12

Experiences in Deploying a Weapon Detector in a Smart City

Juan Daniel Muñoz[1], Hugo Albandea Merino[1], Jesus Ruiz-Santaquiteria[1], Oscar Deniz[1], and Micaela Verucchi[2]

[1]VISILAB, University of Castilla-La Mancha, Spain
[2]Hipert Srl., Italy

Abstract

This work presents the ongoing development of a handgun detection system for a smart city. This detector is expected to continuously analyse images taken by a surveillance camera installed at the entrance hall of a public building. The detector is deployed on a Jetson Orin Nano-based device with constrained computational resources. The development process began with an analysis of hardware limitations, which guided the design of lightweight yet effective deep learning models. Several processing pipelines were considered: a two-stage approach involving person detection followed by hand-region classification, and a direct handgun detection approach. Custom datasets were built and used to train models adapted to the device, while carefully avoiding overfitting. Performance was systematically evaluated using metrics such as mean Average Precision (mAP) and frames per second (FPS). The best trade-off between accuracy and inference speed was obtained with a YOLOv8s model trained exclusively for handgun detection, achieving 75.15% AP50 on the test set and sustaining 20 FPS. The detector was further integrated with MQTT for seamless communication with ThingsBoard, enabling the automatic transmission of detection events. Additional necessary work included

fully headless execution and real-time parameter adjustment through terminal commands. Current work focuses on refining detection performance and exploring alternative models to further improve efficiency and accuracy.

Keywords: weapon detection, edge AI, deep learning, real time, computer vision, MQTT.

12.1 Introduction and Background

Gun violence remains a major threat to public safety, especially in crowded urban areas where rapid intervention is critical [1]. Detecting firearms in real time has thus become a key objective of intelligent surveillance systems. Traditional methods rely on controlled screening environments (such as baggage inspection or millimetric wave scanners) where image acquisition is highly structured [2]. In contrast, open-scene detection introduces challenges such as varying illumination, occlusions, and limited computational capacity at the edge.

Recent advances in deep learning have significantly improved visible firearm detection in CCTV footage, enabling reliable recognition in unconstrained environments [3]. Mehta et al. [4] proposed a multi-purpose YOLOv3-based system capable of detecting both guns and fire in real time, processing video streams at 45 frames per second, and demonstrating high robustness across several datasets, achieving a maximum accuracy of 89.3%. Similarly, Bhatti et al. [5] compared multiple deep learning detectors, including VGG16, Faster R-CNN, and YOLOv4, and found YOLOv4 to deliver the best performance for handgun identification in surveillance videos, reaching an mAP of 91.72%.

Despite their accuracy, these appearance-based methods often produce false alarms, motivating pose-aware approaches. One study [6] showed that integrating body pose information into CNN detectors, such as RetinaNet and YOLOv3, reduces false positives and improves precision (YOLOv3 achieved a precision of 96.23%). Building on this idea, another work [7] incorporated a full-body pose classifier to distinguish threatening from non-threatening postures, further enhancing reliability in real-world scenes.

Beyond visible imagery, researchers have also addressed concealed weapon detection. Khan et al. [8] introduced a real-time 3D radar imaging framework using a modified U-Net to localize hidden weapons regardless of orientation, achieving both accuracy and speed suitable for security checkpoints. Complementary work combined thermal imaging and deep learning

in a two-stage pipeline for wearable devices [9], enabling mobile detection of firearms with reduced false positives and practical real-time performance. The method achieved an mAP@50–95 of 64.52% on a custom thermal dataset.

Nevertheless, most of these methods remain confined to laboratory settings. The present work advances this field by focusing on the real-time deployment of an efficient handgun detection system on an edge device integrated into a communication framework for automated event reporting. This direction aligns with ongoing efforts in smart city research, such as the CLASS project [10], which developed an edge–cloud analytics framework validated in the Modena Automotive Smart Area (MASA); the HAura platform [11], which supports privacy-preserving inference on the edge; and the AI-CAM initiative [12], which promotes cooperative, infrastructure-assisted perception. Together, these developments provide the technological foundation for the real-world implementation described in this work, to be tested within the MASA environment as part of ongoing experimentation in intelligent urban surveillance.

12.2 MASA and the HAura system

Since 2018, the University of Modena and Reggio Emilia (UNIMORE) and the Municipality of Modena have been jointly developing the Modena Automotive Smart Area (MASA). MASA is both an infrastructure and a physical test area of approximately 2 km^2 within the city of Modena (see Figure 12.1). It was conceived as a living laboratory for smart city technologies, designed to generate actionable data that can enhance urban mobility, sustainability, and safety. The initiative builds upon the expertise of the HiPeRT Lab, a research group within the Department of Physics, Computer Science, and Mathematics (FIM) of UNIMORE, directed by Prof. Marko Bertogna. The Lab has established a strong track record in high-performance real-time computing on embedded platforms, particularly in the domain of autonomous driving. In 2020 gave rise to Hipert s.r.l., which continues to industrialise and scale these technologies, focusing on autonomous robotics. MASA thus represents a natural extension of this research, where the smart city itself becomes a technological enabler and support system for connected and automated mobility.

A central technological element within MASA is the HAura system, developed and industrialised by Hipert s.r.l. HAura is a modular and distributed platform for detection, analysis, and event management in real time (see Figure 12.2). Its architecture is composed of multiple HAura Edge units

Figure 12.1 Modena Automotive Smart Area (MASA).

deployed in the urban environment. Each Edge device integrates dual RGB cameras, GPS, and a high-performance embedded computing board, enabling on-site execution of AI-based detection and tracking of different categories of road users. Anonymized metadata (including position, class, velocity, and direction) are generated and transmitted with minimal latency. The HAura Aggregator then consolidates data from multiple Edge units, removes redundancies, and provides a unified geo-referenced view of the monitored area. Importantly, the Aggregator incorporates predictive models capable of forecasting potential collisions and disseminating warnings in real time. At the system boundary, On-Board Units (OBUs) act as recipients of these alerts, which may correspond to connected vehicles or autonomous vehicles. In this perspective, the OBU's perception is extended by the "eyes" of the city infrastructure, thereby augmenting situational awareness and safety.

The design of the MASA and the HAura emphasizes low-latency operation, with end-to-end communication times of less than 100 milliseconds

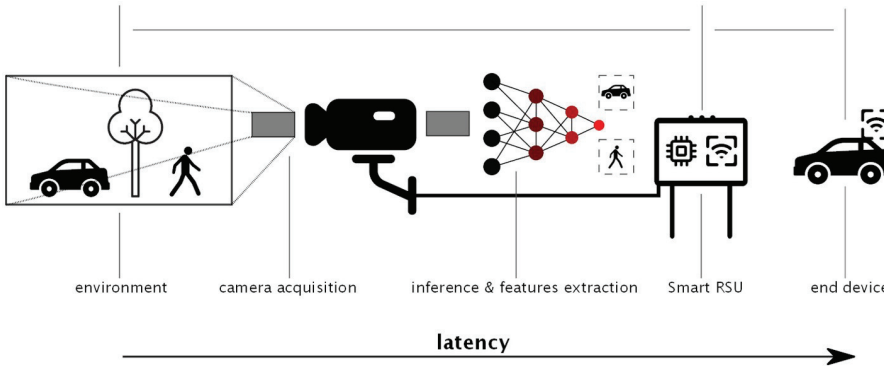

Figure 12.2 Overview of the HAura edge-to-end architecture within MASA, showing the data flow from camera acquisition to inference, smart RSU processing, and end-device communication with increasing latency [12].

between the detection of a road user and the reception of the corresponding alert by an OBU, within the MASA. Such real-time guarantees are essential for safety-critical urban applications. Furthermore, the system is designed with scalability and extensibility in mind. Each HAura Edge unit relies on a general-purpose computing board (based on NVIDIA's Jetson Orin Nano), enabling the deployment of additional software packages and future functionalities. The integration of Over-The-Air (OTA) updates, which now cover both application-level software and the operating system, ensures that the installed infrastructure can be securely and continuously upgraded. As such, MASA provides a replicable model of a smart city platform, capable of hosting advanced services such as smart parking, traffic management, environmental monitoring, and support for cooperative, connected, and automated mobility.

12.3 Dataset Creation

The development of the weapon detection system began with the construction of a dedicated dataset. Images were collected from a wide variety of sources, including publicly available internet repositories, video material from YouTube, frames extracted from commercial video games, and video recordings produced in the VISILAB laboratory at the University of Castilla-La Mancha (UCLM). Particular attention was given to acquiring images that resembled the perspective of a surveillance camera, typically positioned at a medium distance and slightly elevated relative to the subjects. This

Figure 12.3 Test Set images comparison. Left: test set A; middle: test set B; right: test set C.

choice was motivated by the intended application of the system in smart city environments (in MASA).

In order to increase the size and variability of the dataset, several data augmentation techniques were applied. These included geometric transformations such as rotations, as well as photometric adjustments to simulate different environmental and acquisition conditions. The initial dataset contained 7,783 images. After augmentation, this number increased to 46,632, which was subsequently divided into 37,308 images for training and 9,324 for validation. Images and data augmentation techniques presented in [13] were used.

Furthermore, to develop and evaluate a region classifier, the bounding boxes (BBoxes) corresponding to each annotated weapon instance were cropped and stored as independent images. This procedure produced an additional dataset consisting of 54,180 images for training and 11,610 for validation.

To assess model generalisation, a distinct test set of 310 images was assembled, referred to as *Test Set A*. To evaluate robustness under different visual conditions, two derivative sets were also created: *Test Set B*, which consists of the same images as Test Set A but with reduced brightness (simulating low light), and *Test Set C*, generated by simulating greater distance between the camera and the subject. Examples of these sets are shown in Figure 12.3. All quantitative results presented later in this article are computed with respect to these three test sets.

12.4 Architectures and Experimental Platform

Several deep learning architectures were employed in this work, selected for their balance between accuracy and computational efficiency, with consideration for the constraints of embedded platforms.

For object detection, we adopted YOLOv8s (You Only Look Once, version 8, small variant), a single-stage detector that predicts bounding boxes and class probabilities in one forward pass [14]. This design achieves higher inference speed than two-stage methods while maintaining competitive accuracy. The small variant of YOLOv8 [15] offers an effective trade-off between performance and computational cost, making it suitable for real-time inference on the Jetson Orin Nano.

To estimate human poses, we evaluated pose-based YOLO models and MediaPipe Pose. Pre-trained YOLOv8n, YOLOv11n, and YOLOv11x [16] models were compared to assess the trade-off between model size, accuracy, and latency. The smaller (n) variants prioritize efficiency, whereas the larger v11x model provides greater precision at higher computational cost. In parallel, we experimented with MediaPipe Pose [17] a lightweight two-stage framework predicting 33 body keypoints. It is specifically optimized for real-time applications on CPUs and mobile-class devices, requiring significantly fewer computational resources than conventional deep learning–based detectors. This made it a valuable baseline to compare against more computationally demanding architectures, particularly in scenarios where inference latency is critical.

For weapon classification within cropped hand regions, two convolutional neural networks were explored: EfficientNet [18] and MobileNetV2 [19]. EfficientNet employs compound scaling to balance depth, width, and resolution, achieving high accuracy with reduced complexity. MobileNetV2, based on depthwise separable convolutions and inverted residuals, offers even greater efficiency and is particularly suited to real-time edge applications, albeit with slightly lower accuracy.

All experiments were performed on the NVIDIA Jetson Orin Nano Developer Kit [20], integrating an Ampere GPU with 1,024 CUDA cores and 32 Tensor Cores, and 8 GB of LPDDR5 memory.

12.5 Detector and Communication Workflow

Two detection strategies were developed to address real-time weapon recognition under the computational constraints of the target hardware. Each was first implemented as an independent prototype and later adapted for integration into the final system. The first approach followed a single-stage object detection pipeline using YOLOv8s applied directly to full camera frames. This configuration aimed to localize and classify weapons in one step, maximizing frame rate on the Jetson Orin Nano. The small (s) variant

offered an optimal balance between accuracy and efficiency, preventing the detector from monopolizing system resources required for concurrent smart city processes. The second approach implemented a two-stage pipeline to improve robustness and reduce false positives. A pose-based model (either a YOLO-based pose detector (v8n, v11n, v11x) or MediaPipe Pose) was first used to locate individuals and extract keypoints. Based on these keypoints, regions around the wrists and hands were cropped and analysed by a classifier (either EfficientNet or MobileNetV2). The former achieved higher accuracy, while the latter offered lower latency, enabling flexible trade-offs between precision and computational cost. This design explicitly linked detected weapons to human subjects, minimizing erroneous detections on background objects, albeit at a reduced frame rate compared to the single-stage method.

Both pipelines were developed and tested locally on a Jetson Orin Nano Developer Kit, enabling rapid prototyping before remote deployment. For the latter, the system was first adapted for headless operation via secure SSH access. Runtime parameters could be adjusted dynamically through a curses-based terminal interface, ensuring flexible configuration without restarting services. Moreover, upon detecting a weapon, the system generates a structured JSON message containing detection metadata (timestamp, confidence, device ID) and publishes it via MQTT to a ThingsBoard broker. This enables reliable, real-time transmission and logging of smart city events for visualization, analytics, while decoupling edge processing from backend services for improved scalability and resilience.

Both detection strategies were encapsulated within Docker containers to ensure portability, reproducibility, and consistent performance across Jetson devices and deployment environments.

12.6 Training Process

The training process was designed to maximize detection and classification performance while minimizing overfitting. To this end, extensive experimentation was carried out with key hyperparameters, primarily the number of epochs, the learning rate, and the batch size. Hyperparameter values were adjusted iteratively, guided by validation accuracy and average precision (AP) on the test sets. This approach ensured that improvements in training performance did not come at the expense of model generalisation. The main hyperparameters selected can be seen in Table 12.1.

Table 12.1 Hyperparameters chosen to train the models used in the experiments

Trained Model	Epochs	Batch size	Input Resolution (pixels)	Learning rate
Single-stage detector	30	16	640x640	10^{-2}
Region classifier	30	16	224x224	10^{-6}

12.7 Results

The evaluation of object detection models typically relies on two complementary aspects: accuracy and efficiency. In this work, accuracy is measured using Average Precision (AP), while efficiency is assessed in terms of inference speed, expressed as frames per second (FPS).

To make AP meaningful, several fundamental concepts must first be defined [21].

- Ground truth. Human-annotated bounding boxes specifying object locations and class labels. These serve as the reference for evaluating detections.
- True Positive (TP). A detection correctly matching a ground-truth object, determined by the Intersection over Union (IoU): the overlap area between a predicted box B_p and a ground-truth box B_{gt} divided by their union,

$$IoU = \frac{\text{area}\left(B_p \cap B_{\text{gt}}\right)}{\text{area}\left(B_p \cup B_{\text{gt}}\right)} \tag{12.1}$$

A prediction is considered a TP if its IoU exceeds a predefined threshold (e.g., 0.5), following standard one-to-one matching rules.

In addition to IoU, we also employed the Intersection over Minimum area (IoMin) metric [22]. IoMin is defined as:

$$\text{IoMin}\left(B_p, B_{\text{gt}}\right) = \frac{B_p \cap B_{\text{gt}}}{\min\left(\text{Area}\left(B_p\right), \text{Area}\left(B_{\text{gt}}\right)\right)} \tag{12.2}$$

Unlike IoU, which penalises size discrepancies, IoMin produces high scores when the predicted box is fully contained within the ground-truth region, even if their sizes differ substantially.

This distinction is crucial in our pose-based detection setup, where hand regions are cropped using fixed-size bounding boxes around detected keypoints. Such crops may include irrelevant background or omit parts of the forearm, leading to artificially low IoU values even when the handgun is correctly captured. IoMin mitigates this issue by prioritising whether the

object of interest (the handgun) lies within the predicted region, aligning with our operational goal of early weapon detection in surveillance scenarios. Figure 12.4 illustrates the difference between IoU and IoMin.

- False Positive (FP). A predicted box with no sufficient overlap with any ground-truth object, or a duplicate detection for an already matched instance.
- False Negative (FN). A ground-truth object not detected by the model.
- True Negative (TN). Not typically defined in object detection, as the set of possible negative boxes is effectively infinite; evaluation therefore focuses on TP, FP, and FN.

Given the counts above, precision and recall quantify two complementary aspects of detector behaviour:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{12.3}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{12.4}$$

Precision measures the proportion of correct detections (penalising false alarms), while recall measures the proportion of detected ground-truth objects (penalising misses). A strict detector tends to have high precision but low recall, whereas a permissive one exhibits the opposite.

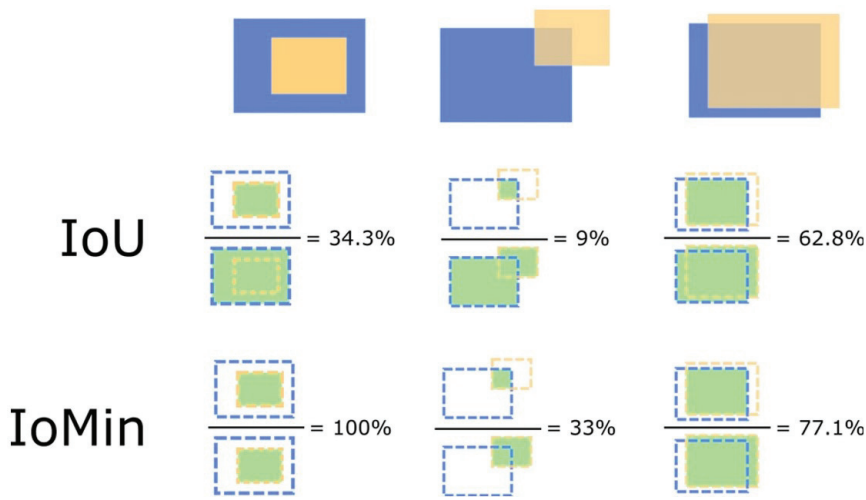

Figure 12.4 Examples of IoU and IoMin values for 3 different overlaps between detection and ground truth [22].

By varying the detection confidence threshold, a precision–recall (PR) curve is obtained. Average Precision (AP) is then computed as the area under the PR curve, summarising the trade-off between precision and recall across all thresholds (see Equation (12.5)).

$$AP = \int_0^1 \text{Precision}(R)dR \tag{12.5}$$

where R represents recall.

When multiple classes are involved, mean Average Precision (mAP) is reported as the arithmetic mean of per-class AP values (see Equation (12.6)).

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{12.6}$$

where AP_i is the average precision for class i.

In this study, only one class ("Handgun") is evaluated, so mAP reduces to a single AP value.

In addition to accuracy metrics, detector speed is a critical factor. A low frame rate (FPS) can cause frames to be skipped, risking missed detections if an armed individual passes quickly through the camera's view. The required FPS thus depends on the subject's movement speed: faster motion demands higher FPS to ensure at least one frame captures the weapon.

To establish a realistic reference, we analysed surveillance footage from the 2023 Nashville school shooting, measuring how long the weapon remained visible across three camera views (examples of these views are shown in Figure 12.5). From these durations, we estimated the minimum FPS necessary to guarantee at least one processed frame per scene, with results presented in Table 12.2.

This criterion only guarantees a single processed frame during the visibility window. In practice, detectors require multiple frames with the weapon visible. A more realistic requirement is therefore $\text{FPS}_{min}^{[f_0]} = \text{N/T}$, where N is the required number of frames (e.g., 5–10). For instance, with T = 5s in

Table 12.2 Duration of weapon visibility in each camera view during the Nashville school shooting (2023) and the corresponding absolute minimum FPS required to ensure at least one processed frame contains the handgun.

Camera view	**Visibility T(s)**	**Minimum FPS (1/T)**
Entrance Hall	5	0.2
Corridor	27	0.037
Lobby	10	0.1

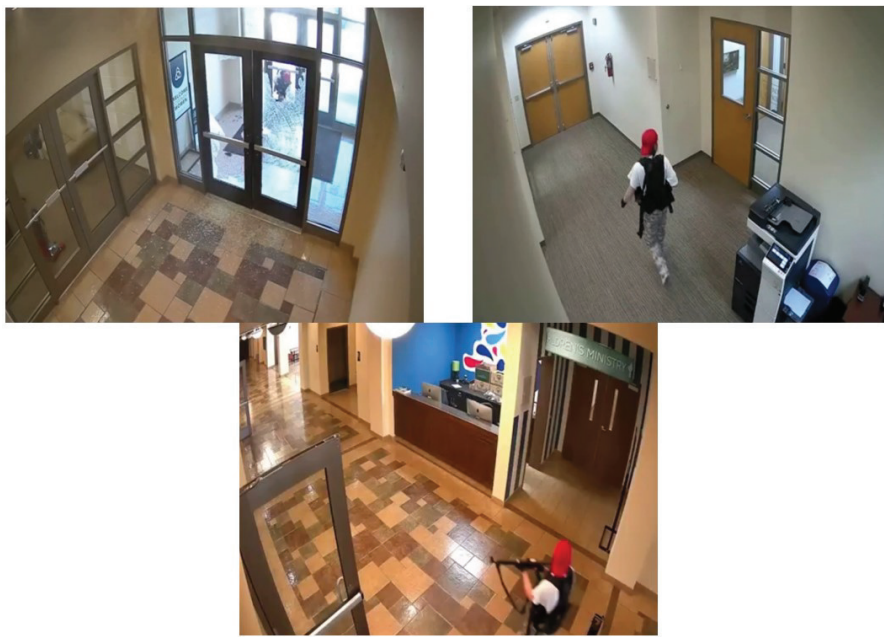

Figure 12.5 Surveillance camera views from the Nashville school shooting (2023) used to estimate visibility times: entrance hall (top left), corridor (top right), and lobby (bottom). The analysed video was taken from Wikipedia.

the entrance hall, the requirement becomes 1 FPS for N = 5 and 2 FPS for N = 10. These values remain modest compared to the 15–20 FPS typically achieved on the deployment hardware, but they better reflect the need for redundancy and reliable detection.

One should also account for the detector's per-frame detection probability "p". If the model only detects the weapon with probability p on any single processed frame, then multiple frames are needed to achieve a desired overall reliability. The probability of at least one success across N independent frames is:

$$P(\geq 1 \text{ detection }) = 1 - (1 - p)^N \tag{12.7}$$

Let us suppose that we want a target reliability of P_{target}. An example value of 0.95 would indicate that we want a 95% probability of detecting the weapon at least once while it is in view. Then the required number of frames is:

$$N = \left\lceil \frac{\log\left(1 - P_{\text{target}}\right)}{\log(1 - p)} \right\rceil \tag{12.8}$$

For example, if p = 0.5, then N = 5 frames are required to reach 95% reliability; if p = 0.8, N = 2 frames suffice. Combining this requirement with the visibility window T gives the operational frame-rate threshold FPS_{min} [f0] = N/T. Now, we considered a target probability P_{target} = 0.99 and three representative per-frame detection probabilities p ∈ {0.75, 0.80, 0.85}. Obtaining N values using Equation 8 and using FPS_{min} = N/T across the three Nashville scenes (Table 12.2), results are shown in Table 12.3.

After establishing the evaluation metrics and the theoretical requirements for real-time operation, we now present the results obtained with the different detection strategies. Comparing the two main pipelines explained, the goal of these experiments was to systematically analyse the trade-offs between accuracy, measured by AP_{50} across the three test sets, and efficiency, expressed in frames per second (FPS) on the target Jetson Orin Nano platform.

The results in Table 12.4 highlight the trade-off between accuracy and inference speed across the evaluated strategies. Each strategy achieves the FPS_{min} value for each Nashville camera view. The single-stage YOLOv8s detector provided balanced performance, sustaining 20 FPS with an AP_{50} around 0.70–0.75 across the test sets. This makes it a robust option for scenarios where real-time responsiveness is essential. The two-stage pipeline with YOLOv11x pose + EfficientNet classification achieved the highest accuracy (AP_{50} above 0.81) but at the cost of significantly reduced speed (3.5 FPS), which may limit its usability in continuous surveillance applications. Using lighter pose models (YOLOv11n) improved efficiency (6 FPS) but with some loss of precision.

The MediaPipe-based pipeline delivered intermediate efficiency (10 FPS) but suffered from poor accuracy, especially under degraded conditions (Test Set C), indicating that its lightweight design sacrifices robustness for handgun

Table 12.3 Operational frame-rate thresholds (FPS_{min}) required to achieve 99% detection reliability (P_{target} = 0.99) for three per-frame detection probabilities (p = 0.75, 0.80, 0.85) across the three Nashville camera views.

Camera View	Visibility T (s)	p	Required Frames N	FPS_{min} = N/T
Entrance Hall	5	0.75	4	0.80
		0.80	3	0.60
		0.85	3	0.60
Corridor	27	0.75	4	0.15
		0.80	3	0.11
		0.85	3	0.11
Lobby	10	0.75	4	0.40
		0.80	3	0.30
		0.85	3	0.30

Table 12.4 Comparison of detection strategies showing accuracy (AP_{50} on three test sets) and inference speed (FPS) on the Jetson Orin Nano platform

Strategy	**Models used**	**%AP 50 Test Set A**	**%AP 50 Test Set B**	**%AP 50 Test Set C**	**FPS**
Full frame weapon detection	YOLO 8s (detection)	75.15	71.45	70.24	20
Region proposal + classification	YOLO 11x (pose) + EfficientNet (classification)	83.11	80.99	81.42	3.5
Region proposal + classification	YOLO 11n (pose) + EfficientNet (classification)	75.09	74.75	78.71	6
Region proposal + classification	MediaPipe (pose) + EfficientNet (classification)	65.42	49.59	32.65	10
Region proposal + classification	YOLO 8n (pose) + MobileNetV2 (classification)	57.36	53.06	45.22	20

detection. Finally, the combination of YOLOv8n pose with MobileNetV2 classification achieved the highest efficiency (20 FPS) but with the lowest accuracy ($AP_{50} < 0.58$).

Overall, the comparison confirms that no single configuration dominates both metrics. For practical deployment on the Jetson Orin Nano, the YOLOv8s full-frame detector emerges as the best compromise, offering sufficient accuracy together with real-time performance (>15 FPS). The two-stage pipelines provide useful insights into how accuracy can be improved, but their higher computational cost makes them less attractive under strict resource constraints.

12.8 Future work

Future work will involve deploying the handgun detection system within the Modena Automotive Smart Area (MASA) using the HAura Edge device for on-site inference and integration with the smart city network. This will allow evaluation under realistic conditions, considering lighting changes, occlusions, and movement, while assessing interoperability, latency, and

communication reliability. Concurrently, efforts will focus on improving accuracy and efficiency through optimized training, model compression, and hyperparameter tuning. Long-term testing will monitor continuous operation to ensure stability, scalability, and sustained performance on the Jetson Orin Nano platform.

12.9 Conclusion

This work presented the development of a real-time handgun detection system designed for deployment in smart city environments. The proposed detector was implemented on a Jetson Orin Nano platform, addressing the constraints of embedded computing while maintaining a balance between accuracy and inference speed. Two complementary detection strategies were explored: a single-stage approach based on YOLOv8s and a two-stage pipeline combining pose estimation with region classification. Experimental results demonstrated that the YOLOv8s full-frame detector achieved the best trade-off, sustaining 20 FPS with an AP_{50} of 75.15%, making it suitable for continuous surveillance applications on resource-limited devices.

Beyond model optimization, the system was integrated into an MQTT-based communication framework, enabling automatic event reporting and real-time configuration. This end-to-end design (from dataset creation to deployment) illustrates a practical approach to embedding AI-driven perception in urban infrastructures. The forthcoming integration of the detector into a HAura Edge unit within the Modena Automotive Smart Area (MASA) will extend this work to large-scale, real-world testing. These experiments will validate system robustness under dynamic conditions and confirm its role as a functional component of the HAura smart city ecosystem, paving the way for future research on intelligent, privacy-aware, and responsive urban surveillance solutions.

Acknowledgements

This work was partially funded by Horizon Europe project dAIEdge, Grant n. 101120726, by the European Commission.

References

[1] Gun Violence Archive, “Gun Violence Archive,” Gunviolencearchive.org, Apr. 21, 2024. https://www.gunviolencearchive.org/

[2] S. A. Ali Shah, M. Ahmad Al-Khasawneh, and M. I. Uddin, "Review of Weapon Detection Techniques within the Scope of Street-Crimes", in 2021 2nd International Conference on Smart Computing and Electronic Enterprise (ICSCEE), 2021, pp. 26–37, https://doi.org/10.1109/ICSCEE50312.2021.9498007.

[3] S. Yellapragada et al., "CCTV-Gun: Benchmarking Handgun Detection in CCTV Images", arXiv [cs.CV]. 2023, https://doi.org/10.48550/arXiv.2303.10703.

[4] P. Mehta, A. Kumar, and S. Bhattacharjee, "Fire and Gun Violence based Anomaly Detection System Using Deep Neural Networks", in 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC), 2020, pp. 199–204, https://doi.org/10.1109/ICESC48915.2020.9155625.

[5] M. T. Bhatti, M. G. Khan, M. Aslam, and M. J. Fiaz, "Weapon Detection in Real-Time CCTV Videos Using Deep Learning", IEEE Access, vol. 9, pp. 34366–34382, 2021, https://doi.org/10.1109/ACCESS.2021.3059170.

[6] J. Salido, V. Lomas, J. Ruiz-Santaquiteria, and O. Deniz, "Automatic Handgun Detection with Deep Learning in Video Surveillance Images", Applied Sciences, vol. 11, no. 13, 2021, https://doi.org/10.3390/app11136085.

[7] J. Ruiz-Santaquiteria, O. Deniz, N. Vállez, A. Velasco Mata, and G. Bueno, "Improving handgun detectors with human pose classification", 09 2022, pp. 1040–1047, https://doi.org/10.17979/spudc.9788497498418.1040.

[8] N. S. Khan, K. Ogura, E. Cosatto, and M. Ariyoshi, "Real-time Concealed Weapon Detection on 3D Radar Images for Walk-through Screening System", in 2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2023, pp. 673–681, https://doi.org/10.1109/WACV56688.2023.00074.

[9] J. D. Muñoz, J. Ruiz-Santaquiteria, O. Deniz, and G. Bueno, "Concealed Weapon Detection Using Thermal Cameras", Journal of Imaging, vol. 11, no. 3, 2025, https://doi.org/10.3390/jimaging11030072.

[10] R. Cavicchioli, R. Martoglia, and M. Verucchi, "A Novel Real-Time Edge-Cloud Big Data Management and Analytics Framework for Smart Cities", JOURNAL OF UNIVERSAL COMPUTER SCIENCE, 01 2022, https://doi.org/10.3897/jucs.71645.

[11] C. Scribano, I. S. Olmedo, M. Verucchi, D. P. Paudel, M. Bertogna, and L. Van Gool, "On-the-Edge Inference Enabled Vision System for Smart

Cities", in SMART 2025: The Fourteenth International Conference on Smart Cities, Systems, Devices and Technologies, 2025, pp. 24–28, http s://hdl.handle.net/11380/1387818.

[12] G. Ferraro et al., "Towards Smart Cities with AI-CAM: Assisted by Infrastructure Cooperative Awareness Messages," 2025 IEEE Conference on Standards for Communications and Networking (CSCN), Bologna, Italy, 2025, pp. 1-6, https://doi.org/10.1109/CSCN67557.2025.11230591.

[13] J. Ruiz-Santaquiteria, A. Velasco-Mata, N. Vallez, O. Deniz, and G. Bueno, "Improving handgun detection through a combination of visual features and body pose-based data", Pattern Recognition, vol. 136, p. 109252, 2023, https://doi.org/10.1016/j.patcog.2022.109252.

[14] P. Jiang, D. Ergu, F. Liu, Y. Cai, y B. Ma, "A Review of Yolo Algorithm Developments," Procedia Computer Science, vol. 199, pp. 1066-1073, 2022, https://doi.org/10.1016/j.procs.2022.01.135.

[15] M. Yaseen, "What is YOLOv8: An In-Depth Exploration of the Internal Features of the Next-Generation Object Detector", arXiv [cs.CV]. 2024, https://doi.org/10.48550/arXiv.2408.15857.

[16] R. Khanam and M. Hussain, "YOLOv11: An Overview of the Key Architectural Enhancements", arXiv [cs.CV]. 2024, https://doi.org/10.48550/arXiv.2410.17725.

[17] C. Lugaresi et al., "MediaPipe: A Framework for Building Perception Pipelines", arXiv [cs.DC]. 2019, https://doi.org/10.48550/arXiv.1906.08172.

[18] M. Tan and Q. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," in Proceedings of the 36th International Conference on Machine Learning (ICML), PMLR, vol. 97, pp. 6105–6114, May 2019, https://doi.org/10.48550/arXiv.1905.11946.

[19] K. Dong, C. Zhou, Y. Ruan, and Y. Li, "MobileNetV2 Model for Image Classification," in 2020 2nd International Conference on Information Technology and Computer Application (ITCA), 2020, pp. 476–480, https://doi.org/10.1109/ITCA52113.2020.00106.

[20] F. P. Scalcon et al., "AI-Powered Video Monitoring: Assessing the NVIDIA Jetson Orin Devices for Edge Computing Applications", in 2024 IEEE Transportation Electrification Conference and Expo (ITEC), 2024, pp. 1–6, https://doi.org/10.1109/ITEC60657.2024.10598994.

[21] R. Padilla, S. L. Netto, and E. A. B. da Silva, "A Survey on Performance Metrics for Object-Detection Algorithms", in 2020 International

Conference on Systems, Signals and Image Processing (IWSSIP), 2020, pp. 237–242, https://doi.org/10.1109/IWSSIP48289.2020.9145130.

[22] A. Velasco-Mata, J. Ruiz-Santaquiteria, N. Vallez, and O. Deniz, "Using human pose information for handgun detection", Neural Computing and Applications, vol. 33, no. 24, pp. 17273–17286, Dec. 2021, https://doi.org/10.1007/s00521-021-06317-8.