This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3755382

Resource-Elastic Multi-Stream Edge-Cloud Cooperative Object Detection for Surveillance Video Analytics

Rui Lai, Guanghong Gong, Haitao Yuan, *Senior Member, IEEE,*
Jia Zhang, *Senior Member, IEEE*, and Rajkumar Buyya, *Fellow, IEEE*

***Abstract*—The number of networked surveillance cameras keeps growing, driving demand for real-time video analytics in distributed vision systems. However, high-accuracy detectors are generally too computationally demanding for individual cameras, and the aggregate workload can increasingly exceed edge-server capacity as deployments scale. Edge-cloud cooperation addresses this compute shortfall but must accommodate limited uplink bandwidth together with edge and cloud processing latency. This paper formulates multi-stream edge-cloud detection as a resource-elastic control problem with joint control of edge-compute, uplink, and cloud-compute queues, and develops ECOD, whose queue-aware Lyapunov controller adapts each stream's operating point to the current backlogs. To enable fine-grained resource elasticity, ECOD introduces an object-region-supervised (ORS) neural video codec that directs low-rate bits toward detector-relevant object regions. Its training objective jointly masks detector-feature and pixel-reconstruction losses with annotated object regions, preserving detection-critical structures at low rates while retaining standard pixel-domain reconstruction. Hardware-in-the-loop (HIL) evaluation on OD-VIRAT and UA-DETRAC shows that, with eight streams sharing a 0.5 Mbps uplink, ECOD achieves in-deadline AP50 scores of 0.661 and 0.599, respectively, compared with 0.647 and 0.520 for the best baseline.**

***Index Terms*—Video analytics, object detection, edge-cloud offloading, task-oriented coding, Lyapunov optimization.**

I. Introduction

THE growing deployment of networked surveillance cameras has increased demand for continuous, real-time video analytics. Object detection is a core workload in such systems. High-accuracy detector configurations based on Faster R-CNN [1] and the YOLO family [2] can impose computational and memory demands beyond the real-time capabilities of resource-constrained Internet-of-Things (IoT) devices [3]–[5]. Pruning, quantization, and distillation can reduce detector size [6]–[9], as can compact backbones such as SqueezeNet and the MobileNet family [10]–[13]. However, the resulting lightweight detectors can lose accuracy in surveillance video, where targets are frequently small, blurred, or crowded [14]–[16]. A nearby edge server can provide real-time inference for a small camera deployment, but aggregate demand can exceed local capacity as the deployment grows. Edge-cloud cooperation frameworks address this limitation by offloading computationally intensive tasks to the cloud [17]–[19]. The edge-cloud cooperation framework, however, must operate within the capacity of the edge-to-cloud uplink [20] while accounting for edge and cloud processing latency [21], [22].

Existing systems adapt along two axes. Content-based systems, such as Glimpse [23], Reducto [24], DDS [25], EdgeDuet [26], and EVA [27], decide what visual content to offload: they skip or filter frames, select regions, or crop objects, and then encode the selected content with a standard codec, so their savings come from discarding content. Resource-based systems, such as VideoStorm [28], AWStream [29], and JCAB [30], select frame rates, resolutions, or detector models in response to resource conditions, using profile-driven selection or aggregate latency feedback. These two adaptation axes highlight the need to jointly consider what to transmit and how to process it when multiple camera streams share limited communication and computing resources.

To tackle this problem, this paper presents ECOD, a multi-stream edge-cloud cooperative object detection system. Its controller selects per-stream operating points online based on the edge-GPU, uplink, and cloud-GPU backlogs, enabling resource-elastic offloading. To reduce uplink bandwidth while preserving detection accuracy, ECOD employs an object-region-supervised (ORS) neural video codec. Rather than optimizing full-frame rate-distortion, the codec focuses its reconstruction quality on annotated objects, encouraging the allocation of limited bits to structures relevant to detection. The controller trades off uplink usage against edge and cloud computation when selecting an operating point. Each operating point specifies three parameters (codec rate, keyframe interval, and detector fidelity) which together govern edge-side computation, uplink bandwidth, and cloud-side computation. Specifically, codec rate selects a neural coding operating point, keyframe interval controls cloud invocation frequency, and detector fidelity determines per-invocation computation. The main contributions are as follows.

This work was supported by the Fundamental Research Funds for the Central Universities, the National Natural Science Foundation of China under Grant 62473014, and the Beijing Natural Science Foundation under Grant L233005. (Corresponding author: Haitao Yuan)

R. Lai, G. Gong, and H. Yuan are with the School of Automation Science and Electrical Engineering, Beihang University, Beijing, China. (e-mail: ggh@buaa.edu.cn; yuan@buaa.edu.cn).

J. Zhang is with the Department of Computer Science, Southern Methodist University, Dallas, TX 75206, USA. (e-mail: jiazhang@smu.edu).

R. Buyya is with the Cloud Computing and Distributed Systems (CLOUDS) Lab, School of Computing and Information Systems, University of Melbourne, Melbourne, VIC 3010, Australia (e-mail: rbuyya@unimelb.edu.au).

Copyright (c) 2026 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org.

Authorized licensed use limited to: University of Melbourne. Downloaded on November 20,2026 at 03:46:09 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3755382

1) This work introduces an ORS codec that concentrates low-rate bits on detector-relevant object regions while reconstructing standard pixel-domain frames, so one decoded frame serves detector models of varying fidelity without re-encoding.
2) This work formulates multi-camera edge-cloud cooperative object detection as resource-elastic joint control of edge-encoding, uplink, and cloud-compute queues under uplink bandwidth constraints, and derives a drift-plus-penalty policy that jointly weighs their normalized backlogs and selects each stream's codec rate, keyframe interval, and detector fidelity subject to task-specific latency constraints.
3) Multi-stream HIL evaluation on OD-VIRAT and UA-DETRAC assesses ECOD under different uplink bandwidths and stream counts. With eight streams sharing a 0.5 Mbps uplink, ECOD achieves in-deadline AP50 scores of 0.661 and 0.599, respectively, compared with 0.647 and 0.520 for the best baseline.

The rest of the paper is organized as follows. Section II reviews related work. Section III formulates the problem, and Section IV details the codec and the controller. Section V reports experiments, and Section VI concludes the paper and outlines future work.

II. Related Work

This section reviews research on neural coding for machine vision and offloading-based video analytics.

A. Neural Coding for Machine Vision

Compression is a major determinant of communication cost in edge-cloud cooperative analytics, and neural codecs substantially improve rate-distortion performance. Learned image codecs replace hand-designed transforms in standards such as JPEG and HEVC with neural networks trained end-to-end for rate-distortion. Ballé *et al*. introduce factorized-prior and scale-hyperprior models [31], [32]. Subsequent context and transformer designs further improve rate-distortion performance [33], [34], while efficient variants target real-time use [35]. However, these methods primarily optimize pixel-domain reconstruction quality for human viewing rather than downstream detection accuracy. Video coding for machines (VCM) presents a general paradigm that bridges feature coding for machine vision and video coding for human vision [36]. Following this paradigm, Shao *et al*. encode temporally correlated task features for edge video analytics, enabling server-side inference without reconstructing the input video [22]. VNVC develops a feature-based neural video coding framework whose intermediate representation supports both direct machine analysis and pixel-domain reconstruction [37].

ECOD introduces an ORS codec that reconstructs pixel-domain frames. It adapts the real-time inter-frame codec DCVC-RT [38] by using annotated object masks to weight the detector-feature and pixel reconstruction losses, concentrating codec optimization on detection-critical regions rather than dense background responses. In contrast to feature-domain approaches, which operate entirely in the feature space, object-region supervision in ECOD is only applied during training; at deployment, the codec still outputs standard pixel-domain keyframes. The same reconstruction can therefore be processed by detector models and input resolutions of different computational costs without re-encoding, enabling detector fidelity to participate in online resource control. This design also enables the reconstructed frames to support downstream modules that rely on appearance features, such as object tracking and re-identification, thereby providing greater extensibility than a task-specific latent interface.

B. Offloading-based Video Analytics

Offloading video analytics to edge or cloud servers enables computationally intensive, high-accuracy detectors to process camera streams, but transmitting high-resolution video incurs bandwidth and latency overheads [39]. Existing systems proceed along two axes: content-based control, which decides what to transmit in response to video content, and resource-based control, which selects configurations in response to resource conditions.

Content-based control. Glimpse [23] and Reducto [24] are representative frame-level reduction systems. Glimpse tracks objects using cached cloud detections and selectively uploads trigger frames, while Reducto employs lightweight frame differencing to decide which frames are likely to affect query results. Both reduce uploaded frames while keeping the codec and cloud detector pipeline fixed. DDS [25] suppresses spatial redundancy through a two-stage feedback mechanism: the client first sends a low-quality frame, and the server identifies important regions and returns their coordinates, which the client then re-encodes at higher quality and transmits. This requires an initial server-side pass and a second transmission, adding latency and cloud-compute demand. EdgeDuet [26] detects large objects locally and offloads high-resolution tiles for small-object detection. However, overlapping tiles duplicate transmitted pixels and increase cloud-side inference cost. EVA [27] tracks objects and uploads selected object crops at their most informative moments. Similarly, overlapping crops duplicate transmitted pixels, while the cloud workload increases with the number of uploaded objects. These systems filter what is transmitted, discarding the information of frames or regions that they do not upload. ECOD instead keeps spatial content adaptation within coding: the ORS codec preserves the full field of view while concentrating low-rate bits on detection-relevant regions, so these coding savings do not require cropping out image regions.

Resource-based control. VideoStorm [28] schedules multiple queries on shared GPUs using quality-lag profiles obtained by offline resource-quality estimation. AWStream [29] measures the available bandwidth online and selects the highest-accuracy Pareto configuration that the measured bandwidth can sustain, avoiding queue buildup. JCAB [30] jointly selects frame rate, resolution, and CNN model and allocates uplink bandwidth across streams sharing one edge node, using a single virtual latency queue driven by Lyapunov drift-plus-penalty and analytic accuracy functions. Beyond video analytics, freshness-aware resource management has also been

Authorized licensed use limited to: University of Melbourne. Downloaded on November 20,2026 at 03:46:09 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3755382

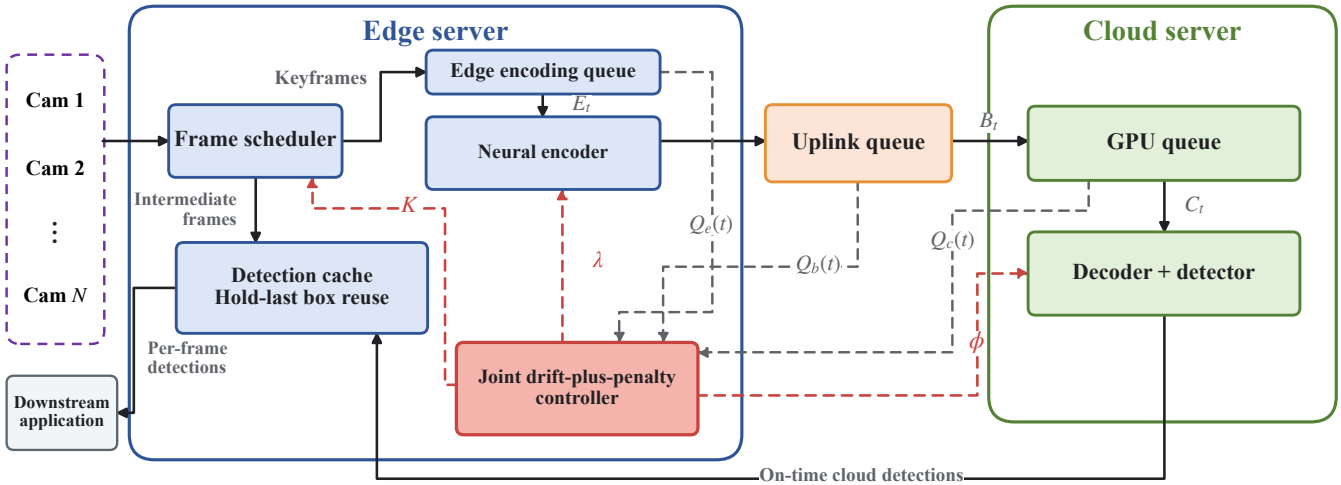

Fig. 1. System workflow of ECOD. For each stream, the edge scheduler sends keyframes at the selected neural codec rate, while intermediate frames reuse the latest on-time cloud detections from the local cache. On-time results refresh the cache, which supplies per-frame detections to the downstream application. The cross-stream controller observes all three backlogs and jointly selects (λ, K, ϕ) from the admissible operating points using the drift-plus-penalty criterion.

studied in UAV-assisted edge networks through game-theoretic optimization [40], [41]. Related work jointly considers information age, link quality, data completeness, and energy consumption, providing a complementary perspective on resource coordination [42]. For multi-stream video analytics, ECOD uses resource-specific queue backlogs to jointly select codec rate, keyframe interval, and detector fidelity. Together with the coding-based content adaptation above, this joint queue control enables ECOD to adapt to limited uplink bandwidth while accounting for edge and cloud computation.

III. Problem Formulation

A. System Model

This work considers a bandwidth-constrained fixed-camera surveillance deployment with three layers: N cameras, one edge node, and one cloud GPU dedicated to these streams. More complex edge-cloud deployments can be represented as collections of this minimal camera-edge-cloud model. The cameras, indexed by $i \in \{1, \ldots, N\}$, continuously capture video streams and send them to the edge node. At slot t of duration δ, camera i observes a frame $x_t^i \in \mathbb{R}^{H \times W \times 3}$. The edge server performs per-stream processing, including frame scheduling, box reuse between keyframes, and encoding of the selected frames. These encoded keyframes are then transmitted over a common edge-to-cloud uplink to the cloud, where they are decoded and processed by a high-accuracy detector. The application receives per-frame cached detections; a new keyframe result must return within τ_d of its source frame to refresh the cache.

Under limited uplink bandwidth, the controller jointly accounts for edge encoding, uplink transmission, and cloud decoding and detection workloads. The following formulation considers one edge node serving the N streams, with E_t seconds of encoding service available in slot t. The N streams share the available uplink rate B_t, measured in bits per second; encoded keyframes accumulate whenever their aggregate arrivals exceed the service available in a slot. They also share a cloud GPU that supplies C_t GPU-seconds of service in slot t. Cloud workload consists of decoding and detection, and accumulates whenever the admitted workload exceeds this service.

The cloud detector is invoked only on selected frames. Here, a keyframe is a frame selected for offloading and may be intra- or inter-coded. Between two cloud detections, the edge reuses the most recent cloud results for intermediate frames. Specifically, for stream i, the controller selects a keyframe interval K_t^i: one frame in every K_t^i frames is scheduled for encoding and upload, while intermediate frames reuse the most recent cloud boxes locally. As a result, increasing K_t^i directly reduces edge encoding demand, uplink traffic, and cloud inference demand.

For an uploaded keyframe x_t^i, the edge server selects a codec rate and produces a bitstream whose size is denoted by $R(x_t^i)$. When the ORS codec is selected, its encoder E_θ maps the frame to a bitstream and its decoder D_ψ reconstructs the cloud input as $\hat{x}_t^i = D_\psi(E_\theta(x_t^i))$. The decoded output is a pixel-domain image, enabling standard pixel-input detectors to process $\hat{x}_t^i$ directly, while this decoded image can be fed to detectors with different input resolutions or model variants. This flexibility makes it possible to support multiple detector fidelities, as validated in Section V. The controller can then trade detection accuracy against cloud-compute cost by selecting the detector fidelity, without modifying the coding interface. Finally, for any selected detector ϕ, the result $\phi(\hat{x}_t^i)$ is scored against the ground truth Y_t^i by an accuracy functional $A(\phi(\hat{x}_t^i), Y_t^i)$.

B. Task-Oriented Coding: A Rate-Task-Distortion View

A reconstruction-oriented codec preserves pixel or perceptual fidelity. Its rate-distortion tradeoff is commonly charac-

Authorized licensed use limited to: University of Melbourne. Downloaded on November 20,2026 at 03:46:09 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3755382

TABLE I
KEY NOTATION IN THE PROBLEM FORMULATION.

Symbol	Meaning
N	Number of camera streams
x_t^i	Frame of stream i at slot t
δ	Slot duration
τ_d	Per-frame detection deadline
τ_r	Time budget for reference initialization or recovery
B_t	Shared effective uplink rate at slot t
C_t	GPU-seconds of cloud service available in slot t
E_t	Seconds of edge-encoding service available in slot t
$Q_e(t)$	Unfinished edge-encoding workload in seconds
$Q_b(t)$	Shared uplink backlog in bits
$Q_c(t)$	Shared unfinished GPU workload in seconds
S_e, S_b, S_c	Per-run fixed service scales in edge seconds, bits, and cloud GPU-seconds
o	Per-stream operating point
$\mathcal{O}$	Finite set of candidate operating points
λ	Codec rate
K	Keyframe interval
ϕ	Cloud detector configuration
$\hat{r}_t^i(o), \hat{\kappa}(o)$	Projected complete keyframe bits and GPU workload
$\rho_t^i(o)$	Projected transmitted bits per source frame
$\kappa(o)$	Average GPU-seconds per source frame
$\hat{e}(o), \epsilon(o)$	Profiled edge seconds per keyframe and per source frame
$a(o)$	Profiled AP50 including box reuse
V	Accuracy-reward weight in the control objective
$L_i^t(o)$	Projected end-to-end latency for stream i
$\mathcal{F}_i^t$	Admissible operating-point set

terized by

$$\min_{\theta,\psi} \ \mathbb{E}[R_\theta(x)] \quad \text{s.t.} \quad \mathbb{E}[d_p(x,\hat{x})] \le D, \tag{1}$$

where x denotes a generic input frame or keyframe, $R_\theta(x)$ is its coded length under encoder parameters θ, ψ denotes the decoder parameters, and d_p denotes a reconstruction distortion, such as mean squared error or a perceptual loss. Such an objective allocates bits based on visual quality, which may not align with the image components that the detector actually uses. This work defines distortion in the feature space of the frozen cloud detector and restricts it to annotated object regions rather than the full frame. Let $F_\ell(\cdot)$ denote the scale-ℓ feature map of the frozen detector. From the annotation Y, let $M_\ell(Y)$ be the object-region mask projected to feature scale ℓ. The task distortion is defined as

$$d_\phi(x,\hat{x};Y) = \frac{1}{L}\sum_{\ell=1}^{L} w_\ell \frac{\left\| M_\ell(Y) \odot \big(F_\ell(x) - F_\ell(\hat{x})\big)\right\|_F^2}{\|M_\ell(Y)\|_1 + \epsilon}. \tag{2}$$

Here, w_ℓ weights detector scales, $\odot$ denotes element-wise multiplication, and ϵ avoids division by zero when no object region is present at a scale. This formulation yields the following rate-task-distortion function:

$$R_\phi(D) = \min_{\theta,\psi} \ \mathbb{E}[R_\theta(x)] \quad \text{s.t.} \quad \mathbb{E}[d_\phi(x,\hat{x};Y)] \le D. \tag{3}$$

C. Adaptive Offloading: Operating-Point Control

At each keyframe decision, the controller selects an operating point $o \in \mathcal{O}$ for the corresponding stream before encoding and cloud offloading:

$$o = (\lambda, K, \phi), \tag{4}$$

where λ is the codec rate, K is the keyframe interval, and ϕ is the cloud detector configuration (model and input resolution). Each codec rate identifies a profiled neural coding operating point with a fixed model and quality setting. Each operating point is profiled by its reset-frame size, steady GOP-average bitstream size $\bar{r}(o)$, edge-encoding time $\hat{e}(o)$, frame-averaged GPU workload $\kappa(o)$, and detection utility $a(o)$. Reset-frame sizes are measured directly, while continuation-frame sizes are derived from the measured GOP-average and reset-frame sizes. Let $\hat{r}_t^i(o)$ denote the projected complete bitstream size under the current codec state of stream i. It accounts for reference initialization, periodic intra refresh, and reference reuse when changing codec rate. For long-term control, $\Delta r_t^i(o)$ is set to $[\hat{r}_t^i(o) - \bar{r}(o)]/G$ for reference initialization or a codec-quality change that requires an additional intra frame, and zero otherwise, where G is the number of coded frames in a GOP. Periodic intra refresh is already included in $\bar{r}(o)$. Let $\hat{\kappa}(o)$ denote the GPU service time for independently processing a keyframe at operating point o. The corresponding frame-averaged loads are

$$\rho_t^i(o) = \frac{\bar{r}(o) + \Delta r_t^i(o)}{K}, \qquad \kappa(o) = \frac{\hat{\kappa}(o)}{K}, \tag{5}$$

The corresponding frame-averaged edge workload is

$$\epsilon(o) = \frac{\hat{e}(o)}{K}. \tag{6}$$

The utility $a(o)$ is AP50 pooled over the scene's calibration frames, with detection performed every K frames and the latest boxes reused on intervening frames.

To derive the configuration score, we approximate the edge, uplink, and cloud loads by averaging each profiled workload over the selected interval K. In this approximation, the edge backlog evolves as

$$Q_e(t{+}1) = \max\Big(Q_e(t) + \textstyle\sum_{i=1}^N \epsilon(o_t^i) - E_t,\, 0\Big). \tag{7}$$

Using the same approximation, the uplink backlog evolves as

$$Q_b(t{+}1) = \max\Big(Q_b(t) + \textstyle\sum_{i=1}^N \rho_t^i(o_t^i) - B_t\delta,\, 0\Big), \tag{8}$$

The corresponding cloud backlog evolves as

$$Q_c(t{+}1) = \max\Big(Q_c(t) + \textstyle\sum_{i=1}^N \kappa(o_t^i) - C_t,\, 0\Big), \tag{9}$$

Following the standard Lyapunov-optimization definition [43], a backlog process $Q(t)$ is strongly stable if

$$\limsup_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}[Q(t)] < \infty. \tag{10}$$

Let S_e, S_b, and S_c be fixed scales used to express edge, uplink, and cloud workloads in comparable dimensionless units, and define

$$u_b^i(t,o) = \frac{\rho_t^i(o)}{S_b}. \tag{11}$$

Authorized licensed use limited to: University of Melbourne. Downloaded on November 20,2026 at 03:46:09 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3755382

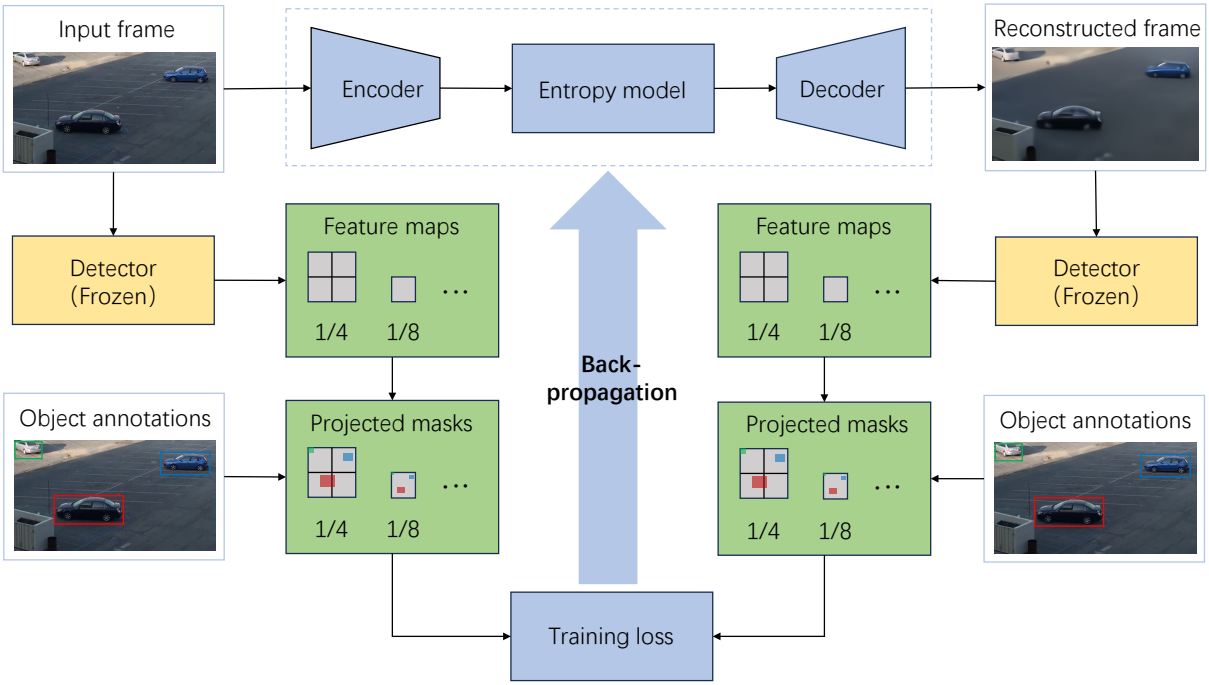

Fig. 2. Training framework of the ORS codec. Multi-scale detector features and projected object masks define the training loss, which is back-propagated through the neural codec.

$$u_e(o) = \frac{\epsilon(o)}{S_e}, \qquad u_c(o) = \frac{\kappa(o)}{S_c}. \tag{12}$$

For a fixed uplink rate B, we set $S_b{=}B\delta$ bits and $S_e{=}S_c{=}\delta$ seconds. The quantities $u_b^i(t,o)$, $u_e(o)$, and $u_c(o)$ equal the transmission, edge-processing, and cloud-processing times per source frame, respectively, divided by the slot duration. Within this approximation, we formulate a relaxed long-run objective that balances profiled detection utility and normalized resource cost subject to stability constraints on the modeled backlogs. The resource-cost term favors operating points that use fewer resources at the same detection accuracy, while $V > 0$ controls the tradeoff between detection accuracy and resource cost. The resulting optimization problem is formulated as follows:

$$\begin{aligned}
\max_{\{o_t^i\}} \quad & \liminf_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\Bigg[\sum_{i=1}^{N}\big(Va(o_t^i) \\
& \qquad -u_e(o_t^i) - u_b^i(t,o_t^i) - u_c(o_t^i)\big)\Bigg] \\
\text{s.t.} \quad & o_t^i \in \mathcal{O}, \qquad \forall i,t, \\
& \limsup_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[Q_e(t)] < \infty, \\
& \limsup_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[Q_b(t)] < \infty, \\
& \limsup_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[Q_c(t)] < \infty.
\end{aligned} \tag{13}$$

Section IV-C derives the configuration score and describes how it is used for online selection.

IV. Method

This section instantiates the formulation with two components: a real-time inter-frame neural codec adapted via object-region supervision, and a queue-aware cross-stream drift-plus-penalty controller that selects each stream's operating point online under uplink bandwidth and end-to-end latency constraints. Fig. 1 illustrates how these components integrate into the multi-stream edge–cloud pipeline.

A. Object-Region-Supervised Neural Video Codec

Codec architecture. Fixed-camera surveillance video often exhibits substantial temporal redundancy because much of the background remains similar across adjacent frames, even at the reduced frame rates used by video analytics systems. An intra codec re-encodes that redundant background on every frame, whereas an inter-frame codec predicts it from the previous decoded frame and can spend more bits on foreground changes. This temporal structure explains why standard inter-frame codecs such as H.264 remain competitive on surveillance video, and why the neural codec must also exploit inter-frame redundancy to achieve a favorable rate-accuracy trade-off. This work therefore builds on a neural video backbone rather than a neural image codec.

The codec adopts the low-delay neural video backbone of DCVC-RT [38] and keeps its inter-frame coding path. The first frame in each group of pictures (GOP) is coded intra, and subsequent frames are coded as inter frames using the preceding decoded frame as reference. For each inter frame, a learned analysis transform, conditioned on the propagated reference feature, maps the current frame to a latent representation. A conditional entropy model with a hyperprior

Authorized licensed use limited to: University of Melbourne. Downloaded on November 20,2026 at 03:46:09 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3755382

encodes the latent into a bitstream, and a synthesis transform reconstructs a pixel-domain frame from the decoded latent. The codec is initialized from pretrained DCVC-RT weights and then fine-tuned through the frozen detector ϕ with object-region supervision: only the codec parameters are updated, and annotated object masks weight the training losses defined below.

Training objective. As shown in Fig. 2, the codec is trained through the frozen detector with a feature term, a pixel term, and a rate term,

$$\mathcal{L} = \gamma \frac{1}{L} \sum_{\ell=1}^{L} w_\ell \frac{\left\| M_\ell(Y) \odot \left(F_\ell(x) - F_\ell(\hat{x})\right) \right\|_F^2}{\|M_\ell(Y)\|_1 + \epsilon} + \beta \frac{\left\| M_x(Y) \odot (x - \hat{x}) \right\|_2^2}{\|M_x(Y)\|_1 + \epsilon} + \eta\, R(\hat{y}), \tag{14}$$

As in (2), $F_\ell(\cdot)$ is the frozen detector feature map and $M_\ell(Y)$ is the object-region mask at feature scale ℓ; $M_x(Y)$ denotes the corresponding mask at pixel resolution. The coefficient γ weights the feature term, and w_ℓ presets the relative weights of detector scales, with larger weight on the finest scale for small objects. The masks keep the feature and pixel losses from being dominated by static background and texture. If a frame has no annotated object, the feature and pixel terms vanish and the rate term encourages compact background coding.

The pixel term, weighted by β, preserves the pixel-domain structure of object regions in $\hat{x}$, allowing standard pixel-input detectors to consume the decoded frame and supporting the multi-fidelity interface. The rate term is weighted by η and uses an entropy-model estimate of the bits needed to code the quantized latent:

$$R(\hat{y}) = \sum_j -\log_2 p_{\hat{y}}(\hat{y}_j), \tag{15}$$

where j ranges over all quantized latent symbols and $p_{\hat{y}}(\hat{y}_j)$ is the probability assigned to the j-th symbol. This differentiable rate estimate controls the bitrate during fine-tuning. At runtime, the range Asymmetric Numeral Systems (rANS) entropy coder [44] produces the actual bitstream, whose measured size defines the transmitted rate, while the built-in quality index allows one trained model to trace the rate-accuracy frontier.

A predictive (P) frame is an inter-coded frame reconstructed from a previously decoded reference frame; its loss is computed against the current source frame. Therefore, reconstruction errors in earlier frames enter the reference used by later P-frames and can accumulate along a GOP. If training uses only short clips, the codec mostly learns to predict from near-clean references and may not remain robust when many inter frames are chained at inference. This work therefore fine-tunes the codec on GOP-32 clips, with one intra frame followed by thirty-one inter frames, and accumulates (14) over the inter frames in the training GOP. This exposes the model to multiple levels of reference distortion during training and improves robustness when deployed on longer inter-frame sequences.

B. Edge-Cloud Pipeline

The pipeline is designed such that the cloud detects uploaded keyframes, while the edge reuses the resulting boxes on the intervening frames. On a keyframe, the edge encodes the frame with the selected codec and uploads the bitstream. The cloud decodes the bitstream to a pixel-domain frame, runs the detector at the selected configuration, and returns the boxes. Between keyframes, the edge CPU directly reuses the most recent cloud detections without applying a motion model, providing per-frame results without additional uplink traffic or cloud inference.

Deploying the encoder at the edge and reconstructing pixel-domain frames in the cloud provides two additional system properties. First, each keyframe is uploaded once and decoded once, so the cloud runs one detector forward per keyframe rather than requiring per-region, per-tile, or two-pass cloud inference. Second, because the decoded output is a pixel-domain image, the same uploaded representation can be processed by detectors with different model sizes or input resolutions without re-encoding. This multi-fidelity detector support makes detector fidelity a controllable cloud-compute variable. Section V quantifies the accuracy and compute trade-offs of this interface.

C. Adaptive Offloading Control

We derive a queue-aware scoring criterion from a drift-plus-penalty upper bound for the approximate model in (13). Let $\Phi(t)=\{Q_e(t), Q_b(t), Q_c(t)\}$ denote the queue state. The normalized quadratic Lyapunov function is

$$\mathcal{L}(\Phi(t)) = \frac{1}{2}\left(\frac{Q_e(t)}{S_e}\right)^2 + \frac{1}{2}\left(\frac{Q_b(t)}{S_b}\right)^2 + \frac{1}{2}\left(\frac{Q_c(t)}{S_c}\right)^2. \tag{16}$$

The corresponding conditional one-slot drift is

$$\Delta\mathcal{L}(\Phi(t)) = \mathbb{E}\{\mathcal{L}(\Phi(t+1)) - \mathcal{L}(\Phi(t)) \mid \Phi(t)\}. \tag{17}$$

For the shared uplink queue, squaring (8) gives

$$\begin{aligned}\left(\frac{Q_b(t+1)}{S_b}\right)^2 &\le \left(\frac{Q_b(t)}{S_b} + \sum_{i=1}^{N} \frac{\rho_t^i(o_t^i)}{S_b} - \frac{B_t\delta}{S_b}\right)^2 \\ &\le \left(\frac{Q_b(t)}{S_b}\right)^2 + \left(\sum_{i=1}^{N} \frac{\rho_t^i(o_t^i)}{S_b}\right)^2 + \left(\frac{B_t\delta}{S_b}\right)^2 \\ &\quad + 2\frac{Q_b(t)}{S_b}\left(\sum_{i=1}^{N} \frac{\rho_t^i(o_t^i)}{S_b} - \frac{B_t\delta}{S_b}\right).\end{aligned} \tag{18}$$

Similarly, squaring (9) gives

$$\begin{aligned}\left(\frac{Q_c(t+1)}{S_c}\right)^2 &\le \left(\frac{Q_c(t)}{S_c} + \sum_{i=1}^{N} \frac{\kappa(o_t^i)}{S_c} - \frac{C_t}{S_c}\right)^2 \\ &\le \left(\frac{Q_c(t)}{S_c}\right)^2 + \left(\sum_{i=1}^{N} \frac{\kappa(o_t^i)}{S_c}\right)^2 + \left(\frac{C_t}{S_c}\right)^2 \\ &\quad + 2\frac{Q_c(t)}{S_c}\left(\sum_{i=1}^{N} \frac{\kappa(o_t^i)}{S_c} - \frac{C_t}{S_c}\right).\end{aligned} \tag{19}$$

Authorized licensed use limited to: University of Melbourne. Downloaded on November 20,2026 at 03:46:09 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3755382

Applying the same bound to (7) gives

$$\left(\frac{Q_e(t+1)}{S_e}\right)^2 \le \left(\frac{Q_e(t)}{S_e}\right)^2 + \left(\sum_{i=1}^{N}\frac{\epsilon(o_t^i)}{S_e}\right)^2 + \left(\frac{E_t}{S_e}\right)^2 + 2\frac{Q_e(t)}{S_e}\left(\sum_{i=1}^{N}\frac{\epsilon(o_t^i)}{S_e} - \frac{E_t}{S_e}\right). \tag{20}$$

For a fixed N, the operating-point set and service processes are bounded. Hence, the quadratic terms admit the uniform bound

$$\mathcal{B} = \frac{1}{2}\sup_{t,\{o_t^i\}}\left\{\left(\sum_{i=1}^{N}\frac{\rho_t^i(o_t^i)}{S_b}\right)^2 + \left(\frac{B_t\delta}{S_b}\right)^2 + \left(\sum_{i=1}^{N}\frac{\kappa(o_t^i)}{S_c}\right)^2 + \left(\frac{C_t}{S_c}\right)^2 + \left(\sum_{i=1}^{N}\frac{\epsilon(o_t^i)}{S_e}\right)^2 + \left(\frac{E_t}{S_e}\right)^2\right\} < \infty. \tag{21}$$

Applying (21), summing (18), (19), and (20), and taking the conditional expectation gives

$$\Delta\mathcal{L}(\Phi(t)) \le \mathcal{B} + \mathbb{E}\Bigg\{\frac{Q_b(t)}{S_b}\left(\sum_{i=1}^{N}\frac{\rho_t^i(o_t^i)}{S_b} - \frac{B_t\delta}{S_b}\right) + \frac{Q_c(t)}{S_c}\left(\sum_{i=1}^{N}\frac{\kappa(o_t^i)}{S_c} - \frac{C_t}{S_c}\right) + \frac{Q_e(t)}{S_e}\left(\sum_{i=1}^{N}\frac{\epsilon(o_t^i)}{S_e} - \frac{E_t}{S_e}\right)\Bigg|\ \Phi(t)\Bigg\}. \tag{22}$$

The drift-plus-penalty expression corresponding to the regularized objective in (13) is

$$\Psi(\Phi(t)) = \Delta\mathcal{L}(\Phi(t)) + \mathbb{E}\Bigg\{\sum_i\left[\frac{\epsilon(o_t^i)}{S_e} + \frac{\rho_t^i(o_t^i)}{S_b} + \frac{\kappa(o_t^i)}{S_c} - Va(o_t^i)\right] \mid \Phi(t)\Bigg\}. \tag{23}$$

Substituting (22) into (23) and dropping the terms independent of the current operating-point choices gives

$$\min_{\{o_t^i\}}\sum_i\left[\left(1+\frac{Q_b(t)}{S_b}\right)\frac{\rho_t^i(o_t^i)}{S_b} + \left(1+\frac{Q_c(t)}{S_c}\right)\frac{\kappa(o_t^i)}{S_c} + \left(1+\frac{Q_e(t)}{S_e}\right)\frac{\epsilon(o_t^i)}{S_e} - Va(o_t^i)\right]. \tag{24}$$

The constant term in each resource coefficient discourages unnecessary resource use when the queues are empty, while the backlog-dependent terms increase the cost of the more congested resource. At each keyframe decision, the controller evaluates this score using the observed backlogs and screens candidates using their predicted latency. The predicted end-to-end latency is expressed as

$$L_i^t(o) = L_{e,i}^t(o) + L_{u,i}^t(o) + L_{c,i}^t(o) + L_{r,i}^t, \tag{25}$$

Algorithm 1 Queue-aware resource-elastic offloading control at slot t

Require: queues $Q_e(t), Q_b(t), Q_c(t)$; E_t, B_t, C_t; fixed scales S_e, S_b, S_c; operating-point profiles over $\mathcal{O}$

for each stream i whose keyframe decision is due **do**
  $\mathcal{F}_i^t \leftarrow \{o \in \mathcal{O} \setminus \mathcal{R}_i^t :\ L_i^t(o) \le \tau_d\}$
  if any candidate requires reference initialization or recovery **then**
    $\mathcal{F}_i^t \leftarrow \mathcal{F}_i^t \cup \{o \in \mathcal{R}_i^t :\ L_i^t(o) \le \tau_r\}$
  end if
  if $\mathcal{F}_i^t = \varnothing$ **then**
    reuse the latest boxes; retry after $\min_{o\in\mathcal{O}} K(o)$ frames; **continue**
  end if
  select o_t^i from $\mathcal{F}_i^t$ according to (27)
  enqueue the keyframe for edge encoding and schedule the next decision after $K(o_t^i)$ frames
end for
update the observed edge, uplink, and cloud backlogs from admitted keyframes and completed service

where the four terms represent edge processing and queueing, uplink transmission and queueing, cloud processing and queueing, and result return, respectively. Processing times are obtained from the operating-point profiles, and transmission time is estimated as $\hat{r}_t^i(o)/B_t$. Queueing delays are predicted by propagating completion times through edge encoding, uplink transmission, and cloud processing under FIFO service, accounting for the remaining work of admitted requests and when each request becomes ready for the next stage. The predicted latency is compared with a time budget that depends on the task type. Ordinary detection tasks use the detection deadline, while reference initialization or recovery tasks use a separate budget because they establish the reference needed for subsequent decoding. These tasks use an intra-coded frame when no valid reference is available or a periodic refresh is due. The admissible set $\mathcal{F}_i^t$ contains the operating points whose predicted end-to-end latency satisfies

$$L_i^t(o) \le \begin{cases}\tau_r, & \text{initialization or recovery,}\\ \tau_d, & \text{ordinary detection.}\end{cases} \tag{26}$$

Here, τ_r is the recovery time budget, while τ_d is the detection deadline. Only detection results returned within τ_d update the cache. If $\mathcal{F}_i^t$ is nonempty, the controller selects

$$o_t^i = \arg\min_{o\in\mathcal{F}_i^t}\left[\left(1+\frac{Q_b(t)}{S_b}\right)\frac{\rho_t^i(o)}{S_b} + \left(1+\frac{Q_c(t)}{S_c}\right)\frac{\kappa(o)}{S_c} + \left(1+\frac{Q_e(t)}{S_e}\right)\frac{\epsilon(o)}{S_e} - Va(o)\right]. \tag{27}$$

Algorithm 1 summarizes this decision procedure. In Algorithm 1, $\mathcal{R}_i^t \subseteq \mathcal{O}$ denotes the configurations that require reference initialization or recovery.

Authorized licensed use limited to: University of Melbourne. Downloaded on November 20,2026 at 03:46:09 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3755382

V. Numerical Experiments and Analysis

This section first describes the experimental setup and baselines. It then validates the operating-point space by examining the resource-accuracy tradeoffs enabled by codec type and rate, detector fidelity, and keyframe interval. After the training-objective ablation, it evaluates multi-stream accuracy and real-time service under uplink constraints, followed by sensitivity to the reward weight and recovery time budget.

A. Experimental Setup

Hardware. The edge-side experiments run on a workstation with an Intel i7-12700KF CPU and an NVIDIA RTX 3070 Ti GPU; the cloud node is a server with dual Intel Xeon Silver 4314 CPUs and an NVIDIA RTX 4090 GPU.

Dataset and detector. The evaluation uses OD-VIRAT [45], a fixed-camera surveillance dataset. The cloud detectors use YOLOv8s and YOLOv8m and run at 640 and 1280 px. The ORS codec is trained on 25311 frames, with its rate and loss settings chosen on a separate validation set containing 4443 frames. An additional 7491 OD-VIRAT frames are used to evaluate ECOD's performance. For multi-stream replay, these frames are split chronologically within each sequence: the initial portion, approximately one third, is used to construct the operating-point profiles, which remain fixed during evaluation, while the remaining 5000 frames are used for evaluation. AP50 is the primary metric of the dataset; AP75, mAP, and F1 accompany it in Table II. mAP averages AP over IoU thresholds from 0.50 to 0.95 in steps of 0.05. F1 is the harmonic mean of precision and recall, evaluated at an IoU threshold of 0.50 and a confidence threshold of 0.30.

UA-DETRAC [46] is used as a supplementary dataset to evaluate ECOD's robustness across different surveillance scenarios. ORS adaptation and checkpoint selection use the official training partition, and evaluation uses the official test partition. The UA-DETRAC multi-stream replay uses the same temporal split as OD-VIRAT for its official test sequences, with the first third of each sequence used for profiling and the remainder for evaluation.

Controller and HIL evaluation. The multi-stream experiments replay N streams ($N\in\{4,8,12\}$), using one source-frame interval as a slot, over a shared uplink and one cloud GPU with a per-stream round trip drawn from a 15 to 25 ms distribution with a 20 ms mean, and a 200 ms deadline. OD-VIRAT uses 30 fps at 1280×720, and UA-DETRAC uses 25 fps at 960×540; shared bandwidth is $B\in\{0.25,0.5,0.75,1\}$ Mbps. During HIL evaluation, transmission and processing delays are determined using the actual encoded bitstream sizes and measured execution times, while operating-point profiles are used for candidate selection. Frames from each stream arrive at the original video frame rate, and the shared uplink serves queued packets causally at B_t. The main HIL comparisons and sensitivity analyses report means over four replay seeds. The reported multi-stream results use $V=10$, while sensitivity to V is examined in Section V-G. The main comparisons use $\tau_r=1$ s, whose sensitivity is examined in Section V-H. In-deadline AP50 scores the available detection cache against each evaluated source frame; only keyframe results returned within $\tau_d=200$ ms refresh that cache. Admitted requests continue to be served after expiry.

Implementation. The codec builds on the public DCVC-RT implementation [38], and the cloud detectors are implemented with YOLOv8 [47]; both are run in PyTorch [48]. The multi-stream neural candidate set contains pretrained DCVC-RT at QP0 and ORS at QP4 and QP8, crossed with $K\in\{1,5,10\}$ and YOLOv8s/m at $640/1280$ px, giving 36 operating points. When switching between pretrained DCVC-RT and ORS, the encoder and decoder use their latest common reconstructed frame to initialize the selected codec's reference state. The selected codec clears its cached reference features and derives new features from this image.

Training and compute. ORS uses the COCO-pretrained YOLOv8m as its feature supervisor. The codec model is optimized with Adam using an initial learning rate of 2×10^{-5} and cosine annealing. The loss weights are $\gamma=1$, $\beta=0.3$, and $\eta=4$, with $w_\ell=2$ for the finest detector scale and $w_\ell=1$ for the other scales. The analysis and synthesis transforms run on the GPU, and their measured encode and decode times are included in the operating-point profiles used by the controller.

B. Baselines

ECOD is compared with H.265-only, No-backlog, and JCAB. JCAB [30] is adapted to the same neural operating-point pool as ECOD. To place JCAB under the common experimental interface, this work sets its energy weight to zero because endpoint energy is not considered in this paper, and maps its processing-capacity constraint to the profiled load of one shared GPU. This work retains JCAB's single long-term latency virtual queue, joint multi-stream configuration selection, closed-form bandwidth allocation, and Markov approximation, and executes its allocated bandwidth shares through causal weighted-fair service. All systems use the same deadline, cached-box prediction rule, and cloud execution path. NO-BACKLOG uses the same operating-point set and admission rules as ECOD but removes the backlog-dependent terms from the selection criterion. H.265-ONLY uses NVIDIA's dedicated hardware video encoder (NVENC) for H.265 encoding at QP $36,40,44,48,51$ with the same K and detector options, applying ECOD's scoring and admission rules. In the main comparisons, ECOD, No-backlog, and H.265-only use $V=10$.

C. Evaluation of Control Variables

1) Effect of Codec Type and Rate: The first experiment evaluates how codec type and rate affect the rate-accuracy tradeoff of cloud detection. ORS, pretrained DCVC-RT [38], and H.264/H.265 are compared across rate levels using the same frames and YOLOv8m detector at 1280 px, including both intra-coded and predicted frames. The neural codecs use QP values of $0,4,8,16,24,$ and 32, while H.264 and H.265 use QP values of $36,40,44,48,$ and 51. Coded sizes are measured from actual bitstreams for all codecs. As a reference,

Authorized licensed use limited to: University of Melbourne. Downloaded on November 20,2026 at 03:46:09 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3755382

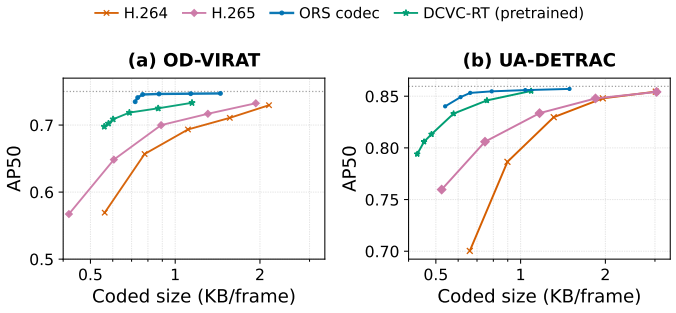

Fig. 3. Rate-accuracy comparison of the ORS codec, pretrained DCVC-RT, H.264, and H.265 on (a) OD-VIRAT and (b) UA-DETRAC, measured by AP50. The dotted lines mark the uncompressed-input references.

uncompressed input achieves AP50 values of 0.750 on OD-VIRAT and 0.8595 on UA-DETRAC, as shown in Fig. 3(a) and (b), respectively.

On OD-VIRAT and UA-DETRAC, adjacent frames at the native frame rate are similar, allowing the evaluated inter-frame codecs to maintain high detection accuracy at sufficiently high bitrates, while differing in the number of bits required to achieve comparable accuracy. As shown in Fig. 3(a), the ORS codec exhibits a nearly flat rate-accuracy curve: it holds AP50 0.735 at its lowest rate of 0.72 KB per frame and 0.747 at 1.45 KB, staying within 0.015 of the uncompressed reference even at the low-rate end. At QP40, H.264 and H.265 achieve AP50 0.711 and 0.717 at 1.563 and 1.306 KB per frame, respectively, whereas ORS achieves 0.735 at 0.721 KB per frame.

Fig. 3(b) reports results on the UA-DETRAC test set. ORS achieves AP50 0.8532 at 0.662 KB per frame, compared with 0.8459 at 0.759 KB for pretrained DCVC-RT. At these operating points, ORS reduces the coded size by 12.8% while improving AP50 by 0.73 percentage points, demonstrating its rate-accuracy advantage in traffic surveillance.

2) Effect of Detector Fidelity: To test the multi-fidelity interface, each frame is encoded once by the ORS codec at `qp4`, decoded, and processed by all four detector configurations. Table II reports detection accuracy on OD-VIRAT and UA-DETRAC. On OD-VIRAT, AP50 rises from 0.498 for `s@640` to 0.741 for `m@1280`, a span of 0.243 AP50. On UA-DETRAC, larger models improve AP50 at each input size, while increasing resolution alone does not consistently improve accuracy. The `m@1280` configuration achieves the highest AP50, AP75, mAP, and F1 on both datasets.

3) Effect of Keyframe Interval: This experiment measures how the keyframe interval affects detection accuracy, thereby quantifying the accuracy-load trade-off available to the controller. At interval K, the cloud detector runs once every K frames, and the edge reuses its most recent boxes for the intervening $K-1$ frames. Tracking is not used between keyframes because it adds per-frame edge computation and is susceptible to drift when detector updates are sparse [49]. ECOD instead directly reuses the latest cloud detections, making the resource cost of K explicit and measurable. Table III reports AP50 at $K \in \{1, 5, 10\}$ on OD-VIRAT and UA-DETRAC for the ORS codec at `qp4` with `m@1280`. Holding the codec, quantization setting, and detector fixed isolates the effect of K.

TABLE II
DETECTOR-FIDELITY GRID ON OD-VIRAT AND UA-DETRAC FRAMES DECODED BY THE ORS CODEC AT `QP4`.

Detector	AP50	AP75	mAP	F1
OD-VIRAT				
`s@640`	0.498	0.219	0.249	0.623
`s@1280`	0.694	0.268	0.335	0.737
`m@640`	0.684	0.245	0.332	0.708
`m@1280`	**0.741**	**0.286**	**0.363**	**0.778**
UA-DETRAC				
`s@640`	0.825	0.683	0.573	0.829
`s@1280`	0.823	0.677	0.569	0.834
`m@640`	0.833	0.708	0.592	0.835
`m@1280`	**0.849**	**0.716**	**0.598**	**0.848**

TABLE III
EFFECT OF KEYFRAME INTERVAL ON ALL-FRAME AP50 FOR THE ORS CODEC AT `QP4` WITH `M@1280`.

Keyframe interval K	OD-VIRAT	UA-DETRAC
1	0.741	0.849
5	0.724	0.780
10	0.700	0.636

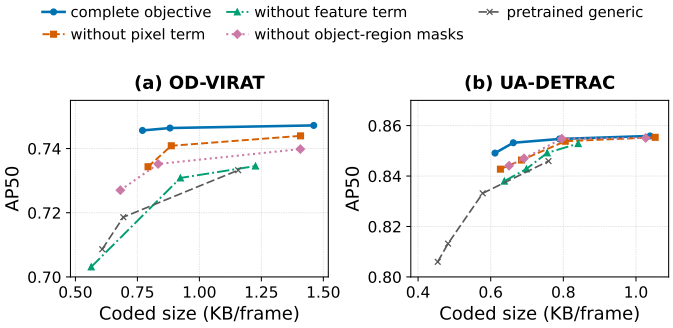

Fig. 4. Rate-accuracy comparison of the training-objective variants on (a) OD-VIRAT and (b) UA-DETRAC.

As shown in Table III, increasing K from 1 to 10 reduces the nominal frequency of frame uploads and cloud detections by $10\times$, while AP50 decreases from 0.741 to 0.700 on OD-VIRAT and from 0.849 to 0.636 on UA-DETRAC. The larger decrease on UA-DETRAC indicates greater sensitivity to sparse detector updates, highlighting the scene-dependent accuracy cost of reducing uplink and cloud-compute load through K.

D. Training-Objective Ablation

Fig. 4 compares the complete objective with four variants on OD-VIRAT and UA-DETRAC. The first removes the pixel term while retaining the feature and rate terms. The second removes the feature term while retaining the pixel and rate terms. The third removes the object-region masks and applies the feature and pixel terms over the full frame. The fourth uses the public pretrained DCVC-RT video model without task-specific fine-tuning.

Authorized licensed use limited to: University of Melbourne. Downloaded on November 20,2026 at 03:46:09 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3755382

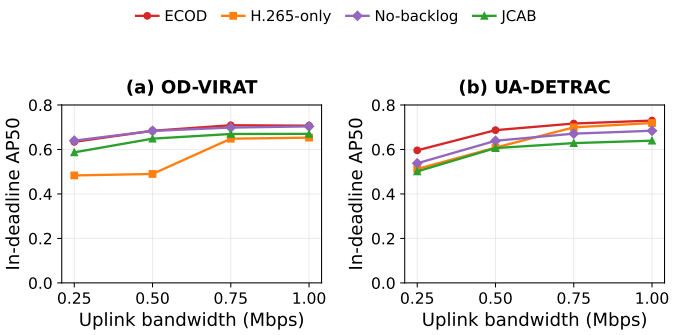

Fig. 5. In-deadline AP50 versus shared uplink bandwidth at $N = 4$ on (a) OD-VIRAT and (b) UA-DETRAC.

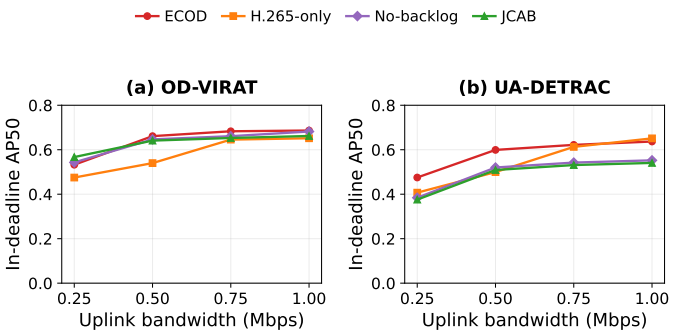

Fig. 6. In-deadline AP50 versus shared uplink bandwidth at $N = 8$ on (a) OD-VIRAT and (b) UA-DETRAC.

On OD-VIRAT, at a matched rate of 0.80 KB per frame, the complete objective reaches AP50 0.746. Removing the pixel term lowers AP50 to 0.735, showing that the pixel term adds to the gain provided by the feature term. Removing the object-region masks and applying both terms over the full frame further reduces AP50 to 0.733, demonstrating the benefit of object-region supervision. The largest change occurs when the feature term is removed: AP50 falls to 0.721, nearly the same as the pretrained generic codec at 0.722. Moreover, the codec without the feature term requires about 1.23 KB per frame to reach AP50 0.735. Together, these results show that the feature term provides the main task gain, while the pixel term and object-region masks provide additional improvements at comparable rates.

On UA-DETRAC, the complete objective achieves AP50 0.8491 at 0.612 KB per frame, compared with 0.8380 at 0.638 KB when the feature term is removed. Removing the pixel term or the object-region masks yields AP50 0.8427 at 0.627 KB and 0.8441 at 0.651 KB, respectively, showing the advantage of combining these terms at low coded sizes. At the high-rate end, the full-frame objective reaches AP50 0.8551, close to the complete objective's 0.8559, indicating that the benefit of object-region weighting is most evident under tighter rate constraints. Across both datasets, the complete objective provides a favorable rate-accuracy tradeoff in the low-rate regime targeted by ORS.

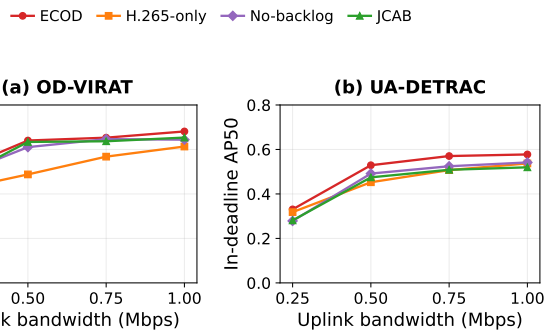

Fig. 7. In-deadline AP50 versus shared uplink bandwidth at $N = 12$ on (a) OD-VIRAT and (b) UA-DETRAC.

E. Multi-Stream Detection under Uplink Constraints

Figs. 5, 6, and 7 report multi-stream detection performance at $N = 4, 8, 12$, respectively. At $N = 8, B = 0.5$ Mbps, ECOD achieves higher AP50 than all three baselines on both datasets. Compared with No-backlog, ECOD improves AP50 from 0.647 to 0.661 on OD-VIRAT and from 0.520 to 0.599 on UA-DETRAC, supporting the benefit of backlog feedback for configuration selection. ECOD also outperforms H.265-only and JCAB under this condition.

Increasing uplink bandwidth generally improves detection accuracy, but the gains diminish as bandwidth increases. For ECOD on OD-VIRAT at $N = 8$, increasing bandwidth from 0.25 to 0.5 Mbps raises AP50 by 0.129, whereas increasing it from 0.75 to 1 Mbps adds only 0.003.

Increasing the number of streams reduces detection accuracy when the available resources remain fixed. At $B = 0.5$ Mbps, increasing N from 4 to 12 lowers ECOD AP50 from 0.684 to 0.640 on OD-VIRAT and from 0.687 to 0.529 on UA-DETRAC. This decline is consistent with increased contention for the shared edge, uplink, and cloud resources, with a larger accuracy reduction observed on UA-DETRAC.

F. Real-Time Service Performance

Table IV presents a comparison of real-time performance on OD-VIRAT and UA-DETRAC with ten evaluation metrics at two operating conditions: N=8, B=0.5 Mbps and N=12, B=0.25 Mbps. The deadline-miss ratio (DMR) measures the fraction of admitted keyframe detection tasks that miss the detection deadline τ_d. The tail latency, L_{tail}, is the end-to-end latency within which 95% of detection tasks are completed. Besides, $\bar{Q}_c$ measures the remaining decoding and detection workload of requests that have arrived at the cloud. The mean edge backlog, $\bar{Q}_e$, measures the remaining serial service workload, while $\bar{T}_e$ is mean edge service time per admitted request, excluding waiting. Cache age is the average time between the capture of the frame underlying the cached detection and the current time.

Under the bandwidth constraints reported in Table IV, mean edge backlog remains below 1.7 ms for all methods, indicating little task accumulation at the edge even with 12 streams. Mean edge service times range from 5.95 to 20.35 ms, showing that edge processing is short relative to the 200 ms detection

Authorized licensed use limited to: University of Melbourne. Downloaded on November 20,2026 at 03:46:09 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3755382

TABLE IV
REAL-TIME SERVICE AT TWO COMMON OPERATING CONDITIONS ON OD-VIRAT AND UA-DETRAC.

Method	AP50	DMR (%)	L_{tail} (ms)	$\bar{Q}_b$ (KB)	$\bar{Q}_c$ (ms)	$\bar{Q}_e$ (ms)	$\bar{T}_e$ (ms)	U_{G} (%)	ΔM_{p} (MB)	Age (ms)
OD-VIRAT, $N = 8$, $B = 0.5$ Mbps										
ECOD	0.661	9.28	218.8	7.06	17.15	1.630	15.95	13.66	1029	348.1
H.265-only	0.540	11.92	233.0	18.51	7.21	1.229	10.19	5.37	1319	560.0
No-backlog	0.647	13.61	302.7	8.89	43.41	1.653	15.92	14.51	1046	442.1
JCAB	0.641	20.11	309.4	7.31	28.39	1.653	15.60	12.89	1025	424.2
OD-VIRAT, $N = 12$, $B = 0.25$ Mbps										
ECOD	0.492	24.32	237.9	20.14	4.92	0.454	20.35	15.39	1151	1159.6
H.265-only	0.412	19.15	266.7	16.83	5.34	0.515	12.57	7.27	1983	1451.0
No-backlog	0.471	32.33	271.5	20.09	9.96	0.413	19.53	17.13	1130	1151.7
JCAB	0.450	30.84	277.1	20.24	8.66	0.443	19.43	16.79	1205	1332.0
UA-DETRAC, $N = 8$, $B = 0.5$ Mbps										
ECOD	0.599	7.66	206.3	3.61	44.09	0.078	10.37	10.27	929	259.7
H.265-only	0.500	16.65	220.0	8.83	12.52	0.579	5.95	4.87	1327	465.6
No-backlog	0.520	14.48	215.2	3.65	93.51	0.138	10.28	8.88	935	331.6
JCAB	0.509	15.36	216.2	3.45	56.71	0.152	10.06	11.31	882	356.4
UA-DETRAC, $N = 12$, $B = 0.25$ Mbps										
ECOD	0.331	21.48	237.3	7.43	2.58	0.067	10.63	12.21	954	1124.9
H.265-only	0.318	17.86	225.8	7.11	2.43	0.602	5.99	6.86	1980	974.1
No-backlog	0.279	25.54	271.2	7.05	3.17	0.056	10.33	11.79	942	1488.6
JCAB	0.281	24.25	258.4	6.87	2.74	0.057	10.10	12.15	950	1216.7

deadline under these conditions. Table IV also reports U_{G} as the mean device-level edge GPU utilization above the idle baseline, and ΔM_{p} as the sampled peak device memory usage minus the mean idle usage. The idle baseline is measured before loading the codec models.

Compared with H.265-only, ECOD achieves higher AP50 on both datasets across the reported conditions, but requires longer edge encoding time. At $N = 8, B = 0.5$ Mbps on OD-VIRAT, AP50 increases from 0.540 to 0.661 while DMR decreases from 11.92% to 9.28%, showing that the additional encoding cost does not preclude an improvement in timely detection. At $N = 12, B = 0.25$ Mbps, however, ECOD retains higher AP50 but has higher DMR than H.265-only on both datasets (24.32% versus 19.15% on OD-VIRAT and 21.48% versus 17.86% on UA-DETRAC). The comparison therefore reveals a tradeoff between detection accuracy and deadline compliance as resource constraints become tighter.

Compared with No-backlog, ECOD reduces mean cloud backlog by more than half on both datasets at $N = 8, B = 0.5$ Mbps. This reduction is accompanied by DMR decreases of 4.33 percentage points on OD-VIRAT and 6.82 percentage points on UA-DETRAC. ECOD also achieves lower mean cloud backlog and DMR than No-backlog on both datasets at $N = 12, B = 0.25$ Mbps. Because the two methods differ only in the backlog-dependent score terms, these results support the role of backlog feedback in limiting cloud workload accumulation and improving deadline compliance.

Compared with JCAB, ECOD achieves lower tail latency and cache age on both datasets at $N = 8, B = 0.5$ Mbps. For example, on OD-VIRAT, ECOD reduces L_{tail} from JCAB's 309.4 to 218.8 ms and mean cache age from 424.2 to 348.1 ms. Besides, ECOD also achieves lower tail latency and mean cache age than JCAB on both datasets at $N = 12, B = 0.25$ Mbps. These results show that ECOD reduces tail latency and keeps cached detections more up to date than JCAB.

TABLE V
REWARD-WEIGHT SENSITIVITY ON OD-VIRAT AT $N = 8$.

	$B = 0.25$ Mbps			$B = 0.5$ Mbps		
V	AP50	DMR (%)	L_{tail} (ms)	AP50	DMR (%)	L_{tail} (ms)
5	0.506	15.15	215.0	0.661	8.68	202.8
10	0.533	17.63	222.1	0.661	9.28	218.8
20	0.523	20.18	229.3	0.654	10.51	227.6

TABLE VI
REWARD-WEIGHT SENSITIVITY ON UA-DETRAC AT $N = 8$.

	$B = 0.25$ Mbps			$B = 0.5$ Mbps		
V	AP50	DMR (%)	L_{tail} (ms)	AP50	DMR (%)	L_{tail} (ms)
5	0.484	12.37	219.9	0.625	6.21	204.2
10	0.475	12.34	218.5	0.599	7.66	206.3
20	0.434	23.62	227.6	0.536	12.08	213.3

G. Sensitivity to the Reward Weight V

Tables V and VI report ECOD's sensitivity to $V \in \{5, 10, 20\}$ at $N = 8$ and $B \in \{0.25, 0.5\}$ Mbps on OD-VIRAT and UA-DETRAC, respectively. All other settings remain unchanged.

On OD-VIRAT at $B = 0.5$ Mbps, ECOD AP50 varies from 0.654 to 0.661, while increasing V from 5 to 20 raises DMR from 8.68% to 10.51% and L_{tail} from 202.8 to 227.6 ms. At $B = 0.25$ Mbps, AP50 peaks at 0.533 for $V = 10$, while DMR and tail latency increase across the tested values of V. Thus, increasing V beyond 10 provides no further accuracy benefit on OD-VIRAT and worsens timeliness at both bandwidths.

On UA-DETRAC, increasing V from 5 to 20 lowers ECOD AP50 from 0.484 to 0.434 at 0.25 Mbps and from 0.625 to 0.536 at 0.5 Mbps, accompanied by higher DMR. Tail latency

Authorized licensed use limited to: University of Melbourne. Downloaded on November 20,2026 at 03:46:09 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3755382

TABLE VII
RECOVERY-BUDGET SENSITIVITY ON OD-VIRAT.

	$B = 0.25$ Mbps			$B = 0.5$ Mbps		
τ_r (s)	AP50	DMR (%)	L_{tail} (ms)	AP50	DMR (%)	L_{tail} (ms)
0.2	0.000	N/A	N/A	0.132	0.79	93.5
0.5	0.537	11.07	215.4	0.641	3.92	203.7
1.0	0.533	17.63	222.1	0.661	9.28	218.8
2.0	0.517	17.91	227.6	0.656	9.47	221.6

TABLE VIII
RECOVERY-BUDGET SENSITIVITY ON UA-DETRAC.

	$B = 0.25$ Mbps			$B = 0.5$ Mbps		
τ_r (s)	AP50	DMR (%)	L_{tail} (ms)	AP50	DMR (%)	L_{tail} (ms)
0.2	0.000	N/A	N/A	0.039	0.40	75.8
0.5	0.432	8.55	212.2	0.615	4.76	198.0
1.0	0.475	12.34	218.5	0.599	7.66	206.3
2.0	0.464	13.32	221.0	0.590	7.04	203.5

also increases, from 219.9 to 227.6 ms and from 204.2 to 213.3 ms, respectively. The change is not strictly monotonic at 0.25 Mbps, where $V = 10$ gives slightly lower DMR and tail latency than $V = 5$. Increasing V places more weight on the profiled utility $a(o)$ relative to resource costs, whereas online AP50 also depends on whether detection results arrive in time to update the cache. On UA-DETRAC, the accompanying increase in deadline misses suggests that favoring configurations with higher profiled accuracy can offset their accuracy benefits when uplink capacity is limited. Overall, sensitivity to V depends on the scene: accuracy varies modestly on OD-VIRAT but declines more substantially on UA-DETRAC, while increasing V from 5 to 20 worsens timeliness on both datasets.

H. Sensitivity to the Recovery Time Budget

Tables VII and VIII evaluate $\tau_r \in \{0.2, 0.5, 1, 2\}$ s on OD-VIRAT and UA-DETRAC at $N = 8$, $V = 10$, and $B \in \{0.25, 0.5\}$ Mbps. Within each bandwidth setting, only τ_r varies, while the detection deadline remains $\tau_d = 200$ ms. DMR is measured against τ_d for all admitted tasks, including initialization and recovery tasks.

With $\tau_r = 0.2$ s, initialization and recovery must meet the same deadline as ordinary detection. At 0.25 Mbps, no tasks are admitted on either dataset, while at 0.5 Mbps AP50 reaches only 0.132 on OD-VIRAT and 0.039 on UA-DETRAC. For the cases with no admitted tasks, DMR and L_{tail} are undefined and reported as N/A. The resulting low AP50 shows that an overly short recovery budget can prevent reference establishment and subsequent detection, even when DMR among admitted tasks is low.

On OD-VIRAT at 0.5 Mbps, increasing τ_r from 0.5 to 1 s improves AP50 from 0.641 to 0.661, but raises DMR from 3.92% to 9.28% and L_{tail} from 203.7 to 218.8 ms. At 0.25 Mbps, however, 0.5 s already yields the highest AP50 among the tested budgets, and further relaxation reduces accuracy while increasing DMR and tail latency.

On UA-DETRAC at 0.25 Mbps, increasing τ_r from 0.5 to 1 s improves AP50 from 0.432 to 0.475, accompanied by a DMR increase from 8.55% to 12.34%. At 0.5 Mbps, 0.5 s instead provides the highest AP50 (0.615), the lowest DMR (4.76%), and the shortest tail latency (198.0 ms) among budgets of 0.5–2 s. On both datasets and at both bandwidths, extending τ_r from 1 to 2 s provides no further accuracy gain. Overall, the recovery budget must be long enough to establish references, but the accuracy and timeliness tradeoff depends on the scene and bandwidth, and a larger budget is not necessarily better.

VI. CONCLUSIONS AND FUTURE WORK

The continued growth of networked surveillance deployments has increased demand for continuous, real-time object detection across distributed camera streams. High-accuracy detectors require more compute than individual cameras can provide, and they increasingly burden edge servers as deployments grow. At the same time, edge-cloud execution must also handle uplink congestion and shared edge and cloud computation. Existing edge-cloud detection methods reduce uploaded content or adapt processing configurations to manage resource demand. This paper presents ECOD, a resource-elastic edge-cloud detection system that jointly controls edge-encoding, uplink and cloud-compute queues. Based on the current backlogs, its Lyapunov controller selects each stream's codec rate, keyframe interval, and detector fidelity to balance detection utility against resource costs subject to task-specific latency constraints. To provide fine-grained resource elasticity, ECOD introduces an ORS codec that directs low-rate bits toward detector-relevant regions while reconstructing pixel-domain frames.

Experiments on OD-VIRAT show that the ORS codec achieves AP50 0.735 at 0.72 KB per frame. With eight streams sharing a 0.5 Mbps uplink, ECOD achieves in-deadline AP50 scores of 0.661 on OD-VIRAT and 0.599 on UA-DETRAC, compared with 0.647 and 0.520 for the best baseline, respectively.

Future work will investigate lightweight tracking between cloud detections, network-condition prediction, and cross-device resource scheduling for larger multi-sensor deployments.

REFERENCES

[1] S. Ren, K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[2] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, 2016, pp. 779–788.

[3] Z. -Q. Zhao, P. Zheng, S. -T. Xu and X. Wu, "Object Detection with Deep Learning: A Review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3212–3232, Nov. 2019.

[4] Y. Wu, H. Guo, C. Chakraborty, M. R. Khosravi, S. Berretti and S. Wan, "Edge Computing Driven Low-Light Image Dynamic Enhancement for Object Detection," *IEEE Transactions on Network Science and Engineering*, vol. 10, no. 5, pp. 3086–3098, Oct. 2023.

[5] V. H. Kim and K. K. Choi, "A Reconfigurable CNN-Based Accelerator Design for Fast and Energy-Efficient Object Detection System on Mobile FPGA," *IEEE Access*, vol. 11, pp. 59438–59445, Jun. 2023.

Authorized licensed use limited to: University of Melbourne. Downloaded on November 20,2026 at 03:46:09 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3755382

[6] Y. Cheng, D. Wang, P. Zhou and T. Zhang, "Model Compression and Acceleration for Deep Neural Networks: The Principles, Progress, and Challenges," *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 126–136, Jan. 2018.

[7] H. Cheng, M. Zhang and J. Q. Shi, "A Survey on Deep Neural Network Pruning: Taxonomy, Comparison, Analysis, and Recommendations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 12, pp. 10558–10578, Dec. 2024.

[8] P. Xiao, C. Zhang, Q. Guo, X. Xiao and H. Wang, "Neural Networks Integer Computation: Quantizing Convolutional Neural Networks of Inference and Training for Object Detection in Embedded Systems," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 17, no. 10, pp. 15862–15884, Aug. 2024.

[9] H. Wang, T. Jia, Q. Wang and W. Zuo, "Relation Knowledge Distillation by Auxiliary Learning for Object Detection," *IEEE Transactions on Image Processing*, vol. 33, pp. 4796–4810, Dec. 2024.

[10] F. Iandola, S. Han, M. Moskewicz, K. Ashraf, W. Dally, K. Keutzer, "SqueezeNet: AlexNet-level Accuracy with 50x Fewer Parameters and $<$ 0.5 MB Model Size," 2016, arXiv:1602.07360.

[11] A. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, "Mobilenets: Efficient Convolutional Neural Networks for Mobile Vision Applications" 2017, arXiv:1704.04861.

[12] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 2018, pp. 4510–4520.

[13] A. Howard, M. Sandler, G. Chu, L. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q. V. Le, H. Adam, "Searching for MobileNetV3," in *2019 IEEE/CVF International Conference on Computer Vision*, Seoul, Korea, 2019, pp. 1314–1324.

[14] C. Dong, C. Pang, Z. Li, X. Zeng and X. Hu, "PG-YOLO: A Novel Lightweight Object Detection Method for Edge Devices in Industrial Internet of Things," *IEEE Access*, vol. 10, pp. 123736–123745, Nov. 2022.

[15] G. Cheng, X. Yuan, X. Yao, K. Yan, Q. Zeng, X. Xie, J. Han, "Towards Large-Scale Small Object Detection: Survey and Benchmarks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 11, pp. 13467–13488, Nov. 2023.

[16] X. Wu, W. Li, D. Hong, R. Tao and Q. Du, "Deep Learning for Unmanned Aerial Vehicle-Based Object Detection and Tracking: A survey," *IEEE Geoscience and Remote Sensing Magazine*, vol. 10, no. 1, pp. 91–124, Mar. 2022.

[17] Y. -Y. Chen, Y. -H. Lin, Y. -C. Hu, C. -H. Hsia, Y. -A. Lian and S. -Y. Jhong, "Distributed Real-Time Object Detection Based on Edge-Cloud Collaboration for Smart Video Surveillance Applications," *IEEE Access*, vol. 10, pp. 93745–93759, Nov. 2022.

[18] Y. Wang, C. Yang, S. Lan, L. Zhu and Y. Zhang, "End-Edge-Cloud Collaborative Computing for Deep Learning: A Comprehensive Survey," *IEEE Communications Surveys & Tutorials*, vol. 26, no. 4, pp. 2647–2683, Apr. 2024.

[19] H. Yuan, J. Bi, Z. Wang, J. Zhang, M. Zhou and R. Buyya, "Multi-perspective and Energy-Efficient Deep Learning in Edge Computing," *IEEE Internet of Things Journal*, vol. 13, no. 3, pp. 3988–4003, Feb. 2026.

[20] R. Xu, S. Razavi and R. Zheng, "Edge Video Analytics: A Survey on Applications, Systems and Enabling Techniques," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 4, pp. 2951–2982, Fourth Quarter 2023.

[21] J. Huang, J. Wan, B. Lv, Q. Ye and Y. Chen, "Joint Computation Offloading and Resource Allocation for Edge-Cloud Collaboration in Internet of Vehicles via Deep Reinforcement Learning," *IEEE Systems Journal*, vol. 17, no. 2, pp. 2500–2511, Jun. 2023.

[22] J. Shao, X. Zhang and J. Zhang, "Task-Oriented Communication for Edge Video Analytics," *IEEE Transactions on Wireless Communications*, vol. 23, no. 5, pp. 4141–4154, May 2024.

[23] T. Y.-H. Chen, L. Ravindranath, S. Deng, P. Bahl, and H. Balakrishnan, "Glimpse: Continuous, Real-Time Object Recognition on Mobile Devices," in *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, Seoul, South Korea, 2015, pp. 155–168.

[24] Y. Li, A. Padmanabhan, P. Zhao, Y. Wang, G. H. Xu, and R. Netravali, "Reducto: On-camera Filtering for Resource-efficient Real-time Video Analytics," in *Proc. Annu. Conf. ACM Special Interest Group Data Commun.*, Virtual Event, 2020, pp. 359–376.

[25] K. Du, A. Pervaiz, X. Yuan, A. Chowdhery, Q. Zhang, H. Hoffmann, J. Jiang, "Server-driven Video Streaming for Deep Learning Inference," in *Proc. Annu. Conf. ACM Special Interest Group Data Commun.*, Virtual Event, 2020, pp. 557–570.

[26] Z. Yang, X. Wang, J. Wu, Y. Zhao, Q. Ma, X. Miao, L. Zhang, and Z. Zhou, "EdgeDuet: Tiling Small Object Detection for Edge Assisted Autonomous Mobile Vision," *IEEE/ACM Transactions on Networking*, vol. 31, no. 4, pp. 1765–1778, Aug. 2023.

[27] Z. Wang, X. He, Z. Zhang, Y. Zhang, Z. Cao, W. Cheng, W. Wang and Y. Cui, "Edge-Assisted Real-Time Video Analytics With Spatial–Temporal Redundancy Suppression," *IEEE Internet of Things Journal*, vol. 10, no. 7, pp. 6324–6335, Apr. 2023.

[28] H. Zhang, G. Ananthanarayanan, P. Bodik, M. Philipose, P. Bahl, and M. J. Freedman, "Live Video Analytics at Scale with Approximation and Delay-Tolerance," in *Proc. 14th USENIX Symp. Netw. Syst. Design Implement.*, Boston, MA, USA, 2017, pp. 377–392.

[29] B. Zhang, X. Jin, S. Ratnasamy, J. Wawrzynek, and E. A. Lee, "AWStream: Adaptive Wide-Area Streaming Analytics," in *Proc. Annu. Conf. ACM Special Interest Group Data Commun.*, Budapest, Hungary, 2018, pp. 236–252.

[30] S. Zhang, C. Wang, Y. Jin, J. Wu, Z. Qian, M. Xiao, and S. Lu, "Adaptive Configuration Selection and Bandwidth Allocation for Edge-Based Video Analytics," *IEEE/ACM Transactions on Networking*, vol. 30, no. 1, pp. 285–298, Feb. 2022.

[31] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," in *Proc. 5th Int. Conf. Learn. Represent. (ICLR)*, Toulon, France, 2017.

[32] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," in *Proc. 6th Int. Conf. Learn. Represent. (ICLR)*, Vancouver, BC, Canada, 2018.

[33] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, "Learned image compression with discretized Gaussian mixture likelihoods and attention modules," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Seattle, WA, USA, 2020, pp. 7939–7948.

[34] J. Liu, H. Sun, and J. Katto, "Learned image compression with mixed transformer-CNN architectures," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Vancouver, BC, Canada, 2023, pp. 14388–14397.

[35] G.-H. Wang, J. Li, B. Li, and Y. Lu, "EVC: Towards real-time neural image compression with mask decay," in *Proc. 11th Int. Conf. Learn. Represent. (ICLR)*, Kigali, Rwanda, 2023.

[36] L.-Y. Duan, J. Liu, W. Yang, T. Huang, and W. Gao, "Video coding for machines: A paradigm of collaborative compression and intelligent analytics," *IEEE Transactions on Image Processing*, vol. 29, pp. 8680–8695, Aug. 2020.

[37] X. Sheng, L. Li, D. Liu, and H. Li, "VNVC: A versatile neural video coding framework for efficient human-machine vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 7, pp. 4579–4596, Jul. 2024.

[38] Z. Jia, B. Li, J. Li, W. Xie, L. Qi, H. Li, and Y. Lu, "Towards Practical Real-Time Neural Video Compression," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Nashville, TN, USA, 2025, pp. 12543–12552.

[39] S. -W. Kim, K. Ko, H. Ko and V. C. M. Leung, "Edge-Network-Assisted Real-Time Object Detection Framework for Autonomous Driving," *IEEE Network*, vol. 35, no. 1, pp. 177–183, Jan. 2021.

[40] Z. Han, Y. Yang, W. Wang, L. Zhou, T. N. Nguyen and C. Su, "Age Efficient Optimization in UAV-Aided VEC Network: A Game Theory Viewpoint," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 12, pp. 25287–25296, Dec. 2022.

[41] W. Wang, G. Srivastava, J. C.-W. Lin, Y. Yang, M. Alazab and T. R. Gadekallu, "Data Freshness Optimization Under CAA in the UAV-Aided MECN: A Potential Game Perspective," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 11, pp. 12912–12921, Nov. 2023.

[42] Y. Yang, X. Wei, R. Xu and L. Peng, "Joint Optimization of AoI, SINR, Completeness, and Energy in UAV-Aided SDCNs: Coalition Formation Game and Cooperative Order," *IEEE Transactions on Green Communications and Networking*, vol. 6, no. 1, pp. 265–280, Mar. 2022.

[43] M. J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*. San Rafael, CA, USA: Morgan & Claypool, 2010.

[44] J. Duda, "Asymmetric numeral systems: Entropy coding combining speed of Huffman coding with compression rate of arithmetic coding," *arXiv preprint arXiv:1311.2540*, 2013.

[45] H. Ullah, A. Khan, A. Munir, and H. Kalva, "OD-VIRAT: A large-scale benchmark for object detection in realistic surveillance environments," *arXiv preprint arXiv:2507.12396*, 2025.

[46] L. Wen, D. Du, Z. Cai, Z. Lei, M.-C. Chang, H. Qi, J. Lim, M.-H. Yang, and S. Lyu, "UA-DETRAC: A new benchmark and protocol

Authorized licensed use limited to: University of Melbourne. Downloaded on November 20,2026 at 03:46:09 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2026.3755382

for multi-object detection and tracking," *Computer Vision and Image Understanding*, vol. 193, Art. no. 102907, Apr. 2020.

[47] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLOv8," version 8.0.0, 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[48] A. Paszke et al., "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems*, vol. 32, Vancouver, BC, Canada, 2019, pp. 8024-8035.

[49] H. Luo, W. Xie, X. Wang, and W. Zeng, "Detect or Track: Towards Cost-Effective Video Object Detection/Tracking," in *Proc. AAAI Conf. Artif. Intell.*, Honolulu, HI, USA, 2019, pp. 8803–8810.



Rui Lai received the B.E. degree and the M.E. degree in mechanical engineering from Huazhong University of Science and Technology, Wuhan, China, in 2019 and 2021, respectively. He is currently pursuing the Ph.D. degree in the School of Automation Science and Electrical Engineering, Beihang University, Beijing, China. His research interests include edge computing, evolutionary optimization and machine learning.



Guanghong Gong received her B.S., M.S. and Ph.D. degrees from Beihang University, in 1990, 1993 and 1997, respectively. She is currently a professor in the Key Laboratory of Advanced Simulation Aeronautical Technology, Beihang University. She is also a managing director of the China Simulation Society. She has won the Science and Technology Progress Award of the Ministry of Education and the National Defense Science and Technology Progress Award for many times. Her research interests are virtual reality, artificial intelligence and distributed interactive simulation.



Haitao Yuan (S'15–M'17–SM'21) received the Ph.D. degree in Computer Engineering from New Jersey Institute of Technology (NJIT), Newark, NJ, USA in 2020. He is currently a Deputy Director in the Department of Strategic Development, Wenchang International Aerospace City, Hainan, China. He is currently an Associate Professor at the School of Automation Science and Electrical Engineering at Beihang University, Beijing, China, and he is named in the world's top 2% of Scientists List since 2022. His research interests include the Internet of Things, deep learning, data-driven optimization, and computational intelligence algorithms. He received the Chinese Government Award for Outstanding Self-Financed Students Abroad, the 2021 Hashimoto Prize from NJIT, the Best Work Award in the 17th International Conference on Networking, Sensing and Control, and the Best Student Work Award Nominee in the 2024 IEEE International Conference on Systems, Man, and Cybernetics. He received the Best Paper in Theory and Technology from IEEE Robotics & Automation Society and IEEE Systems, Man & Cybernetics Society. He is an associate editor for IEEE Transactions on Systems, Man, and Cybernetics: Systems, IEEE Internet of Things Journal, and Expert Systems With Applications.



Jia Zhang (Senior Member, IEEE) received a PhD degree in computer science from the University of Illinois at Chicago, Chicago, IL, USA in 2000. She is currently the Cruse C. and Marjorie F. Calahan Centennial Chair in Engineering, Professor of Department of Computer Science in the Lyle School of Engineering at Southern Methodist University. Her research interests emphasize the application of machine learning and information retrieval methods to tackle data science infrastructure problems, with a recent focus on scientific workflows, provenance mining, software discovery, knowledge graphs, and interdisciplinary applications of all of these interests in earth science. She is a senior member of the IEEE.



Rajkumar Buyya (Fellow, IEEE) is a Redmond Barry Distinguished Professor and Director of the Cloud Computing and Distributed Systems (CLOUDS) Laboratory at the University of Melbourne, Melbourne, VIC, Australia. He received the B.E. and M.E. degrees in Computer Science and Engineering from Mysore and Bangalore Universities in 1992 and 1995, respectively, and the Ph.D. degree in Computer Science and Software Engineering from Monash University, Melbourne, Australia, in 2002. He has authored over 800 publications and seven textbooks and is one of the highly cited authors in computer science and software engineering worldwide.

Authorized licensed use limited to: University of Melbourne. Downloaded on November 20,2026 at 03:46:09 UTC from IEEE Xplore. Restrictions apply.
© 2026 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.