# Online Optimization of Offloading Video Analytics Tasks to Multiple Edges for Accuracy Maximization

Yu Liang[†*], Sheng Zhang[‡], Jie Wu[§]
[†]School of Computer and Electronic Information and the School of Artificial Intelligence, Nanjing Normal University
[‡]State Key Laboratory for Novel Software Technology, Nanjing University
[§]Center for Networked Computing, Temple University, US
[*]Corresponding author: liangyu@njnu.edu.cn

***Abstract*—Real-time video analytics (VA) presents challenges due to its computational intensity and latency sensitivity, especially when processed on mobile devices with limited local resources. We propose to offload VA tasks to edge servers with diverse computational capabilities. We present a "detect + track" approach with on-device object tracking and edge-assisted object detection. We formulate a long-term nonlinear integer programming to maximize the overall accuracy within detection frequency and latency constraints. We then design a queue-based online optimization algorithm to solve it: relax the original problem from the integer domain to the real domain, then employ a queue-based adaptation and randomized rounding strategy. Via rigorous proof, both dynamic regret regarding detection accuracy and the real-time requirement are ensured. Evaluation results also demonstrate the effectiveness of our approach.**

***Index Terms*—Video analytics, resource allocation, online optimization**

## I. Introduction

In recent years, with the rapid development of technologies such as mobile augmented reality, the Internet of Things, and vehicular networks, the demand for applications like virtual reality devices and traffic monitoring systems has steadily increased [1, 2, 25]. Due to limited local computing resource [3], these terminal devices can only perform limited analyses. Offloading a large number of video analytics tasks to multiple remote servers introduces significant delays. In many typical application scenarios, there is an urgent requirement for high-accuracy, low-latency analytics results, imposing higher demands on rapid and precise analytics of video streams. Designing the video analytics task offloading algorithms facing multiple challenges as follows:

Firstly, frame-by-frame video analytics is a computationally intensive task that consumes significant computing resources [4, 5] and can lead to notable delays [6, 24]. For instance, neural networks like YOLO [7], Faster-RCNN [8], and Mask-RCNN [9] often require hundreds of milliseconds for object detection per frame, and higher-accuracy CNN models may need even longer detection time. For high-frame-rate videos, this processing approach can cause long detection delays, making it difficult to meet the real-time requirement of video analytics tasks. Moreover, offloading video analytics tasks to edge servers introduces additional video transmission delay, further increasing the total processing time. Therefore, reducing the processing time of video analytics tasks to meet real-time requirements has become a research focus.

Secondly, the computing resources on devices are typically limited, unable to support high-precision neural networks, and can only perform lightweight computational inference, failing to provide high-precision analytics results. Moreover, offloading video analytics tasks to edge servers results in longer delays. Thus, how to maximize video analytics accuracy while ensuring real-time performance is another pressing issue.

Thirdly, in real-time video stream analytics applications, video frames usually arrive online and need to be processed immediately upon arrival. This requires preemptive offloading decisions for video frames, i.e., determining whether and which server to offload the incoming video frame in advance. Therefore, making preemptive offloading decisions based on existing video analytics results for upcoming video frames presents another research challenge.

Prior research can be classified into two broad types: edge-assisted video analytics task offloading [5, 16, 18, 22, 23, 26, 27, 29] and frame-filtering object detection [19–21]. However, none of them fully solved the aforementioned three challenges.

In this paper, we adopt the "detect + track" processing strategy, which involves offloading some frames to edge servers for object detection, and performing tracking for the other frames on the device, thus achieving high-accuracy video analytics while meeting real-time requirements. We propose a queue-based online optimization algorithm OLA for offloading real-time video analytics tasks in a multi-edge environment, aiming to maximize analytics accuracy while minimizing overall latency. Via rigorous proof, both dynamic regret regarding detection accuracy and the real-time requirement are ensured. Evaluations also demonstrate the effectiveness of our approach.

## II. System Model

Fig. 1 illustrates the considered scenario, which contains a mobile device and multiple edge servers, denoted by $\mathcal{S} = \{1, ..., s, ...S\}$. Each edge server is equipped with different CNN models primarily used for VA tasks (in this paper, we mainly consider a typical VA task, object detection). The mobile device, with limited resources, performs object tracking using optical flow [10]. We consider a set of epochs $\mathcal{T} = \{1, .., t, .., T\}$, where the time duration for each epoch is fixed, such as one second. $\mathcal{F}_t$ represents the set of video frames within epoch $t$. For example, if the time span of each epoch is one second and the frame rate is 30 fps, then

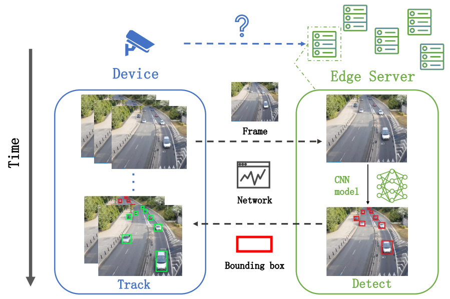

Fig. 1. Considered scenario and system architecture

the number of video frames contained within each epoch is $|\mathcal{F}_t| = 30, \forall t \in \mathcal{T}$. At the beginning of each epoch, we must determine which video frames are offloaded and to which servers they should be offloaded. If a frame is offloaded to a server for inference using a CNN model, bounding boxes for detected objects are generated; otherwise, the frame updates the bounding boxes locally via object tracking.

**Decision Variables.** We introduce a binary decision $x_{f,s} \in \{0,1\}$, $\forall f \in \mathcal{F}_t$, $\forall s \in \mathcal{S}$, to indicate whether frame $f$ should be offloaded to edge server $s$ for object detection. Furthermore, we use $\boldsymbol{x}_t$ to represent the set of decisions for epoch $t$.

**Accuracy Model.** Objects positions are typically represented by bounding boxes defined by two-point coordinates. For frame $f$, the bounding boxes tracked by the mobile device are denoted as $\mathcal{B}_f^D$, those detected on an edge server $s$ as $\mathcal{B}_f^s$, and the ground-truth bounding boxes as $\mathcal{B}_f^G$. The detection accuracy is measured by the overlapped area between a bounding box from $\mathcal{B}_f^D$ or $\mathcal{B}_f^s$ and a bounding box from $\mathcal{B}_f^G$. Hence, $\forall b \in \mathcal{B}_f^D \cup \mathcal{B}_f^s$, $\forall b^* \in \mathcal{B}_f^G$, the accuracy of $b$ respect to $b^*$, denoted by $IoU_{b,b^*}$, is defined as

$$IoU_{b,b^*} = x_{f,s} \cdot IoU_{b,b^*}^s + (1 - x_{f,s}) \cdot IoU_{b,b^*}^D, \tag{1}$$

where $IoU_{b,b^*}^s$ denotes the detection accuracy on server $s$, and $IoU_{b,b^*}^D$ denotes the tracking accuracy on the device. Hence, for $\forall f \in \mathcal{F}_t$, $\forall s \in \mathcal{S}$, the overall accuracy of all objects in the ground truth from frame $f$ on server $s$ is measured by

$$Acc_{f,s} = \sum_{b^* \in \mathcal{B}_f^G} \max_{b \in \mathcal{B}_f^I} \{x_{f,s} \cdot IoU_{b,b^*}^s + (1 - x_{f,s}) \cdot IoU_{b,b^*}^D\}, \tag{2}$$

where the maximum is due to the possible outputs of multiple bounding boxes for the same object.

**Latency Model.** Latency is a critical performance metric, directly affecting whether the system can meet the real-time requirements. We use $L^D$ to denote the local processing latency, i.e., object tracking on the device. Data transmission latency is determined by the data size of a frame and the current network bandwidth conditions. Data processing latency refers to the time it takes for an edge device to perform object detection, primarily depending on the hardware capacity of edge server and the complexity of the deployed CNN model. Therefore, the overall latency for frame $f$ is defined as

$$L_{f,s} = x_{f,s} \cdot \left(\frac{d}{b_{s,t}} + e_s\right) + (1 - x_{f,s}) \cdot L^D, \tag{3}$$

in which $d$ denotes the data size of a frame, $b_{s,t}$ is the bandwidth between edge server $s$ and the device in epoch $t$, and $e_s$ denotes the processing delay for server $s$. Thus, $(\frac{d}{b_{s,t}} + e_s)$ means the service latency for frame $f$ on server $s$, which is also denoted by $L_s^S$.

**Overall Formulation.** We aim to maximize the overall accuracy under a latency budget. The overall problem $\mathbb{P}^G$ with respect to the ground truth is formulated as follows:

$$\begin{aligned}
&\min \sum_{f,t,s} \Big(1 - \sum_{b^* \in \mathcal{B}_f^G} \max_{b \in \mathcal{B}_f^I} \{x_{f,s} \cdot IoU_{b,b^*}^s + (1 - x_{f,s}) \cdot IoU_{b,b^*}^D\}\Big) \\
&s.t. \quad C_1 : \sum_{t \in \mathcal{T}} \sum_{f,s} x_{f,s} \ge \sum_{t \in \mathcal{T}} \sum_{f} \tau, \\
&\qquad\;\; C_2 : \sum_{f,s} \{x_{f,s} L_s^S + (1 - x_{f,s}) L^D\} \le L_{max}, \forall t \in \mathcal{T}, \\
&\qquad\;\; C_3 : \sum_{s} x_{f,s} \le 1, \forall f \in \mathcal{F}_t, \\
&var. \quad C_4 : x_{f,s} \in \{0,1\}, t \in \mathcal{T}, f \in \mathcal{F}_t, s \in \mathcal{S}.
\end{aligned} \tag{4}$$

Constraint $C_1$ is the minimum detection frequency constraint in the long run, which ensures that at least $\tau$ frames are offloaded to edge servers for detection in each epoch. Constraint $C_2$ restricts the maximum latency to real time limit $L_{max}$ per epoch. Constraint $C_3$ restricts that only one edge server can be selected per online decision. $C_4$ restricts the domain of our decision variables.

Since the ground truth cannot be obtained before the analytics is completed, the existing detection results need to be utilized to replace the ground truth. Based on the similarity between consecutive frames and frequent detection, the latest detection results can be very close to ground truth. Therefore, we reformulate $\mathbb{P}^G$ based on the observable inputs as $\mathbb{P}^D$:

$$\begin{aligned}
&\min \sum_{f,t,s} \Big(1 - \sum_{b^* \in \mathcal{B}_f^G} \max_{b \in \mathcal{B}_f^I} \{x_{f,s} \widetilde{IoU}_{b,b^*}^s + (1 - x_{f,s}) \widetilde{IoU}_{b,b^*}^D\}\Big) \\
&s.t. \quad C_1, C_2, C_3, C_4.
\end{aligned} \tag{5}$$

where $\widetilde{IoU}_{b,b^*}^D$ denotes the tracking accuracy from observable inputs, i.e., the IoU between the tracking result and the latest detection result. The same measure is used in $\widetilde{IoU}_{b,b^*}^s$ for evaluating the target detection accuracy on edge server $s$.

The problem $\mathbb{P}^D$ remains intractable despite using the observable inputs in place of the ground truth. Since the domain of $x_{f,s} \in \{0,1\}$, $\mathbb{P}^D$ belongs to 0-1 integer program, which is known as NP-complete. In addition, since the frames always arrive in an online manner, it requires the algorithm to make the offloading decision in advance and achieve high accuracy with latency constraints satisfied.

## III. Online Algorithm Design

In order to solve the above problems, we design a queue-based online optimization algorithm, OLA, which "learns" the video analytics accuracy results from previous offloading decisions and uses this information to make better decisions for the current epoch. In terms of algorithm design, firstly, we relax the domain of the original problem $\mathbb{P}^D$ from the integer domain to the real domain. Then, we use a queue to measure the cumulative violation of the long-run constraints, i.e., the cumulative number of violations of the frequency constraints detected in the long run, and the problem is decoupled into a series of subproblems in each epoch. Finally, we propose a randomized rounding strategy to revise the decisions into a feasible integer provisioning.

**Transformation.** For the ease of presentation, we define:

$$f_t^G(\boldsymbol{x}_t) = \sum_{f,s}(1 - \sum_{b^* \in \mathcal{B}_f^G} \max_{b \in \mathcal{B}_f^I}\{x_{f,s} IoU_{b,b^*}^s + (1-x_{f,s}) IoU_{b,b^*}^D\}),$$

$$f_t(\boldsymbol{x}_t) = \sum_{f,s}(1 - \sum_{b^* \in \mathcal{B}_f^G} \max_{b \in \mathcal{B}_f^I}\{x_{f,s} \widetilde{IoU}_{b,b^*}^s + (1-x_{f,s}) \widetilde{IoU}_{b,b^*}^D\}),$$

$$g_t(\boldsymbol{x}_t) = \sum_f (\tau - \sum_s x_{f,s}), \forall t \in \mathcal{T},$$

$$h_1^t(\boldsymbol{x}_t) = \sum_{f,s}\{x_{f,s} \cdot L_s^S + (1-x_{f,s}) \cdot L^D\} - L_{max}, \forall t \in \mathcal{T},$$

$$h_2^f(\boldsymbol{x}_t) = \sum_s x_{f,s} - 1, \forall f \in \mathcal{F}_t.$$

Since $\mathbb{P}^D$ is NP-hard, we relax the domain of $\mathbb{P}^D$ from the integer domain to the real domain and get problem $\widetilde{\mathbb{P}}^D$:

$$\begin{aligned} \min \quad & \sum_{t=1}^T f_t(\tilde{\boldsymbol{x}}_t) \\ s.t. \quad & \sum_{t=1}^T g_t(\tilde{\boldsymbol{x}}_t) \le 0, \\ \forall t: \quad & \boldsymbol{h}_t(\tilde{\boldsymbol{x}}_t) \le 0, \tilde{\boldsymbol{x}}_t \in \widetilde{\mathcal{X}}, \end{aligned} \tag{6}$$

where $\tilde{\boldsymbol{x}}_t$ is the fraction version of $\boldsymbol{x}_t$, $\widetilde{\mathcal{X}} = [0,1]^{|\mathcal{F}_t| \times S}$ is the real domain, and $\boldsymbol{h}_t = [h_1^t, ..., h2^f, ...]^\mathsf{T}$.

Since $f_t$ is equivalent to a linear function, the constraints $g_t$ and $\boldsymbol{h_t}$ also consist of linear functions. Also, the domain of $\tilde{\boldsymbol{x}}_t$ is a convex set. Thus, $\widetilde{\mathbb{P}}^D$ is a convex optimization problem.

**Queue-based Online Optimization.** Due to the detection frequency constraint over a long-term scope in problem $\widetilde{\mathbb{P}}^D$, it is considered as a long-term optimization problem. We propose a queue-based approach to decouple the problem. First, we introduce a virtual queue $Q_t$ to measure the cumulative degree to which the detection frequency constraint is exceeded. $\forall t$:

$$Q_{t+1} = \max\{-g_t(\boldsymbol{x}_{t+1}), Q_t + g_t(\boldsymbol{x}_{t+1})\}, \tag{7}$$

where $Q_{t+1}$ is updated per epoch by the increment $g_t(\boldsymbol{x}_{t+1})$ to measure the cumulative violation of the detection frequency constraint. In other words, minimizing $Q_t$ is beneficial as it allows more video frames to be offloaded to edge servers for inference, thereby enhancing video analytics accuracy. The original problem is then decoupled into a series of subproblems, where the optimization objective includes detection accuracy, the violation of detection frequency constraint and the regular term, while strictly restraining a latency constraint. At the end of each epoch $t$, we optimize $\mathbb{P}_{t+1}$ to make the decision for the next epoch; here, $\alpha > 0$ is the step size:

$$\begin{aligned} \min_{\boldsymbol{x} \in \widetilde{\mathcal{X}}} \quad & \{f_t(\boldsymbol{x}) + (Q_t + g_{t-1}(\boldsymbol{x}_t))g_t(\boldsymbol{x}) + \alpha\|\boldsymbol{x} - \boldsymbol{x}_t\|_2^2\} \\ s.t. \quad & \boldsymbol{h}_t(\boldsymbol{x}) \le 0. \end{aligned} \tag{8}$$

**Algorithm 1** Queue-based OnLine Optimization Alg. (OLA)

```
Input: $\tau$, $L_{max}$
1:  Initialize a proper step size $\alpha$, $Q_1 = 0$, $\boldsymbol{g}_0(\cdot) = 0$;
2:  Initialize $\hat{\boldsymbol{x}}_1$ by offloading video frames to fixed edge
    server, $\dot{\boldsymbol{x}}_1 = \ddot{\boldsymbol{x}}_1 = \hat{\boldsymbol{x}}_1$;
3:  for $t = 1$ to $T$ do
4:      Deploy the provisioning $\hat{\boldsymbol{x}}_t$;
5:      Solve subproblem $\mathbb{P}_{t+1}^0$ to obtain $\dot{\boldsymbol{x}}_{t+1}$;
6:      Solve subproblem $\mathbb{P}_{t+1}$ to obtain $\ddot{\boldsymbol{x}}_{t+1}$;
7:      Update the virtual queue:
        $Q_{t+1} = \max\{-g_t(\dot{\boldsymbol{x}}_{t+1}), Q_t + g_t(\dot{\boldsymbol{x}}_{t+1})\}$;
8:      if $(\dot{\boldsymbol{x}}_{t+1} - \ddot{\boldsymbol{x}}_{t+1})^\mathsf{T}(\dot{\boldsymbol{x}}_t - \ddot{\boldsymbol{x}}_{t+1}) \ge 0$ then $\tilde{\boldsymbol{x}}_{t+1} = \ddot{\boldsymbol{x}}_{t+1}$;
9:      else $\tilde{\boldsymbol{x}}_{t+1} = \dot{\boldsymbol{x}}_{t+1}$;
10:     Obtain $\hat{\boldsymbol{x}}_{t+1}$ by rounding $\tilde{\boldsymbol{x}}_{t+1}$ randomly;
11: end for
```

Since the bandwidth $b_{s,t}$ between the device and server $s$ varies over time, the instantaneous constraint $h_1^t$ is time-varying. However, designing an online optimization algorithm with guaranteed performance with time-varying constraints is challenging [12]. To address this issue, we introduce a tight constraint for the latency constraint. Although the network bandwidth fluctuates over time, there exists a lower bound $b_{s,0}$, i.e., $b_{s,0} \le b_{s,t}$. Therefore, there is $L_{s,t}^E \le L_{s,0}^E$. Accordingly, the delay constraint $h_1^t$ is transformed into a time-independent tight constraint using the lower bound $b_{s,0}$. Thus, $\forall t \in \mathcal{T}$:

$$h_0(\boldsymbol{x}_t) = \sum_{f \in \mathcal{F}_t}\{x_{f,t,s} \cdot L_{s,0}^S + (1 - x_{f,t,s}) \cdot L^D\} - L_{max}. \tag{9}$$

To facilitate the algorithm design and meet the analytics requirements, we transform the original subproblem $\mathbb{P}_{t+1}$ into $\mathbb{P}_{t+1}^0$ with time-independent constraints:

$$\begin{aligned} \min_{\boldsymbol{x} \in \widetilde{\mathcal{X}}} \quad & \{f_t(\boldsymbol{x}) + (Q_t + g_{t-1}(\boldsymbol{x}_t))g_t(\boldsymbol{x}) + \alpha\|\boldsymbol{x} - \boldsymbol{x}_t\|_2^2\} \\ s.t. \quad & \boldsymbol{h}_0(\boldsymbol{x}) \le 0. \end{aligned} \tag{10}$$

Since constraints for $\mathbb{P}_{t+1}^0$ are tighter than those for $\mathbb{P}_{t+1}$, i.e., the constraints of $\boldsymbol{h}_0(\boldsymbol{x})$ must be satisfied with $\boldsymbol{h}_t(\boldsymbol{x})$. Therefore, any feasible solutions of $\mathbb{P}_{t+1}^0$ also applies to $\mathbb{P}_{t+1}$.

Given that $\mathbb{P}_{t+1}$ is solved in the real domain, while the original problem seeks solutions in the integer domain, we have to convert the solution $\tilde{\boldsymbol{x}}_t$ from the real domain into a feasible solution $\hat{\boldsymbol{x}}_t$ in the integer domain. Thus, we propose a randomized rounding strategy. For each element $x_i$ in $\tilde{\boldsymbol{x}}_t$, it is rounded to 1 with a probability of $x_i$, otherwise to 0.

We propose the queue-based OnLine optimization Algorithm (OLA), shown in Alg. 1. In lines 4-7, OLA separately solves $\mathbb{P}_{t+1}^0$ and $\mathbb{P}_{t+1}$ for each epoch, obtaining the respective solutions $\dot{\boldsymbol{x}}_{t+1}$ and $\ddot{\boldsymbol{x}}_{t+1}$. We then use $\dot{\boldsymbol{x}}_{t+1}$ to update the virtual queue $Q_t$. In lines 8-10, OLA derives the integer solution for the original problem. Specifically, if $(\dot{\boldsymbol{x}}_{t+1} - \ddot{\boldsymbol{x}}_{t+1})^\mathsf{T}(\dot{\boldsymbol{x}}_t - \ddot{\boldsymbol{x}}_{t+1}) \ge 0$, we choose $\ddot{\boldsymbol{x}}_{t+1}$. Otherwise, $\dot{\boldsymbol{x}}_{t+1}$ with a tight constraint. Finally, the rounding strategy converts $\tilde{\boldsymbol{x}}_{t+1}$ into integer $\hat{\boldsymbol{x}}_{t+1}$.

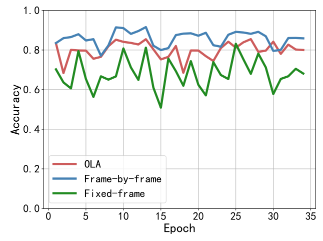

Fig. 2. Video analytics accuracy per epoch

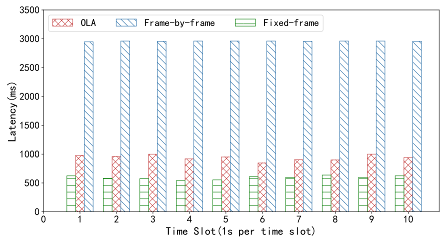

Fig. 3. Overall latency

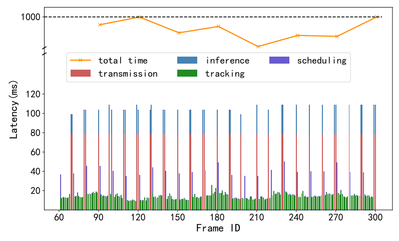

Fig. 4. Latency per frame using OLA

## IV. Theoretical Analysis

Due to space limitations, we show only theoretical results. For details, please refer to supplemental materials [31].

**Lemma 1.** *With the definition of dynamic regret and constraint violation in integral and real domains, we have:*

$$Reg_T \le \widetilde{Reg}_T, \quad Vio_T = \widetilde{Vio}_T. \tag{11}$$

**Lemma 2.** *The solution of the objective function $f_t$ over the real domain is bounded:*

$$\sum_{t=1}^{T} f_t(\tilde{\boldsymbol{x}}_t) \le \sum_{t=1}^{T} f_t(\dot{\boldsymbol{x}}_t) + 2\max_t\{Q_t\}\sum_{t=1}^{T}\theta_t. \tag{12}$$

**Theorem 1.** *With previous assumptions and lemmas, the integral dynamic regret is upper-bounded as:*

$$Reg_T \le \widetilde{Reg}_T = o(T), \tag{13}$$

*if $V_{\boldsymbol{\lambda}}$, $V_f$, $V_g$, $V_{\boldsymbol{x}}$ are subliner with respect to $T$.*

**Theorem 2.** *With previous assumptions and lemmas, the integral constraint violation is upper-bounded as:*

$$Vio_T = \widetilde{Vio}_T = o(T), \tag{14}$$

*if $\widetilde{V}_g$, $V_{\boldsymbol{\lambda}}$ are sublinear; $V_f$, $V_g$, $V_{\boldsymbol{x}}$ are subquadratic with $T$.*

## V. Experiments and Conclusion

**Data and Settings.** We use the Jetson AGX Xavier as the terminal device, employing OpenCV (i.e., the optical flow method) to perform object tracking on the device. We implement OLA with an efficient convex programming solver CVXPY from Python. In each epoch, two convex optimization problems need to be solved: $\mathbb{P}_{t+1}^0$ and $\mathbb{P}_{t+1}$. Jetson AGX Orin is used as the edge server, with three different edge servers deploying YOLOv5s, YOLOv5m, and YOLOv5l [13] as inference models for object detection.

We perform evaluation on the real-world road monitoring videos [14], with the raw videos processed to 1080p and 30fps using the FFmpeg tool [17]. Each epoch contains 10 frames. The network bandwidth is based on a real-world trace [15], varying from 30 to 70 Mbps. Additionally, to ensure that the bounding boxes for the target objects can be corrected with frequent detection results, the video task offloading frequency $\tau$ in the long-term constraint is set as 0.1.

We compare OLA with the following algorithms. (1) Frame-by-Frame offloading: it ignores latency constraints and aims to maximize analytics accuracy by offloading all frames to the server deploying the smallest model, i.e., YOLOv5s. (2) Fixed-Frame offloading: it randomly selects a server to offload frames at a fixed frequency, e.g., every 15 frames.

**Experiment Results.** Fig. 2 shows the video analytics accuracy for different algorithms. OLA achieves an average accuracy of 79.9%, representing a 17.3% improvement over Fixed-Frame (68.12%). This is because OLA can offload more video frames, which helps correct cumulative tracking errors between consecutive frames. Frame-by-Frame uses CNN models to detect objects in every frame, resulting in the highest average accuracy of 85%, while OLA's accuracy is only 6% lower. Fig. 3 shows the overall latency for analyzing one second of video for different algorithms. OLA has an average latency of 944ms, which strictly adheres to the instantaneous latency constraint and meets the real-time requirements for video analytics. Although Frame-by-Frame achieves the highest accuracy, its overall latency reaches 2,997ms, which is 3.17 times that of OLA, with only a 6% increase in accuracy. Since the Fixed-Frame offloads fewer video frames, it has the lowest latency but also the lowest accuracy.

Fig. 4 shows the latency for processing each frame using OLA. The average latency for online scheduling is 28ms, the average transmission latency for a single frame is 78ms, the average inference latency on an edge server is 26ms, and the average tracking latency for a single frame on the device is 14ms. These time costs may vary with network bandwidth fluctuations and changes in video content. Out of 240 frames in an 8-second video, a total of 38 frames were offloaded, with an average latency of 103ms for each offloaded frame, including transmission and inference time. The total video processing time per second is strictly controlled within one second, meeting the real-time requirements for video analytics.

**Conclusion.** This work adopts a "detect + track" approach, offloading video frames to edge servers for object detection at a relatively high frequency while continuously tracking detected objects on the terminal device using tracking technology between consecutive frames. We propose a queue-based online optimization algorithm, OLA. Experiments demonstrate that OLA can achieve high-accuracy video analytics results while meeting real-time requirements, and rigorous theoretical proof verifies that the dynamic regret bound and constraint violation bound grow sublinearly over time.

**Acknowledgments.** This work was supported in part by NSFC (62202233), Double Innovation Plan of Jiangsu Province (JSSCBS20220409), Grant from State Key Laboratory for Novel Software Technology, Nanjing University (KFKT2024B18).

## References

[1] Q. Zhang, H. Sun, X. Wu and H. Zhong, "Edge Video Analytics for Public Safety: A Review," in *Proceedings of the IEEE*, vol. 107, pp. 1675-1696, 2019.

[2] I. E. Olatunji and C.-H. Cheng, "Video Analytics for Visual Surveillance and Applications: An Overview and Survey," *MLP*, pp. 475–515, 2019.

[3] M. J. Shafiee, B. Chywl, F. Li, and A. Wong, "Fast YOLO: A Fast You Only Look Once System for Real-time Embedded Object Detection in Video." *arXiv:1709.05943*, 2017.

[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] X. Ran, H. Chen, X. Zhu, Z. Liu, and J. Chen, "Deep-Decision: A Mobile Deep Learning Framework for Edge Video Analytics," in *IEEE INFOCOM*, 2018, pp. 1421–1429.

[6] "Detectron2 Model," https://github.com/facebookresearch/detectron2/blob/master/MODEL ZOO.md/, 2020.

[7] 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.

[8] 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.

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

[10] 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.

[11] Rezatofighi, Hamid, et al. "Generalized intersection over union: A metric and a loss for bounding box regression." in *IEEE CVPR*, 2019, pp. 658–666.

[12] X. Cao, J. Zhang, and H. V. Poor, "A Virtual-Queue-Based Algorithm for Constrained Online Convex Optimization With Applications to Data Center Resource Allocation," *JSTSP*, vol. 12, no. 4, pp. 703–716, 2018.

[13] X. Zhu, S. Lyu, X. Wang, and Q. Zhao, "TPH-YOLOv5: Improved YOLOv5 Based on Transformer Prediction Head for Object Detection on Drone-captured Scenarios." *arXiv:2108.11539*, 2021..

[14] 4K Road traffic video for object detection and tracking - free download now!, (Jul. 16, 2018). Accessed: Mar. 27, 2024. [Online Video]. Available: https://www.youtube.com/watch?v=MNn9qKG2UFI

[15] R. Netravali et al., "Mahimahi: Accurate Record-and-Replay for HTTP," in *USENIX ATC*, 2015, pp. 417–429.

[16] W. Zhang et al., "Elf: accelerate high-resolution mobile deep vision with content-aware parallel offloading," in *ACM MobiCom*, 2021, pp. 201–214.

[17] FFmpeg, https://ffmpeg.org/.

[18] A. Graves, "Long Short-Term Memory," in Supervised Sequence Labelling with Recurrent Neural Networks, vol. 385, pp. 37–45.

[19] T. Y.-H. Chen, L. Ravindranath, S. Deng, P. Bahl, and H. Balakrishnan, "Glimpse: Continuous, Real-Time Object Recognition on Mobile Devices," in *ACM SenSys*, 2015, pp. 155–168.

[20] M. Hanyao, Y. Jin, Z. Qian, S. Zhang, and S. Lu, "Edge-assisted Online On-device Object Detection for Real-time Video Analytics," in *IEEE INFOCOM*, 2021, pp. 1–10.

[21] 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 *ACM SIGCOMM*, 2020, pp. 359–376.

[22] J. Li, L. Liu, H. Xu, S. Wu, and C. J. Xue, "Cross-camera inference on the constrained edge," in *IEEE INFOCOM 2023*. IEEE, 2023, pp. 1–10.

[23] 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, 2022.

[24] R. Xu, S. Razavi, and R. Zheng, "Edge Video Analytics: A Survey on Applications, Systems and Enabling Techniques," IEEE Communications Surveys & Tutorials, 2023.

[25] J. Ye, H. Yeo, J. Park, and D. Han, "AccelIR: Task-Aware Image Compression for Accelerating Neural Restoration," in *IEEE/CVF CVPR*, 2023, pp. 18 216–18 226.

[26] P. Dai, Y. Chao, X. Wu, K. Liu, and S. Guo, "Context-aware offloading for edge-assisted on-device video analytics through online learning approach," *IEEE Transactions on Mobile Computing*, vol. 23, no. 12, pp. 12 761–12 777, 2024.

[27] K. Du, A. Pervaiz, X. Yuan, A. Chowdhery, Q. Zhang, H. Hoffmann, and J. Jiang, "Server-driven video streaming for deep learning inference," in *ACM Special Interest Group on Data Communication (SIGCOMM)*, 2020, pp. 557–570.

[28] M. Hanyao, Y. Jin, Z. Qian, S. Zhang, and S. Lu, "Edge-assisted online on-device object detection for real-time video analytics," in *IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2021, pp. 1–10.

[29] K. Huang and W. Gao, "Real-time neural network inference on extremely weak devices: agile offloading with explainable ai," in *ACM Annual International Conference on Mobile Computing And Networking (MobiCom)*, 2022, pp. 200–213.

[30] Z. Wang, S. Zhang, J. Cheng, Z. Wu, Z. Cao, and Y. Cui, "Edge-assisted adaptive configuration for serverless-based video analytics," in *IEEE international Conference on Distributed Computing Systems (ICDCS)*, 2023, pp. 248–258.

[31] https://cs.nju.edu.cn/sheng/SupplementalOLA.pdf.