ARCIS: Autonomous Real-time Critical Infrastructure Security

# Abstract

ARCIS (Autonomous Real-time Critical Infrastructure Security) is an AI-driven   
weapon detection platform designed to enhance safety in critical infrastructures such as airports, schools, power plants, and government buildings. The system integrates real-time computer vision models deployed on edge devices (Raspberry Pi and Jetson Nano) with a web-based monitoring dashboard that enables security personnel to visualize, analyze, and respond to threats in real time.   
ARCIS leverages YOLO-based deep learning algorithms to detect weapons, classify threat levels, trigger alarms, and store evidence frames. The accompanying ARCIS website provides advanced analytics, historical data visualization, system health monitoring, and role-based access management.

# 1. Introduction

Ensuring safety in critical infrastructures requires proactive surveillance and   
rapid response to threats. Conventional systems rely heavily on manual monitoring of CCTV feeds, which is error-prone and inefficient, especially under fatigue. ARCIS addresses these limitations through an autonomous, AI-powered framework that combines edge computing with cloud-based monitoring.   
  
The ARCIS project includes two main components:   
1. ARCIS Detection System – A YOLO-based real-time object detection model deployed on   
 lightweight edge devices (Raspberry Pi 4 and Jetson Nano) connected to surveillance cameras. The YOLO model is being stored on an outside VM with its own GPU to run it and transmits back whether a detection occurred (or not).   
2. ARCIS Website – A centralized monitoring platform for security teams to analyze detections, access historical analytics, manage user roles and monitor system health metrics.   
  
Together, these components allow for instant local responses and centralized situational awareness across multiple facilities. This dual-layered approach ensures   
both responsiveness and strategic oversight, empowering security operators with actionable insights.

# 2. Background

The increasing frequency of public safety threats highlights the importance of   
smart surveillance solutions. Conventional CCTV monitoring requires constant human supervision,   
which is unsustainable in high-security environments. Recent advances in deep learning, especially   
with architectures such as YOLO (You Only Look Once), have made it possible to detect and classify   
objects, including weapons, with remarkable speed and accuracy.   
  
ARCIS builds upon this by combining YOLO’s efficiency with edge computing and cloud technologies.   
Edge devices allow low-latency, real-time response at the source of detection, while cloud integration   
enables large-scale analytics and centralized data management. The ARCIS website bridges the gap between   
local detection and centralized monitoring, ensuring data from multiple devices is aggregated,   
analyzed, and made actionable.   
  
This integration of AI, IoT, and cloud computing positions ARCIS as a modern, scalable security   
solution for protecting critical assets and populations.

# 3. Methods

ARCIS employs a multi-layered architecture:   
  
3.1 Detection System:   
- Model Training: The YOLOv5/YOLOv8 family was trained on a curated dataset from Roboflow with ~25,000 labeled images across three classes: person, weapon, and person holding weapon.   
- Edge Deployment: Trained weights are deployed onto Raspberry Pi and Jetson Nano devices.   
- Inference Pipeline: Live video feeds are analyzed frame-by-frame and processed on the VM server side. Communication is handles via HTTP requests on a TCP port. detections with confidence scores above a threshold trigger bounding boxes and alarms.   
- Alarm Mechanism: Using Pygame, edge devices issue local audio alerts to notify the user instantly of detected threats.   
  
3.2 ARCIS Website:   
- Frontend: Developed with React (TypeScript) and Chakra UI, providing a responsive, accessible, and mobile-friendly design.   
- Backend: Node.js API with PostgreSQL database supports detection data storage, retrieval, and analytics.   
- Authentication: Firebase Authentication with role-based access controls ensures secure login, account management, and domain-restricted usage.   
- Features:   
 - Live dashboards showing active threats with confidence levels.   
 - Analytics and visualizations of historical detections, classified by weapon type and location.   
 - System performance monitoring (CPU, GPU, RAM usage).   
 - Tools for manual entries, false-positive deletions, and operator comments.   
  
3.3 Integration:   
Edge devices forward detection metadata (class, timestamp, confidence, frame snapshot) to the backend, where it is aggregated and made visible to operators via the web interface. This ensures local fast responses as well as centralized, coordinated monitoring across facilities.

# 4. Experiments and Results

4.1 Training and Validation:   
- Dataset: ~250,000 annotated images.   
- Training: Conducted over 50 epochs using YOLOv8n.   
- Metrics: mAP@0.5 reached ~85%, with strong recall for weapons. Some misclassification occurred with small objects in cluttered environments, highlighting dataset limitations.   
  
4.2 Edge Device Performance:   
- Raspberry: Achieved stable 30 FPS.   
- Jetson: Achieved 10-15 FPS with TensorRT acceleration, enabling near real-time performance.   
- Google Cloud GPU (Tesla T4): Undetermined at this point, could easily run and analyze both devices twice although up to a certain limit. Scaling up the system would require either upgrading the VM GPU, adding more VM that could communicate with each other or attaching a modern GPU to the edge devices.   
  
4.3 Website Functionality:   
- Secure Firebase login tested with multiple roles.   
- Dashboard successfully updated with real-time detections and analytics.   
- Historical records were visualized with clear detection trends.   
- System health monitoring was validated, accurately reporting device usage metrics.   
  
Results indicate that the system can provide immediate alerts in local deployments while scaling effectively through cloud GPUs for enterprise use cases.

# 5. Discussion

Our system demonstrates the feasibility of merging edge computing with centralized   
monitoring for critical infrastructure protection.   
  
Strengths:   
- Immediate local detection and response through alarms.   
- Centralized monitoring platform for multi-site awareness.   
- Scalable design with compatibility across Raspberry Pi, Jetson Nano, and cloud GPUs.   
- User-friendly dashboard designed for non-technical security staff.   
  
Limitations:   
- Scaling up the system would require a small restructure of the communication system, so far its limited to the analytic capabilities of 1 cloud server.   
- False positives remain an operational challenge; retraining with more diverse data could be necessary.   
- Reliance on stable internet connections may reduce system reliability in some scenarios.

# 6. Conclusion

ARCIS provides an integrated and dual-layered solution to modern security challenges.   
By combining AI-based real-time detection with a monitoring dashboard, it enables both local immediacy and centralized oversight. The system illustrates that low-cost devices like Raspberry Pi   
can act as effective capture clients, while GPUs in the cloud or Jetson Nano devices perform   
higher-throughput inference.

# 7. Future Work

ARCIS can be improved and expanded in several ways:

- Implement GPS chips to improve communication options between devices and handle cases of friendly troop recognition via depth perception.

- Implement 2.9 inch LCD screens attached to the user’s helmets to better display and visualize the location of incoming threats.

- Add to each individual system a controller such as GPIO button or a USB switch to better control the activation of the camera script from the raspberry.  
- Dataset Expansion: Adding more weapon types, clothing variations, and environmental conditions.   
- Model Optimization: Applying pruning, quantization, and knowledge distillation for lightweight deployments on Raspberry Pi.   
- Advanced Cloud Deployment: Either building automated pipelines with Google Cloud for elastic scaling or attaching more capable GPUs  
- Sensor Fusion: Integrating acoustic sensors for gunshot detection or thermal cameras for low-light environments.   
- Mobile Applications: Extending dashboards to mobile apps with push notifications for real-time alerts.   
- Interoperability: Designing APIs for integration with existing security information systems (SIS) and emergency response platforms.   
  
Future iterations of ARCIS aim to reduce latency, increase accuracy, and improve accessibility for security teams across diverse environments as well as implementing further utilities that will broaden the use cases of the system and its reliability.