The Blind and the Elephant: A Preference-aware Edge Video Analytics Scheduler for Maximizing System Benefit

Liang Zhang
Shanghai Jiao Tong University
Shanghai, China
zhangliang@sjtu.edu.cn

Hongzi Zhu*
Shanghai Jiao Tong University
Shanghai, China
hongzi@cs.sjtu.edu.cn

Yunzhe Li
Shanghai Jiao Tong University
Shanghai, China
yunzhe.li@sjtu.edu.cn

Jiangang Shen
Shanghai Jiao Tong University
Shanghai, China
sjg19970410@sjtu.edu.cn

Minyi Guo
Shanghai Jiao Tong University
Shanghai, China
guo-my@cs.sjtu.edu.cn

ABSTRACT

Video analytics is the killer workload in edge computing, which involves the scheduler's complex decisions to balance analysis performance (latency and accuracy) and resource consumption (network, computation, and energy). Traditional schedulers address this as a single-objective optimization problem with fixed weights, unable to precisely capture unknown system preferences due to intricate pricing rules across various service levels and resource costs, consequently leading to suboptimal system benefit like monetary gain. In this paper, we propose a Bayesian optimization-driven multi-objective scheduler, PaMO, that can proactively explore the system pricing preference by pairwise comparing outcome vectors of all objectives. Moreover, PaMO designs a heuristic scheduling algorithm with a zero-delay jitter guarantee to avoid performance degradation caused by resource contention and uses a revised Bayesian optimization algorithm to make video configuration and scheduling decisions. Experiments on real video analytics workloads show that PaMO can achieve up to 53.9% benefit gain compared to state-of-the-art scheduling methods.

CCS CONCEPTS

• **Computer systems organization** → *Client-server architectures*;
• **Theory of computation** → **Scheduling algorithms**.

KEYWORDS

Adaptive configuration, Video analytics, Edge computing, Periodic scheduling, Bayesian optimization, Multi-objective optimization

ACM Reference Format:
Liang Zhang, Hongzi Zhu, Yunzhe Li, Jiangang Shen, and Minyi Guo. 2024. The Blind and the Elephant: A Preference-aware Edge Video Analytics Scheduler for Maximizing System Benefit. In *The 53rd International Conference on Parallel Processing (ICPP '24), August 12–15, 2024, Gotland, Sweden.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3673038.3673081

*Hongzi Zhu is the corresponding author.

ICPP '24, August 12–15, 2024, Gotland, Sweden

ACM ISBN 979-8-4007-1793-2/24/08
https://doi.org/10.1145/3673038.3673081

1 INTRODUCTION

Nowadays, numerous intelligent applications integral to our daily lives rely on video analytics for prompt decision-making. For instance, the advanced map navigation functionality [1] relies on the real-time analysis of the road condition videos, and the video analysis of the chemical workshop [8] helps managers to identify equipment and personnel security risks. In such application scenarios, video streams collected from distributed cameras are scheduled to an edge video analytics (EVA) system consisting of specific servers located at the edge of the network. Such an EVA system relies on the scheduler to configure video analytics parameters and allocate analytics requests to servers to provide timely and reliable results to users while maximizing the system benefit.

However, it is challenging to make optimal EVA schedule due to the intricate and uncertain relationship between the system benefit and scheduling variables like video parameters and server allocation strategies. Specifically, the system benefit is affected by both the service prices and the system costs. However, diverse pricing rules exist in the system, such as tiered electricity or network traffic prices across different areas or network operators [29], differentiated rental prices for heterogeneous servers [2], and dynamic pricing based on the quality of service (QoS) metrics [30]. Moreover, the resource usage and the quality of service on different scheduling decisions are also difficult to estimate due to potential resource contentions and ever-changing video contents. The above reasons make it challenging for the scheduler to estimate total system benefit accurately across different scheduling configurations, thus hindering its ability to make optimal decisions to maximize system benefit.

In the literature, existing EVA schedulers typically begin by modeling the correlation between various QoS and resource usage metrics, and scheduling variables using polynomial regression techniques, and then solve a single-objective optimization problem defined by the linear weighting of these metrics to make scheduling decisions that maximize system benefit. Among them, the adaptive configuration solutions [13, 19, 32–34] adjust frame sampling rate and resolution of video streams to reduce resource usage while sacrificing less analysis accuracy for optimizing system benefit. It

takes advantage of the robustness of neural networks for small input changes or specific scenes to save resources for other complex video streams that require more resources. The solutions to optimize server allocation strategies [19, 33, 34] adjust mapping relationships between video streams and multiple edge devices to avoid resource contention and improve the quality of service and resource utilization. *However, these solutions lack a thorough discussion for selecting optimal weights or restructuring the system pricing preference on different metrics,* i.e., *the differentiated impact of diverse metrics on system benefit caused by complex and unknown pricing rules.* Some existing classical weight definitions [10], such as Equal weights, Rank order centroid (ROC) weights, Rank-sum (RS) weights, and Pseudo-weights, are not flexible enough to adapt to diverse and dynamic EVA system environments. In summary, no existing EVA scheduler can perceive system pricing preference and make optimal scheduling decisions for maximizing system benefit.

In this paper, we propose a Bayesian optimization-driven multi-objective scheduler for edge video analytics, called PaMO, which interacts with the system decision-maker in advance or in the optimization loop to capture system pricing preference on different metrics or objectives. Specifically, we first fit the outcome function of each objective, including latency, accuracy, network bandwidth, computing power, and energy consumption, using profiling data of video analytics workloads. Then, we train a Gaussian-process-based preference model according to pairwise comparison data of different outcome vectors obtained from the decision maker to characterize the priority of all objectives. Finally, we use the Bayesian optimization framework to iteratively update outcome and preference models and recommend better solutions aimed at maximizing system benefits with the help of an anti-noise acquisition function in each iteration until convergence.

Three main challenges are encountered when designing PaMO. First, it is challenging to quantify the system pricing preference on diverse optimization objectives due to its influence by complex pricing rules. Even for a simple linear weighted of multiple objectives, extensive expert knowledge or experimental measurements are required to determine precise weight values. Alternatively, we ask the decision maker some simple comparative questions on the dataset of pairwise outcome vectors to extract pricing preference information about different objectives. This approach transforms the quantitative task into a qualitative one, alleviating the decision-maker's burden.

Second, it is non-trivial to formulate the relationship between the end-to-end latency and video configuration and scheduling schemes due to potential performance interference among consecutive incoming frames on an overloaded edge server. To tackle this difficulty and accurately fit the outcome model of the latency metric, we propose a group-based heuristic scheduling algorithm that proactively avoids scheduling streams with potential resource conflicts onto the same node to achieve zero jitter of each frame and improve latency stability to facilitate modeling.

Last, it is time-consuming and resource-intensive to search for optimal solutions by traversing an exponential search space formed by three-dimensional decision variables. Specifically, considering that M video streams, whose resolution and frame sampling rate have C_r and C_f knobs respectively, need to be mapped into N edge nodes, the search space size becomes $\left(N * C_r * C_f\right)^M$. To reduce the scheduler's decision response time and save valuable resources at the edge, we propose a revised Bayesian optimization algorithm with batch noisy expected improvement (qNEI) acquisition function to solve the multi-objective optimization problem defined by pre-trained outcome and benefit models efficiently. The qNEI function can reduce the interference caused by inaccurate models to the solution search and expedite problem convergence.

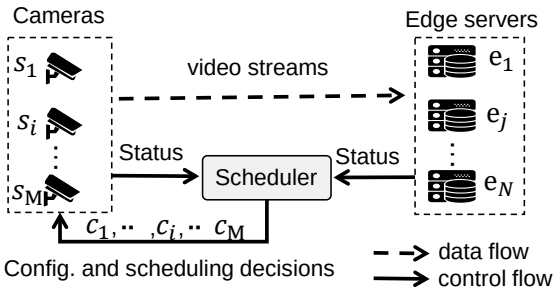


Figure 1: A typical edge video analytics system.

We evaluate the performance of PaMO using video object detection workloads on several NVIDIA Jetson devices. The experimental results show that PaMO greatly enhances system benefits by accurately modeling outcome and preference functions. Moreover, it exhibits strong robustness across various experimental setups. Specifically, PaMO can achieve near-optimal system benefits with low relative errors across diverse system pricing preferences and configurations. Compared with state-of-the-art methods, *i.e.*, JCAB and FACT, PaMO improves system benefit by up to 53.9% and up to 26.5%, respectively.

We highlight the main contributions made in this paper as follows: 1) We formulate a multi-objective optimization problem for EVA scheduler, which includes various outcome functions of objectives, the pricing preference function that reflects the system benefit, and a zero-jitter constraint that is proven to avoid resource contention among video streams. 2) We propose a pairwise comparison-based method to learn the system pricing preference on different objectives and train a Gaussian process model using comparison results to surrogate the preference function. 3) We design a group-based heuristic scheduling algorithm to meet the zero-jitter constraint and propose a revised Bayesian optimization algorithm with the qNEI acquisition function to accelerate the solution of the above optimization problem.

2 PRELIMINARIES

2.1 Video Analytics at the Edge

In edge video analytics scenarios, numerous video streams generated by cameras are transmitted to nearby edge servers via the wireless network for analysis, as shown in Figure 1. Specifically, each camera preprocesses and encodes the original video stream before transmitting it to the designated edge server [1]. The edge server decodes received video streams and processes them using

[1]Modern smart cameras, like [11], typically incorporate both hardware and software related to codecs, in addition to a limited computing capability that support extra preprocessing tasks like resolution adjustments.

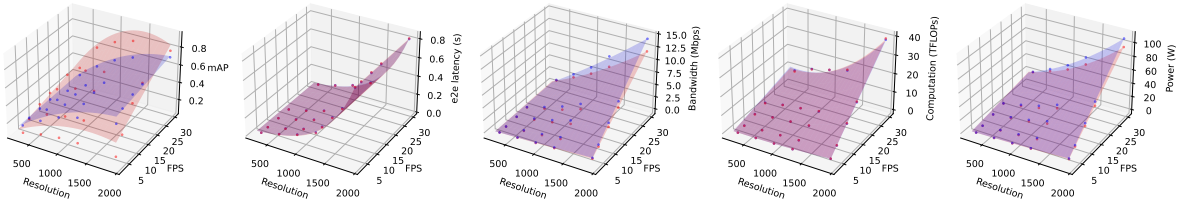


Figure 2: The performance and resource consumption of two video clips from the MOT16 dataset under different configurations. The network bandwidth remained constant at 100Mbps throughout this experiment. Two video clips exhibit a consistent pattern of change in accordance with the configuration adjustments.

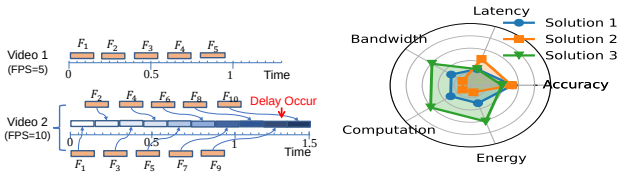
the DNN models' inference service. The scheduler periodically collects performance and resource information including the end-to-end latency and analysis accuracy of video streams, and the network bandwidth, computation and energy usage of all edge servers. According to these real-time data, the scheduler adjusts configuration and scheduling decisions for each video stream, including the video's resolution, frame sampling rate, and the target device's IP address, to optimize analysis performance and resource efficiency. The cameras receive this decision from the scheduler and execute it. This paper assumes that edge servers have uniform resource conditions excluding uplink bandwidth variations and run DNN models with the same structure to provide analytics services. Each edge server can receive multiple video streams simultaneously, and a video stream can also be scattered and scheduled to various servers.

2.2 Adaptive Configuration and Scheduling

To better model the optimization problem of the scheduler, we explore the relationship between video configuration and scheduling decisions and the analytical performance and resource consumption through a series of experiments. First, we profile the performance and resource consumption of two video clips from the MOT16 dataset [21] under various resolution and frame sampling rate configurations. The actual measured data and the fitted surface are shown in Figure 2. The result indicates that various video streams exhibit a consistent pattern of change in accordance with the configuration adjustments, which motivates us to use the similar formula to model different optimization objectives. Moreover, the second subgraph of Figure 2 shows that the end-to-end latency remains constant irrespective of the frame sampling rate when ample resources are available. However, latency experiences an unexpected increase when resource contention arises among multiple streams or frames on a single edge server, as illustrated in Figure 3(a). This unexpected latency accumulation becomes more evident as the frame sampling rate increases. Therefore, the scheduler must conscientiously account for the computing power limits of a single machine during the configuration and scheduling of video streams.

2.3 Necessitate Preference Exploration

Figure 2 shows that larger resolutions and frame rates lead to higher accuracy, but they also take longer to process and consume more resources. Due to this inherent conflicts among diverse objectives,



(a) Latency accumulation caused by resource contention

(b) Pareto optimal solutions

Figure 3: Examples of resource contention and Pareto optimal solutions.

it is impossible to find an optimal solution that minimizes all objectives in the multi-objective optimization problem of video stream analysis. Therefore, we use Pareto optimal solutions to represent those solutions for which no improvement can be made in any of the objectives without causing degradation in at least one of the other objectives. Mathematically, we define a feasible solution x_1 dominate another solution x_2 if $\forall i \in \{1, \ldots, k\}, f_i(x_1) \leq f_i(x_2)$, and $\exists i \in \{1, \ldots, k\}, f_i(x_1) < f_i(x_2)$. A solution x^* (the corresponding outcome $f(x^*)$) is called Pareto optimal if no other solution dominates it. There is a Pareto optimal set for video analytics' multi-objective optimization problem. Figure 3(b) illustrates the outcomes of various objectives for three Pareto optimal solutions, with outcomes normalized to the interval (0,1) for ease of comparison. The results indicate that none of the three solutions exhibits dominance over the others. For instance, while Solution 1 excels over Solution 3 in bandwidth, computation, and energy dimensions, its accuracy falls short, preventing it from dominating Solution 3. Consequently, schedulers need to learn a preference function based on how different objectives affect the system benefit to comprehensively assess the benefits of distinct Pareto optimal solutions and thereby determine an exact optimal solution.

3 PROBLEM FORMULATION

In the context of video stream analysis, the scheduler needs to determine the optimal video configuration and server allocation plan that aligns with system preferences across various objectives for maximizing system benefit. This scheduling problem can be abstracted as a multi-objective optimization problem. Mathematically, the outcome function $f_i : \mathbb{R}^d \rightarrow \mathbb{R}, i \in 1, ..., k$ denotes the benefit associated with the kth objective in different configurations,

while the preference function $g : \mathbb{R}^k \rightarrow \mathbb{R}$ represents the extent to which different objectives affect the system benefit. Consequently, the scheduling optimization problem of the EVA scheduler can be expressed as the formulation:

$$\max_{x \in \chi} g\left(f_1(x), \ldots, f_k(x)\right) \tag{1}$$

where $\mathcal{X} \subset \mathbb{R}^d$ is solution domain consisting of all feasible resolutions, frame sampling rates, and the allocated edge servers (i.e., $d = 3$). Our problem focus on five optimization objectives (i.e.,$k = 5$) including the end-to-end latency, accuracy, network bandwidth, computation and resource consumption. Next, we describe the variables related to the optimization problem and formulate different outcome functions and constraints.

Variable Definition. We consider the video analytics system containing M' video sources and N edge servers who have equivalent computing power (heterogeneous servers can be virtualized as multiple homogeneous VMs or containers). M' video sources might include M^* high-rate streams whose the worst-case processing time of a single frame on one server is greater than their period (the inverse of frame sampling rate), e.g., *Video 2* in Figure 3(a). To avoid additional queue delay caused by conflict between consecutive frames for these high-rate streams, we split each high-rate stream by sampling periodically into $\lceil s_i/(1/p_i) \rceil = \lceil s_i p_i \rceil$ new streams, where s_i and p_i are the frame sampling rate and processing time on a server of video stream i. Consequently, the scheduler need to deal with $M = M' - M^* + \sum_{i=1}^{M^*} \lceil s_i p_i \rceil$ streams that no resource contention when scheduling it to a node alone. We denote these M periodic streams as $\mathcal{T} = \{\tau_1, \tau_2, \ldots, \tau_M\}$, where each stream is characterized by a tuple $\{T_i, r_i, p_i\}$, where T_i is the inter-arrival period (the inverse of frame rate,$T_i = 1/s_i$), r_i is the resolution, and p_i is the average processing time of all frames from the stream i on a server. Besides, let $q_i \in \{0, 1, ..., N\}$ represent the stream i will be allocated to the q_ith server.

Outcome Function Formulation. Drawing insights from Figure 2, we model the relationship between different optimization objectives and frame sampling rate or resolution through either multivariable linear regression or polynomial regression. Specifically, the outcome function of accuracy, network bandwidth, and computing power consumption can be expressed as Equation 2 and 3, respectively. Here, $\epsilon(\cdot)$ is a linear function and $\theta(\cdot)$ is a linear or quadratic function depends on specific experimental data.

$$f_{acc} = \frac{1}{M} \sum_{i=0}^{M} \theta_{\text{acc}}\left(r_i\right) \epsilon_{\text{acc}}\left(s_i\right) \tag{2}$$

$$f_{\text{net}}, f_{\text{com}} = \sum_{i=0}^{M} \theta_{\text{net}}\left(r_i\right) \epsilon_{\text{net}}\left(s_i\right), \sum_{i=0}^{M} \theta_{\text{com}}\left(r_i\right) \epsilon_{\text{com}}\left(s_i\right) \tag{3}$$

In the video analytics pipeline, the total energy consumed is the sum of the energy consumed in the communication and computing process, as shown in Equation 4. We use the total power, i.e., the energy consumed by all video streams in 1 s, to compare the energy efficiency of different configurations. Here, γ represent the transmission energy consumption per bit ($\gamma = 0.5 \times 10^{-5}$ (J) is consistent with existing work [34]). $\theta_{bit}(r_i)$ and $\theta_{eng}(r_i)$ are quadratic functions that represent the data size (bits) and energy consumption (J) of a video frame with resolution r_i, respectively.

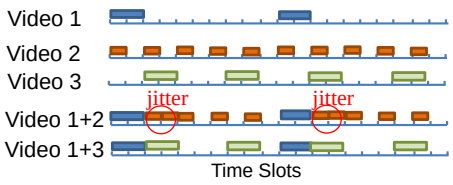


Figure 4: A delay jitter example due to poor scheduling.

$$f_{eng} = \sum_{i=0}^{M} (\gamma \theta_{bit}(r_i) \epsilon_{\text{bit}}\left(s_i\right) + \theta_{eng}(r_i) \epsilon_{\text{eng}}\left(s_i\right)) \tag{4}$$

The end-to-end latency of the video stream includes the computing latency $l_{com} = \theta_{lcom}(r_i)$ and network transmission latency $l_{net} = \theta_{bit}(r_i)/B_{q_i}$, denoted as Equation 5. Here, $p_i = \theta_{lcom}(r_i)$ is a quadratic function that represents the computing time of a video frame with the resolution r_i and B_{q_i} represents the uplink bandwidth of the edge server q_i.

$$f_{ltc} = \frac{1}{M} \sum_{i=0}^{M} (\theta_{lcom}(r_i) + \theta_{bit}(r_i)/B_{q_i}) \tag{5}$$

Constraint Formulation. To mitigate queuing delays induced by computational overload, we add a constraint (denoted by Equation 6) for video analytics optimization problem, i.e., the total processing time for all frames scheduled to a server (potentially originating from multiple video streams) within a 1-second interval should be less than 1 second. Moreover, scheduling video streams with distinct frequencies and processing time per frame to the same server may lead to delay jitter (as shown in Figure 4), i.e., the currently arrived frame has to be postponed because a previous frame occupies the server. The delay jitter introduces additional and irregular end-to-end latencies, so we introduce the constraint 2 (Equation 7) to ensure that the scheduling decision obtained by solving the optimization problem will not cause delay jitter. Here, $gcd(\cdot)$ represents the greatest common divisor of the periods of all streams scheduled to the same node. We explain the rationality of constraint 2 by *Theorem 1* and its proof.

$$Const1: \sum_{\{i|q_i=j\}} p_i s_i \leq 1 \quad \forall \quad j \in [1, \ldots, N] \tag{6}$$

$$Const2: \sum_{\{i|q_i=j\}} p_i \leq \gcd(\{T_i\}_{\{i|q_i=j\}}) \quad \forall \quad j \in [1, \ldots, N] \tag{7}$$

Theorem 1. *A sufficient condition of K periodic streams can be scheduled on the same server with zero delay jitter can be stated as:*

$$\sum_{i}^{K} p_i \leq \gcd(T_1, T_2, ..., T_K) \tag{8}$$

Proof. Let $g = gcd(T_1, T_2, ..., T_K)$ and the start times $o(\tau_1) = 0$ and $o(\tau_k) = \sum_{i=1}^{k-1} p_i, k \in [1, ..., K]$. The stream τ_1 is executed in a subset of the set I_1 of of intervals, defined by $[lg, lg + p_1 - 1], l \in \mathbb{Z}$, and stream τ_k is executed in a subset of the set I_k of intervals. defined by $[lg + \sum_{i=1}^{k-1} p_i, lg + \sum_{i=1}^{k-1} p_i + p_k - 1], l \in \mathbb{Z}$. If $\sum_{i}^{K} p_i \leq g$ holds, $lg + \sum_{i=1}^{k-1} p_i + p_k - 1 \leq lg + g - 1$, i.e., in the lth cycle of length g, the completion time of the last frame is less than the arrival time of the first frame of the next cycle $l + 1$. Hence, no intervals of

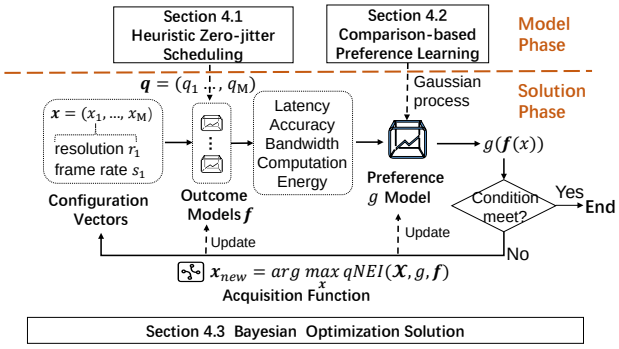


Figure 5: The overall framework of our solution.

$I_k, k \in 1, .., K$ overlap when $\sum_i^K p_i \leq g$ holds, which proves the sufficiency. □

4 ALGORITHM DESIGN

This section describes our solution, PaMO, to multi-objective optimization problem on edge video analytics scheduling, and its overall framework is shown in Figure 5. Firstly, we fit outcome functions of all performance and resource metrics based on the profiling data and formulation in Section 3. We also propose a heuristic algorithm to generate a scheduling decision that meets the computing power and zero delay jitter constraints (*Const*1 in Equation 6 and *Const*2 in Equation 7) while minimizing average communication latency. Based on this scheduling algorithm, we employ a black-box Gaussian process model to reconfigure the outcome function of end-to-end latency. Secondly, PaMO explores the system preference based on pairwise comparison of outcome vectors. The Gaussian process model trained by comparison results is used to surrogate the preference function of the system, and the EUBO acquisition function is used to select more important outcome vectors for accelerating the convergence of the training process. Finally, PaMO solves the multi-objective problem of edge video analytics following a revised Bayesian optimization framework. It iteratively updates outcome and preference models and recommends better solutions using an anti-noise acquisition function qNEI. Through iterative refinement, PaMO gradually converges towards the optimal solution of the multi-objective optimization problem.

4.1 Heuristic Zero-jitter Scheduling

When the frame rate and resolution are known, the scheduling algorithm finds a scheduling scheme that meets the computing power (*Const*1 in Equation 6) and zero delay jitter constraints (*Const*2 in Equation 7). Actually, we show that *Const*2 is a sufficient condition for *Const*1 (*Theorem 2*), which means that the algorithm only needs to ensure that the scheduling result satisfies *Const*2.

Theorem 2. *Const*2 *is a sufficient condition for Const*1.

Proof. Let $|\{i|q_i = j\}| = K_j$ in *Const*1 and *Const*2, we need to prove that when $\sum_{i=0}^{K_j} p_i \leq \gcd(T_1, ...T_{K_j})$ holds, $\sum_{i=0}^{K_j} p_i s_i \leq 1$ must also hold for all $j \in [1, ..., N]$. From the definition of the greatest common divisor, we know that for all $k \in [1, .., K_j]$, the following inequality is always true:

$$gcd(T_1, T_2, ..., T_{K_j}) \leq min\{T_1, T_2, ..., T_{K_j}\} \leq T_k$$

Furthermore, due to $\sum_{i=0}^{K_j} p_i \leq \gcd(T_1, ...T_{K_j})$ holds,

$$\sum_{i=0}^{K_j} p_i s_i = \sum_{i=0}^{K_j} \frac{p_i}{T_i} \leq \sum_{i=0}^{K_j} \frac{p_i}{gcd(T_1, T_2, ..., T_{K_j})} \leq 1$$

This completes the proof of the theorem. □

The discretization of Const2 makes it challenging to be satisfied in the conventional convex optimization process, so we design a group-based heuristic algorithm motivated by *Theorem 3* to proactively generate a scheduling solution that aligns with the requirements of Const2. The overall workflow is shown in Algorithm 1. Specifically, it first sorts all streams based on their periods in ascending order (*line 1*), and then computes each stream's priority according to the number of streams whose period is their period's divisor (*line 2*). The priority mechanism increases the probability of finding a feasible schedule. Next, the algorithm adjusts the order of the streams in ascending order of priority and get the final order (*line 3*). We traverse all streams in the final order and allocate them to their respective groups, ensuring that all streams within the same group meet conditions (a) and (b) in *Theorem 3* (*line 5-19*). Finally, we solve a general assignment problem for minimizing the average communication latency (*line 20*) by a classical Hungarian algorithm to obtain optimal scheduling vector $\mathbf{q}$.

Theorem 3. *Suppose the minimum period of K periodic streams scheduled into the same server is* $T_{min} = min\{T_1, ..., T_K\}$. *The condition (a)* $T_i = t * T_{min} (t \in \mathbf{N}^+)$ *for all* $i \in \{1, ..., K\}$ *and (b)* $\sum_{i=1}^{K} p_i \leq T_{\min}$ *are sufficient for Const2.*

Proof. If the condition (a) and (b) holds, we have $gcd(T_1, ..., T_K) = T_{min}$ and $\sum_{i=1}^{K} p_i \leq T_{\min} = gcd(T_1, ..., T_K)$, i.e., Const2 is satisfied. □

4.2 Comparison-based Preference Learning

Considering the significant impact of system preferences on identifying the optimal solution within the Pareto frontier of multi-objective optimization problems, as analyzed in Section 2, this section delves into the precise modeling of system preferences, i.e., the formulation of preference function. Compared with the traditional preference function formula given by decision-makers who need to have expert knowledge, our method is more user-friendly. It only requires decision-makers to compare the income vector given by the system in pairs, and build the preference function based on Gaussian process according to the comparison results.

Preference Modeling. Assume that $\mathcal{Y}$ is the outcome space of our video analytics problem, in which each element $\boldsymbol{y}$ in $\mathcal{Y}$ is an outcome vector consisting of the values of k outcome functions, i.e. $\boldsymbol{y} = (y_1, y_2, \ldots, y_k)$. We choose V pairs of outcome vectors from the outcome space $\mathcal{Y}$ and asking decision-makers to compare the two outcome vectors in each outcome pair for identifying the better one that in line with their preferences. The comparison results of decision-maker feedback can form a preference set $\mathcal{P}_V = \left\{(\boldsymbol{y}_v^{(1)} \succ \boldsymbol{y}_v^{(2)})\right\}_{v=1}^{V}$ where $\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)} \in \mathcal{Y}$ and $\boldsymbol{y}^{(1)} \succ \boldsymbol{y}^{(2)}$ means

Algorithm 1: Group-based heuristic scheduling

Input : A stream set $\mathcal{T} = \{\tau_1, \tau_2, \ldots, \tau_M\}$, $\tau_i = \{T_i, r_i, p_i\}$, a server set $\mathcal{N} = \{n_1, n_2 ..., n_N\}$ and their uplink bandwidth B_{n_j}

Output : The scheduling vector $\mathbf{q} = [q_1, q_2, .., q_M], q_i \in \mathcal{N}$

Sort all streams in $\mathcal{T}$ by T_i in ascending order: $\mathcal{T} \rightarrow \mathcal{T}'$;
Compute the priority of each stream in set $\mathcal{T}'$: $priority(\tau_i') = I_i = \sum_{j=1}^{i-1} \mathbb{I}(T_i \% T_j = 0)$;
Sort all streams in $\mathcal{T}'$ by I_i in ascending order: $\mathcal{T}' \rightarrow \mathcal{T}''$;
Initial the stream group set $\mathcal{G} = \{G_1, .., G_N\}, G_j = \phi \ for \ all \ j \in [1, ..., N]$;
for $\tau_i'' = \{T_i'', r_i'', p_i''\} \in \mathcal{T}''$ **do**
  for $G_j \in \mathcal{G}$ **do**
    if $G_j = \phi$ **then**
      $G_j = G_j \cup \tau_i''$
    end
    $T_{min} = \min\{T_k | \tau_k \in G_j\}$;
    if $T_i'' = t * T_{min}$ ***and*** $\sum_{\tau_k \in G_j} p_k + p_i'' \leq T_{min}$ **then**
      $G_j = G_j \cup \tau_i''$;
      break;
    end
    if $G_j = G_N$ **then**
      No feasible grouping scheme;
    end
  end
end
Mapping groups $\mathcal{G}$ to servers $\mathcal{N}$ by solving a general assignment problem to optimize the following objective: $\min_{\mathbf{q}} \sum_{G_j \subset \mathcal{G}} \sum_{i \in G_j} \frac{\theta_{bit}(r_i)}{B_{q_i}}$;
return $\mathbf{q} = [q_1, q_2, .., q_M]$

the decision-maker prefers $\boldsymbol{y}^{(1)}$ to $\boldsymbol{y}^{(2)}$. Following previous works [6, 17], we use the following likelihood function to capture the joint probability of the preference relation $\boldsymbol{y}_v^{(1)} \succ \boldsymbol{y}_v^{(2)}$:

$$p(\mathcal{P} \mid \boldsymbol{g}) = \prod_{v=1}^{V} p\left(\boldsymbol{y}_v^{(1)} \succ \boldsymbol{y}_v^{(2)} \mid g\left(\boldsymbol{y}_v^{(1)}\right), g\left(\boldsymbol{y}_v^{(2)}\right)\right) \tag{9}$$

with

$$p\left(\boldsymbol{y}_v^{(1)} \succ \boldsymbol{y}_v^{(2)} \mid g\left(\boldsymbol{y}_v^{(1)}\right), g\left(\boldsymbol{y}_v^{(2)}\right)\right) = \Phi\left(\frac{g\left(\boldsymbol{y}_v^{(1)}\right) - g\left(\boldsymbol{y}_v^{(2)}\right)}{\sqrt{2}\lambda}\right)$$

where λ is a hyperparameter and Φ is the cumulative distribution function of the standard normal distribution.

Then we can formulate the posterior probability of $\boldsymbol{g}$ based on Bayes' theorem, $p(\boldsymbol{g} \mid \mathcal{P}) = \frac{p(g)p(\mathcal{P}|\boldsymbol{g})}{p(\mathcal{P})}$. Finally, Laplace approximation is used to find a GP approximation of the function $\boldsymbol{g}$, which is characterized by a mean function $\boldsymbol{\mu}^g$ and a covariance function $\boldsymbol{K}^g$, i.e., $\boldsymbol{g} \sim GP(\boldsymbol{\mu}^g, \boldsymbol{K}^g)$. We recommend readers to see more details about formula derivation in [6].

Efficiency Improvement. The choice of comparison pairs significantly impacts the convergence rate during the training of Gaussian process models. To maximize training efficiency, our goal is to identify the optimal comparison pair within the intricate outcome space at each iteration of model training.

Recall that we need to recommend the best outcome vector $\boldsymbol{y}^*$ for the decision-maker to maximize benefit, i.e., $\boldsymbol{y}^* \in \text{argmax}_{\boldsymbol{y} \in \mathcal{Y}} \mathbb{E}_v[g(\boldsymbol{y})]$ where $\mathbb{E}_V$ is the conditional expectation given $\mathcal{P}_V$ ($\mathbb{E}_V[\cdot] = \mathbb{E}[\cdot \mid \mathcal{P}_V]$). Therefore, we can quantify the difference in (expected) benefit obtained by the decision-maker due to the additional comparison pair $(\boldsymbol{y}_1, \boldsymbol{y}_2)$ by the following formula:

$$AF\left(\boldsymbol{y}_1, \boldsymbol{y}_2\right) = \mathbb{E}_V\left[\max_{\boldsymbol{y} \in \mathcal{Y}} \mathbb{E}_{V+1}[g(\boldsymbol{y})] - \max_{\boldsymbol{y} \in \mathcal{Y}} \mathbb{E}_V[g(\boldsymbol{y})]\right] \tag{10}$$

with

$$\mathbb{E}_{V+1}[\cdot] = \mathbb{E}\left[\cdot \mid \mathcal{P}_V \cup \{\boldsymbol{y}_1 \succ \boldsymbol{y}_2 \quad or \quad \boldsymbol{y}_1 \prec \boldsymbol{y}_2\}\right]$$

Consequently, we can select the outcome vector pair that maximizes the acquisition function $AF(\boldsymbol{y}_1, \boldsymbol{y}_2)$ for the next iteration of model training. Furthermore, to overcome the difficulty of maximizing the acquisition function due to the nested structure of AF function in Equation 10, we use the expected benefit of the best option (EUBO) that is denoted in Equation 11 to replace Equation 10, which has been proved to be approximately equivalence [17].

$$\text{EUBO}\left(\boldsymbol{y}_1, \boldsymbol{y}_2\right) = \mathbb{E}_V\left[\max\left\{g\left(\boldsymbol{y}_1\right), g\left(\boldsymbol{y}_2\right)\right\}\right] \tag{11}$$

4.3 Bayesian Optimization Solution

In this section, we describe the overall process for determining the optimal configuration and scheduling decision that aligns with system preferences in the context of multi-objective optimization problems for video analysis, as illustrated in Algorithm 2. Our core idea involves leveraging the Bayesian optimization framework to iteratively refine both the outcome and preference models and recommend better solutions. We can reduce the number of iterations that converge to the optimal solution by carefully choosing the acquisition function, thereby mitigating the resources and time consumption expended on collecting outcome data and engaging with decision-makers to obtain comparison results.

Specifically, we first obtain outcome samples under a few configurations and a few comparison samples of outcome vectors through profiling and decision-maker interaction to initialize the outcome and preference model. Then, we use the batch Noisy Expected Improvement (*qNEI*) acquisition function to generate the current optimal recommended solution. We observe the actual outcome vector on this new solution and new comparison results through experiments and decision-maker queries for updating the outcome and preference model. The above process is iterated continuously until the optimal solution converges or reaches the maximum number of iterations.

In our problem, the design of the acquisition function necessitates addressing two key challenges: (1) Due to the unavailability of direct observations of the benefit value closely tied to system preferences, we rely on the preference model fitted by Gaussian process to calculate the benefit value with noise. So the acquisition function is required to identifying the optimal solution based on

Algorithm 2: BO-based configuration decision

Input : A stream set $\mathcal{T} = \{\tau_1, \tau_2, \ldots, \tau_M\}$, a server set $\mathcal{N} = \{n_1, n_2 ..., n_N\}$, convergence threshold δ,$MaxIterNum$

Output : The configuration vector $\boldsymbol{x} = [x_1, x_2, .., x_M], x_i = (s_i, r_i)$

(1) Outcome Function Fitting:
Initialize the configuration set $\mathbf{X} = \{\boldsymbol{x}_u\}_{u=1}^{U}$;
Obtain the outcome set $\mathbf{Y} = \{\boldsymbol{y}_u\}_{u=1}^{U}$ on the configuration set $\mathbf{X}$ by profiling and *Algorithm 1*, where $\boldsymbol{y}_u = [y_{acc}, y_{com}, y_{net}, y_{eng}, y_{lct}]$;
Fit the outcome functions $\boldsymbol{f}_0 = [f_{acc}, f_{com}, f_{net}, f_{eng}, f_{lct}]$ by GP models based on the data set $\mathcal{D}_U = \{(\boldsymbol{x}_u, \boldsymbol{y}_u) | \boldsymbol{x}_u \in \mathbf{X}, \boldsymbol{y}_u \in \mathbf{Y}\}_{u=1}^{U}$;
(2) system preference Modeling:
Constructing pairwise preference dataset $\mathcal{P}_V$:
for $t = 1, 2, ..., V$ **do**
  $(\boldsymbol{y}_1^t, \boldsymbol{y}_2^t) \leftarrow \arg\max_{(\boldsymbol{y},\boldsymbol{y}) \in \mathcal{Y} \times \mathcal{Y}} \text{EUBO}\left(\boldsymbol{y}_1, \boldsymbol{y}_2\right)$;
  $\mathcal{P}_{t+1} \leftarrow \mathcal{P}_t \cup \{\boldsymbol{y}_1^t \succ \boldsymbol{y}_2^t \quad or \quad \boldsymbol{y}_1^t \prec \boldsymbol{y}_2^t\}$
end
Learning preference model $\boldsymbol{g}_0 \sim GP(\boldsymbol{\mu}^g, \boldsymbol{K}^g)$ over $\mathcal{P}_V$
(3) Best Configuration Solving:
Initialization: $\boldsymbol{f} = \boldsymbol{f}_0$, $\boldsymbol{g} = \boldsymbol{g}_0$, $z_p = 0$;
while *true* **do**
  $\boldsymbol{x}^b = [\boldsymbol{x}_1, ..., \boldsymbol{x}_b] \leftarrow \arg\max_{\boldsymbol{x}_{1:b} \in \mathcal{X}^b} qNEI(\boldsymbol{x}, \boldsymbol{f}, \boldsymbol{g})$;
  $\boldsymbol{y}^b = [\boldsymbol{y}_1, ..., \boldsymbol{y}_b] \leftarrow Profile\_and\_Algorithm1(\boldsymbol{x}^b)$;
  Compute benefit value: $z^b = [z_1, ..., z_b] = \boldsymbol{g}(\boldsymbol{y}^b)$;
  Update $\boldsymbol{f}$ over the dataset $\mathcal{D}_U \cup \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{b}$;
  Update $\boldsymbol{g}$ over the dataset $\mathcal{P}_{V+b}$;
  Let $z = \max(z^b)$;
  if $|z - z_p| < \delta$ *or* $IterNum \geq MaxIterNum$ **then**
    return $\boldsymbol{x}'$
  end
  $z_p \leftarrow z$;
  $IterNum \leftarrow IterNum + 1$;
end

noisy observed data. (2) The acquisition function needs to find the optimal solution in the exponential search space of the problem, and each iteration during this process introduces time overhead due to observations or function calculations. A good acquisition function should speed up the iterative process to reduce the response time of the decision. For the above reasons, we choose the qNEI acquisition function defined by Equation 12 in our problem. The *qNEI* acquisition function simultaneously recommends b candidate points in each iteration to facilitate the system to observe benefit values parallelly and maximize the expected improvement with respect to the best value observed so far. See prior work [15] for the detailed derivation of the *qNEI* acquisition function.

$$qNEI(\boldsymbol{x}, \boldsymbol{f}, \boldsymbol{g}) = \int_{\boldsymbol{g}} \alpha_{\text{EI}}\left(\boldsymbol{x} \mid z^* = \max\left(z^U\right)\right) p\left(\boldsymbol{g} \mid \boldsymbol{D}_U\right) d\boldsymbol{g} \quad (12)$$

with

$$\alpha_{\text{EI}}\left(\boldsymbol{x} \mid z^*\right) = \mathbb{E}_V\left[\max\left(0, \max(z_i) - z^*\right) \mid z_i \sim g(\boldsymbol{f}(\boldsymbol{x}_i))\right]$$

where $z^U = \{g\left(f\left(\boldsymbol{x}_u\right)\right)\}_{u=1}^{U}$ represents the observed benefit values over the dataset $\mathcal{D}_U$, and $\boldsymbol{x} = \{\boldsymbol{x}_i\}_{i=1}^{b}$ is the set of b candidate points.

5 EVALUATION

In this section, we evaluate the performance of PaMO based on video object detection workloads, and compare its overall benefits against two single-objective optimization baselines. We implement our algorithms in python 3.8 and BoTorch library [3].

5.1 Experimental Setup

Hardware configuration. We use four Jetson XAVIER NX devices with 6-core CPUs@1.9 GHz, Volta GPU@1100 MHz, and 8GB RAM as edge servers and emulate a group of cameras on a desktop computer with 6 Intel i7-8700 CPUs@3.2 GHz, NVIDIA GeForce GTX1060 and 16GB memory to generate video streams. All machines run Ubuntu 20.04.6 LTS system and are connected to a 450 Mbps TP-LINK Router via WiFi.

Workload configuration. We implement video transmission and analysis between video sources and edge servers based on the Triton Inference Server framework [24]. Specifically, we develop Triton Client code on the desktop to dynamically adjust the frame rate and resolution of video streams, compress the streams, and transmit them to the target server based on decisions returned by the algorithm. On the server side, we leverage Triton's TensorRT backend to execute YOLOv8 detection models [27] trained on the COCO dataset. All videos used in the experiment are sourced from the MOT16 dataset [21]. We use mean average precision (mAP) to evaluate the accuracy of the object detection task.

System benefit metric. We define the system benefit for evaluation as the negative weighted L1 distance between actual outcomes and utopian outcomes of all optimization objectives, which is a higher-is-better metric as defined in Equation 13.

$$U = -\|\boldsymbol{y} - \boldsymbol{y}^*\|_1 \triangleq -\sum_{i \in \{lct,acc,net,com,eng\}} w_i |y_i - y_i^*| \quad (13)$$

where $\boldsymbol{y}$ is the normalized outcome vector. $\boldsymbol{y}^*$ is the utopian outcome vector, which consists of the best outcomes obtained by single-objective optimization. The utopian outcome vector is unattainable in practice due to conflicts between different objectives. We adjust the weight vector $w = [w_i]_{i \in \{lct,acc,net,com,eng\}}$ to construct diverse preference functions, facilitating the comparison of various solutions.

Baselines. We compare PaMO with the following baselines:

- JCAB [34] uses Lyapunov optimization and First-Fit algorithm to make video configuration (resolution and frame sampling rate) and scheduling decision. It targets to maximize the linear weighting function of accuracy and energy consumption.
- FACT [19] uses block coordinate descent (BCD) algorithm to minimize the weighted sum of latency and accuracy by adjusting resolution and server allocation. This method do not consider energy and network bandwidth consumption.

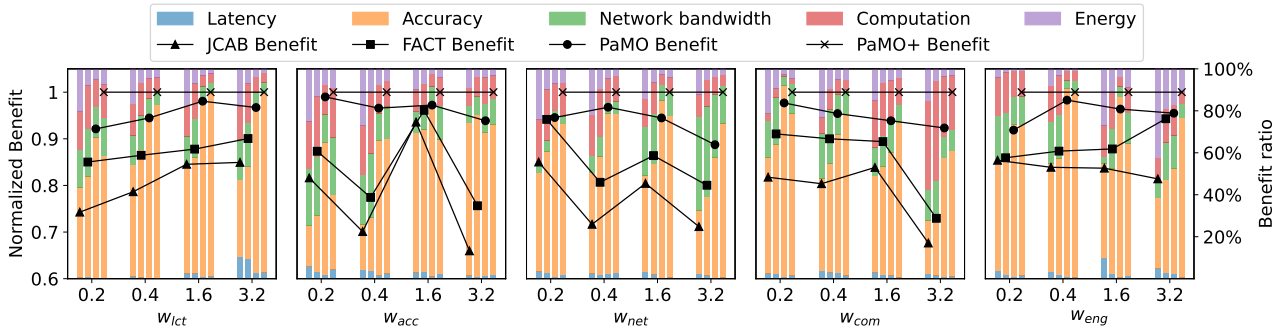


Figure 6: Normalized benefit [2]comparison across different preference functions.

- PaMO+: a variant of PaMO that uses true preference functions (i.e., Equation 13 in our experiments) to find optimal configuration and scheduling solutions.
- $PaMO_{qUCB/qSR/qEI}$: a variant of PaMO that replaces the acquisition function $qNEI$ with the batch upper confidence bound ($qUCB$), the batch simple regret (qSR), and the batch expected improvement (qEI). All acquisition functions are implemented based on Monte-Carlo sampling.

5.2 System Benefit Comparison

Across different preference functions. We construct different system preference functions by adjusting each weight in Equation 13 to (0.2, 0.4, 1.6, 3.2) while keeping the remaining weights at 1. The weights of the corresponding metrics in the optimization objectives of the JCAB and FACT methods are adjusted accordingly. PaMO fits the preference function by comparison-based preference learning methods. The experiment analyzes 8 video streams on five servers. Each method undergoes three repetitions of testing to capture the average value. We normalize the benefit values of various methods using the solutions identified by PaMO+ as benchmarks. The line data in Figure 6 illustrates that PaMO can attain system benefit close to optimal, with errors ranging from 1.02% to 11.26%. Compared with state-of-the-art methods, JCAB and FACT, PaMO enhances system benefit by 3.9% to 42.3% and 0.42% to 26.5% across diverse system preference weights, respectively. Furthermore, the different colored shades in Figure 6 show that the benefit ratio of different objectives. The PaMO solution is closer to the real preference distribution of the system (shown as the solution of PaMO+) than the JCAB and FACT solutions. This result indicates that PaMO can accurately capture the system preference across different preference functions and achieve a higher system benefit than the single-objective optimization methods.

Across different server and video numbers. We evaluate the scheduling performance of PaMO and baselines by two sets of experiments with different server and video numbers. We use trace data to emulate more than four servers. The preference weight of these experiments is set to 1 to reduce the influence of preference learning error on system benefit. The first set of experiments fix the number of video streams at 10, varying the number of servers from 5 to 9. The second set of experiments fix the number of servers at 5, increasing the number of video streams from 7 to 11. We randomly select bandwidth values for servers from (5 Mbps, 10 Mbps, 15 Mbps, 20 Mbps, 25 Mbps, 30 Mbps) to simulate diverse real-world scenarios. The average normalized benefit of three repeated experiments under each configuration is shown in Figure 7. Specifically, compared with JCAB and FACT, PaMO enhances the overall benefit by 13.6% to 53.9% and 6.5% to 16.6% across different server and video numbers, respectively. The decrease in system benefit of PaMO, ranging from 0.0006% to 1.54% compared to PaMO+, stems from the fitting error of the preference function.

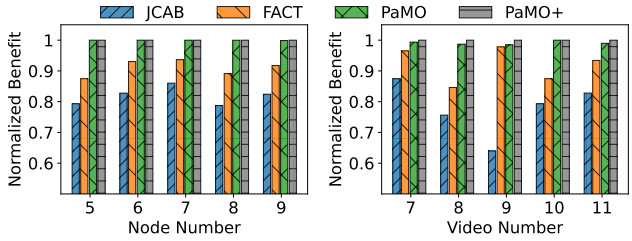


Figure 7: Normalized benefit comparison under different server and video number.

5.3 Component Performance Analysis

Outcome prediction performance. We use the coefficient of determination, i,e, $R^2 = 1-(\sum_i(y_i-\hat{y}_i)^2)/(\sum_i(y_i-\bar{y})^2)$, to evaluate the prediction error of our GP-based outcome models. The number of samples in the training set starts at 200 and increases by 100 until it reaches 500. The pre-trained models over the training set predict the outcome of 20 test samples composed of randomly selected resolution and the frame sampling rate. We repeat the prediction process ten times and plot the results in Figure 8. The figure shows that the R^2 gradually approaches one as the training set increases, which indicates that our outcome models predict more accurately. Apart from computation, other performance prediction models can achieve a prediction error of less than 10% and 5% when the number of training samples reaches 400 and 600, respectively. The prediction model of computation exhibits an average error below 10% when the number of training samples comes to 600.

Preference prediction performance. The output of our preference model is the relative value of the benefit and cannot be directly

[2]Normalized benefit is calculated by $U_{NORM} = 1 - \frac{U-\min(U)}{\max(U)-\min(U)}$, where $\max(U)$ is the benefit value of PaMO+ and $\min(U) = -\frac{1}{2}\sum_{i\in\{lct,acc,net,com,eng\}} w_i$.

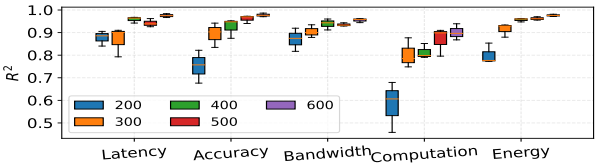


Figure 8: Prediction error of outcome model with different training set sizes.

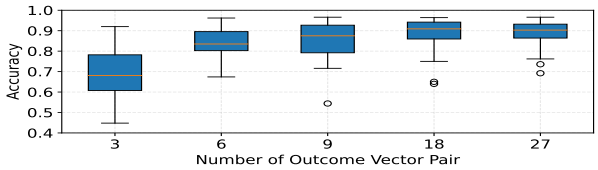


Figure 9: Prediction error of preference model with different training comparison pairs.

compared with the actual benefit value outputted by the true preference function. To assess accuracy, we construct a test dataset with 500 samples, each of which contains two outcome vectors, denoted as $\{(y_1^i, y_2^i)\}_{i=1}^{500}$. We use the real preference function and the preference model to calculate the benefit value of each sample, denoted as z_1, z_2 and $\hat{z}_1, \hat{z}_2$, respectively. Then, we calculate the prediction accuracy by $1/500 \sum_{i=1}^{500}\{1|(z_1^i > z_2^i and \hat{z}_1^i > \hat{z}_2^i) or \quad (z_1^i < z_2^i and \hat{z}_1^i < \hat{z}_2^i)\}$. We evaluate various pre-trained preference models with multiple sizes of training sets ranging from 3 to 27. We repeat the experiment ten times on different test datasets. The results in Figure 9 shows that the preference model's prediction error is less than 10% when the number of training samples reaches 18.

5.4 Sensitivity Analysis

Impact of weight on baseline methods. We evaluate whether the single-objective methods, JCAB and FACT, can learn system preferences by adjusting weights through two sets of experiments with configurations of "5 servers, 8 videos" and "6 servers, 10 videos." In each experiment set, the weight of one objective is adjusted incrementally from 0.05 to 5, while the weight of another objective remains constant at 1. Since the PaMO method uses the Gaussian process model to fit the preference function (with all weights set to 1), it is independent of weight parameters, so weight changes do not affect the benefit value. Figure 10(a) shows that although the benefit values of the JCAB and FACT methods fluctuate with changes in weight parameters, they consistently fall short of reaching or surpassing the benefit values of the PaMO and PaMO+ methods. This observation indicates that, for the multi-objective optimization problem, the preference function with linear weighting fails to accurately capture system preference across different objectives, even with intricate weight tuning. This imprecise characterization will result in deviations of the corresponding single-objective solution algorithms from the optimal value preferred by the system.

Impact of termination threshold on all methods. In a similar experimental configuration with Figure 10(a), we increase the termination threshold δ from 0.02 to 0.2 to compare the performance change of the different methods, where FACT and JCAB adopt the weights that perform best in Figure 10(a). The result in Figure 10(b) shows that the benefit derived from the PaMO method

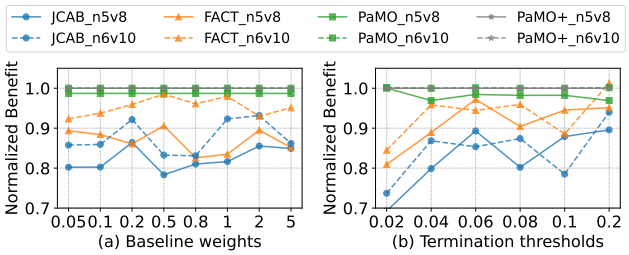


Figure 10: Ablation experiments about baseline weights and termination thresholds. *n5v8* represents the configuration of 5 servers and 8 videos.

is always higher than that obtained by the baseline methods and exhibits less fluctuation with the threshold. Conversely, the baseline methods are sensitive to the termination threshold, requiring additional tuning efforts when applied in practical scenarios.

6 RELATED WORK

Adaptive Video Analytics. Data reduction is an effective way to alleviate the resource pressure brought by video analytics workloads to the edge environment. The exiting works usually adjust some video parameters like frame rate and resolution [13, 19, 32–34], or encode/send only the important part of frames involving new objects [5, 7, 16, 18] to reduce data transmission and computation consumption. Another way to reduce data is to use some cheap models to identify important regions [16] or inference in advance on some frames with salient features [14] on the camera with some low-power GPUs/CPUs. Yoda [31] classifies these adaptive methods and implements a video analytics benchmark to evaluate and discuss the complex interaction between the performance of these methods and video characteristics. Our solution currently uses frame rate and resolution adjustment solutions that are universal to most video streams, and can be followed by adaptive encoding and segmented inference to further improve video analysis performance and resource efficiency.

Multi-objective Optimization. We focus on the preference function formulation of a multi-objective optimization problem, which reflects priorities for various objectives and helps the solver identify a unique optimal solution within the Pareto frontier. The most common form of preference function is the linearly weighted sum of all objective functions. In the existing literature, there are a variety of weight definitions [10], such as Equal weights, Rank order centroid (ROC) weights, Rank-sum (RS) weights, and Pseudo-weights. However, this fixed weight can not accurately capture the system's preference for different objectives, so the problem's solution may deviate from the expected optimal value. The preferential learning method [4, 9, 26], by contrast, has become popular due to its flexibility advantage. This method asks the decision maker to express their preference on a set of outcome vectors and then trains a surrogate model of the preference function based on these preference data. In this paper, we integrate the preference learning process into the Bayesian optimization framework so that the system preference can better guide the solution search process.

Periodic Scheduling Algorithms. Periodic scheduling problems (PSPs) are usually discussed in modern real-time systems like

automotive and avionics. It targets to find a schedule for a set of periodic tasks on single or multiple hardware resources to meet their real-time requirements with different level's jitter constraints [22]. The non-preemptive PSPs focused on this paper, which has been proved as a strongly NP-hard problem [12], can be modeled as an Integer Linear Programming (ILP) problem [28], a Constraints Programming (CP) problem [23] or a Satisfiability Modulo Theories (SMT) problem [25]. The existing work tends to use some heuristic [23] or meta-heuristic [20] algorithms to find feasible solutions within reasonable time. However, these algorithms cannot be directly applied to our problem because they do not consider minimizing communication latency. Our proposed algorithm adopts period-aware grouping and Hungarian algorithm to solve ensures zero jitter delay and latency minimization.

7 CONCLUSION

This paper shows that Bayesian optimization-driven multi-objective schedulers can provide considerable overall benefits over task performance and resource efficiency for video analytics workloads at the edge by adjusting video configuration and server allocation. Our solution, PaMO, notably outperforms the state-of-the-art single-objective schedulers by accurately capturing the influence of various optimization objectives on system benefit. PaMO efficiently identifies optimal solutions by Bayesian optimization iteration with a zero-jitter heuristic scheduling algorithm and an anti-noise acquisition function. This preference-aware multi-objective method will be essential in other performance optimization problems with zero-sum game characteristics. In the future, we aim to delve deeper into the theoretical analysis of Bayesian optimization methods grounded in preference learning. Meanwhile, we are keen on exploring other user-friendly methods for capturing the system's preferences in diverse optimization scenarios.

ACKNOWLEDGMENTS

This work was supported by the National Key Research and Development Program of China under Grant No. 2022YFB4501400.

REFERENCES

[1] AInspiredMinds. 2023. How Google Maps' Immersive View AI is Changing Navigation Forever. https://www.youtube.com/watch?v=zuNUETRtuWo&ab_channel=AInspiredMinds.
[2] Tayebeh Bahreini, Hossein Badri, and Daniel Grosu. 2021. Mechanisms for resource allocation and pricing in mobile edge computing systems. *IEEE Transactions on Parallel and Distributed Systems* 33, 3 (2021), 667–682.
[3] Maximilian Balandat, Brian Karrer, Daniel R. Jiang, Samuel Daulton, Benjamin Letham, Andrew Gordon Wilson, and Eytan Bakshy. 2020. BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization. In *NeurIPS'20*.
[4] Eric Brochu. 2010. *Interactive Bayesian optimization: learning user preferences for graphics and animation*. Ph. D. Dissertation. University of British Columbia.
[5] Tiffany Yu-Han Chen, Lenin Ravindranath, Shuo Deng, Paramvir Bahl, and Hari Balakrishnan. 2015. Glimpse: Continuous, real-time object recognition on mobile devices. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*. 155–168.
[6] Wei Chu and Zoubin Ghahramani. 2005. Preference learning with Gaussian processes. In *Proceedings of the 22nd international conference on Machine learning*. 137–144.
[7] Kuntai Du, Ahsan Pervaiz, Xin Yuan, Aakanksha Chowdhery, Qizheng Zhang, Henry Hoffmann, and Junchen Jiang. 2020. Server-driven video streaming for deep learning inference. In *SIGCOMM'20*. 557–570.
[8] Jonathan E Forman, Christopher M Timperley, Pål Aas, Mohammad Abdollahi, Isel Pascual Alonso, Augustin Baulig, Renate Becker-Arnold, Veronica Borrett, Flerida A Cariño, Christophe Curty, et al. 2018. Innovative technologies for chemical security. *Pure and Applied Chemistry* 90, 10 (2018), 1527–1557.
[9] Javier González, Zhenwen Dai, Andreas Damianou, and Neil D Lawrence. 2017. Preferential bayesian optimization. In *International Conference on Machine Learning*. PMLR, 1282–1291.
[10] Nyoman Gunantara. 2018. A review of multi-objective optimization: Methods and its applications. *Cogent Engineering* 5, 1 (2018), 1502242.
[11] HIKVISION. 2023. DS-2DE7A432IW-AEB(T5) Smart Camera. https://www.hikvision.com/en/products/IP-Products/PTZ-Cameras/Pro-Series/ds-2de7a432iw-aeb-t5-/.
[12] Kevin Jeffay, Donald F Stanat, and Charles U Martel. 1991. On non-preemptive scheduling of periodic and sporadic tasks. In *IEEE real-time systems symposium*. US: IEEE, 129–139.
[13] Junchen Jiang, Ganesh Ananthanarayanan, Peter Bodik, Siddhartha Sen, and Ion Stoica. 2018. Chameleon: scalable adaptation of video analytics. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*. 253–266.
[14] Daniel Kang, John Emmons, Firas Abuzaid, Peter Bailis, and Matei Zaharia. 2017. Noscope: optimizing neural network queries over video at scale. *arXiv preprint arXiv:1703.02529* (2017).
[15] Benjamin Letham, Brian Karrer, Guilherme Ottoni, and Eytan Bakshy. 2019. Constrained Bayesian optimization with noisy experiments. (2019).
[16] Yuanqi Li, Arthi Padmanabhan, Pengzhan Zhao, Yufei Wang, Guoqing Harry Xu, and Ravi Netravali. 2020. Reducto: On-camera filtering for resource-efficient real-time video analytics. In *SIGCOMM'20*. 359–376.
[17] Zhiyuan Jerry Lin, Raul Astudillo, Peter Frazier, and Eytan Bakshy. 2022. Preference Exploration for Efficient Bayesian Optimization with Multiple Outcomes. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 4235–4258.
[18] Luyang Liu, Hongyu Li, and Marco Gruteser. 2019. Edge assisted real-time object detection for mobile augmented reality. In *The 25th annual international conference on mobile computing and networking*. 1–16.
[19] Qiang Liu, Siqi Huang, Johnson Opadere, and Tao Han. 2018. An edge network orchestrator for mobile augmented reality. In *IEEE INFOCOM 2018-IEEE conference on computer communications*. IEEE, 756–764.
[20] Shane D McLean, Silviu S Craciunas, Emil Alexander Juul Hansen, and Paul Pop. 2020. Mapping and scheduling automotive applications on ADAS platforms using metaheuristics. In *ETFA'20*, Vol. 1. IEEE, 329–336.
[21] Anton Milan, Laura Leal-Taixé, Ian Reid, Stefan Roth, and Konrad Schindler. 2016. MOT16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831* (2016).
[22] Anna Minaeva and Zdeněk Hanzálek. 2021. Survey on periodic scheduling for time-triggered hard real-time systems. *ACM Computing Surveys (CSUR)* 54, 1 (2021), 1–32.
[23] Anna Minaeva, Debayan Roy, Benny Akesson, Zdeněk Hanzálek, and Samarjit Chakraborty. 2020. Control performance optimization for application integration on automotive architectures. *IEEE Trans. Comput.* 70, 7 (2020), 1059–1073.
[24] NVIDIA. 2023. Triton Inference Server. https://developer.nvidia.com/triton-inference-server.
[25] Ramon Serna Oliver, Silviu S Craciunas, and Wilfried Steiner. 2018. IEEE 802.1 Qbv gate control list synthesis using array theory encoding. In *2018 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 13–24.
[26] Eero Siivola, Akash Kumar Dhaka, Michael Riis Andersen, Javier González, Pablo García Moreno, and Aki Vehtari. 2021. Preferential batch Bayesian optimization. In *2021 IEEE 31st International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 1–6.
[27] Ultralytics. 2023. YOLOv8. https://github.com/ultralytics/ultralytics.
[28] Marek Vlk, Zdeněk Hanzálek, Kateřina Brejchová, Siyu Tang, Sushmit Bhattacharjee, and Songwei Fu. 2020. Enhancing schedulability and throughput of time-triggered traffic in IEEE 802.1 Qbv time-sensitive networks. *IEEE Transactions on Communications* 68, 11 (2020), 7023–7038.
[29] Chen Wang, Kaile Zhou, and Shanlin Yang. 2017. A review of residential tiered electricity pricing in China. *Renewable and Sustainable Energy Reviews* 79 (2017), 533–543.
[30] Caesar Wu, Rajkumar Buyya, and Kotagiri Ramamohanarao. 2019. Cloud pricing models: Taxonomy, survey, and interdisciplinary challenges. *ACM Computing Surveys (CSUR)* 52, 6 (2019), 1–36.
[31] Zhujun Xiao, Zhengxu Xia, Haitao Zheng, Ben Y Zhao, and Junchen Jiang. 2021. Towards performance clarity of edge video analytics. In *2021 IEEE/ACM Symposium on Edge Computing (SEC)*. IEEE, 148–164.
[32] Ben Zhang, Xin Jin, Sylvia Ratnasamy, John Wawrzynek, and Edward A Lee. 2018. Awstream: Adaptive wide-area streaming analytics. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*. 236–252.
[33] Haoyu Zhang, Ganesh Ananthanarayanan, Peter Bodik, Matthai Philipose, Paramvir Bahl, and Michael J Freedman. 2017. Live Video Analytics at Scale with Approximation and {Delay-Tolerance}. In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*. 377–392.
[34] Sheng Zhang, Can Wang, Yibo Jin, Jie Wu, Zhuzhong Qian, Mingjun Xiao, and Sanglu Lu. 2021. Adaptive configuration selection and bandwidth allocation for edge-based video analytics. *IEEE/ACM Transactions on Networking* 30, 1 (2021), 285–298.