Journal of Building Design and Environment

Science Exploration

Open Access Technical Note

Lightweight head-based multi-object tracking in crowded built environments

Yue Cui[1], Zijie Gao[2,3], Kailai Sun[3,4,*]

[1]Chongqing General Hospital, Chongqing University, Chongqing 401147, China.
[2]National University of Singapore, Singapore 119077, Singapore.
[3]Singapore-MIT Alliance for Research and Technology Centre (SMART), Singapore 138602, Singapore.
[4]Urban Mobility Lab, Massachusetts Institute of Technology, Cambridge, MA 02139, USA.

***Correspondence to:** Kailai Sun, Singapore-MIT Alliance for Research and Technology Centre (SMART), Singapore 138602, Singapore. E-mail: skl24@mit.edu

Received: July 16, 2026 **Accepted:** October 15, 2026 **Published:** October 16, 2026

Cite this article: Cui Y, Gao Z, Sun K. Lightweight head-based multi-object tracking in crowded built environments. J Build Des Environ. 2026;4:202630. https://doi.org/10.70401/jbde.2026.0046

Abstract

Multi-object tracking (MOT) provides an effective approach for occupant monitoring in intelligent built environments. Real-time MOT is challenging on resource-constrained edge devices. This study establishes a systematic evaluation framework for lightweight head-based MOT, examining structured channel pruning and quantization across the complete detection-tracking pipeline. Experiments on Cchead use five detectors and multiple trackers. We comprehensively compare different pruning criteria and pruning strategies, showing that layer-wise pruning with fine-tuning reduces complexity while preserving tracking performance. We further compare post-training quantization (PTQ) and quantization-aware training (QAT) in terms of accuracy preservation, tracking stability, and deployment efficiency; QAT improves $AP_{50:95}$ by up to 13.33% for PPYOLOE-S. Practical deployment on Jetson Nano, Jetson TX2, and Intel Neural Compute Stick 2 achieves latency reductions exceeding 48%. Robustness tests under Gaussian noise show up to 158.41% relative improvement with QAT. These results characterize trade-offs among efficiency, tracking performance, and resilience for edge MOT deployment.

Keywords: Head-based multi-object tracking, edge deployment, structured channel pruning, quantization-aware training, lightweight deep learning

1. Introduction

Accurate occupancy information is important for understanding building usage patterns and supporting intelligent built environments[1]. With the development of smart buildings and sensing technologies, vision-based monitoring has become an effective approach for capturing occupancy distribution in indoor and public spaces[2,3]. Among these approaches, head-based multi-object tracking (MOT) provides rich spatial-temporal information for individual-level monitoring in crowded environments, where full-body observation is often affected by severe occlusion. Head-based tracking has attracted increasing attention because human heads remain relatively stable and observable under dense crowd conditions, providing a robust representation for large-scale pedestrian monitoring in built environments[4,5].

Recent advances in deep learning have significantly improved MOT performance through powerful object detectors and advanced data association algorithms. Modern tracking-by-detection frameworks typically rely on high-capacity convolutional neural networks (CNNs) or transformer-based architectures to achieve accurate detection and identity association. Despite their impressive performance, these models usually require substantial computational resources, memory capacity, and power consumption, making them difficult to deploy on resource-constrained edge devices. This limitation is particularly critical for real-world surveillance scenarios, where low-latency inference, energy efficiency, and on-device processing are required to reduce dependence on cloud computing and protect data privacy.

In crowded built environments, distributed camera systems must balance pedestrian visibility with the computing resources available at each camera location. Head-based tracking can provide pedestrian trajectories when full-body visibility is limited by occlusion. These trajectories may support pedestrian flow analysis, congestion monitoring, and occupancy estimation. Evaluating compression at the system level helps determine whether reductions in detector complexity translate into lower deployment latency without excessive loss of tracking accuracy.

© The Author(s) 2026. This is an Open Access article licensed under a Creative Commons Attribution 4.0 International License (https://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, sharing, adaptation, distribution and reproduction in any medium or format, for any purpose, even commercially, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

To enable efficient deployment of deep learning models on edge platforms, various model compression techniques have been investigated, including knowledge distillation[6], network pruning[7], and quantization[8]. Among these approaches, structured channel pruning is particularly attractive because it directly reduces the physical network architecture by removing redundant channels, thereby decreasing floating-point operations (FLOPs) and accelerating inference without requiring specialized hardware support. However, excessive channel removal may disrupt feature representation and cause significant performance degradation, especially for compact detection networks with limited redundancy. Therefore, selecting appropriate pruning criteria and developing effective fine-tuning strategies remain important challenges for maintaining accuracy after compression.

Another widely adopted optimization technique is neural network quantization, which converts floating-point parameters and activations into low-bit integer representations to reduce memory footprint and improve inference efficiency. Although post-training quantization (PTQ) enables simple deployment, it often introduces considerable accuracy degradation because the original model has not adapted to quantization-induced numerical errors. Quantization-aware training (QAT) addresses this limitation by simulating quantization effects during training, allowing the model to learn more robust representations while preserving the efficiency benefits of low-precision inference. Nevertheless, the effects of pruning and QAT on MOT performance, robustness, and real-world edge deployment efficiency have not been sufficiently investigated.

Previous studies have mainly focused on improving individual components of MOT systems or evaluating compression techniques on image classification and object detection tasks. Comprehensive investigations that jointly consider detector compression, tracking performance preservation, deployment latency, and robustness under practical perturbations remain limited, particularly for head-based MOT in crowded scenes. Moreover, different lightweight detectors may exhibit distinct responses to pruning and quantization due to differences in architecture and model capacity, requiring systematic evaluation rather than adopting a single compression strategy.

In this study, we systematically evaluate lightweight optimization techniques for head-based MOT under edge deployment constraints. Our evaluation covers the complete MOT pipeline and extends to practical deployment and robustness analysis. Structured channel pruning and quantization are evaluated across multiple detector architectures, tracking algorithms, and edge platforms. This comprehensive evaluation provides a clear understanding of how model compression affects both MOT performance and practical deployment.

The main contributions of this work are summarized as follows:

• We establish a systematic evaluation of lightweight head-based MOT by investigating the effects of structured channel pruning and quantization on the complete detection-tracking pipeline.

• We conduct a comprehensive analysis of structured channel pruning, including different channel importance criteria and pruning strategies, and characterize their impacts on detection and tracking performance across multiple detector architectures.

• We compare models obtained using 8-bit PTQ and QAT in terms of detection accuracy and tracking performance, and evaluate the deployment latency of QAT-optimized models.

• We measure end-to-end MOT latency on three edge platforms and separately evaluate detector average precision at an intersection over union (IoU) threshold of 0.5 (AP_{50}) under Gaussian input perturbations.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 presents the proposed framework, dataset, and experimental settings. Section 4 reports the experimental results. Section 5 discusses the findings and limitations, and Section 6 concludes the paper.

2. Related Work

2.1 MOT in crowded scenes

MOT has been extensively studied in computer vision due to its importance in video surveillance, pedestrian analysis, and intelligent transportation systems[9]. Existing MOT methods can generally be categorized into tracking-by-detection[10,11] and joint detection-tracking paradigms[12-14]. Tracking-by-detection frameworks first obtain object detections from individual frames and then perform temporal association to maintain consistent identities. With the rapid development of deep learning-based object detectors, modern MOT systems have achieved significant improvements in detection accuracy and tracking robustness.

In crowded environments, however, pedestrian tracking remains challenging due to severe occlusion, frequent interactions between targets, and large variations in object scale and appearance[4]. Compared with full-body detection, head-based tracking has been demonstrated to be more suitable for dense crowd scenarios because human heads are less affected by occlusion and remain visible under complex viewing conditions. Recent studies have explored head detection and tracking approaches for applications such as crowd counting, surveillance, and public safety monitoring[4,5].

Despite their effectiveness, state-of-the-art MOT systems typically rely on high-capacity detection networks, resulting in considerable computational costs. For example, tracking-by-detection frameworks based on YOLO series detectors[15,16] combined with advanced association algorithms such as ByteTrack[17], Observation-Centric SORT (OC-SORT)[18], and BoT-SORT[19] can achieve excellent tracking performance, but their computational requirements limit deployment on resource-constrained edge devices. Therefore, developing lightweight MOT systems that balance accuracy and efficiency remains an important research direction.

2.2 Lightweight object detection and model compression

Deep neural networks have achieved remarkable success in object detection; however, their increasing model size and computational complexity present significant challenges for real-time edge applications.Recent studies have further advanced deep feature learning for challenging visual restoration and enhancement tasks through adaptive representation learning and multi-level feature fusion strategies[20,21]. However, the increasing model size and computational complexity of deep networks present significant challenges for real-time edge applications. To address this issue, various model compression techniques have been investigated, including knowledge distillation[6,22], lightweight network architecture design[23-25], pruning, and quantization[26,27]. Recent real-time detectors have further improved the accuracy-efficiency trade-off through architecture-level redesign. YOLOv9 introduces Programmable Gradient Information and Generalized Efficient Layer Aggregation Network (GELAN) to improve information preservation and parameter utilization[28], while YOLOv10 adopts non-maximum suppression (NMS)-free end-to-end training and efficiency-oriented model design to reduce inference overhead[29]. Gold-YOLO enhances multi-scale feature fusion through a Gather-and-Distribute mechanism[30], whereas Real-Time DEtection TRansformer (RT-DETR) demonstrates that transformer-based detectors can also achieve real-time performance through an efficient hybrid encoder and improved query selection[31]. These advances highlight the rapid evolution of lightweight detector architectures and motivate further evaluation of compression strategies on newer models.

Among these approaches, network pruning is one of the most widely adopted methods for reducing computational redundancy[32]. Unstructured pruning removes individual weights with small importance values, achieving high sparsity but requiring specialized hardware or sparse computation libraries for acceleration. In contrast, structured pruning removes entire filters or channels, directly reducing the network architecture and improving practical inference efficiency on conventional hardware platforms.

Various criteria have been proposed to estimate channel importance during structured pruning. Norm-based approaches, such as L1-norm and L2-norm pruning[7], measure filter magnitude to identify less important channels and have been widely applied due to their simplicity and effectiveness. Furthermore, Filter Pruning via Geometric Median (FPGM)[33] introduced a redundancy-aware criterion by removing filters located near the geometric median of the filter distribution. Although structured pruning can substantially reduce computational costs, excessive pruning may damage feature representations and cause significant accuracy degradation. Therefore, fine-tuning after pruning is commonly required to recover model performance.

Previous pruning studies have mainly focused on image classification and object detection benchmarks, while their influence on complete MOT pipelines, including detection and identity association, has been less investigated. In particular, different lightweight detectors may exhibit different sensitivity to channel removal due to variations in network capacity and architecture design. A systematic analysis of pruning strategies for head-based MOT remains limited.

2.3 Quantization and edge deployment

Model quantization is another effective approach for deploying deep learning models on resource-constrained devices. By converting floating-point weights and activations into low-bit integer representations, quantization reduces memory consumption and accelerates inference through efficient integer arithmetic operations. Among different quantization schemes, 8-bit integer quantization has become a widely adopted solution for practical edge deployment[8,34].

Existing quantization approaches can generally be divided into PTQ and QAT. PTQ directly converts a pretrained full-precision model into a low-precision representation without additional training. Although PTQ is simple and computationally efficient, it may introduce considerable performance degradation, especially for sensitive detection models. QAT addresses this limitation by simulating quantization effects during training through fake quantization operations, enabling the model to adapt to quantization errors and maintain higher accuracy after deployment.

Recent studies have demonstrated the effectiveness of QAT in classification and object detection tasks. However, the impact of quantization on MOT performance remains insufficiently explored. Since MOT performance depends not only on detection accuracy but also on temporal association stability, small changes in detector outputs may influence identity consistency and tracking robustness. Furthermore, comprehensive evaluations combining pruning, QAT, latency measurement, and robustness analysis on real edge devices are still lacking.

In this work, we bridge these gaps by systematically investigating structured pruning and QAT for head-based MOT under edge deployment constraints. Unlike previous studies that focus on individual compression techniques, we provide an integrated evaluation covering detection accuracy, tracking performance, inference latency, and robustness across multiple edge computing

platforms.

3. Methods

3.1 Overall framework

This work aims to develop an efficient head-based MOT framework for edge devices by reducing computational complexity while preserving detection and tracking accuracy. The proposed framework consists of two complementary optimization strategies: structured channel pruning and QAT. Figure 1 illustrates the overall detector compression and deployment pipeline.

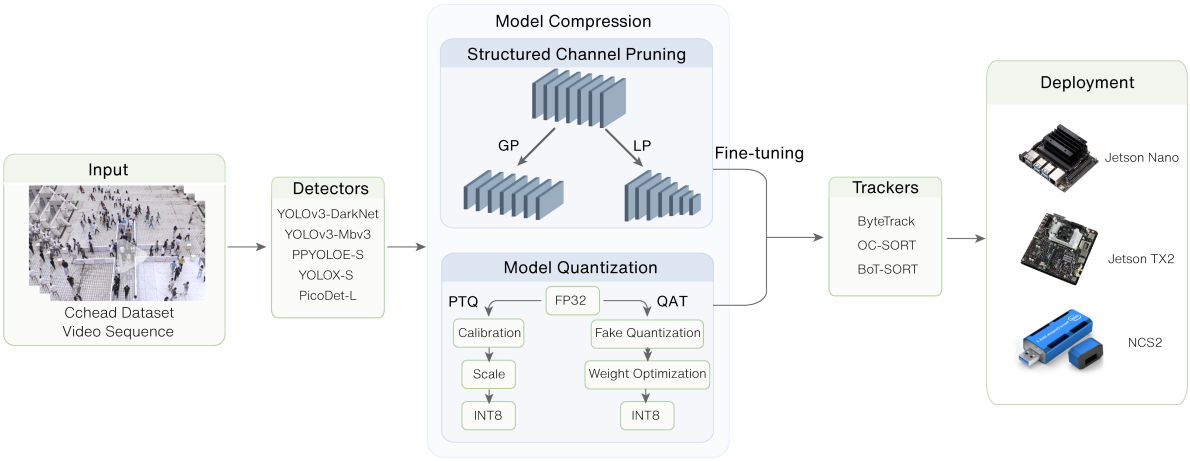


Figure 1. Overview of the proposed detector compression and deployment pipeline for edge-based MOT. MOT: multi-object tracking; OC-SORT: observation-centric SORT; PTQ: post-training quantization; QAT: quantization-aware training; NCS2: Intel Neural Compute Stick 2; INT8: 8-bit integer; FP32: 32-bit floating-point.

Structured channel pruning is applied to remove redundant convolutional channels and reduce model complexity, and the pruned networks are subsequently fine-tuned to recover accuracy degradation caused by reduced model capacity. In parallel, QAT is applied to the baseline detectors to adapt them to low-precision inference by simulating quantization effects during training.

The optimized detector is then integrated into a tracking-by-detection pipeline, where detected heads are associated across consecutive frames to generate complete trajectories. The overall framework enables efficient MOT inference under the constraints of limited computational resources and real-time deployment requirements.

3.2 Structured channel pruning

Structured channel pruning reduces computational complexity by removing redundant convolutional channels. Unlike unstructured pruning, which produces sparse weights requiring specialized acceleration libraries, structured pruning directly modifies network architecture and therefore provides practical speedup on edge devices.

For a convolutional layer with weight tensor shown in Equation (3.1):

$$W \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}} \times K \times K} \tag{3.1}$$

each output channel is treated as an independent filter. The importance score of each channel is calculated using different criteria, as defined in Equation (3.2), Equation (3.3), and Equation (3.4):

$$S_i^{L1} = \sum_j |W_{i,j}| \tag{3.2}$$

where S_i^{L1} represents the importance score of the i-th filter, and W_{ij} denotes the j-th weight element of the i-th filter.

$$S_i^{L2} = \sqrt{\sum_j W_{i,j}^2} \tag{3.3}$$

$$S_i^{FPGM} = \sum_{j=1}^{N} \|W_i - W_j\|_2 \tag{3.4}$$

where W_i and W_j denote the weight vectors of different filters, and N is the number of filters in the layer.

Two pruning strategies are compared in this study. Global pruning applies a uniform pruning ratio to all convolutional layers. In contrast, layer-wise pruning assigns a layer-specific pruning ratio to each convolutional layer, as shown in Equation (3.5):

$$C_l' = (1 - r_l)C_l \tag{3.5}$$

where C_l denotes the original number of channels in layer l, C_l' represents the remaining channels, and r_l is the pruning ratio of layer l. The layer-wise pruning ratios were determined according to the depth and channel width of each convolutional layer. Shallow layers were pruned conservatively or kept unchanged because they contain fewer channels and provide fundamental features for subsequent feature extraction. Wider intermediate and deep layers were assigned larger pruning ratios because they generally contain greater channel redundancy. In addition, the remaining number of channels was preferably kept as a power of two to facilitate efficient computation and memory access. Therefore, pruning ratios of 0.25, 0.50, 0.75, and 0.875 were preferentially considered. The specific pruning configuration was determined separately for each detector according to its architecture. For example, most selected deep layers of YOLOv3-DarkNet were pruned by 0.75, while several later layers were pruned by 0.875. Because the evaluated detectors differ in network depth, channel width, and structural redundancy, detector-specific pruning configurations were adopted, resulting in different levels of computational reduction across models.

The layer-wise pruning configuration for YOLOv3-DarkNet is provided in Table S1. Layers 52-57 and 59-64 were pruned by 75%, while layers 66-71 were pruned by 87.5%. No systematic ablation across pruning intensities was conducted. The selected configurations therefore represent the settings evaluated in this study and have not been established as optimal.

3.3 QAT

In addition to pruning, 8-bit QAT is applied to the baseline detectors to reduce memory consumption and inference cost.

For a floating-point tensor x uniform affine quantization is formulated as Equation (3.6):

$$x_q = \text{round}\left(\frac{x}{S}\right) + Z \tag{3.6}$$

where S and Z denote the scale factor and zero point, respectively. The de-quantization process is given in Equation (3.7):

$$x = S\,(x_q - Z) \tag{3.7}$$

During QAT, fake quantization operators are inserted into the training process to simulate 8-bit integer (INT8) inference, as shown in Equation (3.8), which is the composition of quantization and de-quantization:

$$x_{fq} = S \cdot \text{round}\left(\frac{x}{S}\right) \tag{3.8}$$

Although the forward propagation considers quantization effects, model parameters are still updated using 32-bit floating-point (FP32) gradients. The final optimization objective can be expressed as Equation (3.9):

$$W^* = \arg\min_W L\left(f\left(x; W_q\right), y\right) \tag{3.9}$$

where W_q denotes quantized parameters and L represents the detection loss.

3.4 Lightweight MOT pipeline

The optimized detector is integrated into a tracking-by-detection framework. For each video frame, the detector first generates head bounding boxes and confidence scores. These detections are subsequently associated across consecutive frames using a tracking algorithm to maintain consistent identities.

The tracking process can be formulated as Equation (3.10):

$$T_t = A\left(D_t, T_{t-1}\right) \tag{3.10}$$

where D_t denotes detections at frame t, T_{t-1} represents previous trajectories, and $A(\cdot)$ is the association function.

3.5 Dataset

Experiments were conducted on the Cchead dataset[5], a large-scale head detection and tracking dataset designed for crowded scene analysis. The dataset contains videos collected from surveillance scenarios with significant variations in crowd density, viewpoint, and illumination conditions. Representative examples are shown in Figure 2.

Compared with conventional pedestrian datasets, Cchead dataset focuses on head-level detection, which is more suitable for dense

crowd environments where full-body annotations are frequently affected by severe occlusions. The dataset contains 10 video sequences with 50,528 frames, 2,366,249 annotated head bounding boxes, and 2,358 trajectories. The training and test splits follow the official sequence-level protocol of the Cchead dataset benchmark[5], which avoids temporal leakage between adjacent frames.

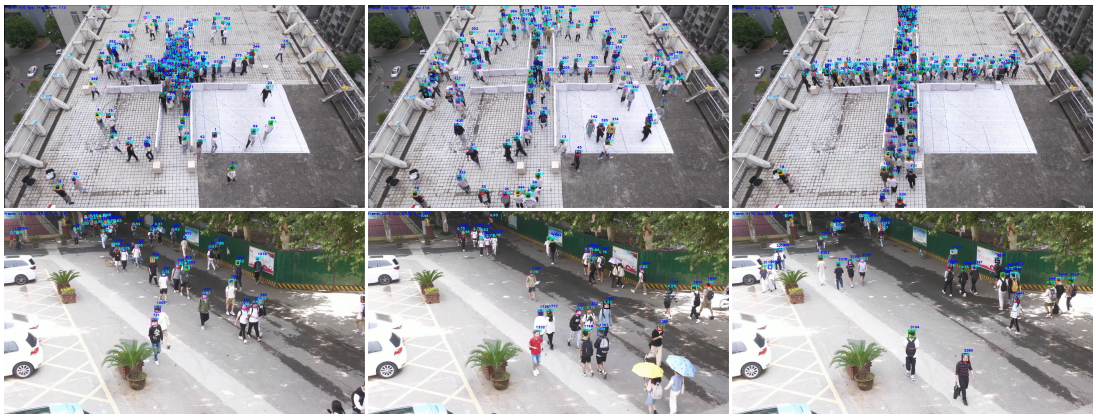


Figure 2. Representative examples from the Cchead dataset. Reproduced from reference[5]. CC BY 4.0. The dataset contains diverse surveillance scenarios with varying crowd densities, viewpoints, illumination conditions, and occlusion levels. Each head is annotated with a bounding box and a unique identity label for MOT. MOT: multi-object tracking.

Some videos were originally recorded using high-resolution cameras with frame rates higher than common surveillance systems. Considering that 25 frames per second (FPS) is widely adopted in practical monitoring applications, videos with higher frame rates were down-sampled to 25 FPS before training and evaluation.

3.6 Implementation details

Five representative object detectors with different architectures and computational scales were selected as baseline models, including YOLOv3-DarkNet[15], YOLOv3-MobileNetV3[35], PPYOLOE-S[36], YOLOX-S[16], and PicoDet-L[37]. For tracking evaluation, the detectors were combined with three representative tracking-by-detection algorithms: ByteTrack[17], OC-SORT[18], and BoT-SORT[19].

All models were initialized from pretrained weights and fine-tuned on the Cchead dataset. During pruning optimization, the compressed models were fine-tuned with a reduced learning rate to recover performance degradation. The pruned models were fine-tuned for 20 epochs with an initial learning rate of 1×10^{-5}, which decayed to 1×10^{-7} following a cosine schedule. Since the models pruned with different criteria converged to similar accuracy after fine-tuning, with differences in AP_{50} below 0.02, the L2-norm criterion was adopted for all subsequent pruning experiments for consistency. For QAT, 8-bit integer quantization was adopted for weights and activations, with per-channel weight quantization and moving-average activation range estimation (window size of 10,000); fake quantization modules were inserted during training to simulate INT8 inference. The PTQ baseline used settings matched to QAT wherever applicable. The 20-epoch schedule and learning rate reported above apply only to post-pruning fine-tuning. The FP32 baseline received no additional training after its initial fine-tuning on Cchead. The original records of the QAT training epochs and learning rate, and the source and size of the PTQ calibration dataset, are no longer available. Since QAT involved further training while the FP32 baseline did not, the comparison cannot separate the effects of quantization-aware adaptation from those of additional fine-tuning.

3.7 Evaluation metrics

For object detection evaluation, AP_{50} and average precision over IoU thresholds from 0.5 to 0.95 ($AP_{50:95}$) were adopted.

For MOT evaluation, Multiple Object Tracking Accuracy (MOTA)[38] and Identification F1 score (IDF1)[39] were used to evaluate tracking accuracy and identity consistency.

For edge deployment evaluation, computational complexity was measured using FLOPs, while end-to-end inference latency was recorded on embedded platforms. The latency includes detection, tracking association, and post-processing time.

3.8 Experimental environment

The experiments were conducted in two different environments for model development and edge deployment evaluation.

For model training and optimization, experiments were performed on a workstation running Ubuntu 20.04 (64-bit), equipped with AMD EPYC 7702 processors and NVIDIA RTX 3090 graphics processing units (GPUs). The models were implemented using Python 3 with PyTorch and PaddlePaddle frameworks, and Compute Unified Device Architecture (CUDA) acceleration was enabled for GPU computation.

For practical deployment evaluation, the optimized models were deployed on resource-constrained edge computing platforms, including NVIDIA Jetson Nano, NVIDIA Jetson TX2, and Intel Neural Compute Stick 2 (NCS2). On Jetson Nano and TX2, models were deployed with JetPack 4.6.1, CUDA 10.2, cuDNN 8.2.1, TensorRT 8.0.1, Paddle-Inference 2.3.2, and PaddleDetection 2.6. For NCS2, models were converted through Paddle-Lite 2.12 and OpenVINO, with a host laptop equipped with an AMD Ryzen 7 6800H central processing unit (CPU). Latency was measured with a batch size of one on a 10-second test video containing on average 60 heads per frame, covering the complete per-frame pipeline of detection, association, and post-processing. Since the three association algorithms exhibited no significant difference in association time, a single latency value is reported for each detector.

4. Results

4.1 Impact of structured channel pruning on detection and tracking performance

To investigate the effectiveness of structured channel pruning for lightweight MOT models, we first evaluated the influence of different channel importance criteria without fine-tuning. As shown in Table 1, directly removing channels resulted in substantial degradation across all detectors and pruning strategies. Specifically, AP_{50} decreased by more than 54% for all evaluated models, indicating that aggressive channel removal significantly affects feature representation when the network is not further optimized. Among different pruning criteria, the performance degradation varied across architectures, suggesting that the evaluated detectors exhibit different sensitivities to channel removal.

Table 1. Impact of different channel pruning criteria on detection performance without fine-tuning.

Detector	NP (AP_{50})	L1-LP		L2-LP		FPGM-LP	
		AP_{50}	Δ(%)	AP_{50}	Δ(%)	AP_{50}	Δ(%)
YOLOv3-DarkNet	0.756	0.320	-57.67	0.268	-64.55	0.342	-54.76
YOLOv3-MobileNetV3	0.676	0.264	-60.95	0.296	-56.21	0.212	-68.64
PPYOLOE-S	0.708	0.201	-71.61	0.168	-76.27	0.156	-77.97
YOLOX-S	0.819	0.170	-79.24	0.207	-74.73	0.185	-77.41
PicoDet-L	0.753	0.232	-69.15	0.195	-74.10	0.201	-73.31

NP denotes the original non-pruned model; L1-LP, L2-LP, and FPGM-LP represent layer-wise channel pruning based on L1-norm, L2-norm, and the FPGM criterion, respectively; Δ(%) indicates relative performance changes (%) compared with NP; Bold indicates the smallest performance degradation among the three pruning criteria; AP_{50}: average precision at an intersection over union (IoU) threshold of 0.5; FPGM: filter pruning via geometric median.

After fine-tuning, the compressed models recovered most of their detection performance, as presented in Table 2. Both layer-wise pruning and global pruning achieved considerable FLOPs reduction while maintaining competitive detection accuracy. Layer-wise pruning reduced computational complexity by 9.52-22.56% across different detectors, while global pruning achieved comparable reductions ranging from 11.27% to 20.96%. In terms of accuracy preservation, layer-wise pruning generally outperformed global pruning. For example, YOLOv3-DarkNet slightly improved its AP_{50} from 0.756 to 0.760 and $AP_{50:95}$ from 0.338 to 0.342 after layer-wise pruning and fine-tuning, whereas global pruning resulted in minor performance degradation. Similar trends were observed for other detectors, demonstrating that preserving the layer-wise structural balance is beneficial for maintaining detection capability after compression.

Table 2. Detection performance comparison between layer-wise and global channel pruning with fine-tuning.

Detector	FLOPs (G)			AP_{50}			$AP_{50:95}$		
	NP	LP+FT	GP+FT	NP	LP+FT	GP+FT	NP	LP+FT	GP+FT
YOLOv3-DarkNet	72.52	56.16 (-22.56)	57.31 (-20.96)	0.756	0.760 (+0.53)	0.751 (-0.66)	0.338	0.342 (+1.18)	0.336 (-0.59)
YOLOv3-MobileNetV3	19.21	16.78 (-12.66)	16.58 (-13.70)	0.676	0.673 (-0.44)	0.643 (-4.87)	0.266	0.262 (-1.50)	0.238 (-10.53)
PPYOLOE-S	7.57	6.43 (-15.06)	6.51 (-14.01)	0.708	0.680 (-3.95)	0.652 (-7.91)	0.315	0.296 (-6.03)	0.263 (-16.51)
YOLOX-S	12.51	11.32 (-9.52)	11.10 (-11.27)	0.819	0.765 (-6.59)	0.731 (-10.74)	0.407	0.362 (-11.06)	0.346 (-15.00)
PicoDet-L	6.72	5.81 (-13.54)	5.88 (-12.50)	0.753	0.732 (-2.79)	0.701 (-6.90)	0.335	0.321 (-4.18)	0.292 (-12.84)

NP denotes the original non-pruned model; LP+FT and GP+FT denote layer-wise and global channel pruning followed by fine-tuning, respectively; Values in parentheses indicate relative changes (%) compared with NP; Bold indicates the better result between LP+FT and GP+FT; FLOPs: floating-point operations; AP_{50}: average precision at an intersection over union (IoU) threshold of 0.5; $AP_{50:95}$: average precision over IoU thresholds from 0.5 to 0.95.

Since detection quality directly affects the subsequent association process in MOT, we further evaluated the impact of pruning on the complete MOT pipeline. The results in Table 3 show that the influence of pruning on tracking performance was highly dependent on the detector architecture. YOLOv3-DarkNet maintained or even improved tracking performance after pruning, with improvements observed across all evaluated trackers. For example, ByteTrack achieved increases of 0.61% and 0.36% in MOTA and IDF1, respectively. In contrast, compact detectors such as PPYOLOE-S and YOLOX-S experienced more noticeable performance degradation after pruning, indicating that lightweight architectures contain fewer redundant features and are more sensitive to channel removal. Overall, these results demonstrate that structured channel pruning combined with fine-tuning can effectively reduce model complexity while preserving practical MOT performance for the evaluated detectors, while the observed differences indicate that compression strategies should be adapted to detector-specific architectural characteristics.

Table 3. MOT performance after structured channel pruning and fine-tuning.

Detector	ByteTrack						OC-SORT						BoT-SORT					
	MOTA			IDF1			MOTA			IDF1			MOTA			IDF1		
	NP	LP+FT	Δ(%)	NP	LP+FT	Δ(%)	NP	LP+FT	Δ(%)	NP	LP+FT	Δ(%)	NP	LP+FT	Δ(%)	NP	LP+FT	Δ(%)
YOLOv3-DarkNet	65.2	65.6	+0.61	55.9	56.1	+0.36	64.1	64.5	+0.62	53.6	54.3	+1.31	66.1	66.4	+0.45	56.2	56.4	+0.36
YOLOv3-MobileNetV3	52.1	52.0	-0.19	47.8	48.1	+0.63	51.7	51.6	-0.19	47.1	46.9	-0.42	52.3	52.2	-0.19	48.4	48.1	-0.62
PPYOLOE-S	42.8	40.3	-5.84	41.4	38.5	-7.00	40.1	37.6	-6.23	38.5	36.1	-6.23	43.2	40.9	-5.32	41.7	40.5	-2.88
YOLOX-S	62.0	58.1	-6.29	56.0	53.5	-4.46	61.2	58.9	-3.76	53.2	51.0	-4.14	62.5	61.1	-2.24	57.3	54.1	-5.59
PicoDet-L	58.9	57.5	-2.38	45.8	45.3	-1.09	56.4	54.9	-2.66	44.2	43.1	-2.49	59.3	58.6	-1.18	46.3	45.2	-2.38

NP denotes the original non-pruned model; LP+FT denotes layer-wise channel pruning followed by fine-tuning; Δ(%) indicates relative changes (%) compared with NP; Bold indicates improvement over the NP baseline; MOTA: multiple object tracking accuracy; IDF1: Identification F1 score. OC-SORT: observation-centric SORT; NP: non-pruned; LP+FT: layer-wise pruning followed by fine-tuning.

4.2 Effectiveness of QAT for edge-based MOT deployment

In addition to pruning, quantization was investigated as a parallel optimization strategy to improve deployment efficiency. We first compared QAT with PTQ, both applied to the original FP32 detectors, in terms of detection performance. As shown in Table 4, QAT consistently achieved better performance preservation than PTQ across different detectors. For AP50, QAT maintained comparable performance to the FP32 baseline, with improvements observed in YOLOv3-DarkNet (+3.17%) and PPYOLOE-S (+4.10%). Similarly, $AP_{50:95}$ was improved for YOLOv3-DarkNet (+1.48%), PPYOLOE-S (+13.33%), and PicoDet-L (+1.49%). In contrast, PTQ resulted in substantial accuracy degradation for all evaluated detectors, with AP_{50} decreases ranging from 18.65% to 45.15%. Under the evaluated settings, the QAT models achieved higher detection accuracy than the PTQ models. However, the FP32 baseline received no additional training, so improvements over this baseline cannot be attributed to quantization-aware adaptation alone. The unavailable QAT training and PTQ calibration details also limit reproducibility.

Table 4. Comparison of detection performance under QAT and PTQ.

Detection Model	AP_{50}			$AP_{50:95}$		
	FP32	QAT	PTQ	FP32	QAT	PTQ
YOLOv3-DarkNet	0.756	0.780 (+3.17)	0.615 (-18.65)	0.338	0.343 (+1.48)	0.241 (-28.70)
YOLOv3-MobileNetV3	0.676	0.675 (-0.15)	0.457 (-32.40)	0.266	0.263 (-1.13)	0.192 (-27.82)
PPYOLOE-S	0.708	0.737 (+4.10)	0.533 (-24.72)	0.315	0.357 (+13.33)	0.213 (-32.38)
YOLOX-S	0.819	0.768 (-6.23)	0.591 (-27.84)	0.407	0.405 (-0.49)	0.256 (-37.10)
PicoDet-L	0.753	0.741 (-1.59)	0.413 (-45.15)	0.335	0.340 (+1.49)	0.179 (-46.57)

FP32 denotes the original 32-bit floating-point model; QAT and PTQ denote the models obtained using quantization-aware training and post-training quantization, respectively; Values in parentheses indicate relative changes (%) compared with the FP32 baseline; Bold indicates improvement over the FP32 baseline; QAT: quantization-aware training; PTQ: post-training quantization; AP_{50}: average precision at an intersection over union (IoU) threshold of 0.5; $AP_{50:95}$: average precision over IoU thresholds from 0.5 to 0.95; FP32: 32-bit floating-point.

To further evaluate whether quantization affects the complete MOT pipeline, we compared tracking performance before and after QAT. The results in Table 5 show that QAT maintained stable MOT performance across different tracking algorithms. For YOLOv3-DarkNet, QAT improved MOTA and IDF1 across most trackers, achieving improvements of up to 2.30% in ByteTrack MOTA and

1.96% in BoT-SORT IDF1. Similar improvements were observed for PPYOLOE-S, where all evaluated tracking metrics increased after QAT. For lightweight detectors such as YOLOX-S and PicoDet-L, QAT introduced minor performance degradation, but the decrease remained limited. Overall, the evaluated QAT models retained broadly comparable tracking performance to the FP32 baselines, with differences across detectors and trackers. These results describe tracking performance on the original evaluation data and do not establish tracking robustness under noisy inputs.

Table 5. MOT performance before and after QAT.

Detector	ByteTrack				OC-SORT				BoT-SORT			
	MOTA		IDF1		MOTA		IDF1		MOTA		IDF1	
	FP32	QAT	FP32	QAT	FP32	QAT	FP32	QAT	FP32	QAT	FP32	QAT
YOLOv3-DarkNet	65.2	66.7 (+2.30)	55.9	56.4 (+0.89)	64.1	65.5 (+2.18)	53.6	52.7 (-1.68)	66.1	67.1 (+1.51)	56.2	57.3 (+1.96)
YOLOv3-MobileNetV3	52.1	52.4 (+0.58)	47.8	48.2 (+0.84)	51.7	51.9 (+0.39)	47.1	47.3 (+0.42)	52.3	52.3 (0.00)	48.4	48.3 (-0.21)
PPYOLOE-S	42.8	43.4 (+1.40)	41.4	41.7 (+0.72)	40.1	40.7 (+1.50)	38.5	39.7 (+3.12)	43.2	43.5 (+0.69)	41.7	41.8 (+0.24)
YOLOX-S	62.0	60.5 (-2.42)	56.0	54.6 (-2.50)	61.2	59.2 (-3.27)	53.2	51.3 (-3.57)	62.5	61.2 (-2.08)	57.3	55.7 (-2.79)
PicoDet-L	58.9	58.2 (-1.19)	45.8	45.6 (-0.44)	56.4	55.9 (-0.89)	44.2	43.4 (-1.81)	59.3	58.5 (-1.35)	46.3	45.2 (-2.38)

FP32 denotes the original 32-bit floating-point model, whereas QAT denotes the model obtained using quantization-aware training; Values in parentheses indicate relative changes (%) compared with the FP32 baseline; Bold indicates improvement over the FP32 baseline; MOTA: multiple object tracking accuracy; IDF1: identification F1 score; QAT: quantization-aware training; OC-SORT: observation-centric SORT; FP32: 32-bit floating-point.

Finally, deployment efficiency was evaluated on the three edge platforms. As shown in Table 6, pruning produced modest latency reductions, whereas the QAT-optimized models showed larger reductions under the tested deployment configurations. On Jetson Nano, end-to-end latency decreased by 32.4%-48.1% relative to the original models. The largest reduction on Jetson TX2 was 47.1%. For the models with both baseline and QAT results on NCS2, reductions ranged from 44.2% to 44.6%. The accuracy-latency trade-offs are illustrated in Figure 3. These measurements show pipeline-level improvements, but actual operator precision was not verified. The latency reductions therefore cannot be attributed specifically to INT8 execution.

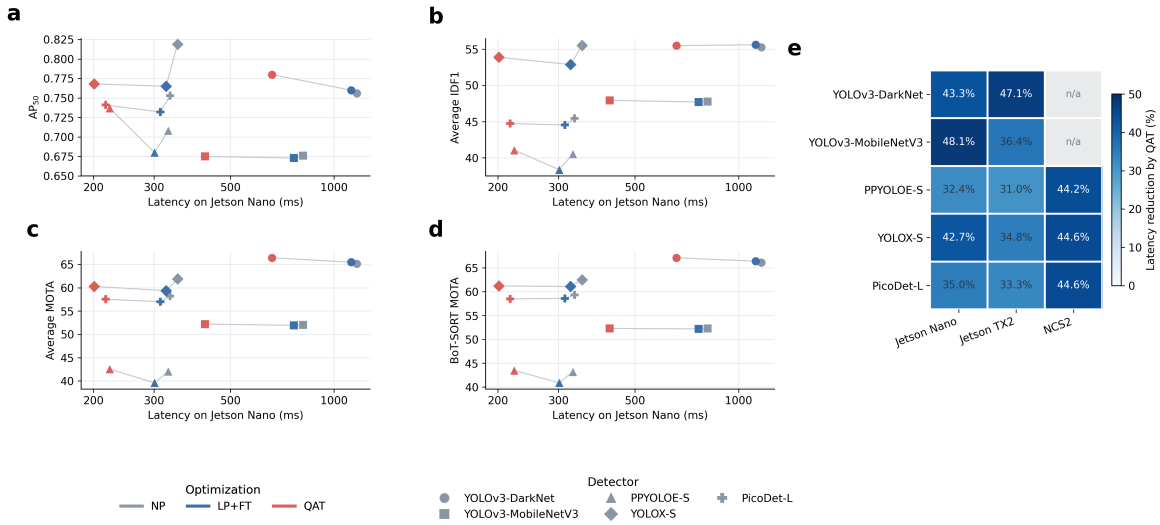


Figure 3. Comprehensive evaluation of lightweight optimization strategies in terms of accuracy-efficiency trade-off and cross-device acceleration. (a) Detection accuracy versus inference latency on Jetson Nano; (b) Average IDF1 versus inference latency; (c) Average MOTA versus inference latency; (d) BoT-SORT MOTA versus inference latency; (e) Latency reduction achieved by QAT across different edge devices. MOTA: multiple object tracking accuracy; QAT: quantization-aware training; IDF1: identification F1 score; AP_{50}: average precision at an intersection over union (IoU) threshold of 0.5.

4.3 Detection robustness under Gaussian noise

To evaluate detection robustness under Gaussian input perturbations, input images were normalized to the range (0, 1) and Gaussian noise was added at the levels reported in Table 7. Detection performance was measured using AP_{50} for the original, pruned, and QAT-optimized detectors, as summarized in Table 7 and Figure 4. MOTA and IDF1 were not evaluated under noisy conditions.

Table 6. End-to-end inference latency comparison of different MOT pipelines on edge computing platforms.

Model	Jetson Nano			Jetson TX2			NCS2		
	NP	LP+FT	QAT	NP	LP+FT	QAT	NP	LP+FT	QAT
YOLOv3-DarkNet	1165	1121 (-3.8)	660 (-43.3)	643	611 (-5.0)	340 (-47.1)	-	-	-
YOLOv3-MobileNetV3	813	765 (-5.9)	422 (-48.1)	379	330 (-12.9)	241 (-36.4)	-	-	930
PPYOLOE-S	330	301 (-8.8)	223 (-32.4)	210	193 (-8.1)	145 (-31.0)	735	701 (-4.6)	410 (-44.2)
YOLOX-S	351	325 (-7.4)	201 (-42.7)	227	209 (-7.9)	148 (-34.8)	782	743 (-5.0)	433 (-44.6)
PicoDet-L	334	313 (-6.3)	217 (-35.0)	213	198 (-7.0)	142 (-33.3)	710	696 (-2.0)	393 (-44.6)

LP+FT denotes the layer-wise pruned model after fine-tuning, and QAT denotes the model obtained using quantization-aware training; QAT identifies the optimization procedure and does not imply verified uniform INT8 execution on each platform; Latency is measured in milliseconds (ms) and includes the complete inference pipeline; Values in parentheses indicate relative latency changes (%) compared with the NP model; "-" indicates that the corresponding model could not be deployed on NCS2 due to hardware resource limitations; Bold indicates the largest relative latency reduction on each platform; MOT: multi-object tracking; QAT: quantization-aware training; NCS2: Intel Neural Compute Stick 2; NP: non-pruned; LP+FT: layer-wise pruning followed by fine-tuning.

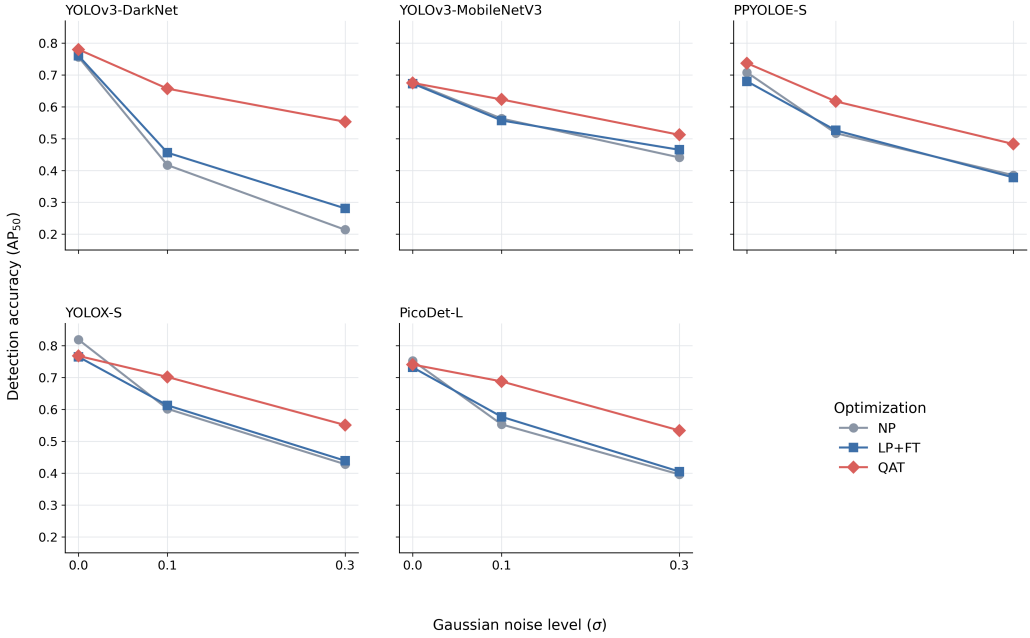


Figure 4. Detection robustness under Gaussian input perturbations. The curves show AP_{50} at the evaluated noise levels (σ = 0, 0.1 and 0.3). QAT: quantization-aware training; AP_{50}: average precision at an intersection over union (IoU) threshold of 0.5; NP: non-pruned; LP+FT: layer-wise pruning followed by fine-tuning.

Table 7. Detection AP_{50} under Gaussian input perturbations.

Model	σ = 0			σ = 0.1			σ = 0.3		
	NP	LP+FT	QAT	NP	LP+FT	QAT	NP	LP+FT	QAT
YOLOv3-DarkNet	0.756	0.760 (+0.53)	0.780 (+3.17)	0.417	0.456 (+9.35)	0.657 (+57.55)	0.214	0.281 (+31.31)	0.553 (+158.41)
YOLOv3-MobileNetV3	0.676	0.673 (-0.44)	0.675 (-0.15)	0.563	0.557 (-1.07)	0.623 (+10.66)	0.441	0.465 (+5.44)	0.512 (+16.10)
PPYOLOE-S	0.708	0.680 (-3.95)	0.737 (+4.10)	0.517	0.526 (+1.74)	0.617 (+19.34)	0.385	0.378 (-1.82)	0.483 (+25.45)
YOLOX-S	0.819	0.765 (-6.59)	0.768 (-6.23)	0.602	0.613 (+1.83)	0.702 (+16.61)	0.428	0.439 (+2.57)	0.551 (+28.74)
PicoDet-L	0.753	0.732 (-2.79)	0.741 (-1.59)	0.553	0.577 (+4.34)	0.688 (+24.41)	0.396	0.405 (+2.27)	0.534 (+34.85)

NP denotes the original non-pruned model; LP+FT denotes the layer-wise pruned model after fine-tuning, and QAT denotes the model obtained using quantization-aware training; Gaussian noise is added to the input images with standard deviation σ; Values in parentheses indicate relative changes (%) compared with the NP baseline; Bold indicates improvement over the NP baseline; All reported performance values are detection AP_{50}; QAT: quantization-aware training; AP_{50}: average precision at an intersection over union (IoU) threshold of 0.5; NP: non-pruned; LP+FT: layer-wise pruning followed by fine-tuning.

Under mild perturbation (σ = 0.1) the QAT-optimized models generally achieved better robustness compared with the original and pruned models. For example, YOLOv3-DarkNet improved from 0.417 to 0.657, corresponding to a relative increase of 57.55% compared with the original model. Similar improvements were observed for PPYOLOE-S, YOLOX-S, and PicoDet-L, where QAT achieved performance gains of 19.34%, 16.61%, and 24.41%, respectively.

In contrast, pruning without QAT provided limited or inconsistent improvements under noisy conditions.

When stronger perturbations were introduced (σ = 0.3) the performance differences between optimization strategies became more significant. The QAT models consistently maintained higher detection accuracy than both non-pruned (NP) and layer-wise pruning followed by fine-tuning (LP+FT) models. In particular, YOLOv3-DarkNet achieved an AP_{50} of 0.553 after QAT, which was substantially higher than the original model (0.214) and the pruned model (0.281), corresponding to an absolute AP_{50} gain of 0.339 over the original model. Similar trends were observed for all other detectors, with QAT achieving relative improvements ranging from 16.10% to 34.85% compared with the NP baseline.

Overall, the QAT-optimized detectors achieved higher AP_{50} than the original detectors at both evaluated nonzero Gaussian-noise levels. These results support improved detection robustness under the tested Gaussian input perturbations. Tracking robustness under noisy conditions and robustness to other forms of input degradation were not evaluated.

5. Discussion

The main objective of this study was to investigate whether lightweight optimization techniques can enable accurate and efficient head-based MOT on resource-constrained edge devices. This study evaluates detection and association performance together with end-to-end latency on edge hardware. Detection robustness under Gaussian input perturbations is assessed separately. The experimental results demonstrate that structured channel pruning and QAT can substantially improve deployment efficiency while maintaining competitive detection and tracking performance. These findings suggest that lightweight optimization should not only focus on reducing model complexity but also consider the stability of the complete tracking pipeline.

Existing model compression studies typically evaluate pruning or quantization on individual classification or object detection models. For example, norm-based pruning[7] and FPGM[33] mainly examine the trade-off between accuracy preservation and computational reduction after channel removal, while INT8 quantization studies such as[8] focus on the accuracy and inference efficiency of quantized models. In contrast, our study evaluates compression in a complete head-based MOT system. Existing lightweight MOT studies have primarily focused on improving efficiency through compact network architectures, lightweight feature extraction[14], or more efficient association strategies. In contrast, this study investigates how established compression techniques affect the complete head-based MOT pipeline, including detection, association, edge deployment, and robustness evaluation. In an MOT system, detection errors can accumulate over time and directly affect identity association stability. Therefore, maintaining tracking consistency after compression is more challenging than preserving detection accuracy alone. By evaluating multiple detectors, tracking algorithms, and edge platforms, this study provides a more comprehensive understanding of how compression strategies influence practical MOT deployment. Since publicly available Cchead results mainly focus on head tracking accuracy and rarely report lightweight MOT performance with consistent detector, tracking, and edge-deployment settings, a fair quantitative comparison with existing lightweight MOT approaches is difficult. Therefore, we focus on controlled comparisons among different compression variants under the same experimental protocol. The robustness experiments further show that QAT models are more resilient to Gaussian input noise. A potential explanation is that fake quantization repeatedly introduces rounding and clipping errors into weights and activations during training, exposing the model to quantization-induced perturbations[40]. This process may encourage more stable feature representations and partly explain the improved robustness observed under noisy inputs. This remains a possible explanation rather than a demonstrated mechanism, since no matched additional-training control was evaluated.

The findings are relevant to distributed vision-based monitoring in crowded built environments, where full-body occlusion and limited computing resources can hinder pedestrian tracking. Lower on-device latency may make pedestrian-flow and occupancy monitoring more practical, while local processing can reduce the need to transmit video to a central server. Such monitoring may support space utilization analysis, facility management, and adaptive building services[1,41]. However, these downstream applications and privacy outcomes were not directly evaluated. The measured latency also does not establish real-time processing at conventional surveillance frame rates.

Several limitations should be acknowledged. First, the evaluated detectors were released between 2018 and 2022 and do not cover the latest edge-oriented architectures, which limits the generalizability of the findings to newer detector families. Second, although the Cchead dataset provides challenging crowded scenarios, validation on additional datasets and real-world deployment environments is still needed. Third, different pruning intensities were not systematically compared in this study. Furthermore, this study evaluates pruning and quantization independently to characterize their individual effects. Although jointly optimizing pruning and quantization may provide additional efficiency improvements, such combinations require extensive architecture-specific tuning and validation, which will be investigated in future work.

Future research will focus on developing adaptive lightweight MOT frameworks that can dynamically adjust computational complexity according to hardware constraints and environmental conditions. The evaluation will also be extended to more recent edge-oriented detector families, including recent YOLO variants and lightweight transformer-based architectures, to further assess the generalizability of the observed pruning and quantization behaviors. Combining pruning, quantization, knowledge distillation, and hardware-aware neural architecture optimization may further improve the balance between accuracy, efficiency, and robustness for next-generation edge intelligence systems.

6. Conclusion

This study investigated lightweight head-based MOT under edge deployment constraints by systematically evaluating structured channel pruning and QAT strategies. A comprehensive evaluation framework was established, covering channel pruning with fine-tuning and INT8 QAT, aiming to reduce computational complexity while maintaining detection and tracking performance.

Experiments on the Cchead dataset showed that structured pruning with fine-tuning reduced model complexity, with detector-dependent effects on detection and tracking performance. QAT models achieved higher detection accuracy than PTQ models under the evaluated settings, although improvements over the FP32 baseline may also reflect additional fine-tuning. The QAT-optimized models showed lower measured end-to-end latency on the tested edge platforms, but these reductions cannot be attributed specifically to INT8 execution. Under the evaluated Gaussian input perturbations, QAT detectors achieved higher AP_{50} than the original detectors. Tracking robustness under noisy conditions was not evaluated.

Overall, this work highlights the importance of jointly considering accuracy, computational efficiency, and deployment reliability when developing lightweight MOT systems for edge intelligence applications. The proposed optimization framework provides a practical solution for deploying crowd monitoring and surveillance systems on resource-constrained devices and offers insights into the design of efficient vision models for real-world edge scenarios.

Supplementary materials

The supplementary material for this article is available at: Supplementary materials.

Declarations

Acknowledgements

The authors would like to thank Xinwei Wang from Tsinghua University for his valuable contributions to the experiments and the construction of the Cchead dataset. During the preparation of this work, the authors used ChatGPT (OpenAI) to improve the readability and language of the manuscript. The AI tool was not used for study design, data processing, data analysis, interpretation of results, or generation of scientific conclusions. All AI-assisted text was reviewed and edited by the authors, who take full responsibility for the content of the manuscript.

Authors contribution

Cui Y, Gao Z: Investigation, formal analysis, data curation, writing-original draft.
Sun K: Conceptualization, writing-review & editing, supervision.

Conflicts of interest

The authors declare no conflicts of interest.

Ethical approval

No new human participants were recruited and no new human-subject data were collected in this study. This study used the publicly released Cchead dataset as secondary data. The original data-collection and ethical procedures were described by Sun *et al*.[5].

Consent to participate

This study did not involve the recruitment of new participants. According to Sun *et al*.[5], informed consent was obtained from participants and volunteers in office, roof, and classroom scenarios before video recording. In public-area scenarios, notices were displayed to inform the public about the recording, with opt-out options where applicable.

Consent for publication

Not applicable.

Availability of data and materials

The Cchead dataset is publicly available at https://github.com/kailaisun/Cchead.

Funding

None.

Copyright

© The Author(s) 2026.

References

1. Qaisar I, Liang W, Sun K, Xing T, Zhao Q. An experimental comparative study of energy saving based on occupancy-centric control in smart buildings. Build Environ. 2025;268:112322. [DOI]
2. Choi H, Um CY, Kang K, Kim H, Kim T. Review of vision-based occupant information sensing systems for occupant-centric control. Build Environ. 2021;203:108064. [DOI]
3. Feng Y, Duives D, Daamen W, Hoogendoorn S. Data collection methods for studying pedestrian behaviour: A systematic review. Build Environ. 2021;187:107329. [DOI]
4. Sundararaman R, De Almeida Braga C, Marchand E, Pettre J. Tracking pedestrian heads in dense crowd. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2021 Jun 20-25; Nashville, USA. Piscataway: IEEE; 2021. p. 3864-3874. [DOI]
5. Sun K, Wang X, Liu S, Zhao Q, Huang G, Liu C. Towards pedestrian head tracking: A benchmark dataset and a multi-source data fusion network. Eng Appl Artif Intell. 2025;158:111265. [DOI]
6. Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network. arXiv:1503.02531 [Preprint]. 2015. [DOI]
7. Li H, Kadav A, Durdanovic I, Samet H, Graf HP. Pruning filters for efficient ConvNets. In: Proceedings of the 5th International Conference on Learning Representations (ICLR); 2017 Apr 24-27; Toulon, France. 2017. Available from: https://openreview.net/forum?id=rJqFGTslg
8. Jacob B, Kligys S, Chen B, Zhu M, Tang M, Howard A, et al. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2018 Jun 18-23; Salt Lake City, USA. Piscataway: IEEE; 2018. p. 2704-2713. [DOI]
9. Luo W, Xing J, Milan A, Zhang X, Liu W, Kim TK. Multiple object tracking: A literature review. Artif Intell. 2021;293:103448. [DOI]
10. Bewley A, Ge Z, Ott L, Ramos F, Upcroft B. Simple online and realtime tracking. In: 2016 IEEE International Conference on Image Processing (ICIP); 2016 Sep 25-28; Phoenix, USA. Piscataway: IEEE; 2016. p. 3464-3468. [DOI]
11. Wojke N, Bewley A, Paulus D. Simple online and realtime tracking with a deep association metric. In: 2017 IEEE International Conference on Image Processing (ICIP); 2017 Sep 17-20; Beijing, China. Piscataway: IEEE; 2017. p. 3645-3649. [DOI]
12. Wang Z, Zheng L, Liu Y, Li Y, Wang S. Towards real-time multi-object tracking. In: Vedaldi A, Bischof H, Brox T, Frahm JM, editors. Computer Vision—ECCV 2020. 16th European Conference, Proceedings; 2020 Aug 23-28; Glasgow, UK. Cham: Springer; 2020. p. 107-122. [DOI]
13. Zhou X, Koltun V, Krähenbühl P. Tracking objects as points. In: Vedaldi A, Bischof H, Brox T, Frahm JM, editors. Computer Vision—ECCV 2020; 2020 Aug 23-28; Glasgow, UK. Cham: Springer; 2020. p. 474-490. [DOI]
14. Zhang Y, Wang C, Wang X, Zeng W, Liu W. FairMOT: On the fairness of detection and re-identification in multiple object tracking. Int J Comput Vis. 2021;129(11):3069-3087. [DOI]
15. Redmon J, Farhadi A. YOLOv3: An incremental improvement. arXiv:1804.02767 [Preprint]. 2018. [DOI]
16. Ge Z, Liu S, Wang F, Li Z, Sun J. YOLOX: Exceeding YOLO series in 2021. arXiv:2107.08430 [Preprint]. 2021. [DOI]
17. Zhang Y, Sun P, Jiang Y, Yu D, Weng F, Yuan Z, et al. ByteTrack: Multi-object tracking by associating every detection box. In: Avidan S, Brostow G, Cissé M, Farinella GM, Hassner T, editors. Computer Vision—ECCV 2022; 2022 Oct 23-27; Tel Aviv, Israel. Cham: Springer; 2022. p. 1-21. [DOI]
18. Cao J, Pang J, Weng X, Khirodkar R, Kitani K. Observation-centric SORT: Rethinking SORT for robust multi-object tracking. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2023 Jun 17-24; Vancouver, Canada. Piscataway: IEEE; 2023. p. 9686-9696. [DOI]
19. Aharon N, Orfaig R, Bobrovsky BZ. BoT-SORT: Robust associations multi-pedestrian tracking. arXiv:2206.14651 [Preprint]. 2022. [DOI]
20. Wen Y, Gao T, Li Z, Zhang J, Zhang K, Chen T. All-in-one weather-degraded image restoration via adaptive degradation-aware self-prompting model. IEEE Trans Multimed. 2025;27:3343-3355. [DOI]
21. Wen Y, Gao T, Chen T, Li Z, Liu M, Liu L. Cross-level interaction and intralevel fusion network for remote sensing image dehazing. IEEE Trans Geosci Remote Sens. 2025;63:1-14. [DOI]
22. Gou J, Yu B, Maybank SJ, Tao D. Knowledge distillation: A survey. Int J Comput Vis. 2021;129(6):1789-1819. [DOI]
23. Sandler M, Howard A, Zhu M, Zhmoginov A, Chen LC. MobileNetV2: Inverted residuals and linear bottlenecks. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2018 Jun 18-23; Salt Lake City, USA. Piscataway: IEEE; 2018. p. 4510-4520. [DOI]
24. Zhang X, Zhou X, Lin M, Sun J. ShuffleNet: An extremely efficient convolutional neural network for mobile devices. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2018 Jun 18-23; Salt Lake City, USA. Piscataway: IEEE; 2018. p. 6848-6856.

[DOI]
25. Han K, Wang Y, Tian Q, Guo J, Xu C, Xu C. GhostNet: More features from cheap operations. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2020 Jun 13-19; Seattle, USA. Piscataway: IEEE; 2020. p. 1577-1586. [DOI]
26. Han S, Mao H, Dally WJ. Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. arXiv:1510.00149 [Preprint]. 2016. [DOI]
27. Liang T, Glossner J, Wang L, Shi S, Zhang X. Pruning and quantization for deep neural network acceleration: A survey. Neurocomputing. 2021;461:370-403. [DOI]
28. Wang CY, Yeh IH, Mark Liao HY. YOLOv9: Learning what you want to learn using programmable gradient information. In: Leonardis A, Ricci E, Roth S, Russakovsky O, Sattler T, Varol G, editors. Computer Vision—ECCV 2024. Cham: Springer; 2025. p. 1-21. [DOI]
29. Wang A, Chen H, Liu L, Chen K, Lin Z, Han J, et al. YOLOv10: Real-time end-to-end object detection. In: Advances in Neural Information Processing Systems 37; 2024 Dec 10-15; Vancouver, Canada. Vancouver: Neural Information Processing Systems Foundation, Inc.; 2024. p. 107984-108011. [DOI]
30. Wang C, He W, Nie Y, Guo J, Liu C, Wang Y, et al. Gold-YOLO: Efficient object detector via gather-and-distribute mechanism. Adv Neural Inf Process Syst. 2023;36:51094-51112. [DOI]
31. Zhao Y, Lv W, Xu S, Wei J, Wang G, Dang Q, et al. DETRs beat YOLOs on real-time object detection. In: 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2024 Jun 16-22; Seattle, USA. Piscataway: IEEE; 2024. p. 16965-16974. [DOI]
32. Vadera S, Ameen S. Methods for pruning deep neural networks. IEEE Access. 2022;10:63280-63300. [DOI]
33. He Y, Liu P, Wang Z, Hu Z, Yang Y. Filter pruning via geometric Median for deep convolutional neural networks acceleration. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2019 Jun 15-20; Long Beach, USA. Piscataway: IEEE; 2019. p. 4335-4344. [DOI]
34. Sun K, Wang X, Miao X, Zhao Q. A review of AI edge devices and lightweight CNN and LLM deployment. Neurocomputing. 2025;614:128791. [DOI]
35. Howard A, Sandler M, Chen B, Wang W, Chen LC, Tan M, et al. Searching for MobileNetV3. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV); 2019 Oct 19-Nov 2; Seoul, Korea (South). Piscataway: IEEE; 2019. p. 1314-1324. [DOI]
36. Xu S, Wang X, Lv W, Chang Q, Cui C, Deng K, et al. PP-YOLOE: An evolved version of YOLO. arXiv:2203.16250 [Preprint]. 2022. [DOI]
37. Yu G, Chang Q, Lv W, Xu C, Cui C, Ji W, et al. PP-PicoDet: A better real-time object detector on mobile devices. arXiv:2111.00902 [Preprint]. 2021. [DOI]
38. Bernardin K, Stiefelhagen R. Evaluating multiple object tracking performance: The CLEAR MOT metrics. J Image Video Proc. 2008;2008:246309. [DOI]
39. Ristani E, Solera F, Zou R, Cucchiara R, Tomasi C. Performance measures and a data set for Multi-target, multi-camera tracking. In: Hua G, Jégou H, editors. Computer Vision—ECCV 2016 Workshops; 2016 Oct 8-10 and 15-16; Amsterdam, The Netherlands. Cham: Springer; 2016. p. 17-35. [DOI]
40. Shin J, So J, Park S, Kang S, Yoo S, Park E. NIPQ: Noise proxy-based integrated pseudo-quantization. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2023 Jun 17-24; Vancouver, Canada. Piscataway: IEEE; 2023. p. 3852-3861. [DOI]
41. Yang B, Liu Y, Liu P, Wang F, Cheng X, Lv Z. A novel occupant-centric stratum ventilation system using computer vision: Occupant detection, thermal comfort, air quality, and energy savings. Build Environ. 2023;237:110332. [DOI]

Publisher's Note

Science Exploration remains a neutral stance on jurisdictional claims in published maps and institutional affiliations. The views expressed in this article are solely those of the author(s) and do not reflect the opinions of the Editors or the publisher.