Edge-assisted Online On-device Object Detection for Real-time Video Analytics

Mengxi Hanyao, Yibo Jin, Zhuzhong Qian[§], Sheng Zhang[§], Sanglu Lu
State Key Laboratory for Novel Software Technology, Nanjing University, China
Email: {{mxhanyao, yibo.jin}@smail.nju.edu.cn, {qzz, sheng, sanglu}@nju.edu.cn}

***Abstract*—Real-time on-device object detection for video analytics fails to meet the accuracy requirement due to limited resources of mobile devices while offloading object detection inference to edges is time-consuming due to the transference of video data over edge networks. Based on the system with both on-device object tracking and edge-assisted analysis, we formulate a non-linear time-coupled program over time, maximizing the overall accuracy of object detection by deciding the frequency of edge-assisted inference, under the consideration of both dynamic edge networks and the constrained detection latency. We then design a learning-based online algorithm to adjust the threshold for triggering edge-assisted inference on the fly in terms of the object tracking results, which essentially controls the deviation of on-device tracking between two consecutive frames in the video, by only taking previously observable inputs. We rigorously prove that our approach only incurs sub-linear dynamic regret for the optimality objective. At last, we implement our proposed online schema, and extensive testbed results with real-world traces confirm the empirical superiority over alternative algorithms, in terms of up to 36% improvement on detection accuracy with ensured detection latency.**

I. Introduction

Recent years have witnessed a rapid increase in the number of mobile AR (Augmented Reality) devices [1], smart phones equipped with cameras [2], and so on, which can only conduct limited computation for basic analytics [3]. In typical scenarios for those various real-time applications [4, 5], the live video streams generated from devices have strong requirement on the fast treatment, e.g., real-time mix reality [6] requires the system to have a comprehensive understanding of different objects and instances as quickly as possible in the real world. As a result, the techniques in computer vision [7, 8] like object detection, classification and related rendering techniques need to be applied nearly in real-time for fast video analytics.

However, detecting objects online for real-time video analytics is non-trivial, which involves the tradeoff between the limited resources of devices and the video transference with nearby edges; the dynamics of edge networks to be tackled for ensured latency for frame transference; and the time-coupled deviation regarding the on-device object tracking. Particularly, online object detection faces multiple challenges as follows:

§ These are corresponding authors.

This work was supported in part by the National Key R&D Program of China (2017YFB1001801), the National Science Foundation of China (61832005, 61872175), the Natural Science Foundation of Jiangsu Province (BK20181252), the Collaborative Innovation Center of Novel Software Technology and Industrialization, and the Nanjing University Innovation and Creative Program for PhD (CXCY19-25).

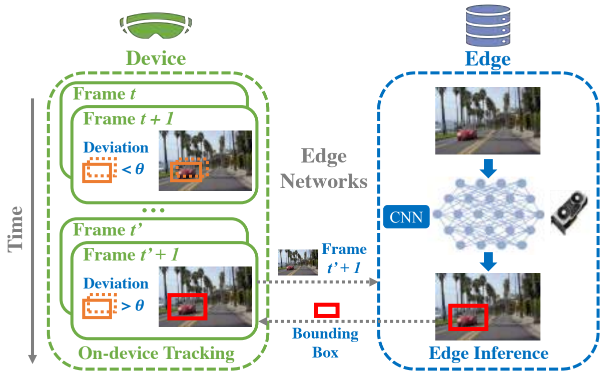


Fig. 1: System architecture: edge-assisted on-device inference

First and foremost, on-device object detection is resource-consuming while the video transference with nearby edges is time-consuming, which entails dynamically navigating the trade-offs between on-device computation and the transference of video frames. More specifically, since analyzing the videos frame by frame is computation-intensive [4, 9], the inference often has high resource demands and the on-device computation easily leads to a long execution delay [10]. For example, neural networks such as YOLO [11], Faster-RCNN [12] and Mask-RCNN [13] often require hundreds of milliseconds for object detection per frame, which is not suitable for a video with tens of frames generated within a second, e.g., 24fps (24 frames per second). Meanwhile, offloading the computation of video inference to nearby edges essentially leads to the video transference of related frames, which is also time-consuming. Therefore, to balance the on-device computation and the video transference over edge networks, as shown in Fig. 1, we resort to the system combining both on-device *object tracking* and frequent *edge-assisted inference* under constrained latency.

However, the frequency of edge-assisted inference is restricted by the condition of edge networks, i.e., the dynamic changes of the bandwidth over time. For instance, the frame size of a high quality video with the resolution higher than 1080 is several megabytes or more, which easily incurs tens of milliseconds for transference per frame. Since the peak bandwidth is often several times higher than the lowest one, the transmission delay of corresponding video data for edge inference over edge networks varies dramatically over time. And for real-time video analytics, the latency of object detection including both device/edge inference and video transference should be strongly ensured. Otherwise, many frames have to be queued for treatment, resulting in unbounded performance and bad QoE. Designing an effective online algorithm to maximize the overall detection accuracy is still a challenging problem.

Moreover, the deviation of online on-device tracking regarding the objects can only be revealed after the actual inference, which hampers us from efficient online decisions. That is, in order to decrease the frequency of edge-assisted inference, one may prefer to request the help of nearby edges only when the deviation of object tracking between two consecutive frames is large, i.e., the deviation exceeds a threshold. Unfortunately, before conducting the actual inference for videos, the deviations during object tracking between two consecutive frames are unknown. The difficulty for online algorithm design caused by such obliviousness to the uncertain inputs further escalates due to the threshold, as the threshold is actually used to control the frequency of edge-assisted analysis and corresponding video transference over edge networks. Intuitively, one may "learn" in an online manner from the "penalty" incurred by the online decisions just made in terms of the detection accuracy, after analyzing all of the frames in a specific epoch, and seek to make better decisions as time goes.

Existing research falls insufficient for addressing the aforementioned challenges. Some works studied edge-assisted on-device inference for lower detection complexity and latency [6, 14–19]. Others focused on the partition of inference models [20–23] between devices and the edge. And the rest investigated the online inference configuration [7, 24–27] adaptively. However, few of them has considered online edge-assisted on-device inference for frames under constrained latency and dynamic edge networks with theoretical performance guarantee.

In this paper, we investigate the problem of optimizing the overall detection accuracy for real-time video analytics over resource-constrained device and dynamic edge networks under constrained detection latency, whose decisions should be made on the fly. We make the following contributions:

We model the desired problem as a non-linear time-coupled program. The proposed problem maximizes the overall object detection accuracy subject to the constraint of overall detection delay within a time period, including both device/edge inference delay as well as the video transference delay. In order to balance the on-device computation and the video transference, we adjust the threshold for triggering edge-assisted inference. That is, when the deviation of related object tracking between two consecutive frames exceeds the threshold, the device needs to request the nearby edge for further analysis and the device corrects the local tracking results by using the edge inference. The blindness that the inference results are only revealed after the decisions makes the problem difficult to be tackled.

We propose and design a novel polynomial-time online algorithm that decouples the proposed problem into a series of subproblems. Each subproblem within an epoch essentially uses the results revealed from the previous one as the penalty to update the latest threshold for triggering edge-assisted inference. In order to balance the transference and inference, a well-designed threshold is studied and proposed to incur less uploaded frames and high detection accuracy under constrained detection latency within an epoch. Through delicately designed transformations, the proposed subproblem within each epoch is actually solved by using linear programming technique on its substitute. Although multiple transformations are used, the algorithmic goal is still unchanged, i.e., comparing the results of our proposed algorithm with its optimum. Via rigorous theoretical analysis, we prove that the dynamic regret, which is a common metric for an online algorithm and characterizes the optimality accuracy relative to a sequence of instantaneous optimizers with known inputs, with regard to our entire online algorithm O^3, only grows sub-linearly along with time.

We implement our online schema O^3 based on on-device tracking technique, Optical Flow [28], and commonly used detection network, Faster RCNN, at nearby edge. The extensive testbed results with real-world traces in terms of both videos and bandwidth over edge networks show that O^3 achieves up to 36% improvement on the detection accuracy compared with multiple state-of-the-art algorithms. O^3 also behaves well for different workloads and various realistic settings, only incurring tens of milliseconds overhead.

TABLE I: Latency and Accuracy in Our Case Study

Detection Methods	Latency per Frame		Detection Accuracy
	max.	***avg.***	
Faster RCNN [12] (*On Edge*)	107ms	83ms	∼100%
MobileNet [29] (*On Device*)	70ms	52ms	53%
Edge-assisted[1] *Optical Flow*	121ms	36.9ms	77%

1. Edge-assisted detection has already considered the transference of frames.

TABLE II: Study on Encoding Consumption[1]

Resolution	Encoding Time (s)	Output Data Size (MB)
1080P	8.143	3.8
720P	5.673	2.7
540P	2.170	1.4

1. We use ffmpeg to encode a 5-second video on a server with 2080 Ti.

(a) Jetson TX2 Kit

(b) Detection Preview

Fig. 2: Case study for 24fps DrivingPOV [30]

II. System Model

In this section, we first introduce our preliminary case study regarding the edge-assisted detection, and then explain our overall system model. Then, we illustrate the proposed control problem with the objective of maximizing the overall accuracy of object detection with constrained detection latency.

A. Preliminary Case Study: Edge-assisted Detection

Sophisticated object detection, often implemented by neural networks, uses each frame of the video as the input to predict the regions of interest (RoI) [31], which are potential locations of objects in the frame. Some works [6, 32] have shown that the techniques of RoI encoding actually reduce the size of raw images and accelerate the transference over the edge network. Unfortunately, the encoding process incurs long computation delays, shown in Table II, which is unsuitable for mobile devices. For example, for a video with the resolution higher than 720p, the encoding time exceeds the overall time scope considered. After the study above, we conclude that a promising approach is *to track the objects on-device while the nearby edge frequently corrects the tracking results*.

Requirements for Object Detection: On one hand, real-time video analytics often has strong latency requirement for object detection, since the frame rate of a video is often 24fps (i.e., 24 frames per second) or higher. That is, the average inference delay for one frame should be controlled within tens of milliseconds. However, the average inference delay of the commonly used object detection network Faster RCNN at edges, even equipped with Dell PowerEdge R740 and Geforce RTX GPU 2080 Ti, is over 83ms, as shown in Table I, which is not suitable for per frame inference ($83*24 > 1000$) in real-time video analytics. On the other hand, although some light-weight detection models are proposed for devices, the accuracy requirement may not be ensured without the help of nearby edges, e.g., the accuracy of object detection by using light-weight MobileNet [29] on Jetson TX2 is only 53%, as shown in Table I, which should be improved as much as possible. Although SOTA frameworks can also achieve a 30.8FPS [33] speed on a Jetson AGX Xavier, most commercial devices do not possess TensorRT module to accelerate video analytics.

Edge-assisted Object Detection: A new trend is to conduct edge-assisted real-time video analytics with light-weight on-device calculation under constrained latency. More specifically, the device tracks the objects itself using light-weight methods, e.g., Optical Flow [28], which calculate the movement of all pixels between consecutive frames. Only when the movement, i.e., the deviation, between two consecutive frames exceeds a threshold, the current frame needs to be uploaded to the edge for further inference. As a result, only 5 frames within a second are uploaded to the edge in our case study as shown in Fig. 2, incurring at most 135ms per frame for both transmission and edge inference. Although edge-assisted inference actually increases the latency, the average latency for all frames is only 36.9ms, i.e., the overall latency is controlled within a second, and the accuracy reaches to 77%, instead of 53% by using just on-device object detection, MobileNet. Note that the overall latency for just edge inference exceeds the deadline requirement, and MobileNet is also not suitable for the videos with higher frame rate than 24fps.

This simple case reveals two important findings: (1) Updating the tracking results frequently with the edge easily incurs high latency regarding both frame transference and edge inference while on-device detection without edge inference leads to low accuracy, and (2) Offloading suitable frames to nearby edges for detecting objects under constrained latency with light-weight on-device tracking method actually improves the accuracy, and is suitable for real-time video analytics.

B. System Settings and Models

We model our target device-edge system for object detection as follows. The system is studied over a series of epochs $\mathcal{T} = \{1, ..., T\}$, and the time duration for each epoch is fixed, e.g., 1 second. Within each epoch, we denote by $\mathcal{F}_t$ the set of frames, e.g., for a video with 24fps, $|\mathcal{F}_t| = 24, \forall t \in \mathcal{T}$. The object detection essentially decides the regions of interest (RoIs) for each frame. We use $\mathcal{B}_{f,t}, \forall f \in \mathcal{F}_t, \forall t \in \mathcal{T}$ to denote the set of regions of interest, and the element $\forall b \in \mathcal{B}_{f,t}$ is actually a rectangle in frame (a.k.a. bounding box [34]), which can be represented by two coordinates of points. Note that, the inference may contain multiple RoIs. To distinguish the regions detected by different roles, i.e., the device, the edge and the ground truth, we use $\mathcal{B}^D_{f,t}$, $\mathcal{B}^E_{f,t}$ and $\mathcal{B}^G_{f,t}$, respectively.

Control Decisions: We introduce our control decision by using θ_t to denote the threshold for triggering edge-assisted inference in terms of the movement between two consecutive frames. If the movement between two consecutive frames reaches θ_t, current frame needs to be uploaded to the edge for further inference, and the bounding boxes are then generated from the edge inference. Otherwise, the device uses its own bounding boxes from object tracking technique locally. We use the binary variable $I_{f,t}$ to indicate whether current frame f in epoch t needs to be uploaded to nearby edge as follows:

$$I_{f,t} = \mathbb{I}[move_{f,prec(f)} \geq \theta_t] = \frac{sgn(move_{f,prec(f)} - \theta_t) + 1}{2},$$

where $move_{f,prec(f)}$ is the movement of all pixels between two consecutive frame $prec(f)$ and f in epoch t; $prec(f)$ is the preceding frame of f; and function $sgn(x)$ indicates the sign of x, i.e., $sgn(x) = 1$ if $x \geq 0$, otherwise $sgn(x) = -1$.

Detection Accuracy Model: For each bounding box b either from $\mathcal{B}^D_{f,t}$ (i.e., the device) or $\mathcal{B}^E_{f,t}$ (i.e., the edge), the accuracy is measured by the overlapped area between b and $b^* \in \mathcal{B}^G_{f,t}$, i.e., the overlapped area between the inference bounding box and the bounding box from the ground truth. Such overlapped area is calculated by using the coordinates of points of both b and b^*. Then, we denote by IoU_{b,b^*} (i.e., the Intersection Over Union) the ratio of the overlapped area (intersection of b and b^*) to the area of bounding box b^*. The accuracy depends on the maximal IoU, i.e., $\max_{b \in \mathcal{B}^I_{f,t}} \{IoU_{b,b^*}\}$ due to the outputs of multiple regions of interests, where $\mathcal{B}^I_{f,t}$ refers to the set of bounding boxes either from the device or the edge. Thus, for all objects in the ground truth, the overall accuracy is measured by $\sum_{b^* \in \mathcal{B}^G_{f,t}} \max_{b \in \mathcal{B}^I_{f,t}} \{IoU_{b,b^*}\}$. Meanwhile, $\forall b \in \mathcal{B}^I_{f,t}$ and $\forall b^* \in \mathcal{B}^G_{f,t}$, the intersection over union IoU_{b,b^*} equals

$$IoU_{b,b^*} = I_{f,t} \cdot IoU^E_{b,b^*} + (1 - I_{f,t}) \cdot IoU^D_{b,b^*}.$$

Detection Latency Model: For the on-device object tracking, we use e^D_t to denote the inference delay by device itself for each frame in epoch t. If the frame is further decided to be uploaded to nearby edge for inference, the delay in terms of both frame transference and edge inference is considered as $e^E_t = \frac{d_t}{b_t} + g_t$, where d_t is the size of a frame in epoch t, b_t is the bandwidth between device and edge in epoch t, and g_t

is the edge inference delay for each frame in epoch t. Thus, the detection latency for one frame f in epoch t is

$$e^I_{f,t} = I_{f,t} \cdot e^E_t + (1 - I_{f,t}) \cdot e^D_t.$$

As a result, the overall detection latency for all the frames in epoch t is considered as follows:

$$\textstyle\sum_f e^I_{f,t} = \sum_f \{I_{f,t} \cdot e^E_t + (1 - I_{f,t}) \cdot e^D_t\},$$

which should be controlled within the duration of an epoch.

C. Problem Formulation and Challenges

Control Problem $\mathscr{P}^G$: With the above system model, we propose the following control problem regarding the real-time edge-assisted on-device object detection, in order to maximize the overall detection accuracy with constrained latency:

$$\begin{aligned}
\max \quad & \sum_{t,f} \sum_{b^* \in \mathcal{B}^G_{f,t}} \max_{b \in \mathcal{B}^I_{f,t}} \{I_{f,t} \cdot IoU^E_{b,b^*} + (1 - I_{f,t}) \cdot IoU^D_{b,b^*}\} \\
s.t. \quad & \sum_{f \in \mathcal{F}_t} \{I_{f,t} \cdot e^E_t + (1 - I_{f,t}) \cdot e^D_t\} \le \sigma, \forall t \in \mathcal{T}, && (1) \\
& I_{f,t} = \mathbb{I}[move_{f,prec(f)} \ge \theta_t], \forall f \in \mathcal{F}_t, t \in \mathcal{T}, && (2) \\
var. \quad & \theta_t \in \mathbb{R}_{\ge 0}, \forall t \in \mathcal{T}. && (3)
\end{aligned}$$

The objective is to maximize the overall detection accuracy. Constraint (1) restricts the overall detection time within the epoch, i.e., bounded by σ. Constraint (2) defines the relationship between binary variable and the threshold for triggering edge-assisted inference. At last, Constraint (3) further specifies the domains of our control variables.

Control Problem $\mathscr{P}^E$ with Observable Inputs: Due to the fact that the actual ground truth is hard to be obtained online, especially when some of the frames are treated locally instead of being uploaded to nearby edges, we have to solve the problem with only observable inputs as follows:

$$\begin{aligned}
\max & \sum_{t,f} \sum_{b^* \in \mathcal{B}^E_{f,t}} \max_{b \in \mathcal{B}^I_{f,t}} \{I_{f,t} \cdot IoU^E_{b,b^*} + (1 - I_{f,t}) \cdot IoU^D_{b,b^*}\} \\
s.t. & \sum_{f \in \mathcal{F}_t} \{I_{f,t} \cdot e^E_t + (1 - I_{f,t}) \cdot e^D_t\} \le \sigma, \forall t \in \mathcal{T}, && (4) \\
& I_{f,t} = \mathbb{I}[move_{f,prec(f)} \ge \theta_t], \forall f \in \mathcal{F}_t, t \in \mathcal{T}, && (5) \\
var. & \ \theta_t \in \mathbb{R}_{\ge 0}, \forall t \in \mathcal{T}, && (6)
\end{aligned}$$

where the best suitable bounding boxes used for comparison are obtained from edge inference $\mathcal{B}^E_{f,t}$ for those uploaded frames, instead of the actual ground truth $\mathcal{B}^G_{f,t}$. Note that, for the frames treated on-device, the whole term $I_{f,t} \cdot IoU^E_{b,b^*}$ is omitted, and there is no need to obtain the actual value of related IoU^E_{b,b^*}. For clarity, we use $\mathcal{P}^G$ and $\mathcal{P}^E$ to represent the objectives of $\mathscr{P}^G$ and $\mathscr{P}^E$ in their canonical forms, respectively. That is, $\mathscr{P}^G = \min:\ \mathcal{P}^G$ and $\mathscr{P}^E = \min:\ \mathcal{P}^E$ defined in their corresponding domains.

Algorithmic Goal: We first introduce the concise notations: $\widetilde{\boldsymbol{\theta}}$ refers to a feasible solution solved from $\mathscr{P}^E$ in an online manner, which is the aggregation of $\{\widetilde{\theta}_t\}$, i.e., the column vector. We use $\boldsymbol{\theta}^*$ to denote the optimal decision that optimizes the objective function in each epoch by observing the corresponding inputs, i.e., the aggregation of $\{\theta^*_t\}$ based on the ground truth. Then, our algorithmic goal is to design the algorithm which produces $\widetilde{\boldsymbol{\theta}}$ in an online manner, while the relationship between $\mathcal{P}^E(\widetilde{\boldsymbol{\theta}})$ and $\mathcal{P}^G(\boldsymbol{\theta}^*)$ should be revealed. We define the dynamic regret [35], i.e., the difference between the long-term objective value of the online decisions that are made without knowing the inputs per epoch and the long-term objective value per epoch by observing the inputs:

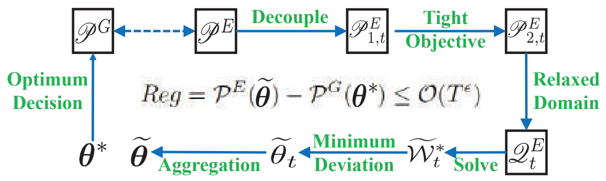


Fig. 3: Roadmap of our proposed online schema

$$Reg := \mathcal{P}^E(\widetilde{\boldsymbol{\theta}}) - \mathcal{P}^G(\boldsymbol{\theta}^*),$$

which is a common metric to measure the performance of an online algorithm compared with its optimum.

However, solving $\mathscr{P}^E$ with observable inputs is still challenging because the deviation in terms of the online object tracking is only revealed after the actual inference.

III. Online Algorithm Design

The intuition of our proposed online schema is that, within each epoch, we should "learn" from the accuracy incurred by the online decision just made, and seek to make a better decision in the next time period, since the accuracy actually shows the inference results from those previous epochs. We design a novel polynomial-time online algorithm for edge-assisted object detection that overcomes the obliviousness to uncertain inference results, which uses observable inputs from the previous epoch through a series of subproblems.

As shown in Fig. 3, the relationships between all of these subproblems proposed are illustrated. Although the key operations in O^3 is simple, all of the proposed subproblems are useful to facilitate the theoretical analysis shown later.

A. Notations and Transformations

The transformations regarding the problem are needed for designing an online control algorithm. For the ease of the presentation, we simplify the representation of $\mathscr{P}^E$ as follows:

$$\begin{aligned}
\min_{\{\mathcal{I}_t\}} \quad & \mathcal{P}^E = \sum_t \mathcal{P}^E_t(\mathcal{I}_t) \\
s.t. \quad & h_t(\mathcal{I}_t) \le 0, \quad \mathcal{I}_t \in \mathcal{X}_{\theta_t}, \quad \theta_t \in \mathbb{R}_{\ge 0}, \quad \forall t \in \mathcal{T},
\end{aligned}$$

where $\mathcal{I}_t$ is the aggregation of $\{I_{f,t}\}$, i.e., the column vector; h_t is the canonical form of Constraint (4); $\mathcal{X}_{\theta_t}$ is the related domain of all feasible values defined in Constraint (5); and $\mathcal{P}^E_t$ is the instantaneous objective per epoch, defined as

$$S - \sum_{f,b^* \in \mathcal{B}^E_{f,t}} \max_{b \in \mathcal{B}^I_{f,t}} \{I_{f,t} \cdot IoU^E_{b,b^*} + (1 - I_{f,t}) \cdot IoU^D_{b,b^*}\},$$

where S is a large constant to ensure $\mathcal{P}^E_t$ is positive.

To solve $\mathscr{P}^E$, the Lagrange multiplier λ_t is introduced as

$$\min_{\mathcal{I}_t} \max_{\lambda_t} \sum_t \left\{\mathcal{P}_t^E + \lambda_t \cdot 0\right\}, s.t.\ h_t(\mathcal{I}_t) \leq 0.$$

Then, we try to minimize the instantaneous objective $\mathscr{P}_{1,t}^E$ with respect to the primal variable $\mathcal{I}_t$:

$$\begin{aligned} \min_{\mathcal{I}} \quad & \mathcal{P}_{1,t}^E = \nabla(\mathcal{P}_{t-1}^E(\mathcal{I}_{t-1}))^\top(\mathcal{I} - \mathcal{I}_{t-1}) + \frac{||\mathcal{I} - \mathcal{I}_{t-1}||^2}{2\alpha} \\ s.t. \quad & h_t(\mathcal{I}) \leq 0, \quad \mathcal{I} \in \mathcal{X}_{\theta_t}, \quad \theta_t \in \mathbb{R}_{\geq 0}, \end{aligned}$$

where it is actually an approximation for epoch t, and obtaining $\mathcal{I}_t$ is essentially to "learn" from previous epoch by using gradient-based approach. Although $\nabla(\mathcal{P}_{t-1}^E(\mathcal{I}_{t-1}))$ is the gradient with respect to $\mathcal{I}_{t-1}$, it is actually a function with respect to θ_{t-1}. As a result, it is still hard to obtain the gradient since Constraint (5) is non-linear and contains a binary indicator. Thus, we try to solve a feasible objective instead of $\mathcal{P}_{1,t}^E$, defined as $\mathscr{P}_{2,t}^E$, to obtain a better decision:

$$\begin{aligned} \min_{\mathcal{I}} \quad & \mathcal{P}_{2,t}^E = \mathcal{P}_{t-1}^E(\mathcal{I}) \\ s.t. \quad & h_t(\mathcal{I}) \leq 0, \quad \mathcal{I} \in \mathcal{X}_{\theta_t}, \quad \theta_t \in \mathbb{R}_{\geq 0}. \end{aligned}$$

However, due to the non-linear property of Constraint (5), it is hard to solve $\mathcal{P}_{2,t}^E$ and obtain the optimum within polynomial time. Thus, the relaxation is needed for solving the proposed subproblem. That is, we further transform $\mathcal{P}_{2,t}^E$ into

$$\begin{aligned} [\mathscr{Q}_t^E] \quad \min_{\mathcal{W}} \quad & \mathcal{Q}_t^E = \mathcal{P}_{t-1}^E(\mathcal{W}) \\ s.t. \quad & h_t(\mathcal{W}) \leq 0, \quad \mathcal{W} \in \widetilde{\mathcal{X}}_{\theta_t}, \quad \theta_t \in \mathbb{R}_{\geq 0}, \end{aligned}$$

where $\mathcal{W}$ is the relaxed variable of $\mathcal{I}$. And the corresponding domain $\widetilde{\mathcal{X}}_{\theta_t}$ is also relaxed so that $\forall f \in \mathcal{F}_t$, $\forall W_f \in \mathcal{W}$, we have $W_f \geq 0$ and $W_f \geq \frac{move_{f,prec(f)} - \theta_t}{M}$, $W_f \leq 1$, where M is a constant to ensure for any θ_t and f, $\frac{move_{f,prec(f)} - \theta_t}{M} < 1$.

B. Online Control Algorithm

The proposed Algorithm 1 for the device contains two part: the first part initializes θ while the second part dynamically adjusts θ according to dynamic edge networks and the inference results incurred in previous epoch for edge-assisted inference.

The first part of Algorithm 1, i.e., lines 1 to 5, initializes $\widetilde{\theta}_1$ for the first epoch based on the first K frames, since the movement of all pixels between two consecutive frames in different videos may vary. After recording the first K frames at the beginning of the video, we use the movement of all pixels and use the average to initialize θ, as shown in line 5 of Algorithm 1, where the function $Average(\cdot)$ is used to calculate the average of all recorded movements. Intuitively, the initialization of θ based on the frames at the beginning of the video actually reflects the scenario in the video, instead of just initialing an arbitrary value of θ for all of the videos. As shown in the experiments later, the value of θ varies according to the scenarios recorded in different videos.

The second part of Algorithm 1, i.e., lines 6 to line 17, is executed in an online manner, and the operations needed for each epoch contain two components. The first component is object tracking for all of the frames in current epoch. For

Algorithm 1 On-device Online Object Detection (O^3)

```
Input: Fixed time duration $\sigma$ for each epoch;
       Parameter $K$ for initializing $\theta$.
 1: for Frame $f = 1, .., K$ do
 2:     Track objects in frame $f$, e.g., using Optical Flow.
 3:     Record the deviation between frames $move_{f,prec(f)}$.
 4: end for
 5: Initialize $\widetilde{\theta}_1 = Average(\{move_{f,prec(f)}\})$.

 6: for Epoch $t = 1, 2, ..., T$ do
 7:     for Frame $f \in \mathcal{F}_t$ do
 8:         Track objects in frame $f$, e.g., using Optical Flow.
 9:         if $move_{f,prec(f)} \geq \widetilde{\theta}_t$ then
10:             Trigger the transference of current frame to edge;
                Edge uses high accuracy model for inference.
11:             Correct object tracking by using edge inference.
12:         end if
13:     end for
14:     Obtain $\widetilde{\mathcal{W}}_{t+1}^*$ by solving subproblem $\mathscr{Q}_{t+1}^E$.
15:     $\Psi_{t+1} = \{$Frames in $\mathcal{F}_t$ with highest $move$ values$\}$,
        s.t. $|\Psi_{t+1}| = \lfloor \sum_f \widetilde{W}_{f,t+1}^* \rfloor$.
16:     $\widetilde{\theta}_{t+1} = move_{f_{min},prec(f_{min})}$,
        s.t. $f_{min} = arg\min_{f \in \Psi_{t+1}} \{move_{f,prec(f)}\}$.
17: end for
```

each frame, the algorithm calculates the movement of the pixels between current frame and the preceding frame by using object tracking technique, like Optical Flow as mentioned in our preliminary case study. Then, if the deviation exceeds $\widetilde{\theta}_t$, current frame is transferred to nearby edge over edge networks for further inference. And after receiving the inference results from nearby edge, the device corrects the object tracking by directly using the inference results received. The second component is the update for the threshold, which uses all of the inference results in current epoch t to construct the subproblem $\mathscr{Q}_{t+1}^E$, and solve $\widetilde{\mathcal{W}}_{t+1}^*$ for the following epoch. Actually, $\widetilde{\mathcal{W}}_{t+1}^*$ shows the preference of decision $\mathcal{I}_{t+1}$. In order to construct $\widetilde{\theta}_{t+1}$ for further $\mathcal{I}_{t+1}$, we use the top $\lfloor \sum_f \widetilde{W}_{f,t+1}^* \rfloor$ frames in $\mathcal{F}_t$ with highest $move$ values, and obtain $\widetilde{\theta}_{t+1}$ based on the minimum, as illustrated in lines 15 to 16 of Algorithm 1. Such construction also facilitates the analysis, and $\mathcal{I}_{t+1}$ actually does not violate the strong latency constraint.

Note that, all of the on-device object tracking, the frame transference over edge networks, the edge inference by using Neural Networks and the overhead of our proposed algorithm are considered within the given time period per epoch. And the linear program with respect to $\sigma|\mathcal{F}_t| + 1$ variables can be solved efficiently by using mature optimization tools.

IV. Performance Analysis

In this section, we measure the performance of our online algorithm with its optimum through the dynamic regret Reg.

Lemma 1 first links the related optimums among proposed subproblems; Theorem 1 then confirms there is no violation regarding the latency; Lemma 2 shows the sub-linear growth in terms of the cumulative difference between the results by using our proposed online algorithm and the optimum for the problem under observable inputs. Theorem 1 further concludes the algorithmic goal as mentioned before.

Except for the decisions of our proposed online algorithm $\widetilde{\boldsymbol{\theta}}$ and the intermediate outputs of $\mathscr{Q}_t^E$, i.e., $\widetilde{\mathcal{W}_t^*}, \forall t$, we also distinguish the results and optimums for other subproblems proposed to facilitate the theoretical analysis. We use $\mathcal{I}_{1,t}^*$ and $\mathcal{I}_{2,t}^*$ to denote the optimum decisions of problems $\mathscr{P}_{1,t}^E$ and $\mathscr{P}_{2,t}^E$, respectively, while we use $\mathcal{I}_t^*$ to denote the optimum decision for objective $\mathcal{P}_t^E$ in the domain of $\mathcal{I} \in \mathcal{X}_{\theta_t}$ and use $\mathcal{W}_{1,t}^*$ to denote the optimum decision for objective $\mathcal{P}_{1,t}^E$ in the domain of $\mathcal{W} \in \widetilde{\mathcal{X}}$. Note that $\widetilde{\mathcal{X}}$ is the domain further larger than $\widetilde{\mathcal{X}}_{\theta_t}$ without the restriction in terms of θ_t. At last, $\boldsymbol{\theta^*}$ and $\boldsymbol{\theta^{E*}}$ represent the optimums of problem $\mathscr{P}^G$ and $\mathscr{P}^E$, respectively, which are all aggregations of decisions.

Assumptions: Before further proceeding, we first introduce some assumptions, which are very common and widely used.

Assumption 1: $\forall t \in \mathcal{T}$, $P_t^E(\cdot)$ has bounded gradient, i.e., $||\nabla \mathcal{P}_t^E(\cdot)|| \le F$, where F is a fixed constant.

Assumption 2: $\forall t \in \mathcal{T}$, all of the domains $\mathcal{X}_{\theta_t}, \widetilde{\mathcal{X}}_{\theta_t}$ and $\mathcal{X}_{\theta_t}$ are bounded, i.e., for any elements a, b within the same domain, we have $||a - b|| \le X$.

Assumption 3: The dynamic changes of edge bandwidth are also bounded, i.e., $\forall t \in \mathcal{T}, b_t \in [b_{min}, b_{max}]$.

Assumption 4: The optimal edge inference is not far away from the ground truth, i.e., $\mathcal{P}^E(\boldsymbol{\theta}^{E*}) \le \mathcal{P}^G(\boldsymbol{\theta}^*) + \epsilon_0$.

Lemma 1. *The relationships between several proposed subproblems in terms of their optimums are illustrated as follows:*

$$\begin{aligned} \mathcal{Q}_t^E(\widetilde{\mathcal{W}_t^*}) = \mathcal{P}_{t-1}^E(\widetilde{\mathcal{W}_t^*}) \le \mathcal{P}_{t-1}^E(\mathcal{W}_{1,t}^*), \\ \mathcal{Q}_t^E(\widetilde{\mathcal{W}_t^*}) \le \mathcal{P}_{2,t}^E(\mathcal{I}_{2,t}^*) \le \mathcal{P}_{1,t}^E(\mathcal{I}_{1,t}^*) + \mathcal{P}_{t-1}^E(\widetilde{\mathcal{I}}_{t-1}). \end{aligned} \tag{7}$$

Proof. See Appendix A. □

Theorem 1. *The latency constraint using the results from our proposed online algorithm holds without any violation, i.e.,*

$$\forall t \in \mathcal{T} : h_t(\widetilde{\mathcal{I}}_t) \le 0, \tag{8}$$

where $\widetilde{\mathcal{I}}_t$ is the decision derived from $\widetilde{\theta}_t$.

Proof. See Appendix B. □

Lemma 2. *The difference between cumulative $\mathcal{P}_t^E(\widetilde{\mathcal{I}}_t)$ and cumulative $\mathcal{P}_t^E(\mathcal{I}_t^{E*})$ only grows sub-linearly with time:*

$$\sum\nolimits_t \{\mathcal{P}_t^E(\widetilde{\mathcal{I}}_t) - \mathcal{P}_t^E(\mathcal{I}_t^{E*})\} \le \mathcal{O}(T^\epsilon), \tag{9}$$

where $\mathcal{I}_t^{E*}$ is the decision derived from θ_t^{E*} and $\epsilon < 1$.

Proof. See Appendix C. □

Theorem 2. *By using O^3, the algorithmic goal, i.e., the dynamic regret only grows sub-linearly with time as follows:*

$$Reg = \mathcal{P}^E(\widetilde{\boldsymbol{\theta}}) - \mathcal{P}^G(\boldsymbol{\theta}^*) \le \mathcal{O}(T^\epsilon), \tag{10}$$

where ϵ is a constant and $\epsilon < 1$ as mentioned in Lemma 2.

Proof. See Appendix D. □

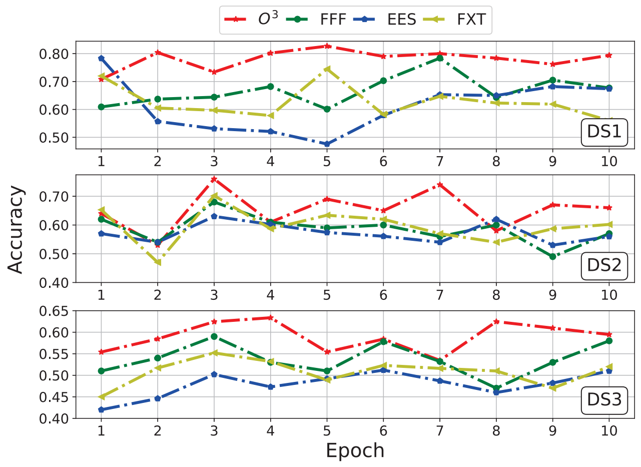


Fig. 4: Accuracy results under real-world video traces

V. Experiment

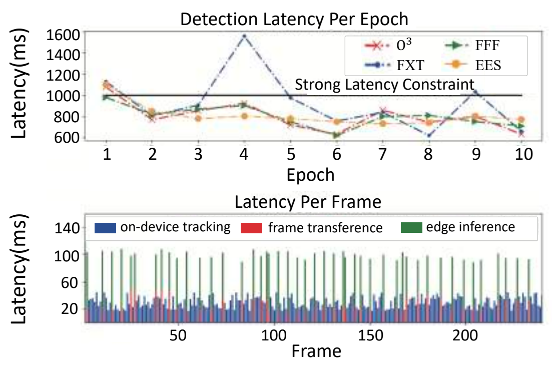
A. Data and Settings

Testbed for Edge-assisted On-device Tracking: We use the Jetson TX2 developer kit with Python environment as the device, where both of the on-device motion tracking module using OpenCV, i.e., Optical Flow, and the module for online decisions using O^3 are implemented and evaluted in the testbed. In order to solve $\mathcal{Q}_{t+1}^E$ at the end of each epoch t for the next time period, we use the tool of cvxopt from Python to obtain $\widetilde{\mathcal{W}_t^*}$ in Algorithm 1 through a linear program. The server, Dell PowerEdge R740 with a GPU GeForce RTX 2080 Ti, is used for edge inference in our testbed based on Detectron 2 Faster RCNN. All of the bounding boxes and related IoU from on-device or edge inference are cached for O^3 within each epoch, whose time duration is one second. The real video clips [30] with a resolution of 1080p and 30fps are taken as the input sources. The network bandwidth varies from 30Mbps to 90Mbps, derived from the real-world trace.

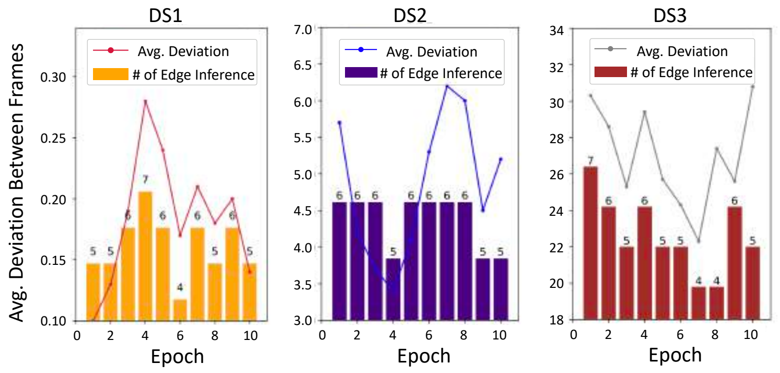
Algorithms and Metrics: Except for the online on-device object detection schema O^3, which adjusts θ for each epoch within the time duration of 1 second, we also compare it with multiple datasets and other algorithms listed as follows:

- **Fixed Threshold (FXT)** initializes θ by using a fixed number of frames at the beginning of the video, and such θ is fixed for all of the following epochs.
- **First Fixed Frames (FFF)** initializes θ_t by using a fixed number of frames at the beginning of each epoch t, and θ_t is then used during the left time of epoch t. Both FXT and FFF use at most 20% frames at the beginning for initialization, i.e., 4 to 6 frames within a second.
- **Every Epoch Sample (EES)** samples a fixed number of frames within an epoch by taking a fixed stepsize, i.e., uniformly sampling the frames. The default proportion of sampled frames is also 20% over with all of the frames, i.e., 6 frames for a 30fps video within a second.

We should mention here that all of the algorithms run online, and none of them could obtain the actual inference results before actual object tracking or edge inference.



(a) Object detection latency

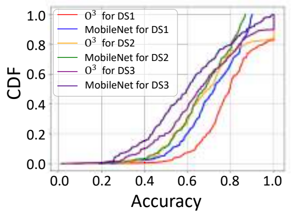


(b) Frames uploaded and the deviation

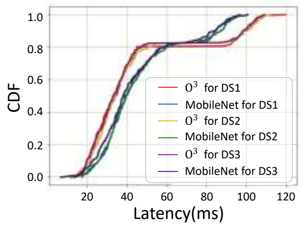
Fig. 5: Further results of our proposed online algorithm

TABLE III: Overall performance upon multiple datasets

Various Datasets	Overall Performance		
	Avg. Deviation	*O^3 Accuracy*	*MobileNet Accuracy*
Dataset 1	0.2	0.77	0.57
Dataset 2	5	0.67	0.56
Dataset 3	28	0.59	0.54



(a) Overall accuracy



(b) Overall latency

Fig. 6: Further CDF results over multiple datasets

B. Testbed Results

Accuracy and Latency: As illustrated in Fig. 4, we plot all of the accuracy results upon three datasets with 10 epochs, i.e., the duration is 10 seconds, compared with other algorithms. Except for the first epoch, O^3 always performs the best using the first dataset (DS1), improving 17.91% on the accuracy regarding the overlapped area of the bounding boxes compared with FFF, improving 25.9% compared with FXT, and improving 26.73% compared with EES. Note that the objects recorded in the video start their states of accelerated motions in epoch four, and O^3 captures such accelerated motions and quickly adjusts the threshold for edge-assisted inference. For the other two datasets, O^3 also outperforms the other algorithms, and it remains robust even when the inter-frame deviations fluctuate over time. Since FXT only adjusts the threshold once at the beginning of the video and EES only samples the frame for adjustment, they fail to capture the accelerated motions of objects. Although FFF adjusts the threshold for each epoch, the overhead for the initialization per epoch is unacceptable and such approach fails to capture the changes between epochs.

Fig. 5(a) depicts the latency for both per epoch and per frame. Except for the first epoch with initialization overhead, O^3 obeys the strong latency constraint. Although frequent edge-assisted inference actually improves the accuracy, it fails to obey the strong latency constraint. And for FXT, when the deviation between two frames is large, it would not adjust the threshold, resulting too many long edge inference. Epoch four contains the objects with accelerated motions, FXT fails to capture the changes with strong latency constraint. We also plot the latency of O^3 per frame in the bottom half of Fig. 5(a). The average latency of on-device tracking is 25ms, the average latency of edge inference is 83ms, and the average latency of frame transference is 28ms, which varies according to the dynamic bandwidth over edge network. The overall number of frames uploaded is 58 over 240 frames and the average latency of them including transference and inference is 111ms, which leads to a suitable frequency of edge-assisted inference.

Scalability of O^3: We evaluate O^3 over multiple datasets. As shown in Table III, the average deviation between two consecutive frames is 0.2, and the accuracy is 77%. Further shown in Fig. 5(b), the overall number of transferred frames is 55. Similarly, for dataset 2, the average deviation is 5 and the number of transferred frames is 57 while for dataset 3, the average deviation is 28 and the number of transferred frames is 53. We find that the number of transferred frames would not change much with the growth of the deviation. Thus, for various datasets, the overall latency can still be controlled within fixed time period for strong latency constraint. When the value of deviation changes dramatically, i.e., the objects move fast, O^3 dynamically adjusts the threshold, resulting in a large number of transferred frames for edge assistance. Since for real scenarios, the changes of deviations are bounded, the overall latency in terms of both inference and transference can still be controlled. And with the growth of deviation, although the results of object tracking need to be corrected by the edge inference frequently, the edge inference actually improves the detection accuracy by increasing a little on average latency.

As shown in Fig. 6, we compare the accuracy and latency of O^3 with the state-of-the-art on-device detection model MobileNet, since just edge inference fails to meet the strong latency constraint. The average improvements on the detection accuracy over these three datasets are 36%, 17% and 8%, respectively. When the values of the deviations are large, i.e., objects move dramatically in the scenario, O^3 dynamically adjusts the threshold to capture the changes. In general, O^3 always performs better than MobileNet regarding the accuracy under the constraint of latency, and O^3 only takes around 90ms to 120ms per call of edge-assisted inference, whose average latency is about 36ms per frame since only 5 or 6 frames are uploaded within a second to nearby edge on average.

In order to update threshold θ, a linear program with $\sigma|\mathcal{F}_t|+1$ variables and $3\sigma|\mathcal{F}_t|$ constraints is needed, which only takes 80ms. To obtain θ, we need to find the minimal one with the highest $move$ values, which only takes less than 1ms. Thus, the overall overhead is acceptable for an epoch, e.g., one second.

VI. Related Work

We summarize prior research in categories, and high-light their drawbacks compared to our work, respectively.

Edge-assisted On-device Inference: Some works studied edge-assisted mechanisms to enable on-device inference. Xu $et\ al.$ [14] used the optical flow estimation for calculating the motions between two consecutive frames. Qiu $et\ al.$ [17] developed augmented vehicular reality system to wirelessly share information with other nearby vehicles. Liu $et\ al.$ [6] proposed to offload the object detection inference to nearby edge based on a fixed deviation threshold. He $et\ al.$ [15] adopted the idea of RoI encoding to reduce the latency and the bandwidth consumption of the streaming process.

Instead of those light-weight models proposed by Howard $et\ al.$ [29] and Li $et\ al.$ [18], Hu $et\ al.$ [21] explored the partition of a complex neural network between devices and the edge to decrease the computation workload on devices. Jeong $et\ al.$ [20] enabled edges to receive DNN partitions in parallel. Stahl $et\ al.$ [22] further focused on the partition of a specific layer, fully connected layer, in neural networks to decrease the transference of intermediate data.

Unfortunately, those mechanisms regarding edge-assisted on-device inference fail to tackle the situation in an online manner where the frames uploaded should cater to the dynamic changes of both edge networks and the changes in the video.

Online Inference Configuration: Other works investigated the online inference configuration adaptively. Cheng $et\ al.$ [25] proposed a high-performance data preprocessing pipeline that selectively offloaded key workloads to FPGAs. Huang $et\ al.$ [26] implemented a deep reinforcement learning-based online offloading for wireless powered mobile-edge computing. Liu $et\ al.$ [24] adaptively adjusted video streaming for virtual reality system on mobile thin clients. Jiang $et\ al.$ [7] achieved higher accuracy video analytics by adapting the video configurations. Wang $et\ al.$ [36] considered joint configuration adaption and bandwidth allocation for video analytics.

Although those works have already considered the online scheduling for adapting inference, these mechanisms fail to meet the strong latency constraint of real-time video analytics and do not have any theoretical performance guarantee, considering limited device resources and dynamic edge networks.

VII. Conclusion

Edge-assisted on-device object detection needs to balance the tradeoff between the on-device object tracking and the video transference over edge networks under the constrained detection latency. In this paper, we formulate this problem by modeling a non-linear time-coupled program, maximizing the overall detection accuracy. Since inference results are only revealed after the decisions, we propose a learning based online algorithm to update the deviation threshold for edge-assisted inference, only taking observable inputs. We rigorously prove that the dynamic regret only grows sub-linearly as time goes. Extensive testbed results show the accuracy improvement compared with other algorithms under constrained latency.

Appendix

A. Proof of Lemma 1

The inequality $\mathcal{P}^E_{t-1}(\widetilde{\mathcal{W}}^*_t) \le \mathcal{P}^E_{t-1}(\mathcal{W}^*_{1,t})$ holds since $\widetilde{\mathcal{W}}^*_t$ is the optimum decision for problem $\mathscr{Q}^E_t$. Note that $\mathcal{W}^*_{1,t}$ is the optimum decision for problem $\mathscr{P}^E_{1,t}$.

Since $\mathcal{I}^*_{2,t}$ is the optimum decision for objective $\mathcal{P}^E_{t-1}$ in domain $\mathcal{X}_{\theta_t}$, which is more tight than the relaxed one $\widetilde{\mathcal{X}}_{\theta_t}$, $\mathcal{Q}^E_t(\widetilde{\mathcal{W}}^*_t) \le \mathcal{P}^E_{2,t}(\mathcal{I}^*_{2,t})$. Since $\mathcal{P}^E_{t-1}(\mathcal{I}) - \mathcal{P}^E_{t-1}(\mathcal{I}_{t-1})$ equals the first term in problem $\mathscr{P}^E_{1,t}$, and the second term in $\mathscr{P}^E_{1,t}$ is positive, we complete the proof of this lemma.

B. Proof of Theorem 1

Proof. Upon the relationship between $\widetilde{\theta}_{f,t}$ and $\widetilde{I}_{f,t}$, we have

$$\sum_{f\in\mathcal{F}_{t-1}} \widetilde{I}_{f,t} = \sum_{f\in\mathcal{F}_{t-1}} \mathbb{I}[move_{f,prec(f)} - \widetilde{\theta}_{f,t}] = \lfloor \sum_{f\in\mathcal{F}_{t-1}} \widetilde{W}^*_{f,t} \rfloor,$$

which implies $\mathbf{1}^\top\widetilde{\mathcal{W}}^*_t - 1 \le \mathbf{1}^\top\widetilde{\mathcal{I}}_t \le \mathbf{1}^\top\widetilde{\mathcal{W}}^*_t$.

For simplicity, let $\mathcal{A}_{f,t} = \sum_{b\in\mathcal{B}^E_{f,t}} \max_{b'\in\mathcal{B}^D_{f,t}} (1 - IoU^D_{b,b'})$. It is actually an observable constant ranging from 0 and $|\mathcal{B}^E_{f,t}|$. Moreover, we define $\mathcal{C}_{f,t} = |\mathcal{B}^E_{f,t}| - \mathcal{A}_{f,t} \ge 0$ since it is the aggregation of all $IoU^E_{b,b'} - IoU^D_{b,b'}$ in a frame.

Then, we can rewrite the objective $\mathcal{P}^E_t(\widetilde{\mathcal{W}}^*_t)$ as follows:

$$S - \sum_{f\in\mathcal{F}_t} \{\mathcal{C}_{f,t}\widetilde{W}^*_{f,t} + |\mathcal{B}^E_{f,t}|\mathcal{A}_{f,t}\}.$$

For the term $\sum_{f\in\mathcal{F}_t} \mathcal{C}_{f,t}\widetilde{W}^*_{f,t}$, using Assumption 3, we have

$$\sum_{f\in\mathcal{F}_t} \mathcal{C}_{f,t}\widetilde{W}^*_{f,t} \le \max_{f,t}\{\mathcal{C}_{f,t}\}\mathbf{1}^\top\widetilde{\mathcal{W}}^*_t \le \max_{f,t}\{\mathcal{C}_{f,t}\}(\mathbf{1}^\top\widetilde{\mathcal{I}}_t + 1)$$
$$\le \max_{f,t}\{\mathcal{C}_{f,t}\}(\sum_{f\in\mathcal{F}_t} \frac{\mathcal{C}_{f,t}}{\mathcal{C}_{f,t}}\widetilde{I}_{f,t} + 1) \le \beta_0 \sum_{f\in\mathcal{F}_t} \mathcal{C}_{f,t}\widetilde{I}_{f,t} + \beta_1,$$

where $\beta_0 = \frac{\max_{f,t}\{\mathcal{C}_{f,t}\}}{\min_{f,t}\{\mathcal{C}_{f,t}\}}, \beta_1 = \max_{f,t}\{\mathcal{C}_{f,t}\}$. Then, for $\mathcal{P}^E_t(\widetilde{\mathcal{W}}^*_t)$,

$$\mathcal{P}^E_t(\widetilde{\mathcal{W}}^*_t) = S - \sum_{f\in\mathcal{F}_t} \{\mathcal{C}_{f,t}\widetilde{W}^*_{f,t} + |\mathcal{B}^E_{f,t}|\mathcal{A}_{f,t}\}$$
$$\ge S - \beta_0 \sum_{f\in\mathcal{F}_t} \mathcal{C}_{f,t}\widetilde{W}^*_{f,t} - \sum_{f\in\mathcal{F}_t} |\mathcal{B}^E_{f,t}|\mathcal{A}_{f,t} - \beta_1 \ge \frac{1}{\beta}\mathcal{P}^E_t(\widetilde{\mathcal{I}}_t),$$

where $1/\beta \le 1 - \{(\beta_0 - 1)\sum_{f\in\mathcal{F}_t} |\mathcal{B}^E_{f,t}|\mathcal{A}_{f,t} + \beta_1\}/\mathcal{P}^E_t(\widetilde{\mathcal{W}}^*_t)$. Since S is a large number to ensure the denominator is larger than the numerator and β_0 is larger than 1, we conclude that

$\beta \geq 1$. Furthermore, we illustrate that the constraint $h_t(\widetilde{\mathcal{I}}_t)$ holds without any violation, i.e., the overall latency $h_t(\widetilde{\mathcal{I}}_t)$ is

$$\sum\nolimits_{f\in\mathcal{F}_t}\{(e_t^E - e_t^D)\widetilde{I}_{f,t} + e_t^D\} = (e_t^E - e_t^D)\mathbf{1}^\top\widetilde{\mathcal{I}}_t + |\mathcal{F}_t|e_t^D$$
$$\leq \sum\nolimits_{f\in\mathcal{F}_t}\{(e_t^E - e_t^D)\widetilde{W}^*_{f,t} + e_t^D\} \leq h_t(\widetilde{\mathcal{W}}_t^*) \leq \sigma. \qquad \square$$

C. Proof of Lemma 2

Proof. Since $\mathcal{P}_{1,t}^E(\mathcal{W})$ is a $1/\alpha$-strongly convex function with respect to $\mathcal{W}$ according to the definition and $\mathcal{W}_{1,t}^*$ is its optimum decision in the domain of $\widetilde{\mathcal{X}}$. Note that, the domain of $\mathcal{W}$ is a convex set instead of $\mathcal{X}_{\theta_t}$. And $\forall a, b$ we have

$$\mathcal{P}_{1,t}^E(b) \geq \mathcal{P}_{1,t}^E(a) + \nabla\mathcal{P}_{1,t}^E(a)^\top(b-a) + ||b-a||^2/2\alpha,$$
$$\nabla\mathcal{P}_{1,t}^E(\mathcal{W}_{1,t}^*)^\top(b - \mathcal{W}_{1,t}^*) \geq 0.$$

By setting $a = \mathcal{W}_{1,t}^*, b = \mathcal{I}_{t-1}^{E^*}$, we have

$$\mathcal{P}_{1,t}^E(\mathcal{I}_{t-1}^{E^*}) \geq \mathcal{P}_{1,t}^E(\mathcal{W}_{1,t}^*) + ||\mathcal{I}_{t-1}^{E^*} - \mathcal{W}_{1,t}^*||^2/2\alpha. \qquad (11)$$

After that, using the definition of $\mathcal{P}_{1,t}^E$, we have $\mathcal{P}_{1,t}^E(\mathcal{W}_{1,t}^*)$

$$= \nabla\mathcal{P}_{t-1}^E(\widetilde{\mathcal{I}}_{t-1})^\top(\mathcal{W}_{1,t}^* - \widetilde{\mathcal{I}}_{t-1}) + ||\mathcal{W}_{1,t}^* - \widetilde{\mathcal{I}}_{t-1}||^2/2\alpha$$
$$= \mathcal{P}_{t-1}^E(\mathcal{W}_{1,t}^*) - \mathcal{P}_{t-1}^E(\widetilde{\mathcal{I}}_{t-1}) + ||\mathcal{W}_{1,t}^* - \widetilde{\mathcal{I}}_{t-1}||^2/2\alpha$$
$$\geq \mathcal{P}_{t-1}^E(\widetilde{\mathcal{W}}_t^*) - \mathcal{P}_{t-1}^E(\widetilde{\mathcal{I}}_{t-1}) + ||\mathcal{W}_{1,t}^* - \widetilde{\mathcal{I}}_{t-1}||^2/2\alpha$$
$$= \nabla\mathcal{P}_{t-1}^E(\widetilde{\mathcal{I}}_{t-1})^\top(\widetilde{\mathcal{W}}_t^* - \widetilde{\mathcal{I}}_{t-1}) + ||\mathcal{W}_{1,t}^* - \widetilde{\mathcal{I}}_{t-1}||^2/2\alpha$$
$$= \mathcal{P}_{1,t}^E(\widetilde{\mathcal{W}}_t^*) - ||\widetilde{\mathcal{W}}_t^* - \widetilde{\mathcal{I}}_{t-1}||^2/2\alpha + ||\mathcal{W}_{1,t}^* - \widetilde{\mathcal{I}}_{t-1}||^2/2\alpha,$$

where the inequality sign holds due to Lemma 1. We should mention here that we omit the residual terms respect to $\widetilde{\mathcal{I}}_{t-1}$ for the second and fourth equation signs since these terms reflect the inherent dynamics of the system and are unchanged.

Combing previous inequality with Inequality (11), we have

$$\mathcal{P}_{1,t}^E(\mathcal{I}_{t-1}^{E^*}) \geq \mathcal{P}_{1,t}^E(\widetilde{\mathcal{W}}_t^*) + \Delta_{t-1},$$

where the parameter Δ_{t-1} in previous inequality is defined as

$$\{||\mathcal{I}_{t-1}^{E^*} - \mathcal{W}_{1,t}^*||^2 + ||\mathcal{W}_{1,t}^* - \widetilde{\mathcal{I}}_{t-1}||^2 - ||\widetilde{\mathcal{W}}_t^* - \widetilde{\mathcal{I}}_{t-1}||^2\}/2\alpha.$$

Adding the term $\mathcal{P}_{t-1}^E(\widetilde{\mathcal{I}}_{t-1})$ to both sides, we have

$$\mathcal{P}_{1,t}^E(\mathcal{I}_{t-1}^{E^*}) + \mathcal{P}_{t-1}^E(\widetilde{\mathcal{I}}_{t-1}) \geq \mathcal{P}_{1,t}^E(\widetilde{\mathcal{W}}_t^*) + \mathcal{P}_{t-1}^E(\widetilde{\mathcal{I}}_{t-1}) + \Delta_{t-1}.$$

We consider the term $\mathcal{P}_{t-1}^E(\mathcal{I}_{t-1}^{E^*})$. Expanding it, we have

$$\mathcal{P}_{t-1}^E(\mathcal{I}_{t-1}^{E^*}) = \mathcal{P}_{t-1}^E(\widetilde{\mathcal{I}}_{t-1}) + \nabla\mathcal{P}_{t-1}^E(\widetilde{\mathcal{I}}_{t-1})^\top(\mathcal{I}_{t-1}^{E^*} - \widetilde{\mathcal{I}}_{t-1})$$
$$= \mathcal{P}_{t-1}^E(\widetilde{\mathcal{I}}_{t-1}) + \mathcal{P}_{1,t}^E(\mathcal{I}_{t-1}^{E^*}) - ||\mathcal{I}_{t-1}^{E^*} - \widetilde{\mathcal{I}}_{t-1}||^2/2\alpha$$
$$\geq \mathcal{P}_{1,t}^E(\widetilde{\mathcal{W}}_t^*) + \mathcal{P}_{t-1}^E(\widetilde{\mathcal{I}}_{t-1}) + \Delta_{t-1} - ||\mathcal{I}_{t-1}^{E^*} - \widetilde{\mathcal{I}}_{t-1}||^2/2\alpha.$$

We then consider three terms in $\Delta_{t-1} - \frac{||\mathcal{I}_{t-1}^{E^*} - \widetilde{\mathcal{I}}_{t-1}||^2}{2\alpha}$, i.e., for $||\mathcal{I}_{t-1}^{E^*} - \mathcal{W}_{1,t}^*||^2 + ||\mathcal{W}_{1,t}^* - \widetilde{\mathcal{I}}_{t-1}||^2 - ||\mathcal{I}_{t-1}^{E^*} - \widetilde{\mathcal{I}}_{t-1}||^2$, it is

$$= (\widetilde{\mathcal{I}}_{t-1} - \mathcal{W}_{1,t}^*)(2\mathcal{I}_{t-1}^{E^*} - \mathcal{W}_{1,t}^* - \widetilde{\mathcal{I}}_{t-1}) + ||\widetilde{\mathcal{I}}_{t-1} - \mathcal{W}_{1,t}^*||^2$$
$$= 2(\widetilde{\mathcal{I}}_{t-1} - \mathcal{W}_{1,t}^*)^\top(\mathcal{I}_{t-1}^{E^*} - \mathcal{W}_{1,t}^*),$$

where the square difference formula is used.

Then, we try to arrange the terms in previous inequality except for $\mathcal{P}_{t-1}^E(\mathcal{I}_{t-1}^{E^*})$ and $\mathcal{P}_{t-1}^E(\widetilde{\mathcal{I}}_{t-1})$. We have

$$-\nabla\mathcal{P}_{t-1}^E(\widetilde{\mathcal{I}}_{t-1})(\widetilde{\mathcal{W}}_t^* - \widetilde{\mathcal{I}}_{t-1}) - ||\mathcal{W}_{1,t}^* - \widetilde{\mathcal{I}}_{t-1}||^2/2\alpha$$
$$\leq ||\nabla\mathcal{P}_{t-1}^E(\widetilde{\mathcal{I}}_{t-1})||\, ||\widetilde{\mathcal{W}}_t^* - \widetilde{\mathcal{I}}_{t-1}|| - ||\mathcal{W}_{1,t}^* - \widetilde{\mathcal{I}}_{t-1}||^2/2\alpha$$
$$\leq \frac{F^2}{2\eta_t} + \frac{\eta_t}{2}||\widetilde{\mathcal{W}}_t^* - \widetilde{\mathcal{I}}_{t-1}||^2 - ||\mathcal{W}_{1,t}^* - \widetilde{\mathcal{I}}_{t-1}||^2/2\alpha,$$

where F is used for bounded gradient in Assumption 1.

Setting $\eta_t = \kappa_t/\alpha$, the sum of the last two terms is 0. For term $\mathcal{P}_{t-1}^E(\widetilde{\mathcal{I}}_{t-1}) - \mathcal{P}_{t-1}^E(\mathcal{I}_{t-1}^{E^*})$, the value is smaller than

$$\frac{\alpha F^2}{2\kappa_t} + \frac{||\mathcal{W}_{1,t}^* - \widetilde{\mathcal{I}}_{t-1}||^2 - 2(\widetilde{\mathcal{I}}_{t-1} - \mathcal{W}_{1,t}^*)^\top(\mathcal{I}_{t-1}^{E^*} - \mathcal{W}_{1,t}^*)}{2\alpha}.$$

Adding two complementary terms on the right, it becomes

$$\alpha F^2/2\kappa_t + \{||\widetilde{\mathcal{I}}_{t-1} - \mathcal{I}_{t-1}^{E*}||^2 - ||\mathcal{W}_{1,t}^* - \mathcal{I}_{t-1}^{E*}||^2\}/2\alpha.$$

Then, for the term $||\widetilde{\mathcal{I}}_{t-1} - \mathcal{I}_{t-1}^{E*}||^2$, we conclude that it is

$$\leq 2X(||\mathcal{I}_t^{E*} - \mathcal{I}_{t-1}^{E*}|| + ||\mathcal{I}_t^{E*}||),$$

where X is used for bounded domain in Assumption 2, and the first inequality sign holds due to the Triangle Inequality.

Thus, for the term $\mathcal{P}_{t-1}^E(\widetilde{\mathcal{I}}_{t-1}) - \mathcal{P}_{t-1}^E(\mathcal{I}_{t-1}^{E^*})$, it is less than

$$\alpha F^2/2\kappa_t + 2X(||\mathcal{I}_t^{E*} - \mathcal{I}_{t-1}^{E*}|| + ||\mathcal{I}_t^{E*}||)/2\alpha.$$

Summing up previous inequality over T, and define the cumulative difference in terms of the optimums, i.e.,

$$V(\{\mathcal{I}_t^{E*}\}) = \sum\nolimits_t\{||\mathcal{I}_t^{E*} - \mathcal{I}_{t-1}^{E*}|| + ||\mathcal{I}_t^{E*}||\},$$

We have the following inequality over T epochs:

$$\sum\nolimits_t\{\mathcal{P}_t^E(\widetilde{\mathcal{I}}_t) - \mathcal{P}_t^E(\mathcal{I}_t^{E^*})\} \leq \alpha F^2 T/2\kappa_{min} + V(\{\mathcal{I}_t^{E*}\})X/\alpha,$$

where $\kappa_{min} = \min_t\{\kappa_t\}$ is a constant.

Properly choosing the step size $\alpha = \sqrt{\frac{V(\{\mathcal{I}_t^{E*}\})}{T}}$, we have

$$\sum\nolimits_t\{\mathcal{P}_t^E(\widetilde{\mathcal{I}}_t) - \mathcal{P}_t^E(\mathcal{I}_t^{E^*})\} \leq \mathcal{O}\{\sqrt{V(\{\mathcal{I}_t^{E*}\})T}\},$$

which sub-linearly grows with respect to T. We should mention here that $V(\{\mathcal{I}_t^{E*}\})$ is the cumulative difference of the optimums, and it is a constant for a given scenario. $\square$

D. Proof of Theorem 2

Proof. Since $\widetilde{\mathcal{I}}_t$ is the decision directly derived from $\widetilde{\theta}_t$ and $\mathcal{I}_t^{E*}$ is the decision derived from θ_t^{E*}, the inequality regarding the cumulative difference in Lemma 2 can be rewritten as

$$\sum\nolimits_t\{\mathcal{P}_t^E(\widetilde{\theta}_t) - \mathcal{P}_t^E(\theta_t^{E*})\} \leq \mathcal{O}\{\sqrt{V(\{\theta_t^{E*}\})T}\},$$

where $V(\{\theta_t^{E*}\})$ also equals $V(\{\mathcal{I}_t^{E*}\})$ with exactly the same meaning. After that, we consider the aggregation for both $\{\widetilde{\theta}_t\}$ and $\{\theta_t^{E*}\}$, where the aggregation refers to the column vector over all of the epochs. Then, we have

$$\mathcal{P}^E(\widetilde{\boldsymbol{\theta}}) - \mathcal{P}^E(\boldsymbol{\theta}^{E*}) \leq \mathcal{O}\{\sqrt{V(\boldsymbol{\theta}^{E*})T}\}.$$

At last, by using Assumption 4 in terms of the accuracy difference between $\mathcal{P}^E$ and $\mathcal{P}^G$, we complete the proof, and conclude the following inequality:

$$\mathcal{P}^E(\widetilde{\boldsymbol{\theta}}) - \mathcal{P}^G(\boldsymbol{\theta}^*) \leq \mathcal{O}\{\sqrt{V(\boldsymbol{\theta}^{E*})T}\} + \epsilon_0 \leq \mathcal{O}\{T^\epsilon\}. \qquad \square$$

References

[1] "Mobile Device Worldwide," https://www.statista.com/statistics/245501/multiple-mobile-device-ownership-worldwide/, 2020.

[2] T. Tuan-Kiet, H.-T. HUYNH, D.-P. NGUYEN, L. Dinh-Dung, T. Thi-Hong, and Y. NAKASHIMA, "Demonstration of a visible light receiver using rolling-shutter smartphone camera," in *IEEE ATC*, 2018, pp. 214–219.

[3] L. Dong, M. N. Satpute, J. Shan, B. Liu, Y. Yu, and T. Yan, "Computation offloading for mobile-edge computing with multi-user," in *IEEE ICDCS*, 2019, pp. 841–850.

[4] 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 *USENIX NSDI*, 2017, pp. 377–392.

[5] G. Ananthanarayanan, V. Bahl, L. Cox, A. Crown, S. Nogbahi, and Y. Shu, "Video analytics-killer app for edge computing," in *ACM MobiSys*, 2019, pp. 695–696.

[6] L. Liu, H. Li, and M. Gruteser, "Edge assisted real-time object detection for mobile augmented reality," in *ACM MobiCom*, 2019, pp. 1–16.

[7] J. Jiang, G. Ananthanarayanan, P. Bodik, S. Sen, and I. Stoica, "Chameleon: scalable adaptation of video analytics," in *ACM SIGCOMM*, 2018, pp. 253–266.

[8] J. Tobin, W. Zaremba, and P. Abbeel, "Geometry-aware neural rendering," in *NIPS*, 2019, pp. 11 559–11 569.

[9] X. Ran, H. Chen, X. Zhu, Z. Liu, and J. Chen, "Deepdecision: A mobile deep learning framework for edge video analytics," in *IEEE INFOCOM*, 2018, pp. 1421–1429.

[10] "Detectron2 Model," https://github.com/facebookresearch/detectron2/blob/master/MODEL_ZOO.md/, 2020.

[11] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *IEEE CVPR*, 2016, pp. 779–788.

[12] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *NIPS*, 2015, pp. 91–99.

[13] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *IEEE ICCV*, 2017, pp. 2961–2969.

[14] L. Xu, J. Jia, and Y. Matsushita, "Motion detail preserving optical flow estimation," in *IEEE PAMI*, vol. 34, no. 9, 2011, pp. 1744–1757.

[15] J. He, M. A. Qureshi, L. Qiu, J. Li, F. Li, and L. Han, "Rubiks: Practical 360-degree streaming for smartphones," in *ACM MobiSys*, 2018, pp. 482–494.

[16] Y. Li, J. Li, W. Lin, and J. Li, "Tiny-dsod: Lightweight object detection for resource-restricted usages," in *arXiv:1807.11013*, 2018.

[17] H. Qiu, F. Ahmad, F. Bai, M. Gruteser, and R. Govindan, "Avr: Augmented vehicular reality," in *ACM MobiSys*, 2018, pp. 81–95.

[18] Y. Li and F. Ren, "Light-weight retinanet for object detection," in *arXiv:1905.10011*, 2019.

[19] A. Wong, M. Famuori, M. J. Shafiee, F. Li, B. Chwyl, and J. Chung, "Yolo nano: A highly compact you only look once convolutional neural network for object detection," in *arXiv:1910.01271*, 2019.

[20] H.-J. Jeong, H.-J. Lee, C. H. Shin, and S.-M. Moon, "Ionn: Incremental offloading of neural network computations from mobile devices to edge servers," in *ACM SoCC*, 2018, pp. 401–411.

[21] C. Hu, W. Bao, D. Wang, and F. Liu, "Dynamic adaptive dnn surgery for inference acceleration on the edge," in *IEEE INFOCOM*, 2019, pp. 1423–1431.

[22] R. Stahl, Z. Zhao, D. Mueller-Gritschneder, A. Gerstlauer, and U. Schlichtmann, "Fully distributed deep learning inference on resource-constrained edge devices," in *Springer SAMOS*, 2019, pp. 77–90.

[23] W. He, S. Guo, S. Guo, X. Qiu, and F. Qi, "Joint dnn partition deployment and resource allocation for delay-sensitive deep learning inference in iot," in *IEEE IoTJ*, 2020.

[24] L. Liu, R. Zhong, W. Zhang, Y. Liu, J. Zhang, L. Zhang, and M. Gruteser, "Cutting the cord: Designing a high-quality untethered vr system with low latency remote rendering," in *ACM MobiSys*, 2018, pp. 68–80.

[25] Y. Cheng, D. Li, Z. Guo, B. Jiang, J. Lin, X. Fan, J. Geng, X. Yu, W. Bai, L. Qu *et al.*, "Dlbooster: Boosting end-to-end deep learning workflows with offloading data preprocessing pipelines," in *ACM ICPP*, 2019, pp. 1–11.

[26] L. Huang, S. Bi, and Y. J. Zhang, "Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks," in *IEEE TMC*, 2019.

[27] N. Chen, S. Quan, S. Zhang, Z. Qian, Y. Jin, J. Wu, W. Li, and S. Lu, "Cuttlefish: Neural configuration adaptation for video analysis in live augmented reality," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 4, pp. 830–841, 2021.

[28] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *IEEE CVPR*, 2017, pp. 2462–2470.

[29] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," in *arXiv:1704.04861*, 2017.

[30] "Video Test Media," https://media.xiph.org/video/derf/, 2020.

[31] Y. Sun, H. Zhu, F. Zhuang, J. Gu, and Q. He, "Exploring the urban region-of-interest through the analysis of online map search queries," in *ACM SIGKDD*, 2018, pp. 2269–2278.

[32] Z. Chai, S. Li, Q. He, M. Chen, and W. Chen, "Fpga-based ROI encoding for HEVC video bitrate reduction," *J. Circuits Syst. Comput.*, vol. 29, no. 11, pp. 2 050 182:1–2 050 182:17, 2020.

[33] H. Liu, R. A. R. Soto, F. Xiao, and Y. J. Lee, "Yolactedge: Real-time instance segmentation on the edge," 2020.

[34] A. Mousavian, D. Anguelov, J. Flynn, and J. Kosecka, "3d bounding box estimation using deep learning and geometry," in *IEEE CVPR*, 2017, pp. 7074–7082.

[35] L. Zhang, T. Yang, J. Yi, R. Jin, and Z.-H. Zhou, "Improved dynamic regret for non-degenerate functions," in *NIPS*, 2017, pp. 732–741.

[36] C. Wang, S. Zhang, Y. Chen, Z. Qian, J. Wu, and M. Xiao, "Joint configuration adaptation and bandwidth allocation for edge-based real-time video analytics," in *IEEE INFOCOM*, 2020, pp. 1–10.