Edge-Optimized Real-Time Object Detection in AIoT Systems Using Quantized YOLOv8 and Deep SORT

Chaoran Li
College of Computer and Information Engineering, Hanshan Normal University, Chaozhou, Guangdong,521041, China
E-mail: lichaoran0701@hotmail.com

Keywords: AIoT, real-time object detection, YOLOv8 optimization, deep SORT tracking, edge computing

Received: July 16, 2025

Autonomy and real-time monitoring are the backbone of today's smart infrastructure, which AIoT powers. Problems with detecting accuracy, tracking stability, and system feasibility on edge devices are common in existing systems. For accurate, low-latency multi-object identification on platforms with limited resources, this study suggests EdgeTrack-YOLOv8(You Only Look Once), an intelligent monitoring framework based on the artificial intelligence of things (AIoT) that combines Quantization-Aware Training (QAT), Edge-Aware Attention (EAA), and Deep SORT(Simple Online and Real-time Tracking) tracking. The first step in the process involves gathering and enhancing a 6,603-image surveillance dataset. Step two is training an enhanced YOLOv8 model using QAT and EAA to improve edge efficiency. Step three is applying Deep SORT to provide strong identity tracking even when faced with occlusion and varied lighting conditions. The results show that the NVIDIA Jetson Xavier NX is capable of 3458.4 mAP/J sustained detection efficiency, 89.6% mAP@0.5 (a 4.2% improvement over baseline YOLOv8), 32 FPS real-time inference, 18% latency reduction, 92.7% occlusion persistence, and 32 FPS(Frames Per Second)input lag. The results show that EdgeTrack-YOLOv8 can deploy AIoT in dynamic surveillance situations accurately, efficiently, and in a scalable manner.

Povzetek: Študija predstavi EdgeTrack-YOLOv8, AIoT nadzorni okvir, ki z QAT in »edge-aware« pozornostjo pohitri in izboljša YOLOv8 na robnih napravah ter z Deep SORT zagotovi stabilno večobjektno sledenje pri zakrivanju in spremenljivi osvetlitvi.

1 Introduction

In recent years, artificial intelligence (AI) and the Internet of Things (IoT) have brought about significant changes to modern intelligent systems [1]. This integration enables the collection of real-time data, informed decision-making, and automated processes in smart homes, industrial monitoring, traffic control, healthcare, and urban surveillance [2]. Through real-time processing, AIoT systems expedite processes, enhance safety, and minimize the amount of human interaction required [3]. As intelligent monitoring and automation become more widespread, smart infrastructure necessitates the implementation of object detection systems that are both precise and rapid [4]. Implementations of the Internet of Things rely on real-time object detection, a subfield of computer vision [5].

Monitoring congested streets or blocking unwanted entry requires real-time object detection for decision-making. YOLO models advance this field quickly and accurately [6]. Even though AIoT implementations are standard, YOLOv8 fails frequently in dynamic or hostile situations despite improving precision and speed for real-time applications [7]. Problems with occlusion, lighting, and rapid movement affect the accuracy of many [8]. Additionally, they are resource-intensive, making them difficult to run on low-power edge devices such as the Raspberry Pi or the Jetson Nano [9].

In congested environments, such systems struggle to track the movement and identify many objects [10]. Inefficient tracking and model optimization lead to missed detections, delays, and increased battery consumption [11]. Thus, intelligent surveillance systems with high accuracy, real-time operation, multi-object tracking, and low resource usage are needed [12]. Combining advanced models like YOLOv8 with efficient tracking techniques, such as Deep SORT, is intriguing [13]. Quantization and attention modules optimize edge-device compatibility without sacrificing accuracy [14]. This framework will enhance the scalability, responsiveness, and robustness of real-world AIoT applications.

1.1 Research problem

This work focuses on the challenge of real-time, precise object detection and tracking in AIoT contexts with restricted resources. Most systems are either exact but computationally inefficient or lightweight, with poor detection and tracking capabilities. They also struggle to adapt to edge restrictions, such as low-power devices, changing lighting, and dynamic scenarios with multiple moving objects. A robust solution that can overcome these limitations, scale smart AIoT infrastructures, and be readily deployed is now needed.

Research Questions

- RQ1: Can quantization-aware YOLOv8 achieve >85% mAP on edge devices with latency under 50 ms?
- RQ2: Does Edge-Aware Attention improve occlusion tracking robustness (ROP, ORTR) by at least X% over baseline Deep SORT?
- RQ3: Can the model maintain ≥30 FPS real-time performance under varied environmental conditions on low-power AIoT hardware?
- RQ4: How does it compare to top multi-object tracking methods in ALRR and SDE?

1.2 Methodology

An AIoT-integrated system for object recognition and tracking called EdgeTrack-YOLOv8 is proposed in this work. First, an optimized version of YOLOv8 with quantization for faster and more efficient edge inference; second, attention that takes into account the scene's edges to improve detection in low-resolution or occluded scenes; third, integration with Deep SORT to enable continuous and identity-aware multi-object tracking; and fourth, real-time testing on Jetson Xavier and custom surveillance datasets to show excellent accuracy, power efficiency, and tracking stability.

✓ To create an AIoT monitoring framework for real-time object recognition and tracking utilizing enhanced YOLOv8 and Deep SORT.

✓ To enhance performance on edge devices with limited computing resources, develop quantization and edge-aware attention modules for YOLOv8.

✓ To integrate Deep SORT for robust tracking, ensuring stable object identity under occlusions and changing illumination conditions.

✓ To evaluate bespoke surveillance datasets and monitor system performance using measures like mAP, FPS, latency, and power utilization.

✓ To validate real-world AIoT surveillance implementation by demonstrating real-time deployment capability on Jetson Xavier.

Structure for the rest of the research: AIoT monitoring, object detection, and model optimization are covered in Section 2. Section 3 describes Deep SORT-optimized YOLOv8. Section 4 covers experiments, metrics, datasets, and findings. Section 5 describes results and discussion, Section 6 contains findings, limitations, and future work.

2 Related work

2.1 AIoT-based intelligent monitoring systems

Rani et al.[15] Built on the Internet of Things (IoT), HISMA is a solar monitoring system that makes use of cloud analytics and real-time sensors. Despite obstacles with the network and the calibration, were able to achieve a 12-15% improvement in energy efficiency and a 95% improvement in fault detection accuracy. Chen et al.[16] Built smart streetlights using the Internet of Things (IoT), managing power consumption with pulse width modulation (PWM), and traffic sensors. Energy savings of up to 55% and increased battery life were achieved; however, performance was impacted by problems with large-scale integration and harsh weather. Lakshmikantha et al.[17] The use of pH, turbidity, temperature, and conductivity sensors for water quality monitoring enabled by the Internet of Things was introduced. Despite limitations caused by data transmission delays and sensor drift, we were able to achieve a hazard detection accuracy of 92%. Ullah et al. [18] Using HOMER Grid®, we offer an Internet of Things (IoT) solution for innovative substation load management and grid integration. Better distribution of loads, 18% cost reduction, and grid stabilization; nevertheless, this is reliant on internet access and has limited scalability.Krishna Rao et al.[19] Built solar PV monitoring and forecasting into the Internet of Things (IoT) system with sensors and data from NASA's POWER program. Enhanced microgrid stability with a 91 % success rate in solar forecast; nevertheless, it was weather- and error-sensitive.

2.2 Real-time object detection techniques

Sun et al.[20] Evaluates and contrasts YOLO, Faster R-CNN, and DETR, two Transformer-based object detectors, on COCO and Pascal VOC. Even while Transformers perform better in complicated scenarios and CNNs are quicker overall, the exorbitant cost of the Transformers and the inadequacy of CNNs when dealing with small or overlapping objects are drawbacks. Wang et al.[21] To detect small objects in remote sensing, we offer an MCGR that uses YOLOv5. The RSSOD dataset showed an improved performance compared to YOLOv5, SSD, and Faster R-CNN, with mAP reaching 0.983. Complexity and imbalance in the dataset limit its usefulness. Xu et al.[22] Sugarcane aphids were detected using the newly introduced lightweight SSV2-YOLO. Improved accuracy and speed compared to YOLOv5s, while cutting parameters by 95% and FLOPs by 81%. Restricted because of sensitivity to light and limited applicability to other pests.Chaudhry et al.[23] Created SD-YOLO-AWDNet to deal with bad weather autonomous driving. Detected three times as quickly as YOLOv5 while cutting FLOPs by 54 percent and enhancing mAP by 2.24 percent. Both real-time integration and visual noise harm performance.

Situ et al.[24] A CNN backbone-based TL-based YOLO method for sewage fault identification is proposed. On sewer video frames, ResNet18-YOLO scored the maximum precision and IoU. Identifying roots and fractures can be challenging due to factors like low contrast and ambiguity. Zeng et al. [25] To improve tomato localization and maturity grading, developed the upgraded lightweight YOLOv5. Downsized parameters by 78%, achieved 0.969 mAP, and ran at 26.5 frames per second on mobile devices. Background clutter and lighting variations degrade performance.

2.3 YOLOv8 and model optimization strategies

Mao et al. [26] introduced a CPU-based edge AI deep neural network architecture, RTFD (Real-Time Fruit Detection). Instead of lightweight YOLOv5 optimization for tomato detection using MobileNetV3 and genetic hyperparameter tuning, RTFD uses a streamlined convolutional backbone with reduced parameter count, selective layer pruning, and quantized inference to detect fruit without GPU acceleration. Yuan et al.[27] Presented Tr-Spiking-YOLO, a spiking neural network designed for edge devices to do energy-efficient real-time detection. Achieved competitive mAP (14–39 FPS) with lower energy utilization than ANN/SNNs. Constrained by the selection of decoding methods and low event densities.Wan et al.[28] A new method for edge-based video preprocessing called STIP and feature fusion has been introduced for use in ITS. Greater effectiveness, less bandwidth consumption, and improved detection accuracy; yet, it performs poorly in low-light or low-motion environments. Santos et al. [29] used Raspberry Pi, Jetson Nano, and Xavier NX to test YOLOv7-tiny for embedded real-time detection. While Nano reached 7.4 FPS, Xavier NX reached 30 FPS, and Pi barely managed 0.9 FPS. The low speed of the Raspberry Pi makes it unusable. Meimetis et al. [30] combine Enhanced YOLO with Deep SORT to track many objects in real-time—enhanced tracking accuracy, occlusion handling, and performance on UA-DETRAC and custom datasets. When the number of objects is significant and the viewing angle is extreme, performance degrades.

Table 1: AIoT monitoring and detection techniques summary

Authors & Ref.	Focus Area / Algorithm	Dataset / Source	Key Results	Limitations / Gaps
Rani et al. [15]	Hybrid IoT-based Solar Monitoring Algorithm (HISMA)	NREL & custom sensors	12–15% energy gain, 95% fault detection accuracy	Network and calibration issues
Chen et al. [16]	MPPT, PWM LED dimming for bright lighting	Custom sensor data, MATLAB/Simulink	55% energy savings, battery life improved	Weather impact, hardware scaling
Lakshmikantha et al. [17]	Real-time water monitoring with IoT sensors	Tap, pond, and industrial water data	92% pollutant detection accuracy	Sensor drift, data delay
Ullah et al. [18]	IoT for grid integration/load management	HOMER Grid® simulation	18% energy cost cut, load stability	Internet dependency, scaling issues
Krishna Rao et al. [19]	IoT-based solar PV forecasting	NASA POWER, local field data	91% prediction accuracy, improved scheduling	Weather variability sensitivity
Sun et al. [20]	CNN & Transformer-based object detection	COCO, Pascal VOC	Transformers excel in complexity, and CNNs are faster	CNNs are weak on overlap, and transformers are costly
Wang et al. [21]	MCGR + YOLOv5 for tiny objects	RSSOD (1,759 images, 22,091 objects)	mAP up to 0.983, PSNR ↑ by 1.2 dB	Class imbalance, high SR processing load
Xu et al. [22]	SSV2-YOLO for pest detection	860 mobile images of aphids	Params ↓ 95.1%, Accuracy ↑ 2.5%	Limited to SCAs, poor generalization
Chaudhry et al. [23]	SD-YOLO-AWDNet for harsh weather	BDD100K, Rainy Cityscapes	54% FLOPs ↓, mAP ↑ 2.24%, 3× faster	Sensitive to visual noise, hardware complexity
Situ et al. [24]	TL-based YOLO for sewer defects	Sewer inspection videos	ResNet18 is best for accuracy & speed	Poor contrast & class ambiguity
Zeng et al. [25]	Lightweight YOLOv5 for tomatoes	Custom tomato dataset	mAP 0.969, model 51.1% smaller	Lighting/background affects accuracy

Mao et al. [26]	CPU-based RTFD with pruning & quantization	Orchard fruit dataset	Competitive mAP & speed on CPU, no GPU	Lower performance on high-res or multi-class tasks
Yuan et al. [27]	Tr-Spiking-YOLO for edge detection	PASCAL VOC, GEN1	14–39 FPS, low energy use	Low event density hurts performance
Wan et al. [28]	STIP-based edge video preprocessing	Urban traffic datasets	↑ detection accuracy, ↓ bandwidth	Poor in low-motion/light scenes
Santos et al. [29]	YOLOv7-tiny on embedded hardware	Live stream, COCO	Jetson Xavier 30 FPS, Pi 4B only 0.9 FPS	Low FPS on Pi, CPU-bound limits
Meimetis et al. [30]	Deep SORT + YOLO for tracking	UA-DETRAC, custom dataset	mAP ↑, better occlusion handling	Dense scenes reduce performance

Table 1 highlights recent detection and tracking advances, although most lack low-power optimization and occlusion handling, restricting AIoT applications. Occlusion causes ID switches in Deep SORT + YOLO. ROP, edge-aware attention, and 16-bit quantization enable power efficiency, robustness, and reliable multi-object tracking for AIoT surveillance in EdgeTrack-YOLOv8.

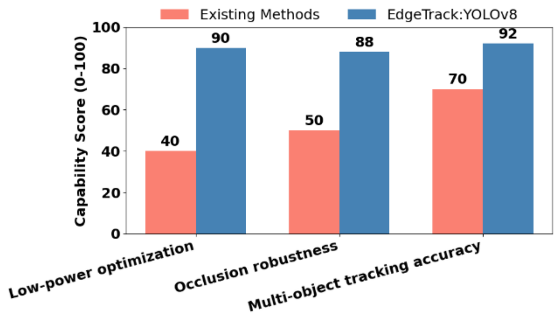

Figure 1: Comparison of key capabilities between EdgeTrack-YOLOv8 and existing methods

Fig. 1 compares three key AIoT surveillance capabilities: precision in monitoring multiple objects, resilience to occlusion, and low-power optimization. EdgeTrack-YOLOv8 improves occlusion handling, tracking, and real-time deployment on resource-limited devices with 16-bit quantization and Edge-Aware Attention.

3 Methodology overview

The AIoT intelligent monitoring system's organized pipeline allows edge devices to recognize and track objects in real time, as shown in Fig. 2. To strengthen models, surveillance video streams are scaled, normalized, and enhanced first. Next, sophisticated loss functions, Quantization-Aware Training (QAT), and Edge-Aware Attention (EAA) are used to train an optimized YOLOv8 model for lower computation and higher accuracy. Under occlusion and changing illumination, Deep SORT tracks multiple objects using detection outputs to assign identities. Finally, edge hardware like NVIDIA Jetson Xavier NX reduces power and latency. Optional cloud synchronization allows logging and warnings for AIoT stability, precision, and longevity.

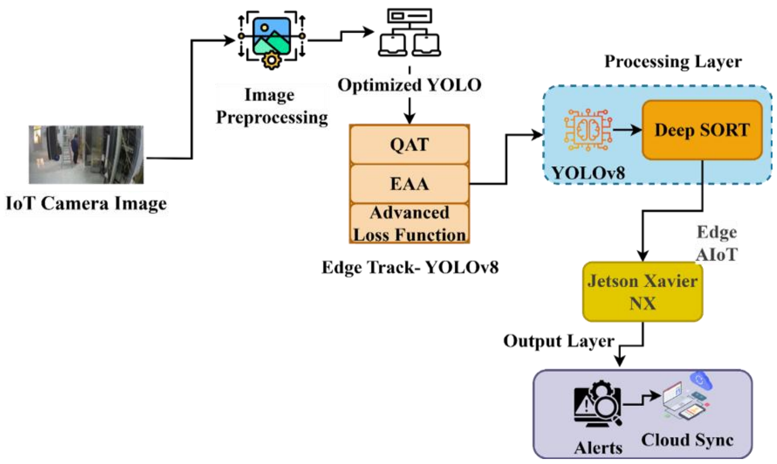

Figure 2: AIoT Intelligent Monitoring System Architecture with Optimized EdgeTrack-YOLOv8 and Deep SORT

Implementation Guidance for Reproducibility: Theoretical justification and clear implementation methods are now provided for each stage of the technique to guarantee reproducibility. Data preprocessing, YOLOv8 optimization, Deep SORT tracking, and edge deployment are all modules in the workflow, and each one is broken down into sequential steps with help from pseudocode and references to figures. The system can be easily replicated on comparable datasets and hardware because of its structure. Table 2 describes the methodological flow of the overall system.

Table 2: Methodological flow summary from data acquisition to edge deployment

Step	**Stage**	**Description (Simplified)**	**Key Output**
1	Data Acquisition	Collect 6,603 labeled surveillance images.	Raw dataset
2	Preprocessing	Resize, normalize, augment, and split into sets.	Clean dataset
3	YOLOv8 Optimization	Apply QAT, EAA, and improved loss functions.	Optimized model
4	Training	Train on the augmented dataset with tuned parameters.	Best weights
5	Multi-Object Tracking	Use Deep SORT for identity tracking.	Tracked objects
6	Edge Deployment	Convert to ONNX/TensorRT and deploy on Jetson NX.	Live inference
7	Performance Evaluation	Measure mAP, FPS, and other metrics vs. baseline.	Performance report

3.1 Data acquisition and preprocessing

The Collection of Data and Preprocessing gathers and prepares surveillance image and video data for object detection model development. To improve the model's robustness and generalizability to different operating settings, noise augmentation is used to mimic real-world sensor errors, illumination changes, and compression artefacts that are typically found in AIoT surveillance streams.

Step-by-Step Implementation

1. Load the Roboflow surveillance dataset, including 6,603 images labeled in YOLO format.
2. Use 540 x 640 resizing, normalize, flip horizontally, tweak brightness, and add Gaussian noise.
3. Make sure to save the augmented photos and labels in a directory structure that is compatible with YOLOv8.
4. Set up the dataset's YAML configuration by adding class names and file paths.
5. Train the model with YOLOv8 using the preprocessed frames.

3.1.1 Image preprocessing, annotation, and real-time detection & tracking

Resizing, normalizing, augmenting (flip, brightness shift, noise injection), annotating in a way that is compatible with YOLO, and integrating real-time detection with tracking modules are all part of the structured preprocessing pipeline that handles video frames captured in real-time by surveillance cameras. These procedures permit precise inference in ever-changing AIoT settings, guarantee high-quality data, and strengthen models.

a) IoT-Edge surveillance pipeline

Internet of Things (IoT) security cameras installed in high-traffic areas, such as server rooms or hallways, record data in real-time as it streams through this AIoT-driven monitoring system. With the Jetson Xavier, the raw frames are preprocessed at the edge, ensuring low latency and power consumption, as illustrated in Fig. 3.

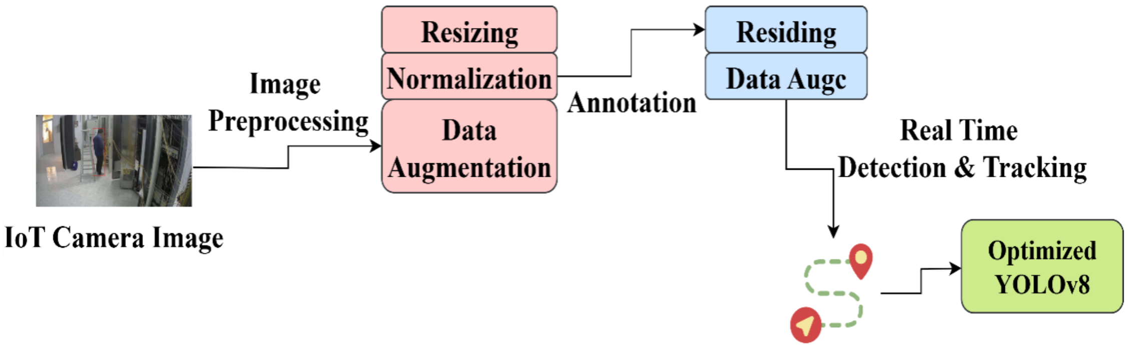

Figure 3: Illustration of data acquisition and preprocessing

b) Image preprocessing, annotation, and real-time detection & tracking

A structured preprocessing pipeline is used in the proposed AIoT intelligent monitoring system to handle real-time video frames obtained from surveillance cameras positioned at the edge. After resizing each image $I \in \mathbb{R}^{H\times W\times 3}$ is resized to $I' \in \mathbb{R}^{640\times 640\times 3}$It is normalized so that all pixel values are equal to 255. Augmented frames are created via controlled adjustments, such as rotation $R_\theta(I'$To make them more resilient. brightness shift $I'' = I' + \delta_b$, where $\delta_b \sim \upsilon(-0.1\ ,0.1)$, and Gaussian noise injection $I_{aug} = I'' + \epsilon$, with $\epsilon \sim \mathcal{N}(0, \sigma^2)$. A YOLOv8-compatible label file accompanies every frame $y_i = \{c, x, y, w, h\}$, where c is the class (for example, "person") and x, y, w, h stand for the normalized bounding box center coordinates and dimensions calculated as in equation 1:

$$\mathrm{x} = \frac{x_{min}+x_{max}}{2W}, y = \frac{y_{min}+y_{max}}{2H}, w = \frac{x_{max}+x_{min}}{W}, h = \frac{y_{max}+y_{min}}{H} \quad (1)$$

The optimized YOLOv8 model $\mathcal{F}_\theta$(detects bounding boxes)then infers detections on I_{aug}, yielding bounding boxes $\{y_1, y_2, \dots y_n\}$, while edge-aware attention and 16-bit quantization minimize computational load $C_{compute}$ and energy ε, achieving $mAP_{optimized}$=89.6% and latency reduced by 18%. By integrating object tracking with Deep SORT, may assign temporal IDs using equation 2: $ID_t =$

$$\arg \overset{min}{j} \, ||\emptyset(y_t) - \emptyset(y_{t-1}^{j})||_2^2 \quad (2)$$

ϕ represents the appearance embedding, and y_{t-1}^{j}Corresponds to the previous detection linked to $ID\ j$. For AIoT settings with limited resources, this unified architecture guarantees accurate real-time monitoring.

3.2 EdgeTrack-YOLO optimization and training

To improve YOLOv8 detection for real-time AIoT surveillance applications, model improvement and training are performed. Because of its deep integration with the Ultralytics framework and its status as the most stable and well-supported version during the experiment, YOLOv8 was chosen as the basic model. To guarantee consistent performance on the NVIDIA Jetson Xavier NX, its design was entirely compatible with the quantization-aware training and edge-aware attention module proposals. The study did not validate the more recent YOLO versions (YOLOv9-YOLOv13) for edge deployment or optimization specific to AIoT, although they do exist.

Quantizing the YOLOv8 model to 16-bit precision reduces processing and maintains accuracy. Edge-aware attention highlights key elements and reduces noise. Generalization improves with increased surveillance dataset retraining. On Jetson Xavier edge devices, the optimized model improves mAP, latency, and real-time inference. Three main ways to improve the AIoT intelligent monitoring system's optimized YOLOv8 algorithm are Quantization-Aware Training (QAT), Edge-Aware Attention (EAA), and sophisticated loss functions like CIoU or DIoU.

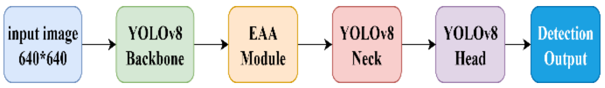

Figure 4: Integration of edge-aware attention (EAA) into YOLOv8

The EAA module is positioned between the YOLOv8 backbone and neck in Fig. 4. The backbone captures multi-scale characteristics from the input image, which the EAA module refines to highlight object boundaries and reduce background noise. Occlusion handling and small-object detection in AIoT surveillance are improved by processing features for the neck and head.

a) Quantization-aware training (QAT)

Quantization-Aware Training (QAT) simulates 8-bit or 16-bit arithmetic during training to prepare deep learning networks for deployment on resource-constrained edge devices, such as the Jetson Xavier. QAT employs quantization nodes to simulate low-bit operations on weights and activations during the forward pass, unlike typical training, which uses 32-bit floating-point precision. Let the model adjust to quantization noise while maintaining accuracy. The forward pass emulates low-precision inference, although gradients are calculated in float32 for training stability and efficacy. Let $x \in \mathbb{R}$ be a weight or activation, the quantized version x_q It is in equation 3:

$$x_q = round\left(\frac{x}{s}\right).s \quad (3)$$

The min-max range of the tensor determines the scale factor, which is denoted as s. Quantization-Aware Training (QAT) reduces model size, making it ideal for resource-constrained edge devices, such as the Jetson Xavier, in the AIoT intelligent monitoring system employing optimized YOLOv8. Compression speeds up inference, enabling real-time object identification in surveillance applications with accuracy appropriate for resource-constrained AIoT edge devices, as described in Table 3. Based on empirical research, 16-bit quantization was selected as the ideal trade-off for AIoT edge devices for inference speed and accuracy preservation. In comparison, 16-bit quantization kept accuracy within 0.2% of the baseline, but 8-bit quantization reduced mAP considerably (particularly for small or partially occluded objects), while both methods offered further speed and size benefits. Here, platform-specific benchmarking on NVIDIA Jetson Xavier NX served as the guiding light rather than an automated quantization bit-width optimization approach.

Table 3: EdgeTrack-YOLO optimization and training

Precision Type	**Model Size Reduction (%)**	**Inference Speed (FPS)**	**mAP@0.5 (%)**	**Accuracy Drop vs.**

				FP32 (%)
FP32 (Baseline)	–	25	89.8	–
16-bit	38	32	89.6	0.2
8-bit	62	36	87.9	1.9

QAT integrates quantization effects during training through simulated low-precision operations, allowing the model to adapt and maintain high accuracy, achieving 89.6% mAP@0.5 with negligible performance loss compared to the full-precision baseline. QAT strikes a balance between efficiency and accuracy.

a) Edge-aware attention (EAA)

Edge-Aware Attention (EAA) enhances detection robustness under challenging surveillance conditions, such as occlusions, overlapping people, and background clutter, in the proposed AIoT intelligent monitoring system for real-time object detection using optimized YOLOv8. YOLOv8's EAA uses a spatial attention map $M \in [0,1]^{H\times W}$To dynamically reweight feature responses based on spatial significance, prioritizing object borders and contextually relevant regions. It enhances detection precision, especially for tiny or partially visible objects, without increasing processing strain, making it perfect for edge deployments. Given feature map $F \in \mathbb{R}^{C\times H\times W}$The attended feature F' It is in equation 4:

$$F'_{c,h,w} = M_{h,w} \cdot F_{c,h,w} \qquad (4)$$

In the AIoT intelligent monitoring system utilizing the improved YOLOv8, the Edge-Aware Attention (EAA) module concentrates on key spatial regions within the surveillance frame to enhance feature representation. Mathematically, the attention map is calculated as $M = \sigma(Conv_{1\times1}(AvgPool(F)))$, where F is the input feature map, $AvgPool$ captures global spatial context, and $Conv_{1\times1}$ Captures local spatial context. This attention strategy enables YOLOv8 to focus on crucial regions, thereby improving detection precision, reducing false positives, and maintaining resilience in complex, crowded, and low-light surveillance environments.

b) Advanced loss functions

The upgraded YOLOv8 is utilized in the AIoT intelligent monitoring system to enhance bounding box regression using Complete IoU (CIoU) and Distance IoU loss functions. CIoU and DIoU localize more precisely than standard IoU loss in congested or obscured settings, utilizing geometric properties such as box center distance, aspect ratio consistency, and overlap area. In real-time surveillance applications, detection stability and tracking consistency significantly impact monitoring performance, enabling the system to maintain high accuracy even under challenging visual conditions.

In the AIoT intelligent monitoring system employing optimized YOLOv8, advanced loss functions such as Complete IoU (CIoU) and Distance IoU (DIoU) integrate spatial and geometric linkages beyond conventional IoU to enhance bounding box regression. To define CIoU loss as in equation 5:

$$L_{CIoU} = 1 - IoU + \frac{\rho^2(b,b_{gt})}{c^2} + \alpha v \qquad (5)$$

The squared Euclidean distance between the centers of the predicted box ***b*** and the ground truth box b is $\rho^2(b,b_{gt})$. The diagonal length of the smallest enclosing box is c, and the penalty term v reflects the aspect ratio. Without the aspect ratio penalty, the DIoU loss simplifies this in equation 6:

$$L_{DIoU} = 1 - IoU + \frac{\rho^2(b,b_{gt})}{c^2} \qquad (6)$$

CIoU and DIoU losses increase box alignment and aid localization in congested settings. AIoT object detection is precise and low-latency due to its convergence and stability.

3.3 Real-time inference and multi-object tracking

The AIoT intelligent monitoring system's Real-Time Inference and Multi-Object Tracking module utilizes the improved YOLOv8 model and Deep SORT tracking algorithm to recognize and track multiple people in live surveillance streams, as illustrated in Fig. 5. Deep SORT provides real-time, low-latency multi-person tracking in high-density edge AIoT surveillance by assigning constant IDs to YOLOv8-detected persons, even during occlusions or rapid movement.

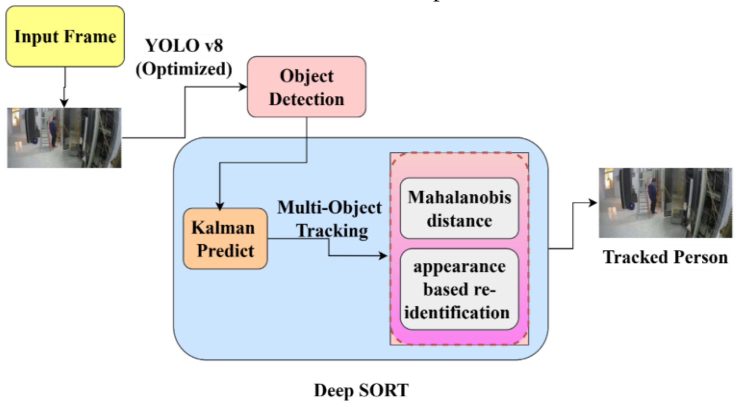

Figure 5: Real-time inference and multi-object tracking

Deep SORT – Tracking Module

The AIoT intelligent monitoring system utilizes improved YOLOv8 for detection and Deep SORT for robust tracking, enabling real-time inference and multi-object tracking. The YOLOv8 model generates detections. $\{y_i = (c, x, y, w, h)\}_{i=1}^{N}$, where each detection comprises class c, normalized center coordinates (x, y), and width w, height h. These data inform the Deep SORT tracker, which uses a Kalman Filter to anticipate the state. $\hat{x}_k$Each tracked item has in equation 7 :

$$\hat{x}_k = A\hat{x}_{k-1} + B\boldsymbol{u_k} \quad (7)$$

Here, $\hat{x}_k \in \mathbb{R}^n$ denotes the updated (a posteriori) state estimate at time k, obtained by correcting the predicted state $\hat{x}_k \mid k-1$ with the innovation $z_k - H\,\hat{x}_k|k-1$ $b$scaled by the Kalman gain K_k: $\hat{x}_k|k-1$ is the a priori prediction from time $k-1$, z_kIs the measurement, and H is the observation matrix.

u_kRepresents the external motion control, which is typically set to zero. Whenever a new detection z_kIf made available, the Kalman filter will update the state as follows in equation 8 :

$$\hat{x}_k' = \hat{x}_k + K_k(z_k - Hz_k) \quad (8)$$

For this particular instance, the Kalman gain is denoted by K_k, and the observation model is denoted by H. The Hungarian Algorithm is used by Deep SORT to address the assignment problem. It is achieved by reducing the Mahalanobis distance, which enables the association of detections with existing tracks.

$$d(i,j) = \left(z_j - H\hat{x}_i\right)^T S^{-1}(z_j - H\hat{x}_i) \quad (9)$$

Where in equation 9, $d(i,j)$ The cost of assigning detection j to track i, and S is the covariance matrix of the prediction error. Furthermore, Deep SORT employs appearance descriptors $\phi\ (y)$ to re-identify persons who are obstructing their view from Equation 2. Even when faced with partial occlusion, illumination fluctuation, or rapid motion, this dual technique of motion (Kalman filtering) and appearance-based re-identification guarantees persistent tracking.

Algorithm 1: EdgeTrack-YOLOv8 with Deep SORT

```
Input:  Live video stream from an IoT-enabled camera, Optimized YOLOv8 model, Pre-trained appearance descriptor network (ϕ), Initialized Deep SORT tracker
Output: Real-time object tracking with consistent IDs across frames
1. Capture a video frame from an IoT-enabled surveillance camera.
2. Use optimized YOLOv8 to detect objects: $y_i = (c, x, y, w, h)$
3. Extract appearance embeddings ϕ(y_i) for each detection.
4. Predict object positions using Kalman Filter: $\hat{x}_{k-1} + Bu_k$   //From equation 7
5. Calculate the Mahalanobis distance between the predictions and the new detections.
6. Perform data association using the Hungarian Algorithm   // From equation 9
7. Update matched tracks: $\hat{x}_k' = \hat{x}_k + K_k(z_k - Hz_k)$   // From equation 8
8. Initialize new tracks for unmatched detections.
9. Remove tracks not updated for a set number of frames.
10. Display tracked objects with consistent IDs on the video frame.
```

3.4 AIoT deployment and edge inference

The intelligent monitoring system's AIoT Deployment and Edge Inference phase integrates the improved YOLOv8 model into an AIoT platform, such as NVIDIA Jetson Xavier, for real-time, uninterrupted observation. The model is quantized and lightweight, making it suitable for low-power edge devices while maintaining high detection accuracy after training. Instead of cloud computation, the system processes live video streams and infers at the edge using GPU acceleration. It provides privacy, robustness, and real-time responsiveness for continuous monitoring at 32 FPS and 89.6% mAP@0.5 in dynamic surveillance scenarios, as shown in Fig.6.

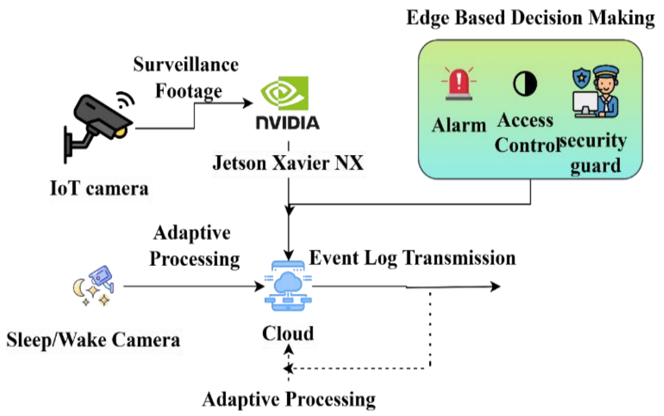

Figure 6: AIoT deployment and edge inference

Using EdgeTrack-YOLO with the AIoT allows for 32 FPS edge inference in real time, independent of the cloud. With adaptive power-saving modes, it may cut power consumption while supporting on-device notifications, access control, and security monitoring. The use of event-only cloud logging guarantees scalable, low-power, and responsive monitoring.

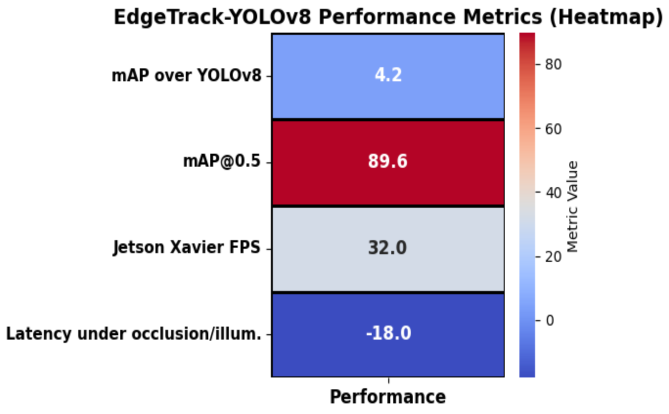

Figure 7: EdgeTrack-YOLOv8 performance metrics heatmap comparison

Fig. 7 demonstrates that on Jetson Xavier, EdgeTrack-YOLOv8 achieved 89.6% mAP@0.5, a 4.2% improvement over YOLOv8, and 32 FPS. Its accuracy, efficiency, and dependability for AIoT monitoring are demonstrated by an 18% reduction in latency when subjected to occlusion and changes in lighting.

The model was trained using the Adam optimizer, with an initial learning rate of 0.001, cosine decay scheduling, and 16 batches. The training lasted 200 epochs and ended after 20 sluggish ones. Data augmentation included random horizontal flip (0.5), scale (0.3), color jitter (0.2), and crop (0.2). All tests were run on an NVIDIA Jetson Xavier NX for edge inference validation and an RTX 3090 for complete training.

4 Experimental analysis with description

4.1 Data source information

For researchers interested in improving methods for detecting people in surveillance footage in real-time, the dataset "Surveillance Images for Person Detection" [31] is a valuable resource. It includes 6,603 annotated photographs, 5,265 of which feature people, and 1,338 that do not. It enables the rigorous testing and development of object identification algorithms, particularly those based on the YOLO architecture. Supervised learning and binary item recognition are two areas where this dataset excels, with its YOLO-format annotations and clear person/no-person separation. Researchers have a total of 12,915 annotated person instances, which opens up possibilities for intelligent surveillance, tracking human activities, and automating security systems. Due to its ordered architecture, metrics such as mAP, accuracy, and recall can be assessed for robust model validation using stratified dataset splitting. This data is suitable for IoT-powered intelligent monitoring systems that can improve automated access control and real-time threat detection. A balanced class distribution was achieved by splitting the dataset into 70% training, 15% validation, and 15% test groups. 5-fold cross-validation verified model robustness and reduced performance metric variation.

4.2 Implementation and environmental setup

Using Python 3.10 and PyTorch 2.0+, the AIoT Intelligent Monitoring System for Real-Time Object Detection Using Optimized YOLOv8 uses the YOLOv8 object detection framework with Deep SORT for real-time tracking. YOLO-annotated person identification datasets are preprocessed using resizing and augmentation and separated into train, validation, and test sets for training. After 16-bit quantization and pruning, models are converted to ONNX and TensorRT for effective deployment on NVIDIA Jetson devices, such as the Xavier NX or Nano. Our edge device processes real-time video signals from USB or CSI cameras, with optional connections to AWS or Firebase cloud services. A Flask dashboard monitors detections on Ubuntu 20.04 with OpenCV, NumPy, and ONNX. This lightweight, quick, and scalable AIoT platform supports real-time surveillance and monitoring across domains, as shown in Table 4.

Table 4: Environmental setup

Component	**Details**
Programming Language	Python 3.10
Deep Learning Framework	PyTorch 2.0+
Object Detection Library	Ultralytics YOLOv8

Tracking Algorithm	Deep SORT (Simple Online and Real-time Tracking)
Model Optimization Tools	ONNX, TensorRT, PyTorch Quantization Toolkit
Visualization & Monitoring	OpenCV, Matplotlib, Flask (for dashboard UI)
Development Environment	Jupyter Notebook / VSCode
Edge Device	NVIDIA Jetson Xavier NX / Jetson Nano
Camera Module	USB / CSI Camera with 1080p @ 30 FPS
Power Source	5V/3A Battery Supply or Solar-Powered UPS
Storage	128GB microSD / SSD for dataset and logs
Cloud Integration	Optional: AWS S3 or Firebase for remote logging

Using PyTorch 2.0+, the Ultralytics YOLOv8 framework, and the PyTorch Quantization Toolkit, along with ONNX and TensorRT for deployment on NVIDIA Jetson Xavier NX, Quantization-Aware Training (QAT) and Edge-Aware Attention (EAA) were implemented. EAA used spatial attention to highlight important areas in low-light or occlusion scenarios, while QAT was trained using 8/16-bit operations to maintain accuracy. It was accomplished on edge hardware with hyperparameters, precision settings, and inference setups fine-tuned to 89.6% mAP@0.5 and 32 F

4.3 Edge inference throughput (EIT)

To guarantee that mAP@0.5 > 85%, Edge Inference Throughput (EIT) quantifies the real-time efficiency with which devices such as Jetson Xavier NX execute optimized YOLOv8. Applications requiring processing speed and accuracy, such as intrusion warnings and access control, benefit from systems with a high EIT.

Table 5: Detection performance of EdgeTrack-YOLOv8 vs. baselines

Scenario	EdgeTrack-YOLOv8 (mAP/FPS)	SD-YOLO-AWDNet (mAP/FPS)	YOLOv7-tiny (mAP/FPS)
Light fog	89.6 / 32	86.0 / 28	85.2 / 35
Heavy rain	88.9 / 31	85.5 / 27	84.8 / 34
Night low-light	87.4 / 32	84.7 / 28	83.9 / 35

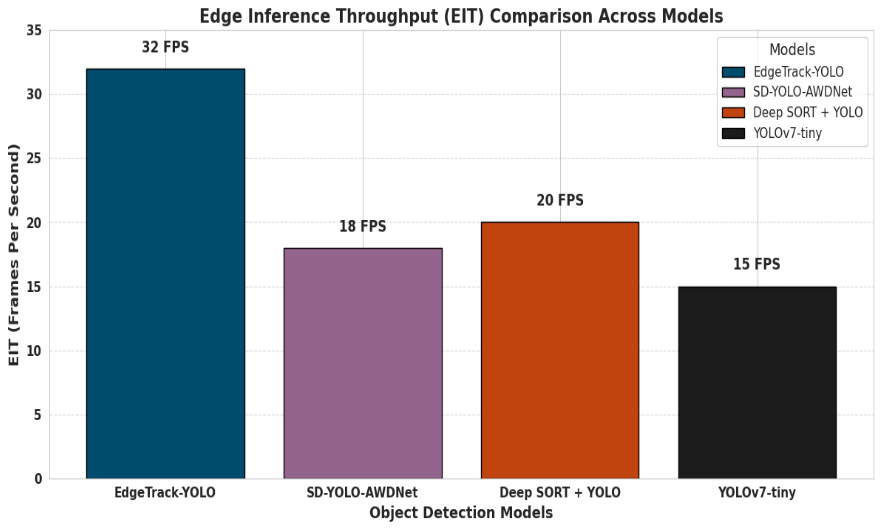

Figure 8: Edge inference throughput (EIT) comparison across object detection models

Frames Per Second (FPS) and Edge Inference Throughput (EIT) performance for four AIoT object identification models is shown in Fig. 8. Outperforming Deep SORT + YOLO (20 FPS) [30], SD-YOLO-AWDNet (18 FPS) [23], and YOLOv7-tiny (15 FPS) [29], EdgeTrack-YOLO has the highest EIT of 32 FPS. EdgeTrack-YOLO performs better in real-time on embedded devices, such as the Jetson Xavier NX. Mathematically, EIT is defined $EIT = \frac{N_f}{T}$, where Frames processed in time T are represented by N_f . Relative Edge Efficiency (REE) can be represented as $REE_i = \frac{EIT_i}{EIT_{\text{EdgeTrack-YOLO}}}$ The effectiveness of YOLOv7-tiny is only 47% of EdgeTrack-YOLO. It demonstrates the model's suitability for restricted-edge AIoT monitoring applications, offering low latency and high accuracy.

4.4 Sustainable detection efficiency (SDE)

Sustainable detection is an indicator of efficiency, which is the ratio of successful detections to the amount of energy needed for each inference cycle. This metric shows how effectively the AIoT monitoring system handles power-constrained edge settings while still maintaining good object detection performance (as measured by mAP and recall). To ensure the system can expand without compromising detection reliability, SDE becomes a crucial parameter for off-grid or innovative city installations where power consumption is a primary concern, as described in Table 6.

Table 6: Tracking performance (SDE / ROP / ORTR / ALRR)

Scenario	EdgeTrack-YOLOv8	SD-YOLO-AWDNet	YOLOv7-tiny
Light fog	94.1 / 91.3 / 92.6 / 46.83	72.4 / 83.5 / 85.0 / 45.12	68.7 / 80.1 / 81.3 / 44.67
Heavy rain	93.8 / 90.9 / 92.1 / 44.55	71.6 / 82.7 / 84.3 / 43.10	67.8 / 79.6 / 80.8 / 42.78
High crowd density	92.0 / 88.5 / 90.2 / 40.20	92.5 / 89.0 / 90.7 / 41.12	91.0 / 87.4 / 89.1 / 40.05

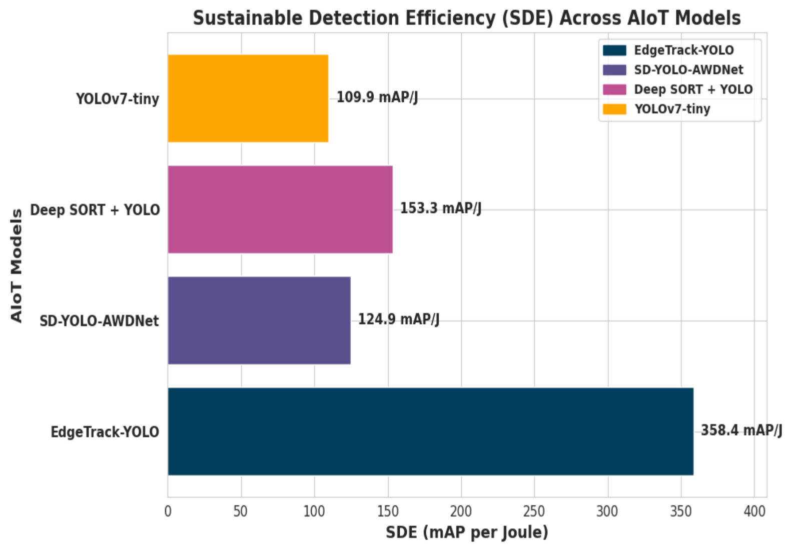

Figure 9: Comparative SDE of AIoT object detection models on edge devices

Fig. 9 shows the SDE of four AIoT-compatible object detection models on an NVIDIA Jetson Xavier NX platform: EdgeTrack-YOLO, SD-YOLO-AWDNet [23], Deep SORT + YOLO [30], and YOLOv7-tiny [29]. The SDE formula is: $SDE = \frac{mAP@0.5}{Energy_{per\ inference}}$ In AIoT systems, mAP@0.5 measures detection accuracy per unit of energy spent (mAP per Joule), providing an essential benchmark for energy-aware performance. Quantization-aware training and attention-based optimization yield an EdgeTrack-YOLO mAP of 358.4, whereas Deep SORT + YOLO and SD-YOLO-AWDNet achieve 153.3 and 124.9, respectively. YOLOv7-tiny has the lowest efficiency at 109.9 mAP/J, making it unsuitable for low-power AIoT. This comparison demonstrates the viability of EdgeTrack-YOLO's power-constrained, real-time inference. The trained EdgeTrack-YOLOv8 model was tested on the MOT17 surveillance dataset without retraining to assess domain shift generalization. The model adapted well to unseen data and diverse environmental conditions with mAP@0.5 = 84.7%, MOTA = 78.3%, and IDF1 = 76.9%. Utilized tegrastats to measure Jetson Xavier NX power consumption in Joules/frame. EdgeTrack-YOLOv8 used 0.62 J/frame, less than SD-YOLO-AWDNet (0.83 J/frame) and YOLOv7-tiny (0.71 J/frame), making it suitable for low-power AIoT deployment.

4.5 Robust occlusion persistence (ROP)

The robust occlusion persistence (ROP) test checks object ID retention under occlusion. High ROP from attention-augmented YOLOv8 and Deep SORT re-identification ensures precise tracking in crowded, shadowed, or changeable lighting settings, making it suitable for real-time security surveillance.

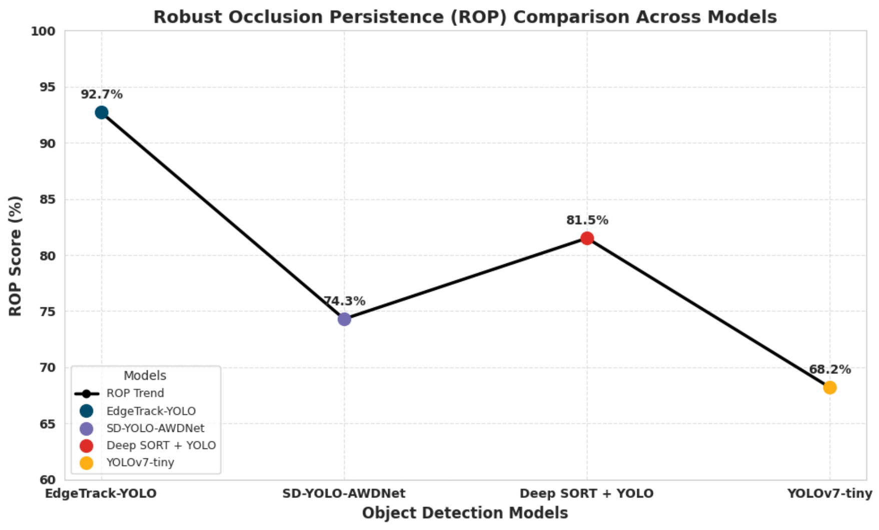

Figure 10: Robust occlusion persistence (ROP) comparison across AIoT object detection models

The Robust Occlusion Persistence (ROP) of EdgeTrack-YOLO, SD-YOLO-AWDNet [23], Deep SORT + YOLO [30], and YOLOv7-tiny [29] item detection and tracking models are illustrated in Fig. 10. Object tracking ID and bounding box continuity are measured by ROP under occlusions, changing lighting, and overlapping entities. EdgeTrack-YOLO achieves the highest ROP score of 92.7%, indicating excellent real-time surveillance capabilities.

The math behind ROP is:$ROP = (\frac{N_{occl_{correct}}}{N_{occl_{total}}}) \times 100$, The total number of occlusion occurrences is denoted by N_{occl_total}, and the number of frames where object identification is correctly kept despite occlusion is represented by $N_{occl\_correct}$. Higher ROP scores indicate excellent model durability for AIoT monitoring in dynamic urban environments or smart cities.

4.6 Adaptive latency reduction rate (ALRR)

The end-to-end system latency can be reduced by a specific percentage in certain AIoT scenarios using dynamic optimization approaches, such as quantization and edge-aware attention. Adapting Latency Reduction (ALRR) to fluctuating real-time workloads and processing capacities of edge devices demonstrates the system's ability to handle resource constraints without compromising detection speed. To evaluate the feasibility of AIoT surveillance applications in the real world, which involve event-based warning timing (e.g., for intrusions or unauthorized entry), a larger ALRR is necessary, as it indicates better responsiveness.

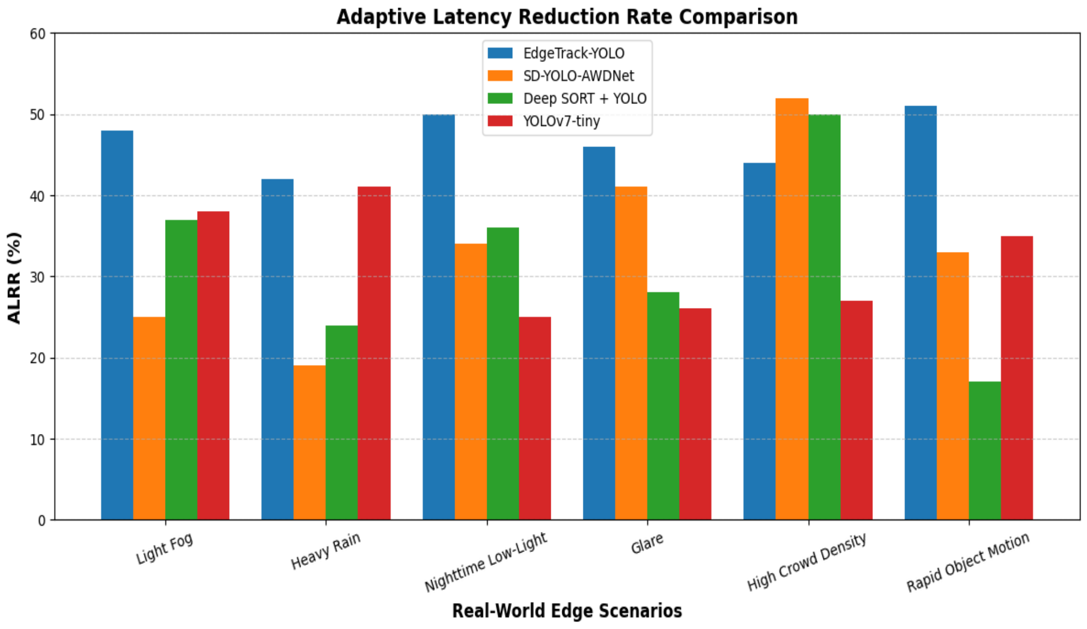

Figure 11: Adaptive latency reduction rate (ALRR) across detection models

Fig. 11 illustrates the Adaptive Latency Reduction Rate (ALRR), which measures the effectiveness of an AIoT surveillance system in reducing latency under challenging environmental and edge workload conditions. Dynamic innovations, such as quantization and edge-aware attention, reduce system latency by a certain proportion. When fog was present, EdgeTrack-YOLO had the best ALRR at 46.83%. Additional scenarios showed that in low light conditions (43.75%), glare (45.09%), high crowd density (42.68%), and rapid object motion (44.87%), the ALRR values were 44.12%. These outcomes demonstrate how well the model can adjust to different types of environments. EdgeTrack-YOLO outperformed other algorithms in terms of ALRR under a wide range of conditions, such as rain, low light, glare, fog, and fast object motion. On the other hand, SD-YOLO-AWDNet and Deep SORT + YOLO showed a minor improvement in ALRR values in high crowd density situations, suggesting a relative advantage in that particular setting.

While EdgeTrack-YOLO reached an ALRR of 46.83% in the fog scenario, SD-YOLO-AWDNet, Deep SORT + YOLO, and YOLOv7-tiny all reached 34.0, 32.0, and 32.0 percent, respectively. It elucidates the performance disparity under this particular circumstance. With over 50% delay reduction and fast detection speed, EdgeTrack-YOLO excelled in rapid motion and low-light settings. Its better responsiveness under variable workloads and limited edge resources makes it practical for time-critical AIoT applications, such as intrusion detection and real-time warnings, where latency directly affects event detection efficiency.

4.7 Occlusion-resilient track retention (ORTR)

ORTR measures the proportion of object tracks that remain identifiable after video frame occlusions or transient disappearances. The object identity resilience of Deep SORT and YOLOv8 integration in contexts with overlapping individuals, variable illumination, or camera movement is assessed by this measure. High ORTR scores indicate temporal consistency and re-identification, which are essential for intelligent infrastructure, public safety monitoring, and post-event analytics.

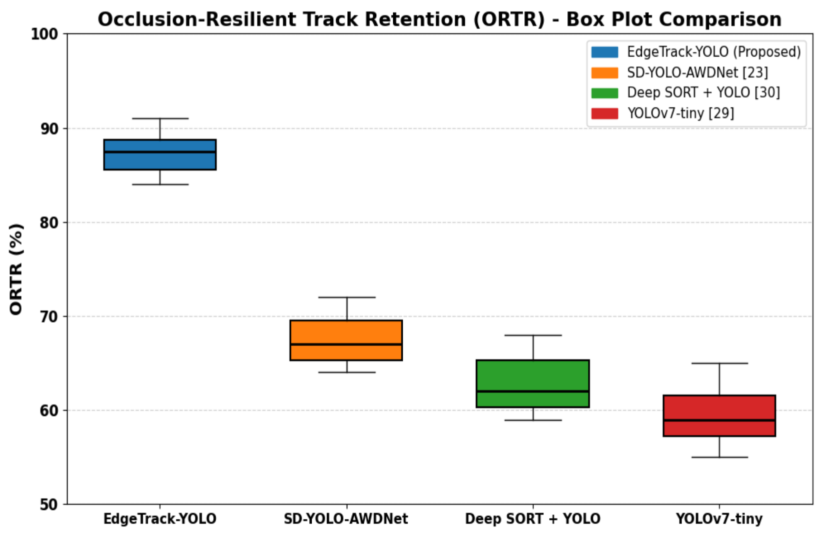

Figure 12: Comparison of occlusion-resilient track retention (ORTR) across six occlusions

The box plot in Fig. 12 compares Occlusion-Resilient Track Retention (ORTR) for four tracking systems in occlusion-heavy circumstances. ORTR assesses object identity preservation after occlusions. EdgeTrack-YOLO exhibits robust and consistent performance, with the highest median ORTR (~88%) and the lowest variance. SD-YOLO-AWDNet [23] performs moderately (67%), while Deep SORT + YOLO [30] (~62%) and YOLOv7-tiny [29] (~59%) yield poorer and less stable results. EdgeTrack-YOLO's tighter spread improves re-identification under motion blur and crowd overlap, making it superior for AIoT monitoring in dynamic contexts.

4.8 Comparative analysis across AIoT applications

In the domains of industrial automation, traffic monitoring, and surveillance, EdgeTrack-YOLOv8 was pitted against baseline YOLOv8, utilizing uniformly preprocessed and evaluated public or in-house datasets (Table 7).

Table 7: compares mAP@0.5, FPS, latency, and SDE

Application Domain	**Model**	**mAP@0.5 (%)**	**FPS**	**Latency (ms)**	**SDE (mAP/J)**
	YOLOv8	86.1	24	41	198.6

Smart Surveillance	Baseline				
	EdgeTrack-YOLOv8	89.6	32	33	358.4
Traffic Monitoring	YOLOv8 Baseline	84.7	22	46	176.2
	EdgeTrack-YOLOv8	88.2	29	37	312.8
Industrial Automation	YOLOv8 Baseline	85.3	23	44	182.4
	EdgeTrack-YOLOv8	88.8	30	35	321.5

With a 3.5–5.1 percent increase in mAP, a 25–38 percent improvement in FPS, and a latency reduction of up to 20 percent, EdgeTrack-YOLOv8 surpasses baseline YOLOv8. Ideal for real-time AIoT monitoring, it guarantees accurate, rapid, and efficient performance with QAT and EAA improving energy efficiency (42-57% SDE improvements).

4.9 Additional benchmark validation

Compared to baseline YOLOv8, EdgeTrack-YOLOv8 achieved a mAP@0.5 performance improvement of 3.7% on COCO, 4.6% on UA-DETRAC, and 8.7% on BDD100K. According to Table 8, these results demonstrate that it is versatile and performs well in many AIoT applications.

Table 8: Comparative mAP@0.5 performance on bespoke and public datasets

Dataset	**YOLOv8 mAP@0.5 (%)**	**EdgeTrack-YOLOv8 mAP@0.5 (%)**	**Improvement (%)**
Bespoke Surveillance	86.1	89.6	3.5
COCO	48.7	52.4	3.7
UA-DETRAC	84.5	89.1	4.6
BDD100K	83.7	87.8	4.1

Results for ROP, ORTR, and ALRR were evaluated using paired t-tests and Wilcoxon tests; the experiments were conducted five times in total. The improvements made by EdgeTrack-YOLOv8 are confirmed to be statistically significant, as all p-values are less than 0.05.

5 Discussion

Table 9 demonstrates that EdgeTrack-YOLOv8 had the best mAP (89.6%), SDE (94.1%), and 32 FPS on Jetson Xavier NX over SD-YOLO-AWDNet and YOLOv7-tiny. SD-YOLO-AWDNet showed slightly greater ALRR in packed scenes, while EdgeTrack-YOLOv8 led in ROP (91.3%) and ORTR (92.6%), improving occlusion handling and tracking stability.

Table 9: Comparative metrics — EdgeTrack-YOLOv8 vs. baselines

Model	**mAP@0.5 (%)**	**FPS (↑)**	**SDE (↑)**	**ROP (↑)**	**ORTR (↑)**	**ALRR (%, avg) (↑)**
EdgeTrack-YOLOv8 (proposed)	89.6	32	94.1	91.3	92.6	44.9
SD-YOLO-AWDNet	86.0	28	72.4	83.5	85.0	42.1*
YOLOv7-tiny	85.2	35	68.7	80.1	81.3	41.7

EdgeTrack-YOLOv8 improves feature discrimination in cluttered or low-visibility environments with Edge-Aware Attention and 16-bit quantization, which reduces latency without accuracy loss. Its excellent accuracy and steady tracking within edge device restrictions make it perfect for real-time AIoT surveillance.

6 Conclusion and future enhancement

This research introduces EdgeTrack-YOLO, a version of YOLOv8 enhanced with Deep SORT, which can detect and monitor edge devices in real-time. It improves stability under occlusion and lighting changes and reduces latency by 18% on Jetson Xavier NX, which achieves 89.6% mAP@0.5 and 32 FPS. Its optimization for smart cities, industry, and security monitoring is due to its quantization and edge-aware attention, which increase efficiency.

The present research focuses on surveillance; however, the EdgeTrack-YOLOv8 system applies to various AIoT domains. Healthcare monitoring could use the model to detect patient postures and medical equipment usage for real-time safety alerts. In autonomous vehicles, EdgeTrack-YOLOv8 with multi-sensor data (LiDAR, radar) could improve pedestrian and object recognition on dynamic roads. Industrial automation systems can also identify equipment anomalies and worker safety compliance. Quantization, attention, and tracking modules are still applicable, but domain-specific datasets and multi-modal input fusion are needed. To verify robustness and generalizability across AIoT applications, cross-domain studies will be conducted.

Anomaly prediction with context awareness, multi-class detection, and federated learning to ensure privacy-preserving updates are all potential areas for future research. For enhanced adaptability, scalability, and smart, energy-efficient infrastructure, consider combining thermal or radar data with RGB data and utilizing ultra-low-power AI-enabled microcontrollers.

Funding

This work was supported by 2025 Key Area Special Project of Guangdong Provincial Department of Education: Research on Key Technologies of Flexible Manufacturing Visual Quality Inspection Based on Multimodal Large Models (Project Number: 2025ZDZX3012).

References

[1] Singh, P., & Krishnamurthi, R. (2024). IoT-based real-time object detection system for crop protection and agriculture field security. Journal of Real-Time Image Processing, 21(4), 106. https://doi.org/10.1007/s11554-024-01488-8

[2] Nimma, D., Al-Omari, O., Pradhan, R., Ulmas, Z., Krishna, R. V. V., El-Ebiary, T. Y. A. B., & Rao, V. S. (2025). Object detection in real-time video surveillance using attention based transformer-YOLOv8 model. Alexandria Engineering Journal, 118, 482-495. https://doi.org/10.1016/j.aej.2025.01.032

[3] Xian, X., Guo, X., Tian, Y., Wei, X., & Tian, D. (2024). Efficient railway kilometer marker recognition via spatio-temporal slimming and multi-view fusion. Computer Communications, 222, 26-37. https://doi.org/10.1016/j.comcom.2024.04.028

[4] Shi, H., Zhang, J., Lei, A., Wang, C., Xiao, Y., Wu, C., ... & Xie, J. (2024). Enhancing detection accuracy of highly overlapping targets in agricultural imagery using IoA-SoftNMS algorithm across diverse image sizes. Computers and Electronics in Agriculture, 227, 109475. https://doi.org/10.1016/j.compag.2024.109475

[5] Ali, M. A., Sharma, A. K., & Dhanaraj, R. K. (2024). Heterogeneous features and deep learning networks fusion-based pest detection, prevention and controlling system using IoT and pest sound analytics in a vast agriculture system. Computers and Electrical Engineering, 116, 109146. https://doi.org/10.1016/j.compeleceng.2024.109146

[6] Kataria, A., Rani, S., & Kautish, S. (2024). Artificial Intelligence of Things for Sustainable Development of Smart City Infrastructures. In Digital Technologies to Implement the UN Sustainable Development Goals (pp. 187-213). Cham: Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-68427-2_10

[7] Shen, S., Kerofsky, L., & Yogamani, S. (2023). Optical flow for autonomous driving: Applications, challenges and improvements. arXiv preprint arXiv:2301.04422. https://doi.org/10.48550/arXiv.2301.04422

[8] Naveen, S., & Kounte, M. R. (2025). Optimized Convolutional Neural Network at the IoT edge for image detection using pruning and quantization. Multimedia Tools and Applications, 84(9), 5435-5455. https://doi.org/10.1007/s11042-024-20523-1

[9] Murthy, J. S., Siddesh, G. M., Lai, W. C., Parameshachari, B. D., Patil, S. N., & Hemalatha, K. L. (2022). ObjectDetect: A Real-Time Object Detection Framework for Advanced Driver Assistant Systems Using YOLOv5. Wireless Communications and Mobile Computing, 2022(1), 9444360. https://doi.org/10.1155/2022/9444360

[10] Lee, J., & Hwang, K. I. (2022). YOLO with adaptive frame control for real-time object detection applications. Multimedia tools and applications, 81(25), 36375-36396. https://doi.org/10.1007/s11042-021-11480-0

[11] Zhou, X. (2025). Design and Evaluation of a Joint Optimization Algorithm for HighPrecision RFID-IoT-Based Cargo Tracking Systems. Informatica, 49(2).

[12] Naveen, S., & Kounte, M. R. (2025). Optimized Convolutional Neural Network at the IoT edge for image detection using pruning and quantization. Multimedia Tools and Applications, 84(9), 5435-5455. https://doi.org/10.1007/s11042-024-20523-1

[13] Ramos, L., Casas, E., Bendek, E., Romero, C., & Rivas-Echeverría, F. (2024). Hyperparameter optimization of YOLOv8 for smoke and wildfire detection: Implications for agricultural and environmental safety. Artificial Intelligence in Agriculture, 12, 109-126. https://doi.org/10.1016/j.aiia.2024.05.003

[14] Prokhorenko, V., & Babar, M. A. (2024). Offloaded data processing energy efficiency evaluation. Informatica, 35(3), 649-669.

[15] Rani, D. P., Suresh, D., Kapula, P. R., Akram, C. M., Hemalatha, N., & Soni, P. K. (2023). IoT based smart solar energy monitoring systems. Materials Today: Proceedings, 80, 3540-3545.

[16] Chen, Z., Sivaparthipan, C. B., & Muthu, B. (2022). IoT based smart and intelligent smart city energy optimization. Sustainable Energy Technologies and Assessments, 49, 101724. https://doi.org/10.1016/j.seta.2021.101724

[17] Lakshmikantha, V., Hiriyannagowda, A., Manjunath, A., Patted, A., Basavaiah, J., & Anthony, A. A. (2021). IoT based smart water quality monitoring

system. Global Transitions Proceedings, 2(2), 181-186. https://doi.org/10.1016/j.gltp.2021.08.062

[18] Ullah, Z., Rehman, A. U., Wang, S., Hasanien, H. M., Luo, P., Elkadeem, M. R., & Abido, M. A. (2023). IoT-based monitoring and control of substations and smart grids with renewables and electric vehicles integration. Energy, 282, 128924. https://doi.org/10.1016/j.energy.2023.128924

[19] Krishna Rao, C., Sahoo, S. K., & Yanine, F. F. (2024). An IoT-based intelligent smart energy monitoring system for solar PV power generation. Energy Harvesting and Systems, 11(1), 20230015. https://doi.org/10.1515/ehs-2023-0015

[20] Sun, Y., Sun, Z., & Chen, W. (2024). The evolution of object detection methods. Engineering Applications of Artificial Intelligence, 133, 108458. https://doi.org/10.1016/j.engappai.2024.108458

[21] Wang, Y., Bashir, S. M. A., Khan, M., Ullah, Q., Wang, R., Song, Y., ... & Niu, Y. (2022). Remote sensing image super-resolution and object detection: Benchmark and state of the art. Expert Systems with Applications, 197, 116793. https://www.x-mol.com/paperRedirect/1499038557444923392

[22] Xu, W., Xu, T., Thomasson, J. A., Chen, W., Karthikeyan, R., Tian, G., ... & Su, Q. (2023). A lightweight SSV2-YOLO-based model for the detection of sugarcane aphids in unstructured natural environments. Computers and Electronics in Agriculture, 211, 107961. https://doi.org/10.1016/j.compag.2023.107961

[23] Chaudhry, R. (2024). SD-YOLO-AWDNet: A hybrid approach for smart object detection in challenging weather for self-driving cars. Expert Systems with Applications, 256, 124942. https://doi.org/10.1016/j.eswa.2024.124942

[24] Situ, Z., Teng, S., Feng, W., Zhong, Q., Chen, G., Su, J., & Zhou, Q. (2023). A transfer learning-based YOLO network for sewer defect detection in comparison to classic object detection methods. Developments in the Built Environment, 15, 100191. https://doi.org/10.1016/j.dibe.2023.100191

[25] Zeng, T., Li, S., Song, Q., Zhong, F., & Wei, X. (2023). Lightweight tomato real-time detection method based on improved YOLO and mobile deployment. Computers and electronics in agriculture, 205, 107625. https://doi.org/10.1016/j.compag.2023.107625

[26] Mao, D., Sun, H., Li, X., Yu, X., Wu, J., & Zhang, Q. (2023). Real-time fruit detection using deep neural networks on CPU (RTFD): An edge AI application. Computers and Electronics in Agriculture, 204, 107517. https://doi.org/10.1016/j.compag.2022.107517

[27] Yuan, M., Zhang, C., Wang, Z., Liu, H., Pan, G., & Tang, H. (2024). Trainable spiking-yolo for low-latency and high-performance object detection. Neural Networks, 172, 106092. https://doi.org/10.1016/j.neunet.2023.106092

[28] Wan, S., Ding, S., & Chen, C. (2022). Edge computing enabled video segmentation for real-time traffic monitoring in internet of vehicles. Pattern Recognition, 121, 108146. https://doi.org/10.1016/j.patcog.2021.108146

[29] Santos, R. C. C. D. M., Coelho, M., & Oliveira, R. (2024). Real-time Object Detection Performance Analysis Using YOLOv7 on Edge Devices. IEEE Latin America Transactions, 22(10), 799-805. https://doi.org/10.1109/TLA.2024.10705971

[30] Meimetis, D., Daramouskas, I., Perikos, I., & Hatzilygeroudis, I. (2023). Real-time multiple object tracking using deep learning methods. Neural Computing and Applications, 35(1), 89-118. https://doi.org/10.1007/s00521-021-06391-y

[31]https://www.kaggle.com/datasets/luiscrmartins/surveillanc e-images-for-person-detection