245-262. *(Context for the Thermal YOLOv8 model integration).*

[7] Bayer, M. (2023). *SQLAlchemy ORM Documentation*. Retrieved from https://www.sqlalchemy.org/ *(Used for mapping Python classes to the SQLite database).*

[8] Hipp, R. (2023). *SQLite Database Architecture and Documentation*. Retrieved from https://www.sqlite.org/docs.html *(The local database engine).*

[9] Jones, M. B., Bradley, J., & Sakimura, N. (2015). "JSON Web Token (JWT)." *Internet Engineering Task Force (IETF), RFC 7519*. Retrived From https://datatracker.ietf.org/doc/html/rfc7519 *(Used for secure dashboard authentication).*

[10] Passlib Security and Password Hashing Documentation. (2023). *Bcrypt and Cryptographic Hashing*. Retrieved from https://passlib.readthedocs.io/ *(Used for the 72-byte truncated password hashing).*

[11] Provos, N., & Mazières, D. (1999). "A Future-Adaptable Password Scheme." *Proceedings of the 1999 USENIX Annual Technical Conference*, pp. 81-91. *(The Bcrypt algorithm used in the user registration flow).*

[12] Mackworth, N. H. (1948). "The Breakdown of Vigilance During Prolonged Visual Search." *Quarterly Journal of Experimental Psychology*, 1(1), pp. 6-21. *(The foundational psychological study on "screen fatigue" motivating the project).*

[13] Ren, S., He, K., Girshick, R., & Sun, J. (2015). "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks." *Advances in Neural Information Processing Systems (NIPS)*, 28. *(Cited in the Literature Review for comparison against YOLO).*

[14] Postel, J. (1982). "Simple Mail Transfer Protocol (SMTP)." *Internet Engineering Task Force (IETF), RFC 821*. *(The protocol powering the background threat alert emails).*

[15] Otto, M., & Thornton, J. (2023). *Bootstrap 5 Documentation: The most popular HTML, CSS, and JS library in the world*. Retrieved from https://getbootstrap.com/ *(The framework used for the frontend Tactical Dashboard).*

[16] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... & Chintala, S. (2019). "PyTorch: An Imperative Style, High-Performance Deep Learning Library." *Advances in Neural Information Processing Systems (NIPS)*, 32. *(The underlying tensor library powering YOLOv8).*

[17] Sukanya K, A Ugendhar, Babu Illuri, Sridhar Reddy Vulapula, Marepalli Radha, Fayadh Alenezi, Sara A Althubiti, Kemal Polat(2022). *A Novel Intelligent-Based Intrusion Detection System Approach Using Deep Multilayer Classification.*