A Reinforcement-Learning-Based Energy-Efficient Framework for Multi-Task Video Analytics Pipeline

Yingying Zhao*, *Member, IEEE,* Mingzhi Dong*, Yujiang Wang†, Da Feng, Qin Lv, Robert P. Dick, *Senior Member, IEEE,* Dongsheng Li, *Member, IEEE,* Tun Lu, *Member, IEEE,* Ning Gu, *Member, IEEE,* and Li Shang, *Member, IEEE*

Abstract—Deep-learning-based video processing has yielded transformative results in recent years. However, the video analytics pipeline is energy-intensive due to high data rates and reliance on complex inference algorithms, which limits its adoption in energy-constrained applications. Motivated by the observation of high and variable spatial redundancy and temporal dynamics in video data streams, we design and evaluate an adaptive-resolution optimization framework to minimize the energy use of multi-task video analytics pipelines. Instead of heuristically tuning the input data resolution of individual tasks, our framework utilizes deep reinforcement learning to dynamically govern the input resolution and computation of the entire video analytics pipeline. By monitoring the impact of varying resolution on the quality of high-dimensional video analytics features, hence the accuracy of video analytics results, the proposed end-to-end optimization framework learns the best non-myopic policy for dynamically controlling the resolution of input video streams to globally optimize energy efficiency. Governed by reinforcement learning, optical flow is incorporated into the framework to minimize unnecessary spatio-temporal redundancy that leads to re-computation, while preserving accuracy. The proposed framework is applied to video instance segmentation which is one of the most challenging computer vision tasks, and achieves better energy efficiency than all baseline methods of similar accuracy on the YouTube-VIS dataset.

Index Terms—energy-efficient, vision, multi-task application, reinforcement learning

I. Introduction

Deep learning has achieved great success on video-based computer vision tasks [1], [2], [3]. Deep models such as MaskTrack R-CNN [2] are widely employed for multi-task video analytics, such as object detection, object classification, and segmentation. Deep models are generally energy-intensive due to the high amount of video stream data to process, which

* Equal contribution.

† Corresponding author.

Y. Zhao, M. Dong, T. Lu, N. Gu, L. Shang are with School of Computer Science, Fudan University, Shanghai, China, Shanghai Key Laboratory of Data Science, Fudan University, Shanghai, China, and Shanghai Institute of Intelligent Electronics & Systems, Shanghai, China.

Y. Wang is with Department of Computing, Imperial College London, London, UK. yujiang.wang14@imperial.ac.uk.

D. Feng is with Alibaba (Beijing) Software Service Company Limited, Beijing, China.

Q. Lv is with University of Colorado Boulder, Boulder, CO, USA.

R. P. Dick is with Department of Electrical Engineering and Computer Science College of Engineering, University of Michigan, Ann Arbor, MI, USA.

D. Li is a senior researcher with Microsoft Research Asia, Shanghai, China and an adjunct professor with School of Computer Science, Fudan University, Shanghai, China.

constrains their adoption in energy-constrained scenarios such as edge computing [4]. However, the ability to perform intelligent video analytics in energy-constrained edge devices is becoming increasingly important with the fast expansion of intelligent Internet-of-Things [4], [5]. There is an urgent need for energy-efficient multi-task video analytics.

This work aims to optimize the energy efficiency of video analytics tasks using a variable-resolution strategy. This is inspired by the observation that abundant data redundancy potentially exists in multi-task video analytics applications. As illustrated in Fig. 1, this enables two widely used computer vision tasks, i.e., object detection and semantic segmentation, to optimize efficiency while maintaining acceptable accuracy across a wide range of data resolutions. Real-world data redundancy offers us opportunities to optimize energy efficiency via variable-resolution analysis.
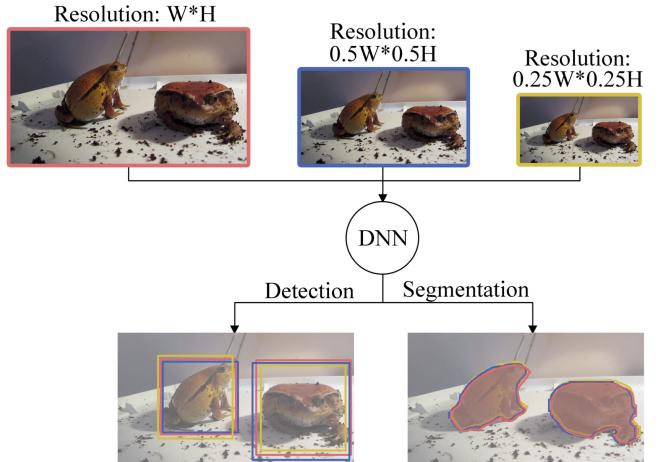

Fig. 1: Object detection and segmentation results of a sample input image with different resolutions: The red/blue/yellow edges indicate the results of the same image with resolution $W * H$, $\frac{W}{2} * \frac{H}{2}$, and $\frac{W}{4} * \frac{H}{4}$. We can see that the detection and segmentation results are very similar for the three resolutions, and the general performance is acceptable across different resolution settings.

Learning appropriate frame resolutions for multi-task video analytics is a challenging problem as appropriate resolutions may vary across different tasks, different scenarios, etc. For instance, as shown in Fig. 2, a Deep Neural Network (DNN) may still work well on object detection with low-resolution images, but it cannot properly address the semantic segmentation, which is more sensitive to resolution. Another example is shown in Fig. 3: even for the same task, i.e., face detection

shown in Fig. 3, it is still difficult to find appropriate frame resolutions due to the varying frame analysis difficulty. We aim to make online decisions on frame resolutions that can lead to globally optimized energy efficiency.
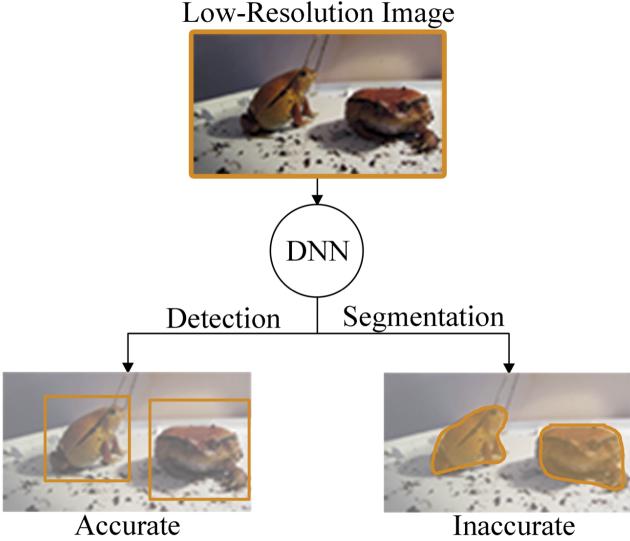
Low-Resolution Image

Fig. 2: Object detection and segmentation results for a low-resolution frame: the bounding boxes of the detection task are still accurate, while the predicted segmentation mask becomes less accurate with unsatisfying visual qualities. Different tasks may require different resolutions to produce adequate results.
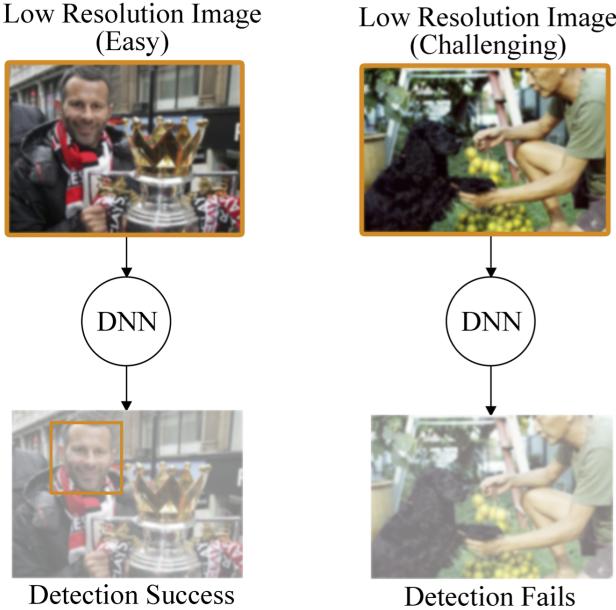

Fig. 3: For two facial images with identical low resolutions, face detection accuracy may be adequate on one (*Left Column*) and fail on another (*Right Column*). This indicates that even for the same video analytics task, the suitable frame resolutions vary with frame analysis difficulty.

The complicated temporal dynamics of video streams also pose a challenge. Reducing resolution has the potential to reduce accuracy. However, such accuracy loss can be effectively compensated for by an estimator that is aware of the historical temporal information in video streams. As shown in Fig. 4, our estimator incorporates historical information from earlier,

high-resolution frames to generate more robust and more accurate predictions, despite that the low-resolution current frame can be misleading to DNN models. Therefore, accurate estimation requires the online analysis of video temporal changes.

In this paper, we propose to use reinforcement learning (RL) to holistically overcome these challenges: (1) complexity variations among different tasks, (2) variable difficulty of different samples, and (3) complicated temporal dynamics. To globally optimize energy efficiency, our RL network learns the best non-myopic policy for determining the spatio-temporal frame resolution of incoming video stream data. Compared with other energy-efficient single-task video analytics solutions [6], [7] that were designed for still images without utilizing temporal information, our work is the first to address the energy consumption optimization problem for multi-task video analytic pipeline, and it is also the first to leverage RL to holistically tackle all these challenges indicated above, and to do end-to-end global efficiency policy optimization.

Our analysis pipeline is illustrated in Fig. 5. Frame images have variable resolution (e.g., $\frac{W}{2} * \frac{H}{2}$, $\frac{W}{4} * \frac{H}{4}$ and $\frac{W}{8} * \frac{H}{8}$, and denoted as non-key frames), or remain unchanged (key frames with resolution $W * H$). To leverage temporal information and compensate for performance reduction with lower resolution, we incorporate contextual optical flow [8] for feature estimation as suggested by Zhu et al. [9]. The energy optimization problem, specifically, determining frame resolutions with respect to multiple tasks and temporal dynamics, is considered as an end-to-end optimization problem and is modeled as a Markov Decision Process (MDP), which is solved using RL [10].

To evaluate the proposed framework, we have applied it to video instance segmentation [2], a synthesis video analytics pipeline consisting of simultaneous detection, segmentation, and tracking of object instances. Video instance segmentation is considered one of the most challenging multi-task video analytics applications, as it requires the predictions of instance-level segmentation masks while simultaneously tracking and identifying each instance. Our experimental results on the YouTube-VIS dataset [2] indicate that our proposed solution is more energy efficient than all baseline methods.

In summary, this work makes the following contributions:

1) This work presents an adaptive-resolution framework for multi-task video analytics in energy-constrained scenarios. The resulting challenges are managed by Reinforcement Learning (RL) algorithms aiming to globally optimize energy efficiency. To the best of our knowledge, this is the first time that RL has been employed to learn a non-myopic policy for such an energy-efficient framework.

2) We have applied the proposed framework to video instance segmentation [2], one of the most challenging multi-task computer vision tasks. Our framework is significantly more energy efficient than all baseline methods of similar accuracy.

The rest of this paper is organized as follows. Section II surveys related work. Section III analyzes the spatio-temporal data redundancy in video stream. Section IV characterizes
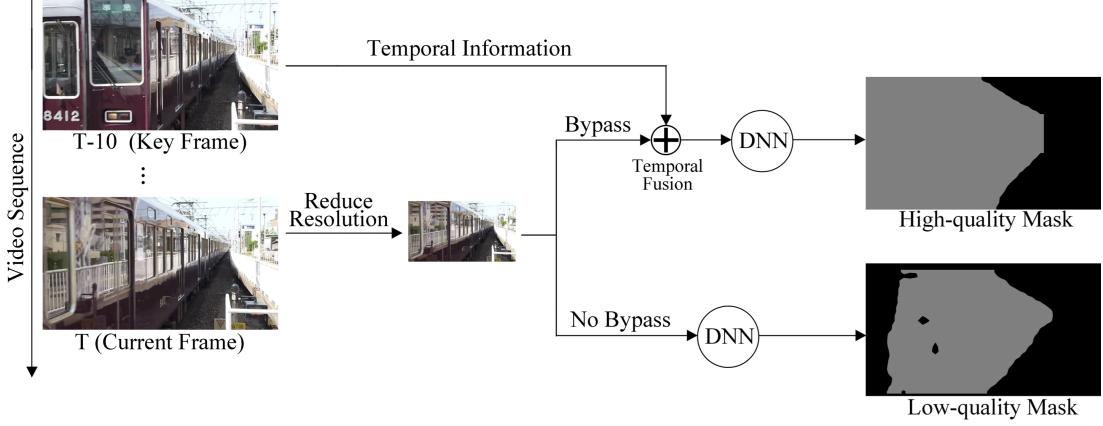
Fig. 4: An illustration of the temporal estimation mechanism for the semantic segmentation task. Without estimation, a low resolution frame (e.g., *T* in the figure) may lack important semantic information and can lead to low-quality segmentation masks. However, we can compensate by exploiting spatio-temporal redundancy to estimate the missing high-resolution information (e.g., *T-10* in the figure), which can produce segmentation results with high visual quality.
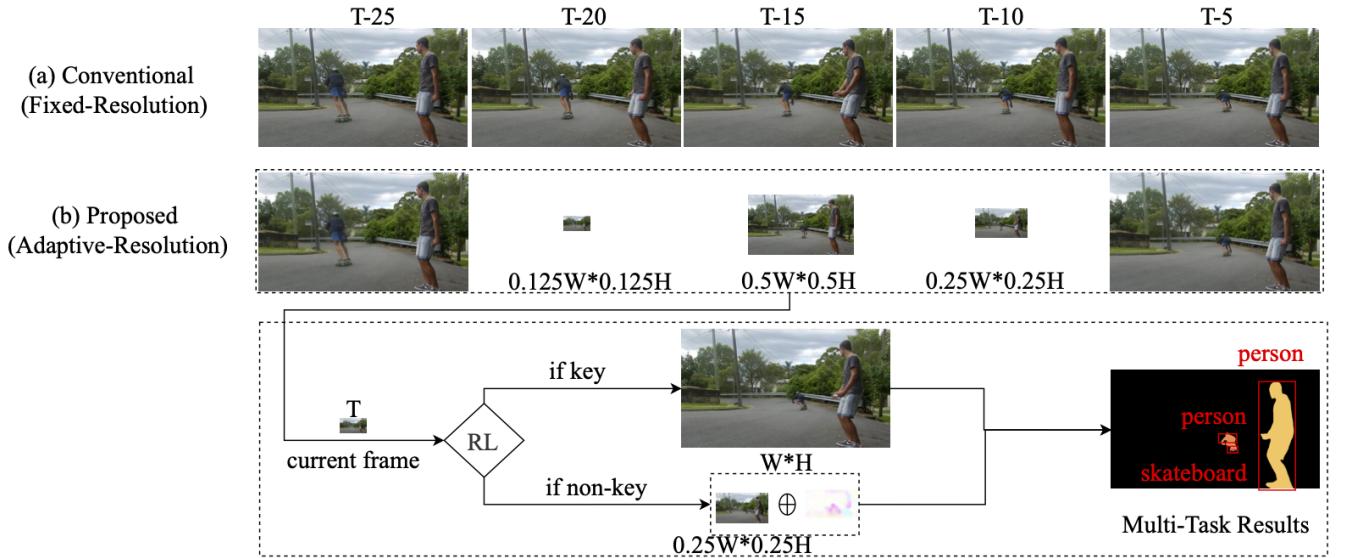

Fig. 5: An illustration of the proposed framework where video frames are shown in every 5 frames. (a) Common fixed-resolution frames in multi-task video analytics pipeline. (b) The proposed adaptive-resolution multi-task video analytics pipeline.

the energy consumption of imaging systems. Section V provides the problem definition. Section VI describes the energy-efficient framework. Section VII presents experimental results. Section VIII concludes this work.

II. RELATED WORK

The most relevant works are those on energy-efficient computer vision and feature propagation with optical flows.

Energy-Efficient Computer Vision: Kulkarni et al. proposed to optimize energy efficiency by varying frame resolution in a multi-camera surveillance network [11], which significantly reduced energy usage (85% or more) while providing comparable reliability. LiKamWa et al. proposed a power model based only on hardware [12]. This model reduces power consumption by 30% for video capturing by optimizing camera clock frequency. Based on the power model proposed in [12], Lubana et al. analyzed sensing energy and described the energy model for imaging systems [6]. This work indicated that system energy consumption depends significantly on the

transferred resolutions in imaging systems, and thus they optimized energy usage by using a multi-phase capture-and-analysis approach in which low-resolution, wide-area captures are used to guide high-resolution, narrow captures, thus eliminating task-irrelevant image data capture, transfer, and analysis. Later, Lubana et al. [7] described an application-aware compressive sensing framework, which reduces channel bandwidth requirements and signal communication latency without substantial performance drop by reducing unimportant data (i.e., pixels) transmission and analysis, thereby compressing application-related data representation. Additionally, a two-stage variable-resolution solution is proposed by Wang et al. [13], which implemented object detection using low-resolution images and recognition using high-resolution images. Their experimental results demonstrated that the resolution can be reduced by 51.4% with comparable recognition accuracy. Feng et al. proposed to detect and track moving objects in video to reduce the data volume in video-based computer vision applications [14].

Our method differs from the prior works in two ways: (1) we consider the complicated temporal dynamics in video streams and leverage a temporal bypassing system to better estimate high-resolution spatial information and (2) our work is end-to-end, considering all the challenging factors in multi-task video analytics (e.g., the complexity variations among different tasks and spatial and temporal dynamics in video data) as a complete system using RL to optimize the energy efficiency of the entire multi-task video analytics pipeline.

Feature Propagation Methods: Zhu et al. [9] presented a Deep Feature Flow (DFF) method that propagates the intermediate features between video frames via optical flow [15]. DFF accelerates the video analytics pipeline by using efficient optical flow calculation instead of computation-intensive feature extraction with backbone networks. Their work schedules the key frames at a fixed interval. In contrast, Wang et al. [16] presented a more flexible key-frame scheduler to accelerate semantic segmentation [17], [18], [19] in videos while preserving the segmentation accuracy. They modeled the key decision process as a deep RL problem and learned an efficient scheduling policy by maximizing the global return, hence the global performance.

Xu et al. demonstrated a dynamic video segmentation network (DVSNet) for fast and efficient video semantic segmentation [20]. They designed a light-weight decision net to determine whether the current frame is sent to the fast warping path or the computational-intensive segmentation path. Xu et al. [20] considered deviations from the current frame and the last key frame to judge whether it is appropriate to schedule a key frame.

In contrast with prior work focusing on single-computer vision tasks, we tackle the more challenging multi-task video analytics problem and use feature propagation with the optical flow to exploit temporal redundancy to estimate high-resolution spatial information. This is controlled by RL-based policy network, concurrently with other challenging components.

III. Data Redundancy Analysis

Video data are inherently redundant, both spatially and temporally. In this section, we characterize data redundancy in video data at different resolutions, and demonstrate that it is possible to reduce resolution to reduce data redundancy, thereby improving energy efficiency while maintaining acceptable task performance.

A. Redundancy Analysis

We first analyze spatial data redundancy by varying frame resolution and evaluating the resulting impact on performance of video analytics tasks. We consider two commonly used computer vision tasks: face detection with still images and video-based object detection. For each task, we uniformly subsample the original image/video frames at several reduced resolutions and determine the resulting accuracy. The down-sampling factor is defined as the ratio of resized image pixels to original image pixels. For face detection, we evaluate the

performance of the S³FD architecture [21] on the WIDER-FACE [22] dataset. For video-based object detection, we evaluate the performance of the MaskTrack R-CNN architecture for object detection on the YouTube-VIS dataset [2]. Since both are detection tasks, mean Average Precision (mAP) is used to quantify performance. We use COCO evaluation metrics[1] to average 10 Intersection over Union (IoU) thresholds from 50% to 95% in 5% intervals. Note that the WIDERFACE dataset divides the samples into three difficulty categories: *easy*, *medium*, and *hard*, which are plotted separately.
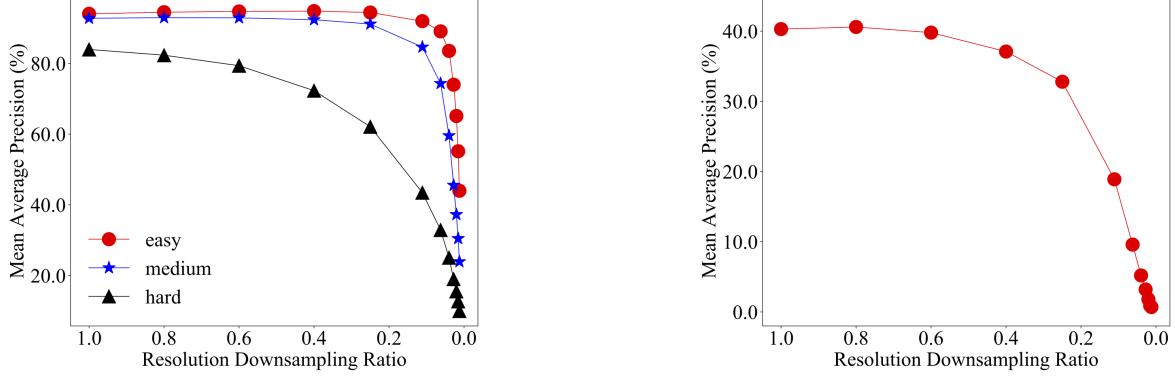
As demonstrated in Fig. 6a and Fig. 6b, the task performance (mAP) degrades gracefully with the decrease in resolution. For instance, when the resolution downsampling ratio is greater than 0.3 (30% of the original pixels), the performance degradation remains insignificant (e.g., 0.4% for the *medium* data). In short, discarding 70% of the original data greatly improves energy consumption with little impact on accuracy. In addition, Fig. 6a demonstrates that, for the *easy* data set, the task performance remains acceptable until the resolution reduces to approximately 0.1 (i.e., 10% of the original pixels), while the accuracy of the *hard* data set deteriorates significantly as the resolution approaches 0.5. This study demonstrates that it is possible to apply spatial resolution reduction with limited impact on performance, yet it remains a challenge to determine the appropriate resolution for individual frames with varying difficulties.

B. Dynamics Analysis

Consecutive video frames generally share a large fraction of similar pixels, which can lead to high temporal redundancy in video stream data. Fig. 7 shows one such example. It is unnecessary to re-compute the whole current frame given that we have already obtained the features of previous frames: we can use the features in the previous frames to accelerate the analysis of the current frame. Such techniques have been studied in the field of video segmentation, and we have adopted the Deep Feature Flow [9] framework following [20], [16] to address temporal redundancy. Specifically, if a current frame is determined as a non-key frame with low resolution, we use FlowNet [8] to obtain the optical flow between the current frame and the last key frame, and the computed optical flow is used to propagate features from the last key frame into the current one such that the performance drop caused by low-resolution frames can be compensated for. More details can be found in Section VI.

To demonstrate the effectiveness of using Deep Feature Flow [9] to exploit the temporal redundancy, we integrate FlowNetC [8] into the MaskTrack R-CNN model and evaluate its performance on object detection and instance segmentation tasks using the YouTube-VIS dataset. As shown in Fig. 8a, the performance of a solo MaskTrack R-CNN model for object detection drops significantly when the downsampling ratio is extremely low (e.g., lower than 0.15). However, when optical flow has been integrated ("MaskTrack R-CNN+FlowNetC" in Fig. 8a), the resulting model can be much more tolerant to

---

[1]https://github.com/cocodataset/cocoapi

(a) Face detection using the S^{3}FD method on the WIDERFACE dataset.

(b) Object detection using MaskTrack R-CNN method on YouTube-VIS dataset.

Fig. 6: Resolution versus mAP on different tasks.

downsampling, demonstrating the importance of using temporal information to eliminate spatio-temporal redundancy in video stream. A similar trend can be observed in Fig. 8b for the instance segmentation task. As a result, we utilize MaskTrack R-CNN+FlowNet for the non-key frames with lower resolutions, while for key frames, we use the MaskTrack R-CNN model, which performs better performance on high-resolution images.

When bypassing temporal information, we can also eliminate a lot of temporal redundancy while retaining acceptable performance, which further demonstrates the feasibility of the adaptive resolution strategy. However, the temporal dynamics in video stream data are usually complicated and therefore difficult to analyze, highlighting the challenges of obtaining suitable frame resolutions. These observations motivated us to develop the RL-based optimization framework.

IV. ENERGY CONSUMPTION ANALYSIS

In this section, we first characterize the energy consumption of imaging systems. Then, we demonstrate that the amount of energy consumption is highly related to the volume of input data.

A. Conventional Image Analysis Framework

A typical imaging pipeline starts with an image sensor that captures and converts the incoming light into electrical signals via a 2-D sensor array, and transfers the signals in the form of data frames to an image signal processor (ISP) and an application processor for digital signal processing and computer vision tasks [6]. Prior work indicates that data transfer, digital signal processing, and computer vision tasks account for more than 90% of the total energy [7], which depends strongly on the amount of data.

B. Energy Model

The energy consumption, E, of an imaging system (per frame) is mainly due to data sensing, communication, and computation [6], as follows:

$$E = E_{sensor} + E_{ISP} + E_{host} + E_{comm}, \qquad (1)$$

where E_{sensor}, E_{ISP}, E_{host}, and E_{comm} denote the energy consumption of image sensor, ISP, host application processor, and the communication interface between the sensor and ISP/application processor, respectively.

(1) Energy consumption of image sensing. The energy consumption of an image sensor is state-dependent (e.g., idle, active, and standby). In the exposure phase (T_{exp}), the image sensor is idle with power $P_{sensor,idle}$. In the active phase (T_{active}), the image sensor processes and outputs pixels, with one pixel per clock period [6]. The time duration T_{active} is therefore determined by the ratio of image frame resolution R_{frame} to external clock frequency f, and the power consumption of the active state $P_{sensor,active}$ is a linear function of sensor resolution R ($R \geq R_{frame}$). The image sensor consumes negligible power in standby mode [12] (0.5–1.5 mW, typically) so no corresponding term is required in the energy model. Sensor energy is calculated as follows:

$$E_{sensor} = P_{sensor,active}T_{active} + P_{sensor,idle}T_{exp}, \qquad (2)$$

where R and $P_{sensor,idle}$ are sensor-specific parameters.

(2) Energy consumption of the ISP. The ISP is active during image processing (T_{ISP}), and idle during image sensing ($T_{exp} + R_{frame}/f$) and other computer vision tasks (T_{app}) [6]:

$$E_{ISP} = P_{ISP,active}T_{ISP} + P_{ISP,idle}(T_{exp} + R_{frame}/f + T_{app}), \qquad (3)$$

where $P_{ISP,idle}$ and $P_{ISP,active}$ are the idle and active power of the ISP, respectively. Prior work has shown that T_{ISP} is a nearly linear function of image resolution, and T_{app} is also strongly dependent on image resolution [6]. Therefore, the energy consumption of the ISP depends strongly on image resolution.

(3) Energy consumption of application processor. The host application processor is active during computer vision task processing and idle otherwise [6]:

$$E_{host} = P_{host,active}T_{app} + P_{host,idle}(T_{exp} + R_{frame}/f + T_{ISP}), \qquad (4)$$

where $P_{host,active}$ and $P_{host,idle}$ are processor-dependent. Equation 4 also suggests that the energy usage of computer
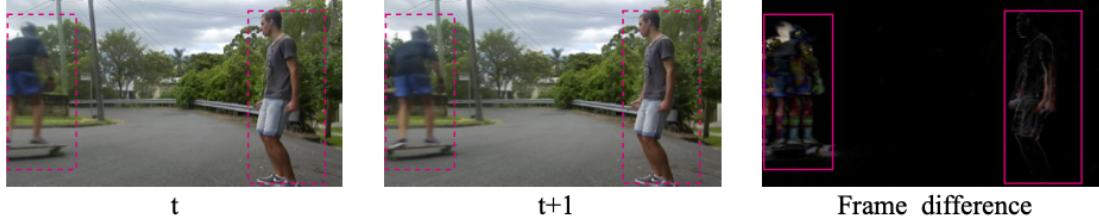
Fig. 7: Comparison of two frames at timestamps t and $t + 1$ in a video sequence. *Left* and *Middle*: Two consecutive video frames selected from the Youtube-VIS dataset. *Right*: Difference between the two frames. The dashed and solid red rectangles highlight the different parts, and we can see that those two frames share a large proportion of similar pixels.

(a) Object detection
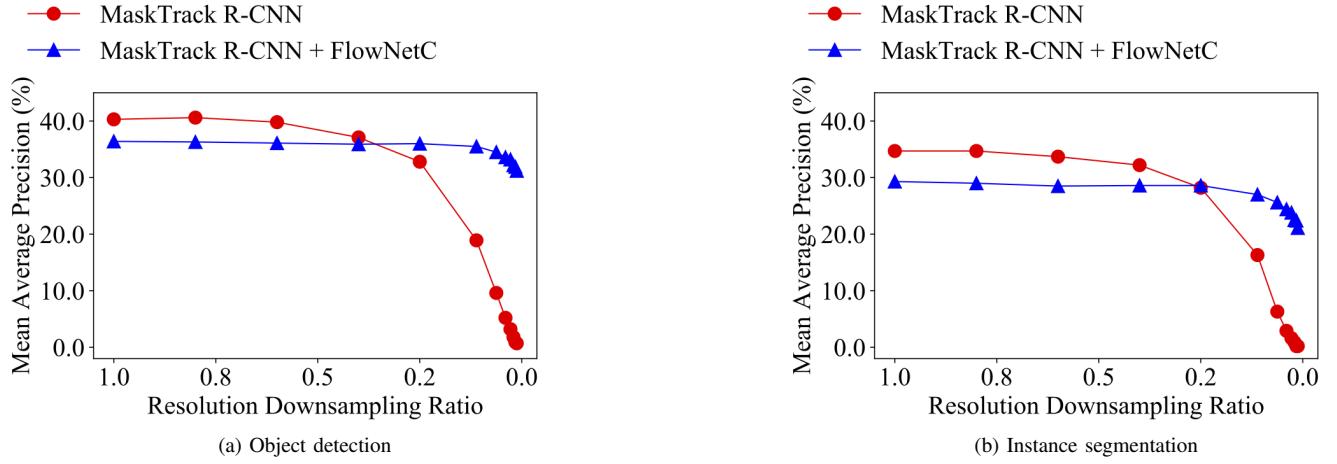
(b) Instance segmentation

Fig. 8: mAP of MaskTrack R-CNN without and with FlowNetC (i.e., temporal bypassing with optical flow) on the YouTube-VIS dataset.

vision tasks strongly depends on image resolution.

(4) Energy consumption of communication interface. E_{comm} is a linear function of the total amount of data transferred (in pixels) [6], which is defined in Equation 5.

$$E_{comm} = k \cdot R_{frame}, \tag{5}$$

where k is a communication interface specific constant.

Equations 1–5 demonstrate that the energy consumption of an imaging system is a strong function of image resolution, specifically, spatial resolution and frame rate. Therefore, data reduction offers the most promising first-line treatment for imaging system energy optimization. However, data reduction may negatively affect video analytics accuracy, which motivates the following study on data redundancy and the impact on video analytics accuracy.

V. PROBLEM DEFINITION

The energy model in Section IV demonstrates that it is feasible to improve energy efficiency by reducing frame resolutions. In Section III, we illustrate that the strategy of adopting reasonably low-resolution frames can be potentially applied to energy-constrained scenarios, since it can effectively reduce the spatial and temporal data redundancy while preserving acceptable accuracy.

However, determining a suitable resolution for each video frame in the multi-task video analytic pipeline is challenging, as we need to consider (1) varying difficulties in different frames, (2) varying task complexities, and (3) complicated temporal dynamics in the video stream. If we fail to consider

any of these, we may end up with resolution decisions that will lead to unsatisfying energy consumption efficiency.

We have developed a holistic approach that simultaneously considers these factors in an end-to-end fashion. Specifically, we formulate the process of estimating the energy-optimal frame resolutions as a Markov Decision Process (MDP), which is explained below.

A. Cumulative Reward

Let $\mathcal{A} = \{a^1, a^2, ..., a^n\}$ be the set of n potential actions where each action represents using a certain frame resolution, e.g., $1/4$ of the original size. We denote the policy of determining frame resolutions as π. Let $\mathbf{s}_t$ be the state to be considered by π at time step t, and let $a_t \in \mathcal{A}$ be the decision on the tth frame's resolution, i.e., $a_t = \pi(\mathbf{s}_t)$. Let ACC_{a_t} be the performance with a certain metric achieved by decision $\mathbf{a_t}$ on that frame, and let E_{a_t} be the energy consumption of that decision. we can define the reward r_t at this time step as

$$r_t = ACC_{a_t} + \lambda \frac{1}{E_{a_t}}, \tag{6}$$

where λ is a hyper-parameter to trade off accuracy ACC_{a_t} and energy consumption E_{a_t}. A larger r_t is generally more desirable. For a video sequence of length m, our goal is to learn an optimal policy π for maximizing the cumulative rewards G that can be written as

$$G = \sum_{t=1}^{m} \gamma^{t-1} r_t = \sum_{t=1}^{m} \gamma^{t-1} \left(ACC_{a_t} + \lambda \frac{1}{E_{a_t}} \right), \tag{7}$$

where $\gamma^{t-1} \in [0,1]$ and $a_t = \pi(\mathbf{s}_t)$. However, it is difficult to determine a non-myopic policy π for realistic video analytics applications. In this paper, we adopt RL to maximize Equation 7, which is described in detail in Section VI.

B. Video Instance Segmentation

For evaluation purposes, we select video instance segmentation [2], a synthesis and challenging multi-task video analytics pipeline that has broad application scenarios. Specifically, instance segmentation consists of three major targets: (1) *object detection* to localize all objects in video frames; (2) *object classification* to assign category labels to the detected objects; and (3) *instance segmentation* to perform pixel-level classification for each instance. For video-based instance segmentation, an additional task named *object tracking* is defined in [2], which traces the object trajectory in video sequences. The YouTube-VIS [2] dataset is widely used for video instance segmentation task evaluation; We use it for evaluation.

MaskTrack R-CNN [2], a variant of Mask R-CNN [1], is used as a baseline method for video instance segmentation. Fig. 9 illustrates the video instance segmentation framework based on MaskTrack R-CNN [2]. An input image of arbitrary size is first fed into the backbone network (or feature extractor) to obtain appropriate feature descriptors, and then a Region Proposal Network (RPN) [23] is leveraged to generate several potential Regions of Interest (RoIs) on those descriptors. RoI align [1] is utilized to convert each RoI candidate with variable size into fixed-size feature maps, e.g., 7×7. After that, those fixed-size feature maps are fed into three branches of networks (referred to as "heads" [1]): (1) a Fully Connected (FC) network head to localize instances with bounding boxes and perform classifications; (2) a Fully Convolutional Network (FCN) to predict segmentation masks for each instance; and (3) a tracking network head for tracking instances in a video sequence. Note that the tracking head is not included in the original Mask R-CNN framework [1] and is inserted by Yang et al. [2] to meet the need of video instance segmentation tasks. For a fair comparison, we use this MaskTrack R-CNN model as the baseline method in this work.

VI. METHODOLOGY

This section describes our reinforcement-learning-based adaptive-resolution framework for video instance segmentation in detail.

A. Framework Overview

As described in Section V, our goal is to develop an adaptive-resolution multi-task video analytics framework that optimizes energy consumption and accuracy. We model the adaptive resolution selection problem as an MDP. To maximize the cumulative reward G in Equation 7, we use RL to dynamically govern the spatial resolution and temporal dynamics of the complete video instance segmentation pipeline.

Let $\mathbf{I} = \{\mathbf{I}_1, \mathbf{I}_2, ..., \mathbf{I}_m\}$ be a video sequence of length m, where $\mathbf{I}_t$ denotes the frame image at time step $t \in \mathbb{Z} \cap [1, m]$. For a frame image $\mathbf{I}_t$ of resolution $w_t * h_t$, where w_t and h_t refer to its width and height, respectively, we define the action set $\mathcal{A} = \{a^1, a^2, \ldots, a^k\}$, where a^1 stands for using its original frame size $w_t * h_t$. $a^2, \ldots, a^k$ refer to downsampling $\mathbf{I}_t$ to a lower resolution, e.g., $\frac{w_t}{2} * \frac{h_t}{2}$, $\frac{w_t}{4} * \frac{h_t}{4}$ and $\frac{w_t}{8} * \frac{h_t}{8}$. We denote the frame where action a^1 is used (i.e., without downsizing the frame image) as the key frame and others as the non-key frames. Therefore, our goal is to find a policy network π_θ that can map the state s_t at each time step t to an appropriate action a_t to maximize the cumulative reward G described in Equation 7. The RL with Double Q-learning (DDQN) [24] is used for optimization. Although various computer vision problems can be solved using this framework, we focus on video instance segmentation.

Specifically, given an incoming frame $\mathbf{I}_t$ at time step t, video instance segmentation performs the following prediction tasks: (1) bounding box prediction $\mathbf{b}_t$, (2) object classification $\mathbf{c}_t$, (3) segmentation mask $\mathbf{s}_t$, and (4) tracking prediction $\mathbf{d}_t$. We follow the MaskTrack R-CNN approach [2] to perform these predictions with several modifications. The first step is to use a feature extractor denoted as $\mathcal{N}_{feat}$ to extract representative feature descriptors $\mathbf{f}_t$, i.e., $\mathbf{f}_t = \mathcal{N}_{feat}(\mathbf{I}_t)$. After that, a Regional Proposal Network (RPN) $\mathcal{N}_{RPN}$ and a RoI Align operation [1] $RoIAlign$ are applied to obtain RoI features $\mathbf{f}'_t$ with identical sizes, i.e., $\mathbf{f}'_t = RoIAlign(\mathcal{N}_{RPN}(\mathbf{f}_t))$. $\mathbf{f}'_t$ is then fed into three task-related branches (i.e., heads): (1) the Bounding Boxes Head (BBbox Head) $\mathcal{N}_{bbox}$; (2) the Segmentation Head $\mathcal{N}_{mask}$; and (3) the Tracking Head $\mathcal{N}_{track}$. These three heads generate the required predictions, i.e., $\{\mathbf{b}_t, \mathbf{c}_t\} = \mathcal{N}_{bbox}(\mathbf{f}'_t)$, $\mathbf{s}_t = \mathcal{N}_{mask}(\mathbf{f}'_t)$ and $\mathbf{d}_t = \mathcal{N}_{track}(\mathbf{f}'_t)$. To evaluate the overall performance on frame $\mathbf{I}_t$, we use the metric described by Yang et al. [2]: the mAP score integrating the performance of all four predictions. mAP is higher for more similar bounding boxes. This MaskTrack R-CNN pipeline is illustrated in Fig. 9.

Following the idea of Deep Feature Flow [9], we also integrate the FlowNet [15] architecture into the MaskTrack R-CNN framework for temporal information inference. Let $\mathcal{F}$ be the FlowNet model, and let $\mathbf{I}_k$ be the last key frame ($a_k = a^1$) where the feature descriptor $\mathbf{f}_k$ is already computed. If the current frame $\mathbf{I}_t$ is determined to be a non-key frame, i.e., $a_t \neq a^1$, we use $\mathcal{F}$ to estimate the optical flow from $\mathbf{I}_k$ to $\mathbf{I}_t$ denoted as $\mathbf{OF}_{k \rightarrow t}$, i.e., $\mathbf{OF}_{k \rightarrow t} = \mathcal{F}(\mathbf{I}_t, \mathbf{I}_k)$, and the feature descriptor $\mathbf{f}_t$ is calculated as follows: $\mathbf{f}_t = \mathcal{W}(\mathbf{OF}_{k \rightarrow t}, \mathbf{f}_k, \mathbf{S}_{k \rightarrow t})$, where $\mathcal{W}$ is a warping function and $\mathbf{S}_{k \rightarrow t}$ is the scale field from I_k to I_t. Zhu et al. [9] give details on the warping function and scale field. If $\mathbf{I}_t$ is determined to be a key frame ($a_k = a^1$), $\mathbf{f}_t$ will be obtained from the feature extractor $\mathcal{N}_{feat}$. The main advantage of using optical flow for non-key frames is that it can compensate for accuracy reductions due to downsampling, as demonstrated in Section III. We refer to the MaskTrack R-CNN + FlowNet architecture as MaskTrackFlow R-CNN, and Fig. 10 illustrates the structure of our MaskTrackFlow R-CNN.

Building on the MaskTrackFlow R-CNN, we design a reinforcement-based policy network π_θ with parameters θ to learn appropriate actions a_t such that the cumulative reward G in Equation 7 can be maximized, as explained below.
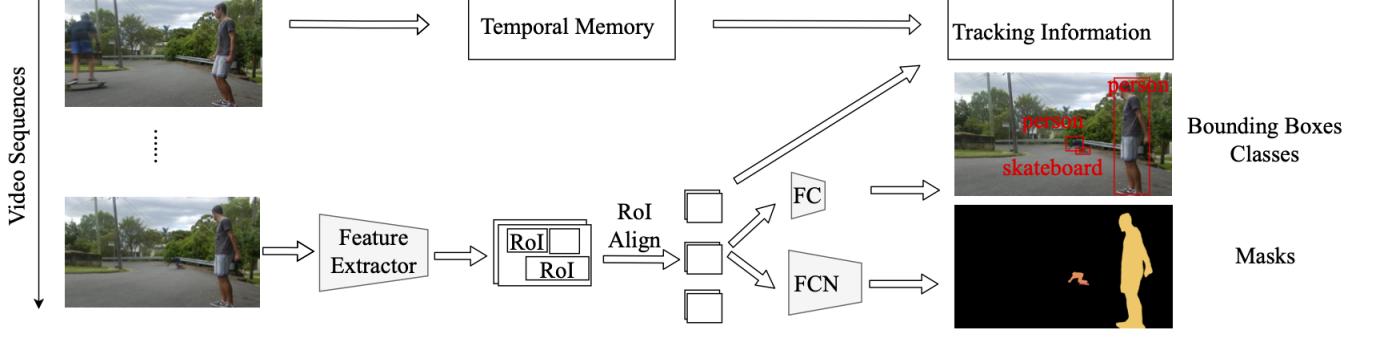
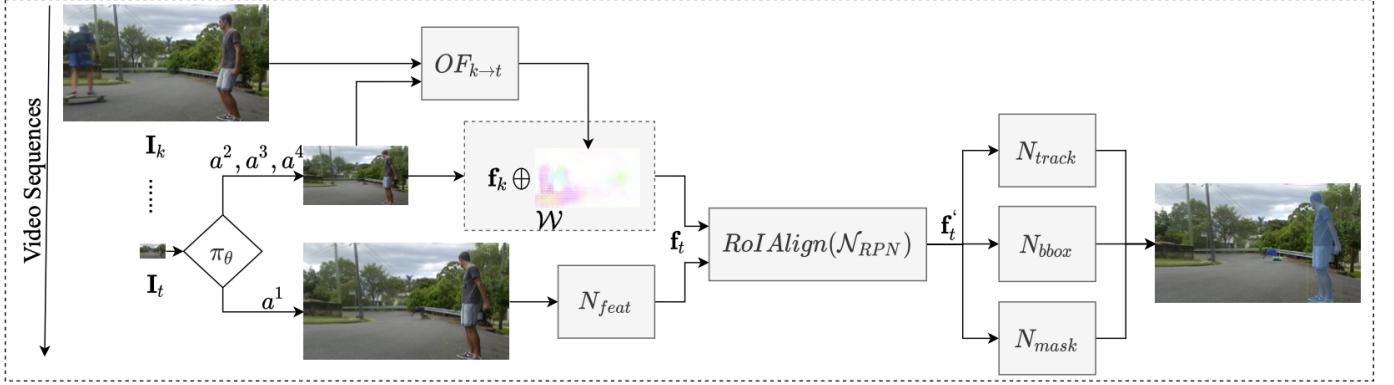Fig. 9: Video instance segmentation pipeline based on the MaskTrack R-CNN framework.

Fig. 10: Flowchart of the proposed MaskTrackFlow R-CNN. We use a policy network π_θ to govern the complete system. For each frame $\mathbf{I}_t$, this policy network π_θ determines an appropriate action from the action set $\mathcal{A}$ that aims to maximize the global objective function. If a non-key action (a^2, a^3 or a^4) is selected, the optical flow between the last key frame $\mathbf{I}_k$ and the current frame $\mathbf{I}_t$ will be used to estimate high resolution information using temporal redundancy. In effect, this propagates key features from $\mathbf{f}_k$ to $\mathbf{f_t}$ to maintain accuracy in the presence of downsampling. Here, $\mathbf{f}_k = \mathcal{N}_{feat}(\mathbf{I}_k)$, and $\mathbf{f_t}$ represents the features of the current frame. If π_θ opts for a key action (a^1), $\mathbf{f_t}$ (feature of current frame) will be obtained directly from $\mathcal{N}_{feat}$, i.e., $\mathbf{f_t} = \mathcal{N}_{feat}(\mathbf{I}_t)$.

B. Policy Network

For a video frame $\mathbf{I}_t$ of resolution $w_t * h_t$, our policy network π_θ gathers useful information at time step t and uses it as the state $\mathbf{s}_t$ to determine an appropriate action $a_t \in \mathcal{A}$. As indicated in Section VI-A, we define the action space $\mathcal{A}$ as

$$\mathcal{A} = \{a^1, a^2, a^3, a^4\}, \tag{8}$$

where a^1 refers to using original frame resolution $w_t * h_t$, a^2, a^3 and a^4 stand for resizing the frame image to lower resolution settings (e.g., $\frac{w_t}{2} * \frac{h_t}{2}$, $\frac{w_t}{4} * \frac{h_t}{4}$ and $\frac{w_t}{8} * \frac{h_t}{8}$). Let $\mathbf{I}_t^{a_t}$ be the frame image after applying action a_t, we define the state $\mathbf{s}_t$ as

$$\mathbf{s}_t = \{\mathbf{f}_t^{a^4}, \mathbf{f}_k - \mathbf{f}_t^{a^4}, \xi\}, \tag{9}$$

where $\mathbf{f}_t^{a^4} = \mathcal{N}_{feat}(\mathbf{I}_t^{a^4})$ represents the feature descriptor for $\mathbf{I}_t^{a^4}$, $\mathbf{f}_k$ is the feature descriptor for the last key frame $\mathbf{I}_k$ ($a_k = a^1$) that was already computed, and ξ is the summary information for historical resolution decisions. Intuitively, the first two terms $\mathbf{f}_t^{a^4}$ and $\mathbf{f}_k - \mathbf{f}_t^{a^4}$ provide the necessary spatial and temporal information for making resolution decisions, and the last term ξ informs the policy network π_θ of the historical decisions. Since the spatial resolution of $\mathbf{f}_k$ and $\mathbf{f}_t^{a^4}$ are not identical, we resize $\mathbf{f}_t^{a^4}$ to the shape of $\mathbf{f}_k$ through bi-linear interpolation such that $\mathbf{f}_k - \mathbf{f}_t^{a^4}$ can be implemented and also to avoid information loss in $\mathbf{f}_k$ of larger size.

The policy network π_θ contains one convolution layer (Conv0) and four fully connected (FC) layers: FC0, FC1, FC2, and FC3, as illustrated in Fig. 11. The tensor $\mathbf{f}_t^{a^4}$ (256 channels) is concatenated with tensor $\mathbf{f}_k - \mathbf{f}_t^{a^4}$ (256 channels) as the input to the first 1*1 convolution layer (Conv0) with 256 output channels. The input channels are squeezed gradually from FC0 to FC2 layers, which are 15,360, 4,096 and 1,024 channels. Following Wang et al. [16], we append the decision history ξ to the input tensor of the FC3 layer, while ξ depends on two terms: a vector with 20 channels containing the last 10 resolution decisions (we use two binary digits to encode a decision since we have a total of four actions here), and a scalar denoting the distance of the current t-th frame from the last key frame (i.e., action is a^1). Appending ξ increases the input channels of the FC3 layer from 256 to 277, which are summarized into four estimated Q values, i.e., $Q(s_t, a^i)$ ($i = 1, \ldots, 4$). Equation 10 defines how to estimate the Q values.

$$Q^\pi(s, a) = \mathbb{E}[G_t | s_t = s, a_t = a]. \tag{10}$$

For video instance segmentation, however, a more task-specific reward function than Equation 6 needs to be defined. We define the reward r_{a_t} of an action a_t on frame $\mathbf{I}_t$ as

$$r_{a_t} = \begin{cases} \lambda \frac{1}{E_{a_t}} + C_0 & a_t = a^1 \\ U_{a_t} - U_{\arg\max_x U_x^t}^t + \lambda \frac{1}{E_{a_t}} + C_0 & a_t \neq a^1 \end{cases}, \tag{11}$$
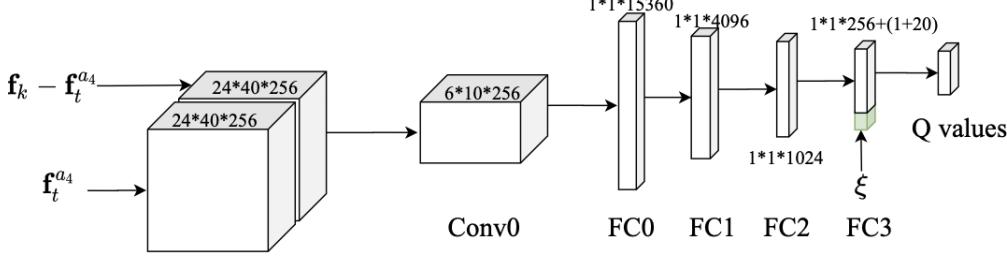
Fig. 11: The architecture of the policy network π_θ that determines an appropriate resolution (based on which action to take) for each frame $\mathbf{I}_t$. The input tensor consists of two parts: (1) the feature descriptor $\mathbf{f}_t^{a^4}$ obtained from the resized frame $\mathbf{I}_t^{a^4}$, and (2) the tensor $\mathbf{f}_k - \mathbf{f}_t^{a^4}$ which is the element-wise difference between the key feature $\mathbf{f}_k$ and the first component $\mathbf{f}_t^{a^4}$. The historical decision information ξ is concatenated to the last fully connected layer (FC3). The output of π_θ is a tensor with four channels. Each channel contains a Q value, and each Q value corresponds to taking a certain action in the action set $\mathcal{A}$. We select the action with the largest Q value as the action to take.

where E_{a_t} is the energy consumption of action a_t, U_{a_t} is the mAP score for the video instance segmentation task achieved on the frame $\mathbf{I}_t$ by action a_t, and $U_{\arg\max_x U_x^t}^t$ is the highest potential mAP on this time step, which is typically obtained when $a_t = a^1$. C_0 is a positive constant added to ensure $r_{a_t} \geq 0$.

Additionally, let P be the total number of episodes in the training process, and let T be the maximum time steps in one episode, we can see that the computational complexity of DDQN is $O(mTP)$, since the agent needs to determine one action (with the maximum Q value) from m actions at each time step, while there can be a maximum of TP time steps during the training process.

The algorithm for the proposed reinforcement-learning-based energy-efficient framework is described in Algorithm 1.

VII. EXPERIMENTS AND RESULTS

This section describes the experimental evaluation of the proposed energy-efficient video analytics pipeline.

A. Dataset

We use the YouTube-VIS dataset[2] [2] to evaluate the performance of our framework. This dataset consists of 2,883 videos, a 40-category label set and 131k instance masks, while the train/validation/test sets contain 2,238/302/343 videos, respectively. The 5th frame for each video snippet is annotated. Each video snippet lasts 3 to 6 seconds with a 30 fps frame rate. The maximum resolution of the original frame is $1,280 \times 720$. Since only the training set's annotation is released, we divide the training set with a 90%/5%/5% ratio for training/validation/testing in the following study.

B. Experimental Settings

1) Evaluation platform: The proposed framework is designed for energy-constrained edge devices. For evaluation purposes, following Lubana et al. [6], we consider an embedded hardware configuration including a Raspberry Pi 3 equipped with a Sony IMX219 image sensor with variable resolution support. The Sony IMX219 supports a maximum $3,280 \times 2,464$ resolution with 12 MHz clock frequency. As pointed out by Lubana et al. [6], the power consumption in

[2]https://youtube-vos.org/dataset/vis/

Algorithm 1 A Reinforcement-Learning-based Energy-Efficient Framework

Input: N Video Frames $\{\mathbf{I}_0, \mathbf{I}_1, \ldots, \mathbf{I}_N\}$ of Resolutions $\{w_0 * h_0, w_1 * h_1, \ldots, w_N * h_N\}$

1: $k \leftarrow 0$ ▷ initialize key frame
2: $\mathbf{f}_k = \mathcal{N}_{feat}(\mathbf{I}_k)$ ▷ obtain key feature
3: $\mathbf{f}_k' = RoIAlign(\mathcal{N}_{RPN}(\mathbf{f}_k))$ ▷ obtain key RoI feature
4: $\{\mathbf{b}_k, \mathbf{c}_k\} \leftarrow \mathcal{N}_{bbox}(\mathbf{f}_k')$ ▷ obtain key bbox and class
5: $\mathbf{s}_k \leftarrow \mathcal{N}_{mask}(\mathbf{f}_k')$ ▷ obtain key segmentation mask
6: $\mathbf{d}_k \leftarrow \mathcal{N}_{track}(\mathbf{f}_k')$ ▷ obtain key tracking prediction
7: $\mathbf{Y}_k \leftarrow \{\mathbf{b}_k, \mathbf{c}_k, \mathbf{s}_k, \mathbf{d}_k\}$ ▷ put together key predictions
8: Initialize ξ ▷ initialize historical information
9: **for** $t = 1$ to N **do**
10:      $\mathbf{f}_t^{a^4} = \mathcal{N}_{feat}(\mathbf{I}_t^{a^4})$ ▷ obtain low-resolution feature
11:      $s_t \leftarrow \{\mathbf{f}_t^{a^4}, \mathbf{f}_k - \mathbf{f}_t^{a^4}, \xi\}$ ▷ collect current state
12:      Estimate Q values and select action using policy network π_θ
13:      $a_t = \max_a Q(s_t, a; \theta)$ ▷ determine current action
14:      Update ξ with a_t ▷ update historical information
15:      **if** $a_t = a^1$ **then**   ▷ if key action
16:          $\mathbf{f}_t = \mathcal{N}_{feat}(\mathbf{I}_t)$ ▷ obtain current feature
17:          $k \leftarrow t$ ▷ update key with current
18:      **else**   ▷ if none-key action
19:          $\mathbf{OF}_{k \to t} = FlowNet(\mathbf{I}_k^{a_t}, \mathbf{I}_t^{a_t})$ ▷ obtain optical flow
20:          $\mathbf{S}_{k \to t} = \mathcal{S}(\mathbf{I}_k^{a_t}, \mathbf{I}_t^{a_t})$ ▷ obtain scale fields
21:          $\mathbf{f}_t = \mathcal{W}(\mathbf{OF}_{k \to t}, \mathbf{f}_k, \mathbf{S}_{k \to t})$ ▷ obtain current feature
22:      **end if**
23:      $\mathbf{f}_t' = RoIAlign(\mathcal{N}_{RPN}(\mathbf{f}_t))$ ▷ obtain current RoI feature
24:      $\{\mathbf{b}_t, \mathbf{c}_t\} \leftarrow \mathcal{N}_{bbox}(\mathbf{f}_t')$ ▷ obtain current bbox and class
25:      $\mathbf{s}_t \leftarrow \mathcal{N}_{mask}(\mathbf{f}_t')$ ▷ obtain current segmentation mask
26:      $\mathbf{d}_t \leftarrow \mathcal{N}_{track}(\mathbf{f}_t')$ ▷ obtain current tracking prediction
27:      $\mathbf{Y}_t \leftarrow \{\mathbf{b}_t, \mathbf{c}_t, \mathbf{s}_t, \mathbf{d}_t\}$ ▷ put together current predictions
28: **end for**

Output: Energy-efficient video analytics results $\{\mathbf{Y}_0, \mathbf{Y}_1, \ldots, \mathbf{Y}_N\}$

state $P_{sensor,idle}$ is 141.8 mW and that in $P_{sensor,active}$ is 8.27 mW/MP·R + 17.364 mW + 113.03 mW. We use a T_{exp} of 20 ms in the following study.

The Raspberry Pi 3 is equipped with an embedded GPU consisting of a dedicated image signal processing pipeline [6]. Following prior work [6], we approximate P_{ISP} using the GPU's power consumption. P_{CPU} and P_{GPU} (W-level, typically) can be directly measured by an ammeter. Time required by the Raspberry Pi ISP pipeline is approximately linear in R_{frame} [6], $T_{ISP} = 0.095 \times R_{frame} + 0.032$ (R_{frame} unit is MP).

The following study focuses on evaluating the energy efficiency and accuracy of our framework compared with existing work. We use the mean Average Precision (mAP) [2]

as the performance metric for video instance segmentation. We also define energy reduction as the ratio of the energy consumption of our method to that of existing work. The energy consumption is calculated using the energy model described in Section IV-B.

2) Training MaskTrackFlow R-CNN: In the MaskTrack-Flow R-CNN architecture, we employ the ResNet-50-FPN [25], [1] as the feature extractor $\mathcal{N}_{feat}$ and we use the Regional Proposal Network described by Yang et al. [2]. We also adopt the same structures for the three heads $\mathcal{N}_{bbox}$, $\mathcal{N}_{mask}$, and $\mathcal{N}_{track}$. For the FlowNet model $\mathcal{F}$, we use the FlowNetC architecture [15], and apply the warping function $\mathcal{W}$ from Deep Feature Flow [9]. Considering the complexity of the proposed MaskTrackFlow model, we use a two-step process to train it. We first train the feature extraction model $\mathcal{N}_{feat}$ and the three heads $\mathcal{N}_{bbox}$, $\mathcal{N}_{mask}$, and $\mathcal{N}_{track}$ on the video instance segmentation task described by Yang et al. [2], without considering the FlowNet model $\mathcal{F}$. We then train the FlowNet model $\mathcal{F}$ while freezing the other components (i.e., feature extractor $\mathcal{N}_{feat}$ and the three heads), following the design in Deep Feature Flow [9].

3) Training the policy network: To avoid unnecessary computation, we separate training of the policy network π_θ from training the MaskTrackFlow model. In other words, the weights of the MaskTrackFlow model are already learned and frozen when we train the policy network π_θ. We use the features extracted by ResNet-50 [1], [25] from the final convolutional layer of the first stage as the feature descriptor for images, e.g., $\mathbf{f}_t^{a^4}$ and $\mathbf{f}_k$ in Equation 9. We use Adam [26] as the optimizer with an initial learning rate of 0.0005. The discount factor (γ) is set to 1, implying that each frame in the video sequence is equally important. The exploration policy uses an ϵ-greedy policy [27] and we set ϵ to decrease from 0.9 to 0.05.

4) Baselines: We compare the proposed reinforcement-based approach of selecting frame resolutions with the following baseline methods:

(1) Downsampling Scan Method [6]: The Digital Foveation method [6] improves system energy efficiency using a multi-round, variable-resolution, variable-region strategy, in which an application-specific estimated accuracy constraint ($cnstrt$) may be used to govern the sensing and analysis process. It was developed for still images and therefore does not make use of temporal information about downsampling resolution. There are several ways it might be extended to video analytics, and we describe one straight-forward extension for use as a base case. We use the variable-resolution concept of Digital Foveation but gradually vary the sensed resolution for frame $\mathbf{I}_t$ from $\frac{w_t}{8} * \frac{h_t}{8}$, $\frac{w_t}{4} * \frac{h_t}{4}$, $\frac{w_t}{2} * \frac{h_t}{2}$ to $w_t * h_t$ if the accuracy reduction has surpassed the constraint $cnstrt$. In this work, we empirically set $cnstrt$ to be 0.2, 0.4, 0.6 and 0.8, respectively. We call this extension to video the Downsampling Scan method.

(2) Adaptive High-Resolution Frame Scheduling (*AdaptiveHFS*): This approach selects the key action a^1 for a frame $\mathbf{I}_t$ when the flow magnitude between $\mathbf{I}_t$ and the last key frame $\mathbf{I}_k$ exceeds a certain threshold Thr, otherwise a certain non-key action (i.e., a^2, a^3, or a^4) is taken. Please refer to Xu

et al. [20] for the definition of flow magnitude. We select Thr from 8 to 12 with an interval of 2. We have three variants of *AdaptiveHFS*, each of which selects a different non-key action to use: *AdaptiveHFS*(a^2), *AdaptiveHFS*(a^3), and *AdaptiveHFS*(a^4).

(3) Fixed-Interval High-Resolution Frame Scheduling (*FixIntervalHFS*). This baseline method selects a certain non-key action (a^2, a^3, or a^4) for every l ($l \in \{1, 2, 3\}$) frames, and the rest is set as key action (a^1). According to which non-key action to take, we also have three variants for the *FixIntervalHFS* approach, which are *FixIntervalHFS*(a^2), *FixIntervalHFS*(a^3), and *FixIntervalHFS*(a^4).

(4) Random High-Resolution Frame Scheduling (*RandomHFS*): This baseline method determines actions for each frame randomly with a hybrid distribution. Specifically, for frame $\mathbf{I}_t$, the probability of selecting the key action a^1 is r where $r \in \{0.9, 0.7, 0.5\}$, and the probability of taking other three non-key actions (a^2, a^3, and a^4) are uniform and sum to $1 - r$.

C. Results

1) RL Training Visualization: Fig. 12a, 12b, and 12c illustrate the average return during RL training where $\lambda \in \{0.4, 0.6, 0.8\}$. Note that we set C_0 in Equation 11 to 825 so all sessions can generate positive and comparable returns. Despite the fluctuations, all three curves steadily increase, indicating that the policy network is learning to maximize global return. The fluctuations of the curves plateau for large episodes (e.g., > 500), which suggests that the upper bound is being approached. When λ grows and the energy consumption term in Equation 11 increases, the maximum return achieved by the training curves also increases, which is consistent with expectations.

2) mAP versus Energy Consumption: Fig. 13a, Fig. 13b, and Fig. 13c illustrate the mAP (performance) versus energy consumption reduction curves for our method and the baselines. The energy consumption reduces significantly (more than 80%) at the cost of slight accuracy drops, no matter which resolution-selection method is used, thus verifying the effectiveness of the proposed adaptive resolution framework. Note that the policy net π_θ only accounts for a very small proportion of the total energy consumption in Fig. 13, which is around 4.2%, thanks to the low-resolution input and its light-weight architecture. Moreover, our method outperforms all other baseline approaches on all the energy consumption intervals, which shows the superiority of our RL-based resolution selector. For realistic computer vision tasks, we can fine-tune the RL models to have different energy consumption rates to suit the varying requirements.

Additionally, as the upper-bound method, MaskTrack R-CNN [2] delivers the highest mAP which is 41.7%. In contrast, our method greatly reduces energy consumption at the cost of slightly reduced accuracy, e.g. our framework achieves 41.4% mAP (only reduced by 0.3%) but saves approximately 84.0% energy consumption at the same time, which is much more energy-efficient.

We also explore how the parameter λ in Equation 6 can affect the proposed framework. λ characterizes the trade-offs

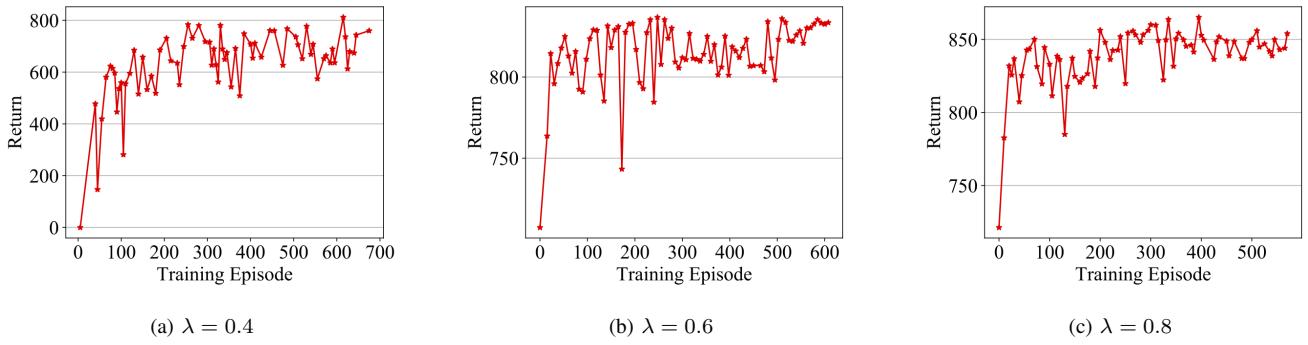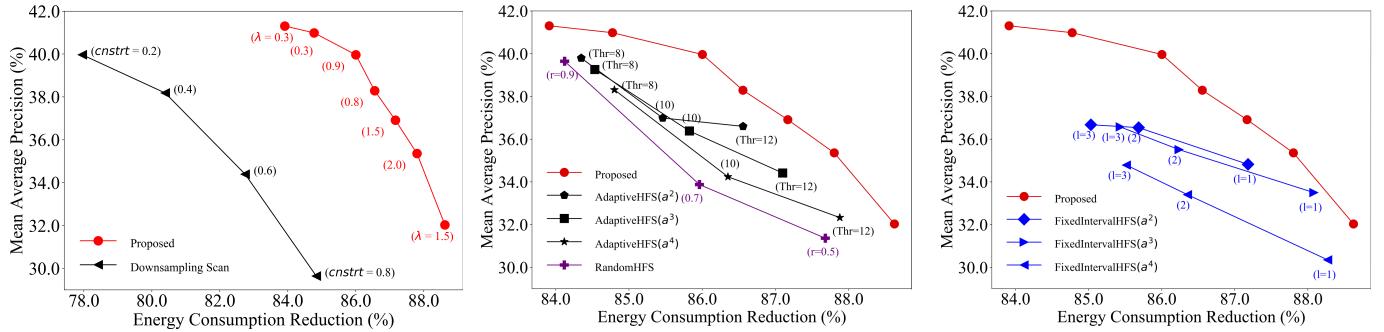(a) $\lambda = 0.4$ (b) $\lambda = 0.6$ (c) $\lambda = 0.8$

Fig. 12: The training curves of average return where λ value is set to 0.4, 0.6, and 0.8, respectively.

(a) Comparison of our method with the Downsampling Scan method. Different λ values utilized by the proposed method are also annotated.

(b) Comparison of our method with the *AdaptiveHFS* and *RandomHFS* baselines

(c) Comparison of our method with the *FixInterval-HFS* baseline

Fig. 13: mAP versus energy consumption reduction between the proposed method and the baselines.

between the accuracy and the energy consumption in our system and is thus one of the most important parameters. Generally, the higher λ is, the more important the energy consumption term in Equation 6 will be. The influences of different λ values can be seen in Fig. 13a, and we can discover a general trend that a larger λ value can lead to a higher energy consumption reduction rate, despite several fluctuations of the accuracy. Such observations are generally in line with our theoretical analysis.

3) Case Studies: This section further clarifies why the proposed method outperforms the Downsampling Scan method, as well describes three cases to provide intuition on why our RL-based method can outperform others.

(1) Comparison with the Downsampling Scan method: As shown in Fig. 13a, compared with the Downsampling Scan method, our extension of Digital Foveation [6] to video, our method achieves significantly better performance and energy reduction results. This is because the Downsampling Scan method attempts multiple downsampling resolutions for each frame, instead of using temporal context to quickly arrive at an appropriate downsampling rate. In contrast, our method uses a light-weight RL-based policy network to dynamically determine appropriate frame resolutions, avoiding an explicit per-frame linear search process, and is therefore better able to efficiently generalize to complicated video scenarios, e.g., VIS. Our method also embodies the multi-round process in Digital Foveation, but it allows the rounds to be divided among video

frames with only one round per frame and it exploits temporal locality in the optimal downsampling rate.

(2) Comparison with the *FixIntervalHFS* baseline: This case study compares our method with the *FixInterval-HFS*(a^2) baseline ($l = 1$) on a video sequence of 90 frames. We first study how the Accumulated Energy Consumption Reduction (AECR, the higher the better) rate varies on this video sequence. As shown in Fig. 14 (up), our method has 82.5% mAP on this sequence, surpassing the 80.0% of *Fix-IntervalHFS*(a^2). Our method also demonstrates lower energy consumption with 87.6% AECR (versus baseline's 87.0%) on the 90th frame. If we inspect the varying trends of AECR on this sequence, we see that although our method selects multiple key actions (a^1) at the beginning, the non-key actions (a^2, a^3) are frequently selected for frames $50 - 90$, thus reducing energy consumption. We plot the prediction results in Fig. 14 ($bottom$) on this temporal range, where the resolution selected by our method produces accurate results.

(3) Comparison with the *AdaptiveHFS* method: Similarly, we report the AECR results for our method and the *AdaptiveHFS*(a^3) ($Thr = 10$) on a 90-frame video sequence. As demonstrated in Fig. 15 (up), the mAP of our method on this sequence is 85.0%, which is significantly better than the 75.1% of the baseline, while our energy efficiency is also better than the *AdaptiveHFS* method (85.9% AECR versus 85.3%). It can be found that although our method has selected multiple key actions (a^1), it also opts for multiple a^4 which are the most energy-saving actions. As a result, our
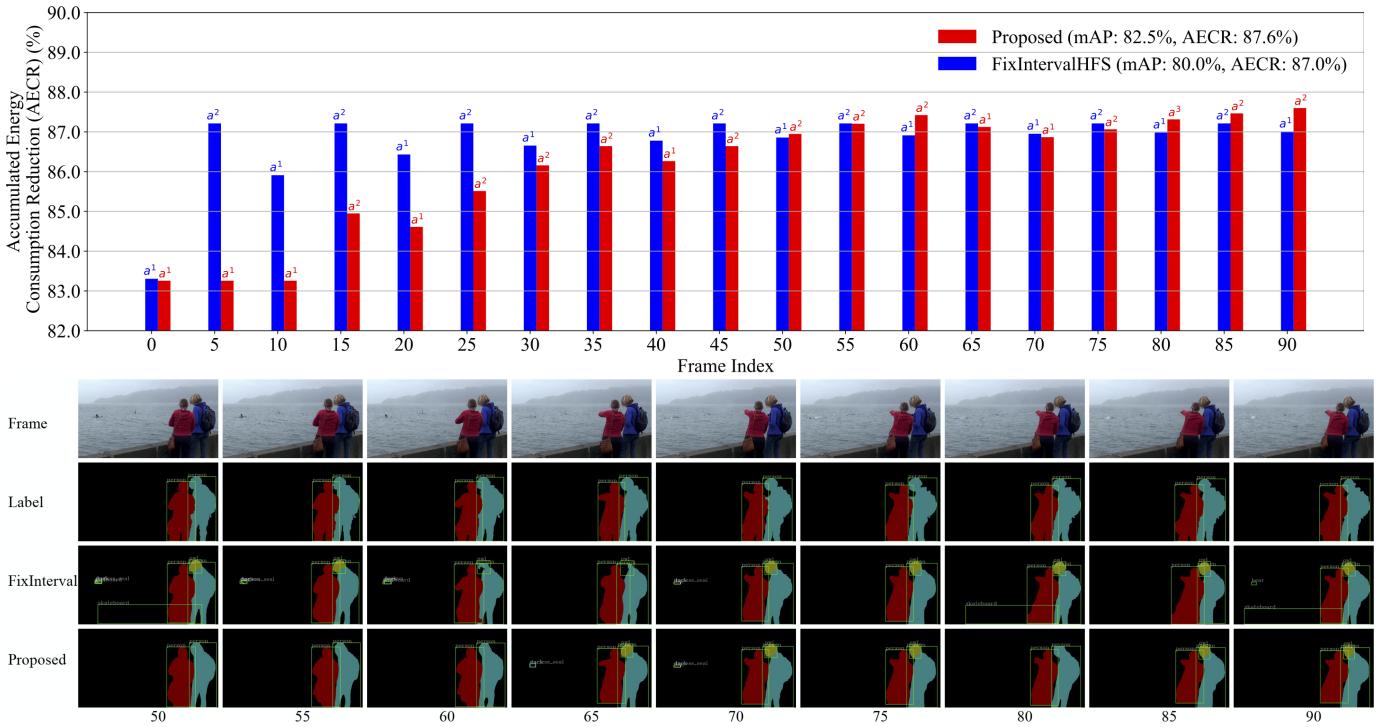
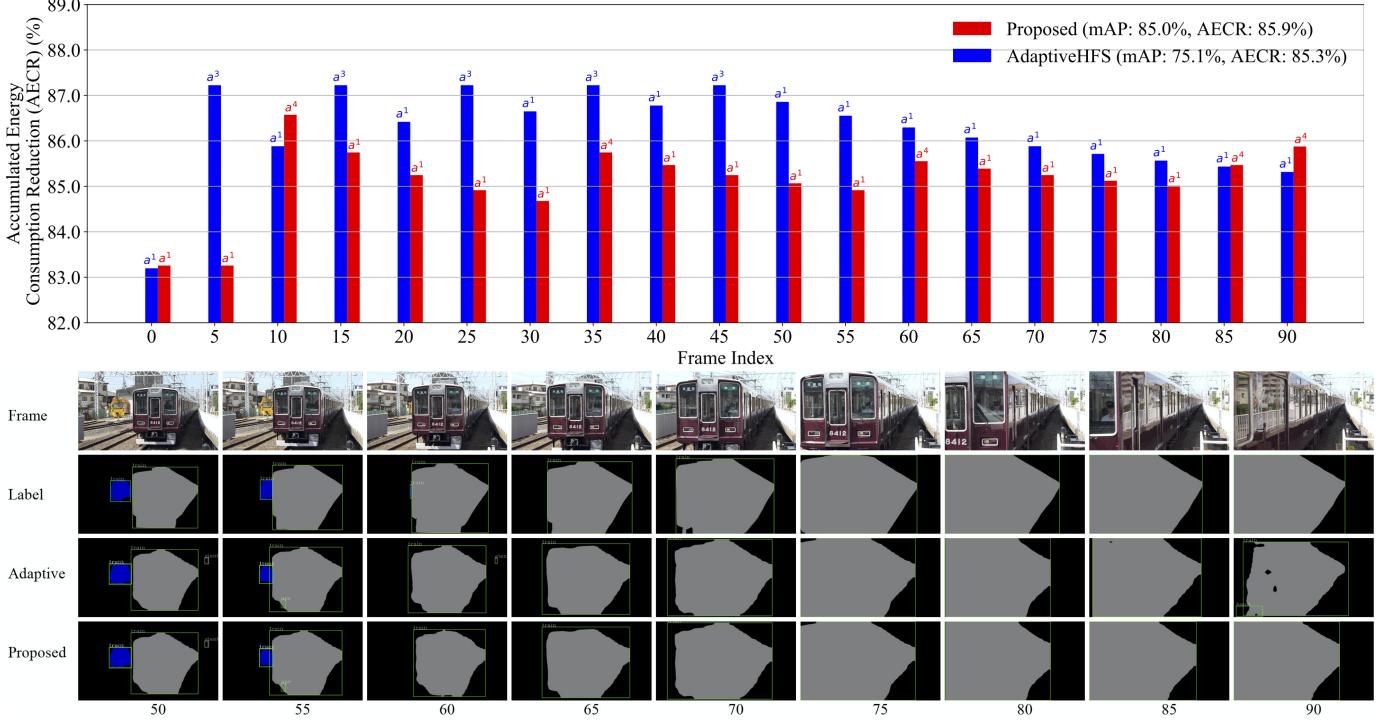Fig. 14: A comparison between the proposed method and the *FixIntervalHFS*(a^2) ($l = 1$) method.

Fig. 15: A comparison between the proposed method and the *AdaptiveHFS*(a^3) ($f = 10$) method.

method produces better prediction results, as shown in Fig. 15 (*bottom*).

(4) Comparison with the *RandomHFS* method. In Fig. 16, we show the comparison of our method with the *RandomHFS* baseline on a 90-frame video. Fig. 16 (*up*) shows that our method also outperforms the *RandomHFS* approach in both mAP and energy consumption. In particular, our method has

selected a large percentage of a^2 actions, while the baseline has frequently selected the key actions a^1. However, more key actions do not necessarily lead to better performance. As illustrated in 16 (*bottom*), our method has obtained better prediction results than the baseline, although the baseline has employed many more key actions. Therefore, we can arguably conclude that our RL-based method can more accurately grasp
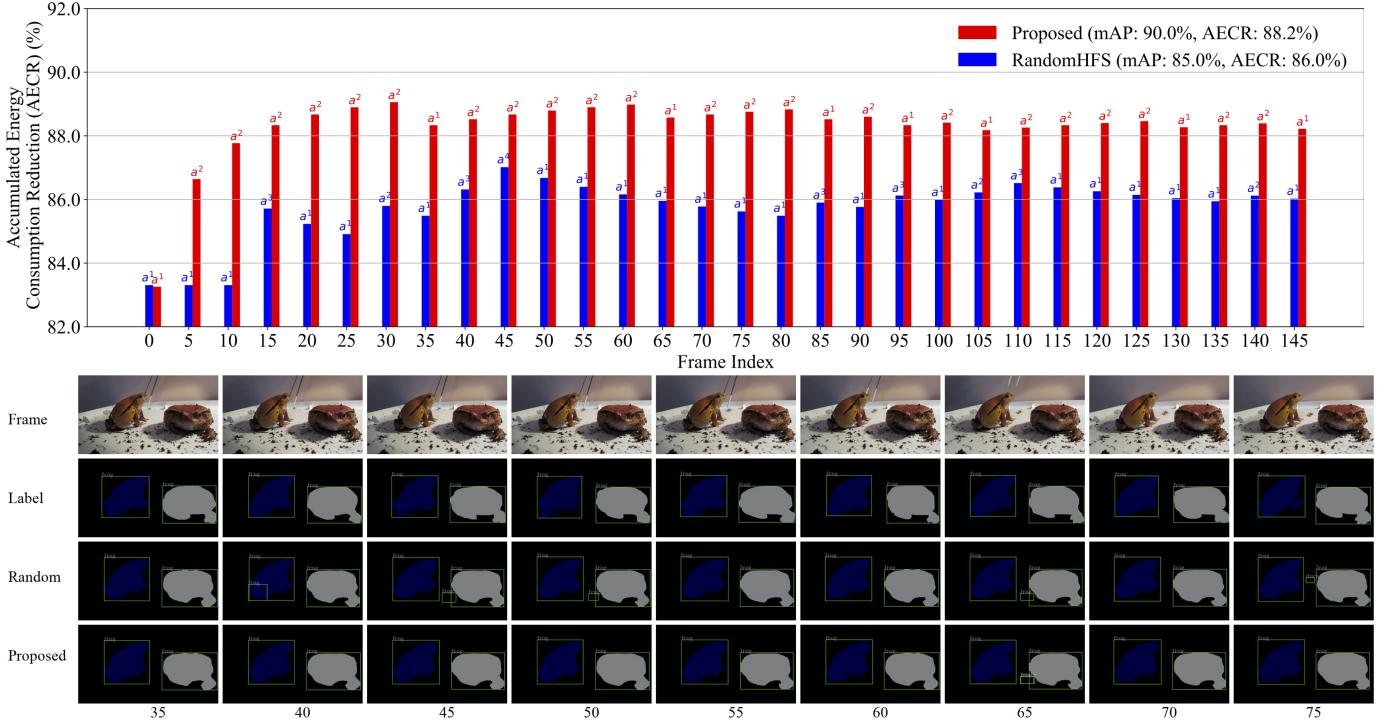
Fig. 16: A comparison between the proposed method and the *RandomHFS* method.

the global video context and thus makes better resolution decisions.

VIII. CONCLUSIONS

This paper describes an adaptive-resolution energy optimization framework for a multi-task video analytics pipeline in energy-constrained scenarios. We described a reinforcement-learning-based method to govern the operation of the video analytics pipeline by learning the best non-myopic policy for controlling the spatial resolution and temporal dynamics to globally optimize system energy consumption and accuracy. The proposed framework is applied to video instance segmentation which is one of the most challenging video analytics problems. Experimental results demonstrate that our method has better energy efficiency than all baseline methods. This framework can be applied to a wide range of computer vision pipelines with a high demand for efficient energy consumption, e.g., various embedded and Internet-of-Things applications.

ACKNOWLEDGMENTS

REFERENCES

[1] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. on Computer Vision*, 2017, pp. 2961–2969.

[2] L. Yang, Y. Fan, and N. Xu, "Video instance segmentation," in *Proc. IEEE Int. Conf. on Computer Vision*, 2019, pp. 5188–5197.

[3] Q. Wang, C. Yuan, and Y. Liu, "Learning deep conditional neural network for image segmentation," *IEEE Trans. on Multimedia*, vol. 21, no. 7, pp. 1839–1852, 2019.

[4] S. Reif, B. Herzog, P. G. Pereira, A. Schmidt, T. Büttner, T. Hönig, W. Schröder-Preikschat, and T. Herfet, "X-Leep: Leveraging cross-layer pacing for energy-efficient edge systems," in *Proc. ACM Int. Conf. on Future Energy Systems*. Virtual Event Australia: ACM, Jun. 2020, pp. 548–553. [Online]. Available: https://dl.acm.org/doi/10.1145/3396851.3402924

[5] R. P. Dick, L. Shang, M. Wolf, and S.-W. Yang, "Embedded intelligence in the Internet-of-Things," *IEEE Design & Test*, vol. 37, no. 1, pp. 7–27, 2019.

[6] E. S. Lubana and R. P. Dick, "Digital Foveation: An energy-aware machine vision framework," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 11, pp. 2371–2380, Nov. 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8493507/

[7] E. S. Lubana, V. Aggarwal, and R. P. Dick, "Machine Foveation: An application-aware compressive sensing framework," in *2019 Data Compression Conf. (DCC)*. IEEE, 2019, pp. 478–487.

[8] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *Proc. IEEE Int. Conf. on Computer Vision*, 2015, pp. 2758–2766.

[9] X. Zhu, Y. Xiong, J. Dai, L. Yuan, and Y. Wei, "Deep feature flow for video recognition," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2017, pp. 2349–2358.

[10] V. Radu, C. Tong, S. Bhattacharya, N. D. Lane, C. Mascolo, M. K. Marina, and F. Kawsar, "Multimodal deep learning for activity and context recognition," *Proc. ACM Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 1, no. 4, pp. 1–27, Jan. 2018. [Online]. Available: https://dl.acm.org/doi/10.1145/3161174

[11] P. Kulkarni, D. Ganesan, P. Shenoy, and Q. Lu, "*SensEye*: a multi-tier camera sensor network," in *Proc. ACM Int. Conf. on Multimedia*. Hilton, Singapore: ACM Press, 2005, p. 229. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1101149.1101191

[12] R. LiKamWa, B. Priyantha, M. Philipose, L. Zhong, and P. Bahl, "Energy characterization and optimization of image sensing toward continuous mobile vision," in *Proceeding Int. Conf. on Mobile Systems, Applications, and Services*. Taipei, Taiwan: ACM Press, 2013, p. 69. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2462456.2464448

[13] Z. Wang, Q. Hao, F. Zhang, Y. Hu, and J. Cao, "A variable resolution feedback improving the performances of object detection and recogni-

tion," *Proc. Institution of Mechanical Engineers Part I J. of Systems & Control Engineering*, p. 095965181772140, 2017.

[14] X. Feng, V. Swaminathan, and S. Wei, "Viewport prediction for live 360-degree mobile video streaming using user-content hybrid motion tracking," *Proc. ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 3, no. 2, pp. 1–22, Jun. 2019. [Online]. Available: https://dl.acm.org/doi/10.1145/3328914

[15] 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 *Proc. IEEE Conf. on Computer Vision and pattern recognition*, 2017, pp. 2462–2470.

[16] Y. Wang, M. Dong, J. Shen, Y. Wu, S. Cheng, and M. Pantic, "Dynamic face video segmentation via reinforcement learning," in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 2020, pp. 6959–6969.

[17] Y. Wang, B. Luo, J. Shen, and M. Pantic, "Face mask extraction in video sequence," *Int. J. of Computer Vision*, vol. 127, no. 6-7, pp. 625–641, 2019.

[18] Y. Wang, M. Dong, J. Shen, Y. Lin, and M. Pantic, "Dilated convolutions with lateral inhibitions for semantic image segmentation," *arXiv preprint arXiv:2006.03708*, 2020.

[19] B. Luo, J. Shen, S. Cheng, Y. Wang, and M. Pantic, "Shape constrained network for eye segmentation in the wild," in *The IEEE Winter Conf. on Applications of Computer Vision*, 2020, pp. 1952–1960.

[20] Y.-S. Xu, T.-J. Fu, H.-K. Yang, and C.-Y. Lee, "Dynamic video segmentation network," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2018, pp. 6556–6565.

[21] S. Zhang, X. Zhu, Z. Lei, H. Shi, X. Wang, and S. Z. Li, "S3fd: Single shot scale-invariant face detector," in *IEEE Int. Conf. on Computer Vision*. Venice: IEEE, Oct. 2017, pp. 192–201. [Online]. Available: http://ieeexplore.ieee.org/document/8237292/

[22] S. Yang, P. Luo, C. C. Loy, and X. Tang, "WIDER FACE: A face detection benchmark," in *IEEE Conf. on Computer Vision and Pattern Recognition*. Las Vegas, NV, USA: IEEE, Jun. 2016, pp. 5525–5533. [Online]. Available: http://ieeexplore.ieee.org/document/7780965/

[23] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[24] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," *arXiv preprint arXiv:1509.06461*, 2015.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. on computer vision and pattern recognition*, 2016, pp. 770–778.

[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[27] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015. [Online]. Available: http://www.nature.com/articles/nature14236