User Preference Based Energy-Aware Mobile AR System with Edge Computing

Haoxin Wang and Jiang Xie
The University of North Carolina at Charlotte, Charlotte, NC 28223, U.S.A.

***Abstract*—The advancement in deep learning and edge computing has enabled intelligent mobile augmented reality (MAR) on resource limited mobile devices. However, today very few deep learning based MAR applications are applied in mobile devices because they are significantly energy-guzzling. In this paper, we design a user preference based energy-aware edge-based MAR system that enables MAR clients to dynamically change their configuration parameters, such as CPU frequency and computation model size, based on their user preferences, camera sampling rates, and available radio resources at the edge server. Our proposed dynamic MAR configuration adaptations can minimize the per frame energy consumption of multiple MAR clients without degrading their preferred MAR performance metrics, such as service latency and detection accuracy. To thoroughly analyze the interactions among MAR configuration parameters, user preferences, camera sampling rate, and per frame energy consumption, we propose, to the best of our knowledge, the first comprehensive analytical energy model for MAR clients. Based on the proposed analytical model, we develop a LEAF optimization algorithm to guide the MAR configuration adaptation and server radio resource allocation. Extensive evaluations are conducted to validate the performance of the proposed analytical model and LEAF algorithm.**

I. Introduction

With the advancement in *Deep Learning* in the past few years, we are able to create intelligent machine learning models to accurately detect and classify complex objects in the physical world. This advancement has the potential to make *Mobile Augmented Reality* (MAR) applications highly intelligent and widely adaptable in various scenarios, such as tourism, education, and entertainment. Thus, implementing MAR applications on popular mobile architectures is a new trend in modern technologies.

However, only a few MAR applications are implemented in mobile devices and are developed based on deep learning frameworks because (i) performing deep learning algorithms on mobile devices is significantly energy-guzzling; (ii) deep learning algorithms are computation-intensive and executing locally in resource limited mobile devices may not provide acceptable performance for MAR clients [1]. To solve these issues, a promising approach is to transfer MAR input image/video frames to an edge server which is powerful enough to execute the deep learning algorithms.

Motivations. Although compared to running a deep learning algorithm locally on a mobile device, edge-based approach may extend the device's battery life to some extent, it is still considerably energy consuming due to conducting multiple pre-processes on the mobile device, such as camera sampling, screen rendering, image conversion, and data transmission [2]. For instance, based on the measurement from our developed MAR testbed, a 3000 mAh smartphone battery is exhausted within approximately 2.3 hours for executing our developed MAR application which continuously transmits the latest camera sampled image frames to an edge server for object detection. Therefore, the energy efficiency of MAR devices becomes a bottleneck, which impedes MAR clients to obtain better MAR performance. For example, decreasing the energy consumption of an MAR device is always at the cost of reducing the object detection accuracy. Therefore, improving the energy efficiency of MAR devices and balancing the trade-offs between energy efficiency and other MAR performance metrics are crucial to edge-based MAR systems.

This work was supported in part by the US National Science Foundation (NSF) under Grant No. 1718666, 1731675, 1910667, and 1910891.

Challenges. An accurate analytical energy model is significantly important for understanding how energy is consumed in an MAR device and for guiding the design of energy-aware MAR systems. However, to the best of our knowledge, there is no existing energy model developed for MAR devices or applications. Developing a comprehensive MAR energy model that is general enough to handle any MAR architecture and application is very challenging. This is because (i) interactions between MAR configuration parameters (e.g., client's CPU frequency and computation model size) and MAR device's energy consumption are complex and lack analytic understandings; (ii) interactions between these configurations and the device's energy consumption may also vary with different mobile architectures.

In addition, designing an energy-aware solution for mobile devices in edge-based MAR systems is also challenging, even after we obtain an analytical energy model. This is because: (i) complicated pre-processes on MAR devices increase the complexity of the problem. Compared to conventional computation offloading systems, besides data transmission, there are also a variety of pre-processing tasks (e.g., camera sampling, screen rendering, and image conversion) necessarily to be performed on MAR devices, which are also energy consuming. For example, over 60% of the energy is consumed by camera sampling and screen rendering, based on observations from our developed testbed. Therefore, we have to take into account the energy efficiency of these pre-processing tasks while designing an energy-aware approach for MAR clients. (ii) Considering the user preference constraint of individual MAR clients also increases the complexity of the problem. For example, maintaining a high detection accuracy for a client who prefers a precise MAR while decreasing its energy consumption is very challenging. As stated previously, reducing the energy consumption of the MAR device without degrading other performance metrics is no easy task. (iii) In practical scenarios, an edge server is shared by multiple MAR clients. Individual client's energy efficiency is also coupled with the radio resource allocation at the edge server. Such a coupling makes it computationally hard to optimally allocate radio resources and improve each client's energy efficiency.

Our Contributions. In this paper, we study these research challenges and design a user preference based energy-aware edge-based MAR system. The novel contributions of this paper are summarized as follows:

1) We design and implement an edge-based MAR system to analyze the interactions between MAR configurations

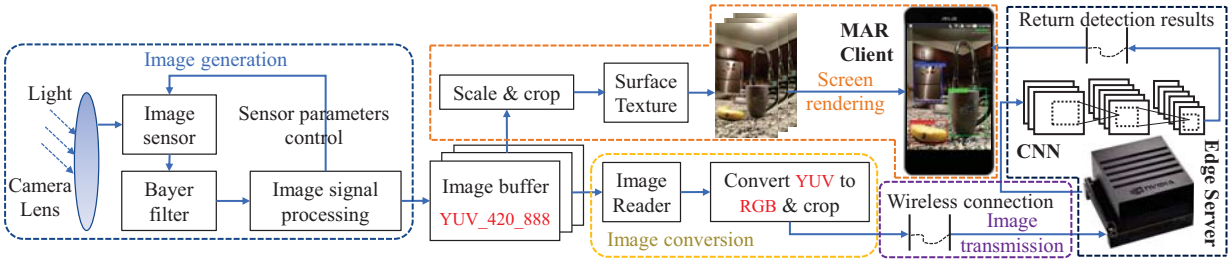


Fig. 1. The processing pipeline of the edge-based MAR system developed in this paper.

and the client's energy consumption. Based on our experimental study, we summarize several insights which can potentially guide the design of energy-aware MAR systems.

2) We propose, to the best of our knowledge, the first comprehensive energy model which identifies (i) the tradeoffs among the energy consumption, service latency, and detection accuracy, and (ii) the interactions among MAR configuration parameters (i.e., CPU frequency and computation model size), user preferences, camera sampling rate, network bandwidth, and per frame energy consumption for a multi-user edge-based MAR system.
3) We propose an energy-efficient optimization algorithm, LEAF, which guides MAR configuration adaptations and radio resource allocations at the edge server, and minimizes the per frame energy consumption while satisfying variant clients' user preferences.

II. Related Work

Energy Modeling. Energy modeling has been widely used for investigating the factors that influence the energy consumption of mobile devices. [3]–[10] propose energy models of WiFi and LTE data transmission with respect to the network performance metrics, such as data and retransmission rates, respectively. [11]–[14] propose multiple power consumption models to estimate the energy consumption of mobile CPUs. However, none of them can be directly applied to estimate the energy consumed by MAR applications. This is because MAR applications introduce a variety of (i) energy consuming components (e.g., camera sampling and image conversion) that are not considered in the previous models and (ii) configuration variables (e.g., computation model size and camera sample rate) that also significantly influence the energy consumption of mobile devices.

Computation Offloading. Most existing research on computation offloading focuses on how to make offloading decisions. [15]–[17] coordinate the scheduling of offloading requests for multiple applications to further reduce the wireless energy cost caused by the long tail problem. [18] proposes an energy-efficient offloading approach for multicore-based mobile devices. However, these solutions cannot be applied to improving the energy efficiency of mobile devices in MAR offloading cases. This is because (i) a variety of pre-processing tasks in MAR executions, such as camera sampling, screen rendering, and image conversion, are not taken into account and (ii) besides the latency constraint that is considered in most existing computation offloading approaches, detection accuracy is also a key performance metric, which must be considered while designing an MAR offloading solution. In addition, although some existing work proposes to study the tradeoffs between the MAR service latency and detection accuracy [19]–[21], none of them considered (i) the energy consumption of the MAR device and (ii) the whole processing pipeline of MAR (i.e., starting from the camera sampling to obtaining detection results).

CPU Frequency Scaling. Our work is also related to CPU frequency scaling. For modern mobile devices, such as smartphones, CPU frequency and the voltage provided to the CPU can be adjusted at run-time, which is called Dynamic Voltage and Frequency Scaling (DVFS). Prior work [15], [22]–[24] proposes various DVFS strategies to reduce the mobile device energy consumption under various applications, such as video streaming [15] and delay-tolerant applications [23]. However, to the best of our knowledge, there have been no efforts factoring in the energy efficiency of MAR applications in the context of mobile device DVFS.

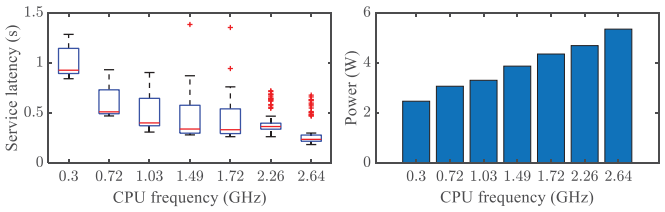
III. Experimental Results on Factors Affecting MAR Client Energy Efficiency

In this section, we describe our preliminary experiments to evaluate the impact of various factors on the energy efficiency of an MAR client, service latency, and detection accuracy in an edge-based MAR system. Specifically, these experimental results provide (i) observations on interactions between energy consumption and MAR configuration parameters, such as MAR client's CPU frequency, computation model size, camera sampling rate, and user preference, (ii) bases of modeling the energy consumption of an MAR client, and (iii) insights on designing an energy-efficient optimization algorithm.

A. Testbed Setup

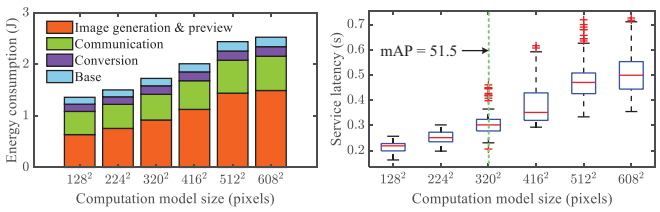
Our testbed consists of three major components: MAR client, edge server, and power monitor. Note that this paper focuses on the MAR application in which an MAR client captures physical environmental information through the camera and sends the information to an edge server for object detection. The detailed processing pipeline is shown in Fig. 1.

Edge Server. The edge server is developed to process received image frames and to send the detection results back to the MAR client. We implement an edge server on an Nvidia Jetson AGX Xavier, which connects to a WiFi access point (AP) through a 1Gbps Ethernet cable. The transmission latency between the server and AP can be ignored. Two major modules are implemented on the edge server: (i) the *communication handler* which establishes a TCP socket connection with the MAR device and (ii) the *analytics handler* which performs object detection for the MAR client. In this paper, the analytics handler is designed based on a custom framework called



(a) (b)

Fig. 2. CPU frequency vs. power and service latency (computation model size: 320^2 pixels).



(a) (b)

Fig. 3. Computation model size vs. energy consumption and service latency.

Darknet [25] with GPU acceleration and runs YOLOv3 [26], a large Convolutional Neural Networks (CNN) model. The YOLOv3 model used in our experiments is trained on COCO dataset [27] and can detect 80 classes.

MAR Client. We implement an MAR client on a rooted Nexus 6 smartphone which is equipped with Qualcomm Snapdragon 805 SoC (System-on-Chip). The CPU frequency ranges from 0.3 GHz to 2.649 GHz. The MAR client transfers the converted RGB frames to the edge server through a TCP socket connection. To avoid the processing of stale frames, the MAR client sends the latest camera captured frame to the server and waits for the detection result before sending the next frame for detection.

Power Monitor. The power monitor is responsible for measuring the power consumption of the MAR client. We use Monsoon Power Monitor [28], which can sample at 5000 Hz, to provide power supply for the MAR device.

Key Performance Metrics. We define three performance metrics to evaluate the MAR system:

- *Per frame energy consumption:* The per frame energy consumption is the total amount of energy consumed in an MAR client by successfully performing the object detection on one image frame. It includes the energy consumed by camera sampling (i.e., image generation), screen rendering (i.e., preview), image conversion, communication, and operating system.
- *Service latency:* The service latency is the total time needed to derive the detection result on one image frame. It includes the latency of image conversion, transmission, and inference.
- *Accuracy:* The mean average precision (mAP) is a commonly used performance metric to evaluate the detection accuracy of a visual object detection algorithm [29], where a greater accuracy is indicated by a higher mAP.

B. The Impact of CPU Frequency on Power Consumption and Service Latency

In this experiment, we seek to investigate how the CPU frequency impacts the power consumption of the MAR device and the service latency. We set the test device to the *Userspace* Governor and change its CPU frequency manually by writing files in the `/sys/devices/system/cpu/[cpu#]/cpufreq` virtual file system with root privilege. The results are shown in Fig. 2. The lower the CPU frequency, the longer service latency the MAR client derives and the less power it consumes. However, the reduction of the service latency and the increase of the power consumption is disproportional. For example, as compared to 1.03 GHz, 1.72 GHz reduces about 2% service latency but increases about 15% power consumption. As compared to 0.3 GHz, 0.72 GHz reduces about 60% service latency, but only increases about 20% power consumption.

Insight: *This result advocates adapting the client's CPU frequency for the service latency reduction by trading as little increase of the per frame energy consumption as possible, where the per frame energy consumption is calculated by the power multiplies the service latency.*

C. The Impact of Computation Model Size on Energy Consumption and Service Latency

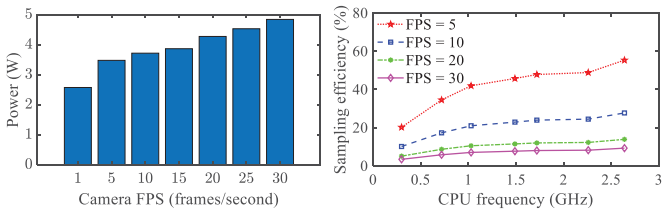
In this experiment, we implement six object detection algorithms based on the YOLOv3 framework [26] with different computation model sizes. The test device works on the default CPU governor, *Interactive*. Increasing the model size always results in a gain of mAP. However, the gain on mAP becomes smaller as the increase of the model sizes [20]. In addition, the per frame energy consumption and the service latency boost 85% and 130%, respectively, when the model size increases from 128^2 to 608^2 pixels, as shown in Fig. 3(a) and 3(b).

Insight: *This result inspires us to trade mAP for the per frame energy consumption and service latency reduction when the model size is large.*

D. The Impact of Camera FPS on Power Consumption

In this experiment, we vary the MAR client's camera FPS to explore how it impacts the device's power consumption, where the camera FPS is defined as the number of frames that the camera samples per second. Fig. 4(a) shows that a large camera FPS leads to a high power consumption. However, as shown in Fig. 1, not every camera captured image frame is sent to the edge server for detection. Because of the need (i) to avoid the processing of stale frames and (ii) to decrease the transmission energy consumption, only the latest camera sampled image frame is transmitted to the server. This may result in the MAR client expending significant reactive power for sampling non-detectable image frames. In Fig. 4(b), we quantify the sampling efficiency with the variation of the camera FPS. As we expected, a large camera FPS leads to a lower sampling efficiency (e.g., less than 2% of the power is consumed for sampling the detectable image frames when the camera FPS is set to 30). However, in most MAR applications, users usually request a high camera FPS for a smoother preview experience, which is critical for tracking targets in physical environments. Interestingly, increasing CPU frequency can reduce the reactive power for sampling, as shown in Fig. 4(b).

Insight: *This result demonstrates that when a high camera FPS is requested, increasing CPU frequency can promote the sampling efficiency but may also boost the power consumption.*



(a) (b)

Fig. 4. Camera FPS vs. power and sampling efficiency (computation model size: 320^2 pixels).

Therefore, finding a CPU frequency that can balance this tradeoff is critical.

E. User Preference

An MAR client may have variant preferences in different implementation cases, including:

- **Latency-preferred.** The MAR application of cognitive assistance [30], where a wearable device helps visually impaired people to navigate on a street, may require a low service latency but can tolerate a relatively high number of false positives (i.e., false alarms are fine but missing any potential threats on the street is costly).
- **Accuracy-preferred.** An MAR application for recommending products in shopping malls or supermarkets may tolerate a long latency but requires high detection accuracy and preview smoothness.
- **Preview-preferred.** The MAR drawing assistant application [31], where a user is instructed to trace virtual drawings from the phone, may tolerate a long latency (i.e., only needs to periodically detect the position of the paper where the user is drawing on) but requires a smooth preview to track the lines that the user is drawing.

Insight: *This observation infers that the user preference's diversity may significantly affect the tradeoffs presented above. For instance, for the accuracy-preferred case, trading detection accuracy for the per frame energy consumption or service latency reduction works against the requirement of the user.*

IV. Proposed System Architecture

Based on the above insights, we propose an edge-based MAR system that can reduce the per frame energy consumption of MAR clients by dynamically selecting the optimal combination of MAR configurations (i.e., CPU frequency and computation model size) and radio resource allocations according to user preferences, camera FPS, and available radio resources at the edge server. To derive the optimal MAR configurations and radio resource allocations, we propose an optimization algorithm (LEAF) that supports low-energy, accurate, and fast MAR applications. LEAF can jointly optimize the CPU frequency, computation model size, and radio resource allocation (explained in detail in Section VI).

Fig. 5 shows the overview of our proposed system. In the first step, MAR clients send their service requests and selected camera FPS and user preferences to an edge server. In the second step, according to the received camera FPS and user preferences, the edge server determines the optimal CPU frequency, computation model size, and allocated radio resource for each MAR client using our proposed LEAF algorithm. The determined CPU frequency and computation

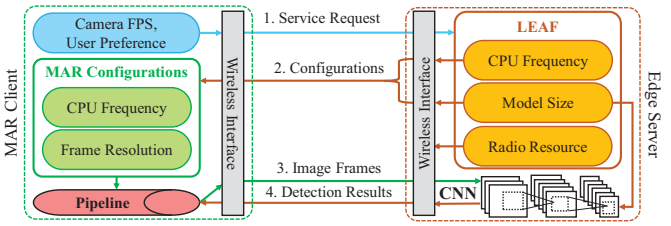


Fig. 5. Overview of the proposed edge-based MAR system.

model size are then sent back to corresponding MAR clients as MAR configuration messages. In the third step, MAR clients set their CPU frequency to the optimal value and resize their latest camera sampled image frames based on the received optimal computation model size. After the CPU frequency adaptation and image frame resizing, MAR clients transmit their image frames to the edge server for object detection. In the final step, the edge server returns detection results to corresponding MAR clients.

However, designing such a system is challenging. From the presented insights in the previous section, the interactions among the MAR system configuration variables, user preference, camera FPS, and the per frame energy consumption are complicated. (i) Some configuration variables improve one performance metric but impair another one. For example, a lower computation model size reduces the service latency but decreases the detection accuracy. (ii) Some configuration variables may affect the same metric in multiple ways. For example, selecting a higher CPU frequency can decrease the per frame energy consumption by increasing the sampling efficiency, but it increases the CPU power, which conversely increases the per frame energy consumption. Unfortunately, there is no analytical model for characterizing these interactions in the MAR system and it is not possible to design a prominent optimization algorithm without thoroughly analyzing these interactions.

V. Proposed Analytical Model and Problem Formulation

In this section, we thoroughly investigate the complicated interactions among the MAR configuration parameters, user preference, camera FPS, and the key performance metrics presented in Section III. We first propose a comprehensive analytical model to theoretically dissect the per frame energy consumption and service latency. The proposed model is general enough to handle any MAR device and application. Then, using the proposed model, we further model multiple fine-grained interactions, whose theoretical properties are complex and hard to understand, via a data-driven methodology. Finally, based on the above proposed models, we formulate the MAR reconfiguration as an optimization problem.

A. Analytics-based Modeling Methodology

We consider an edge-based MAR system with K MAR clients and one edge server, where clients are connected to the edge server via a single-hop wireless network. Denote $\mathcal{K}$ as the set of MAR clients. The per frame service latency of the kth MAR client can be defined as

$$L^k = L^k_{cv} + L^k_{tr} + L^k_{inf}, \quad (1)$$

where L^k_{cv} is the image conversion latency caused by converting a buffered camera captured image frame from YUV

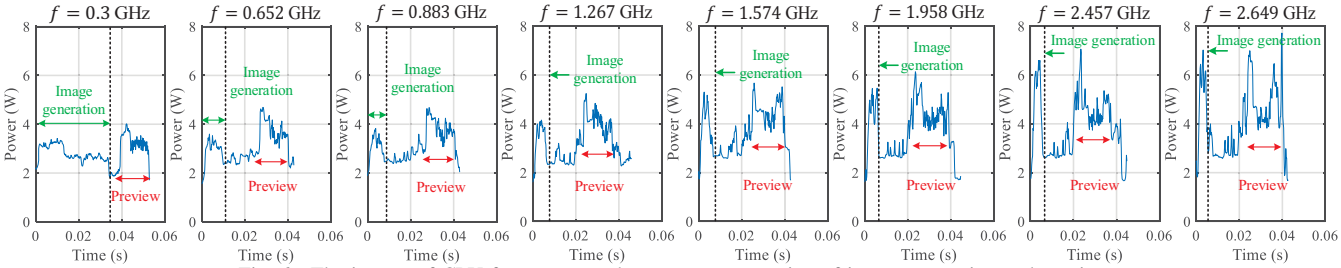


Fig. 6. The impact of CPU frequency on the power consumption of image generation and preview.

to RGB; L^k_{tr} is the transmission latency incurred by sending the converted RGB image frame from the kth client to its connected edge server; and L^k_{inf} is the inference latency of the object detection on the server. According to the MAR pipeline depicted in Fig. 1, the per frame energy consumption of the kth MAR client can be defined as

$$E^k = E^k_{img} + E^k_{cv} + E^k_{com} + E^k_{bs}, \tag{2}$$

where E^k_{img} is the image generation and preview energy consumption incurred by image sampling, processing, and preview rendering; E^k_{cv} is the image conversion energy consumption; E^k_{com} is the wireless communication energy consumption, which includes four phases: *promotion, data transmission, tail*, and *idle*; and E^k_{bs} is the MAR device base energy consumption.

The Model of Image Generation and Preview. Image generation is the process that an MAR client transfers its camera sensed continuous light signal to a displayable image frame. Preview is the process of rendering the latest generated image frame on the client's screen. As these two processes are executed in parallel with the main thread, their execution delays are not counted in the per frame service latency.

As depicted in Fig. 3(a), the energy consumption of image generation and preview is the largest portion of the per frame energy consumption. To understand how energy is consumed in image generation and preview and what configuration variables impact it, we conduct a set of experiments. We find that *the power consumption of image generation and preview highly depends on the CPU frequency.* Fig. 6 shows the power consumption of image generation and preview under different CPU frequencies, where the camera FPS is set to 15. A higher CPU frequency results in a higher average power consumption. In addition, the image generation delay is also closely related to the CPU frequency, where a higher CPU frequency always leads to a shorter delay. However, the delay of rendering a preview is only related to the GPU frequency, which is out of the scope of this paper. Thus, we consider the preview delay as a fixed value with any CPU frequencies. We model the energy consumption of the kth MAR client's image generation and preview within a service latency as

$$E^k_{img} = \left(\int_0^{t^k_{gt}(f_k)} P^k_{gt}(f_k)\, dt + \int_0^{t_{prv}} P^k_{prv}(f_k)\, dt \right) \cdot fps_k \cdot L^k, \tag{3}$$

where P^k_{gt}, P^k_{prv}, t^k_{gt}, t_{prv} are the power consumption of image generation, preview, the delay of image generation, and preview, respectively; f_k is the CPU frequency; fps_k is the camera FPS; P^k_{gt}, P^k_{prv}, and t^k_{gt} are functions of f_k.

The Model of Image Conversion. Image conversion is processed through the MAR client's CPU; hence, the conversion latency and power consumption highly depend on the CPU frequency. We define L^k_{cv} and E^k_{cv} a function of f_k. Therefore, the major source of the power consumption of the image conversion is the CPU computation. The power consumption of mobile CPUs can be divided into two components, $P^k_{cv} = P_{leak} + P^k_{dynamic}$ [13], where P_{leak} is independent and $P^k_{dynamic}$ is dependent upon the CPU frequency. (i) P_{leak} is the power originating from leakage effects and is in essence not useful for the CPU's purpose. In this paper, we consider P_{leak} a constant value ϵ. (ii) $P^k_{dynamic}$ is the power consumed by the logic gate switching at f_k and is proportional to $V_k^2 f_k$, where V_k is the supply voltage for the CPU. Due to the DVFS for the power saving purpose, e.g. a higher f_k will be supplied by a larger V_k, each f_k matches with a specific V_k, where $V_k \propto (\alpha_1 f_k + \alpha_2)$; α_1 and α_2 are two positive coefficients. Thus, the energy consumption of converting a single image frame of the kth MAR client can be modeled as

$$E^k_{cv} = P^k_{cv} L^k_{cv} = (\alpha_1^2 f_k^3 + 2\alpha_1\alpha_2 f_k^2 + \alpha_2 f_k + \epsilon) \cdot L^k_{cv}(f_k). \tag{4}$$

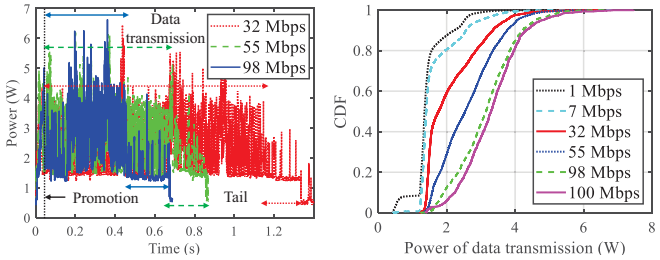
The Model of Wireless Communication and Inference. Intuitively, the wireless communication latency is related to the data size of the transmitted image frame (determined by the frame resolution) and wireless data rate. As the data size of detection results is usually small, we do not consider the latency caused by returning the detection results [20]. In this paper, we use s_k^2 (pixels) to represent the computation model size of the kth MAR client. The client must send image frames whose resolutions are not smaller than s_k^2 to the edge server to obtain the corresponding detection accuracy. Thus, the most efficient way is to transmit the image frame with the resolution of s_k^2 to the server. Denote σ as the number of bits required to represent the information carried by one pixel. The data size of an image frame is calculated as σs_k^2 bits. Let B_k be the wireless bandwidth derived by the kth MAR client. We model the transmission latency of the kth client as

$$L^k_{tr} = \frac{\sigma s_k^2}{R_k}, \tag{5}$$

where R_k is the average wireless data rate of the kth client, which is a function of B_k.

In addition to the computation model size and wireless bandwidth, the transmission latency is also determined by the MAR client's CPU frequency. This is because the image transmission uses TCP as the transport layer protocol, and TCP utilizes substantial CPU capacity to handle congestion avoidance, buffer, and retransmission requests. For example, when the CPU frequency is low, the remaining CPU capacity may not be adequate to process the TCP task; thus, the TCP throughput is decreased. Therefore, R_k is also a function of f_k, i.e., $R_k(B_k, f_k)$. In this paper, $R_k(B_k, f_k)$ is defined as

$$R_k(B_k, f_k) = r_k^{max}(B_k) \cdot r_k^*(f_k), \tag{6}$$



(a) (b)

Fig. 7. MAR client's wireless interface power consumption.

where $r_k^{max}(B_k)$ is the network throughput, which is not affected by the variation of the MAR client's CPU frequency, and is only determined by the bandwidth (more comprehensive model of this part can be found in [3], which is out of the scope of this paper); $r_k^*(f_k)$ represents the impact of the CPU frequency on the TCP throughput.

In WiFi networks, when transmitting a single image frame, the MAR client's wireless interface experiences four phases: promotion, data transmission, tail, and idle. When an image transmission request comes, the wireless interface enters the promotion phase. Then, it enters the data transmission phase to send the image frame to the edge server. After completing the transmission, the wireless interface is forced to stay in the tail phase for a fixed duration and waits for other data transmission requests and the detection results. If the MAR client does not receive the detection result in the tail phase, it enters the idle phase and waits for the feedback from its associated edge server. Fig. 7 depicts the measured power consumption of the MAR client that transmits a 3840×2160 pixel image with different throughput. We find that the average power consumption of the data transmission phase increases as the throughput grows. However, the average power consumption and the duration of promotion and tail phases are almost constant. Therefore, we model the energy consumption of the kth MAR client in the duration that starts from the promotion phase to obtaining the object detection result as

$$E_{com}^k = P_{tr}^k(R_k(B_k, f_k))L_{tr}^k + P_{idle}^k t_{idle}^k + P_{pro}t_{pro} + P_{tail}t_{tail}, \tag{7}$$

where P_{tr}^k, P_{idle}^k, P_{pro}, and P_{tail} are the average power consumption of the data transmission, idle, promotion, and tail phases, respectively; t_{idle}^k, t_{pro}, and t_{tail} are the durations of the idle, promotion, and tail phases, respectively;

$$P_{idle}^k t_{idle}^k = \begin{cases} 0, & L_{inf}^k(s_k^2) \le t_{tail}, \\ P_{bs}^k \cdot (L_{inf}^k(s_k^2) - t_{tail}), & L_{inf}^k(s_k^2) > t_{tail}, \end{cases} \tag{8}$$

where P_{bs}^k is the MAR device's base power consumption; $L_{inf}^k(s_k^2)$ is the inference latency on the edge server, which is determined by the computation model size [20]. Note that our proposed wireless communication model can also be used in other wireless networks (e.g., LTE).

The Model of Base Energy. In this paper, the base energy consumption is defined as the energy consumed by the MAR clients' CPU without any workloads, except running its operating system, and the energy consumed by the screen without any rendering. Because the screen's brightness is not a critical factor that affects the object detection performance, it is considered as a constant value in our proposed power model. Thus, the base power consumption is only a function of the CPU frequency. We model the base energy consumption of the kth MAR client within a service latency as

$$E_{bs}^k = \begin{cases} P_{bs}^k(f_k) \cdot L^k, & L_{inf}^k(s_k^2) \le t_{tail}, \\ P_{bs}^k(f_k) \cdot (L^k - L_{inf}^k(s_k^2) + t_{tail}), & L_{inf}^k(s_k^2) > t_{tail}. \end{cases} \tag{9}$$

TABLE I
THE PROPOSED REGRESSION-BASED MODELS.

	Proposed models	RMSE
$E_{gt}(f)$	$-0.01071f^3 + 0.06055f^2 - 0.1028f + 0.107$	0.002
$E_{prv}(f)$	$0.01094f + 0.04816$	0.002
$P_{cv}(f)$	$0.1124f^3 + 0.01f^2 + 0.2175f + 0.04295$	0.041
$L_{cv}(f)$	$-0.145f^3 + 0.8f^2 - 1.467f + 0.996$	0.025
$r^{max}(B)$	$0.677B$	2.403
$r^*(f)$	$0.07651f^3 - 0.4264f^2 + 0.7916f + 0.4489$	0.013
$P_{tr}(R)$	$0.01821R + 0.7368$	0.052
$L_{inf}(s^2)$	$0.07816s^2 + 0.08892$	0.838
$P_{bs}(f)$	$0.07873f + 0.5918$	0.015

B. Regression-based Modeling Methodology

As shown in Subsection V-A, some interactions or functions in our proposed analytical models still cannot be expressed clearly in an analytic form. This is because of (i) the lack of analytic understandings of some interactions and (ii) specific coefficients/functions that may vary with different MAR device models. For example, in (4), the specific coefficients in $P_{cv}^k(f_k)$ are unknown due to the lack of theoretical knowledge and vary with different MAR device models.

Therefore, we propose a data-driven methodology to address the above challenge, where those interactions with inadequate analytic understandings can be modeled and trained offline via empirical measurements and regression analyses. Note that regression-based modeling methodology is one of the most widely used approaches in developing mobile CPU's property models (e.g., CPU power and temperature variation modeling) and has shown to be effective in estimating CPU properties [11], [12], [15]. We use our testbed to collect measurements. The test MAR device is selected to work at 18 different CPU frequencies ranging from 0.3 to 2.649 GHz. In addition, in order to obtain fine-grained regression models and eliminate the interference among different workloads on the device power consumption, we develop three Android applications; each is applied with a specific function of the MAR client, which includes image generation and preview, image conversion, and image transmission applications. The developed regression models are shown in Fig. 8 and Table I. Note that to obtain a statistical confidence in the experimental results, each data point in Fig. 8 is derived by generating, transmitting, and detecting 1000 image frames and calculating the average values. The root mean square error (RMSE) is applied for calculating the average model-prediction error in the units of the variable of interest [32].

C. Problem Formulation

Based on the above proposed models, we formulate the MAR reconfiguration as a multi-objective optimization problem [33]. We aim to *minimize the per frame energy consumption of multiple MAR clients in the system while satisfying the user preference* (i.e., stated in Section III-E) *of each*. We introduce two positive weight parameters λ_1^k and λ_2^k to characterize the user preference of the kth MAR client, where λ_1^k and λ_2^k can be specified by the client. We adopt

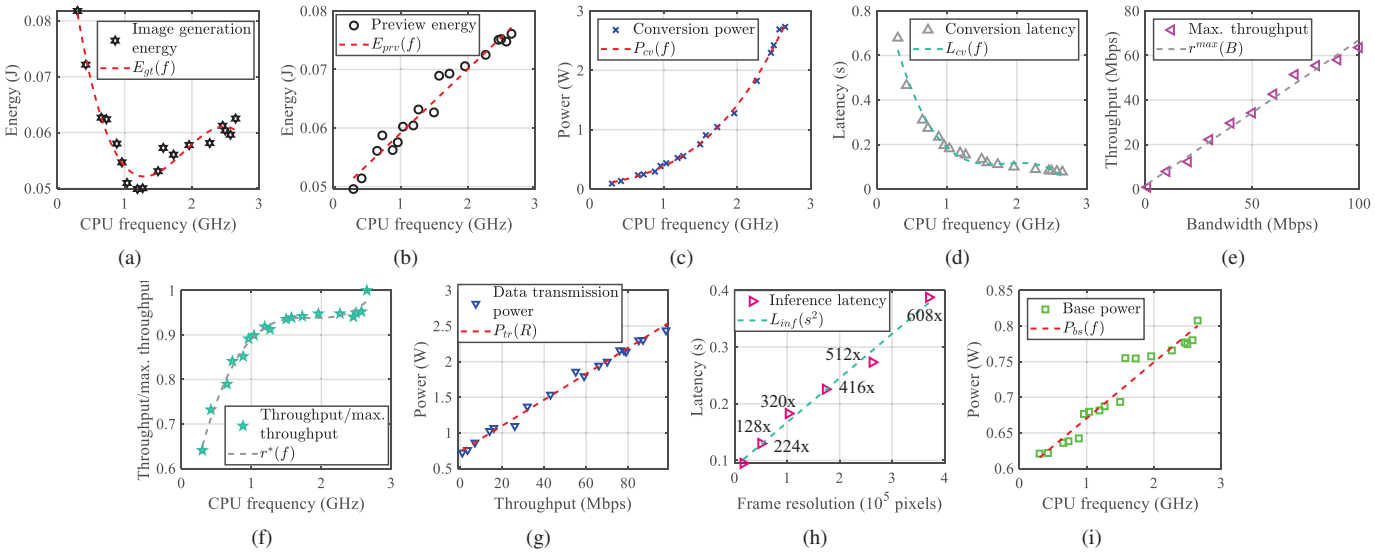


Fig. 8. The proposed regression-based models.

the weighted sum method [34] to express the multi-object optimization problem as

$$\begin{aligned} \mathscr{P}_0: \min_{\{f_k, s_k, B_k, \forall k \in \mathcal{K}\}} \quad & Q = \sum_{k\in\mathcal{K}} (E^k + \lambda_1^k L^k - \lambda_2^k A_k) \\ s.t. \quad & C_1: \sum_{k\in\mathcal{K}} B_k \le B_{max}; \\ & C_2: L^k \le L_{max}^k, \forall k \in \mathcal{K}; \\ & C_3: F_{min} \le f_k \le F_{max}, \forall k \in \mathcal{K}; \\ & C_4: s_k \in \{s_{min}, ..., s_{max}\}, \forall k \in \mathcal{K}; \end{aligned} \tag{10}$$

where A_k is an object detection accuracy function in terms of the kth MAR client selected computation model size s_k^2 (e.g., $A(s_k^2) = 1 - 1.578e^{-6.5\times10^{-3}s_k}$ [20]); L_{max}^k is the maximum tolerable service latency of the kth client; B_{max} is the maximum wireless bandwidth that an edge server can provide for its associated MAR clients. In practical scenarios, an edge server may simultaneously offer multiple different services for its associated users, e.g., video streaming, voice analysis, and content caching. Hence, the edge server may reallocate its bandwidth resource based on the user distribution. In this paper, we assume that B_{max} varies with time randomly. The constraint C_1 represents that MAR clients' derived bandwidth cannot exceed the total bandwidth allocated for the MAR service on the edge server; the constraint C_2 guarantees that the service latency of MAR clients are no larger than their maximum tolerable latency; the constraints C_3 and C_4 are the constraints of the MAR device's CPU frequency and computation model size configurations, where s_k is a discrete variable and its values depend on the available computation models in the MAR system.

VI. Proposed LEAF Optimization Algorithm

As shown in the previous section, problem $\mathscr{P}_0$ is a mixed-integer non-linear programming problem (MINLP) which is difficult to solve [35]. In order to solve this problem, we propose the LEAF algorithm based on the block coordinate descent (BCD) method [36].

To solve problem $\mathscr{P}_0$, we relax the discrete variable s_k into continuous variable $\hat{s_k}$. The problem is relaxed as

$$\begin{aligned} \mathscr{P}_1: \min_{\{f_k, \hat{s_k}, B_k, \forall k \in \mathcal{K}\}} \quad & Q = \sum_{k\in\mathcal{K}} (E^k + \lambda_1^k L^k - \lambda_2^k A_k) \\ s.t. \quad & C_1, C_2, C_3 \\ & \hat{C}_4: s_{min} \le \hat{s_k} \le s_{max}, \forall k \in \mathcal{K}. \end{aligned} \tag{11}$$

According to the BCD method, we propose the LEAF algorithm which solves Problem $\mathscr{P}_1$ by sequentially fixing two of three variables and updating the remaining one. We iterate the process until the value of each variable converges.

$\nabla y(x)$ is denoted as the partial derivative of function y corresponding to variable x. Denote $\text{Proj}_{\mathcal{X}}(x)$ as the Euclidean projection of x onto $\mathcal{X}$; $\text{Proj}_{\mathcal{X}}(x) \triangleq \arg\min_{v\in\mathcal{X}} \|x - v\|^2$.

The procedure of our proposed solution is summarized as:

- Given $\hat{s_k}$ and B_k, we can derive a new f_k according to

$$f_k^{(j+1)} = \text{Proj}_{\mathcal{X}_f}\left(f_k^{(j)} - \gamma_k \nabla Q_k\left(f_k^{(j)}\right)\right), \forall k \in \mathcal{K}; \tag{12}$$

  where $\gamma_k > 0$ is a constant step size and $\mathcal{X}_f$ is the bounded domain constrained by C_3. Based on the BCD method, we repeat (12) until the derived f_k is converged and then update f_k.
- Given f_k and B_k, we can derive a new $\hat{s_k}$ according to

$$\hat{s_k}^{(j+1)} = \text{Proj}_{\mathcal{X}_{\hat{s}}}\left(\hat{s_k}^{(j)} - \eta_k \nabla Q_k\left(\hat{s_k}^{(j)}\right)\right), \forall k \in \mathcal{K}; \tag{13}$$

  where $\eta_k > 0$ is a constant step size and $\mathcal{X}_{\hat{s}}$ is the bounded domain constrained by $\hat{C}_4$. Based on the BCD method, we repeat (13) until the derived $\hat{s_k}$ is converged and then update $\hat{s_k}$.
- Given f_k and $\hat{s_k}$, the problem is simplified to

$$\begin{aligned} \min_{\{B_k, \forall k \in \mathcal{K}\}} \quad & Q = \sum_{k\in\mathcal{K}} (E^k + \lambda_1^k L^k - \lambda_2^k A_k) \\ s.t. \quad & C_1: \sum_{k\in\mathcal{K}} B_k \le B_{max}; \\ & C_2: L^k \le L_{max}^k, \forall k \in \mathcal{K}; \end{aligned} \tag{14}$$

  where constraints C_3 and $\hat{C}_4$ are irrelevant to this problem.

The Lagrangian dual decomposition method is utilized to solve the above problem, where the Lagrangian function is

$$\mathcal{L}(B_k,\mu,\boldsymbol{\beta}) = \sum_{k\in\mathcal{K}}(E^k + \lambda_1^k L^k - \lambda_2^k A_k) + \mu(\sum_{k\in\mathcal{K}} B_k - B_{max}) + \sum_{k\in\mathcal{K}}\beta_k(L^k - L_{max}^k), \tag{15}$$

where μ and $\boldsymbol{\beta}$ are the Lagrange multipliers, (i.e., $\boldsymbol{\beta}$ is a Lagrange multiplier vector), corresponding to constraints C_1 and C_2, respectively. The Lagrangian dual problem can therefore be expressed as

$$\max_{\{\mu,\boldsymbol{\beta}\}} \quad g(\mu,\boldsymbol{\beta}) = \min_{\{B_k,\forall k\in\mathcal{K}\}} \mathcal{L}(B_k,\mu,\boldsymbol{\beta}) \quad s.t. \quad \mu \geq 0, \boldsymbol{\beta} \geq 0. \tag{16}$$

Here, $g(\mu,\boldsymbol{\beta})$ is concave with respect to B_k.

Lemma 1. *The problem $\mathscr{P}_1$ is convex with respect to B_k.*

Proof. For any feasible $B_i, B_j, \forall i,j \in \mathcal{K}$, we have

$$\frac{\partial^2 Q}{\partial B_i \partial B_j} = \begin{cases} 0, & i \neq j, \\ \Psi_i \cdot \frac{\partial^2 (1/r^{max})}{\partial B_i \partial B_j}, & i = j, \end{cases} \tag{17}$$

where $\Psi_i = \frac{\left[fps_i(E_{gt}(f_i)+E_{prv}(f_i))+P_{tr}^i(0)+P_{bs}(f_i)+\lambda_1^i\right]\sigma s_i^2}{r_i^*(f_i)}$ which is positive, and $\frac{\partial^2(1/r^{max})}{\partial B_i \partial B_j} = \frac{2}{0.677 B_i^3} > 0$. Thus, the Hessian matrix $\mathbf{H} = \left(\frac{\partial^2 Q}{\partial B_i \partial B_j}\right)_{K\times K}$ is symmetric and positive definite. Constraint C_1 is linear and C_2 is convex with respect to B_k. Constraints C_3 and C_4 are irrelevant to B_k. Therefore, $\mathscr{P}_1$ is strictly convex with respect to B_k. □

Therefore, based on the Karush-Kuhn-Tucker (KKT) condition [37], the sufficient and necessary condition of the optimal allocated bandwidth for the kth MU can be expressed as

$$B_k^* = \sqrt{\frac{\Phi(f_k, s_k, \beta_k)}{0.677\mu}}, \tag{18}$$

where $\Phi_k = \frac{\left[fps_i(E_{gt}(f_i)+E_{prv}(f_i))+P_{tr}^i(0)+P_{bs}(f_i)+\lambda_1^i+\beta_k\right]\sigma s_i^2}{r_i^*(f_i)}$.

Next, the sub-gradient method [37] is used to solve the dual problem. Based on the sub-gradient method, the dual variables of the kth MAR clients in the $(j+1)$th iteration are

$$\begin{cases} \mu_k^{(j+1)} = \max\left\{0, \left[\mu^{(j)} + \vartheta_k^{\mu}\nabla g(\mu^{(j)})\right]\right\}, \forall k \in \mathcal{K}; \\ \beta_k^{(j+1)} = \max\left\{0, \left[\beta_k^{(j)} + \vartheta_k^{\beta}\nabla g(\beta_k^{(j)})\right]\right\}, \forall k \in \mathcal{K}; \end{cases} \tag{19}$$

where $\vartheta_k^{\mu} > 0$ and $\vartheta_k^{\beta} > 0$ are the constant step sizes.

Based on the above mathematical analysis, we propose an MAR optimization algorithm, LEAF, which can dynamically determines the CPU frequency of multiple MAR devices, selects the computation model sizes, and allocates the wireless bandwidth resources. The pseudo code of the proposed LEAF MAR algorithm is presented in Algorithm 1. First, the LEAF is initialized with the lowest CPU frequency, the smallest computation model size, and evenly allocated bandwidth resources among MAR devices. We then iteratively update f_k, $\hat{s_k}$, and B_k until the LEAF converges (i.e., line 7-8 in Algorithm 1). In addition, $\hat{s_k}$ is a relaxed value of the computation model size. Thus, it may not match any pre-installed computation model in a real system. In this case, the LEAF selects the computation model size s_k that is the closest to the relaxed one $\hat{s_k}$ (i.e., line 10 in Algorithm 1). Since the LEAF MAR algorithm is developed based on the BCD method and follows the convergence results in [36], we claim that the LEAF converges to a local optimal solution.

Algorithm 1: The LEAF MAR Algorithm

Input: λ_1^k, λ_2^k, L_{max}^k, B_{max}, fps_k, and τ, $\forall k \in \mathcal{K}$.
Output: f_k, s_k, and B_k, $\forall k \in \mathcal{K}$.
1 $B_k \leftarrow B_{max}/|\mathcal{K}|$, $\hat{s_k} \leftarrow s_{min}$, $\forall k \in \mathcal{K}$, $i \leftarrow 1$;
2 **while** *True* **do**
3 $f_k \leftarrow$ solving $\mathscr{P}_1$ with fixed $\hat{s_k}$ and B_k;
4 $\hat{s_k} \leftarrow$ solving $\mathscr{P}_1$ with fixed f_k and B_k;
5 $B_k \leftarrow$ solving $\mathscr{P}_1$ with fixed f_k and $\hat{s_k}$;
6 $Q_i \leftarrow \sum_{k\in\mathcal{K}}(E^k + \lambda_1^k L^k - \lambda_2^k A_k)$
7 **if** $|(Q_i - Q_{i-1})/Q_i| \leq \tau$ **then**
8 **break;** ▷ Converges
9 $i \leftarrow i + 1$;
10 $s_k = \arg\min_{s\in\{s_{min}\ldots s_{max}\}} |s - \hat{s_k}|, \forall k \in \mathcal{K}$;
11 **return** f_k, s_k, and B_k, $\forall k \in \mathcal{K}$.

VII. Performance Evaluation

In this section, we evaluate both the proposed MAR analytical energy model and LEAF algorithm. We first validate our analytical model by comparing the estimated energy consumption with the physical energy measurement (obtained from our developed testbed described in Section III). The Mean Absolute Percentage Error (MAPE) is used for quantifying the estimation error. Then, we evaluate the per frame energy consumption, service latency, and detection accuracy of the proposed LEAF algorithm under variant bandwidth and user preferences through data-driven simulations.

A. Analytical Model Validation

The measured power and duration of promotion and tail phases in WiFi are shown in Table II (note that LTE has different values [38]). As shown in Fig. 9, we validate the proposed analytical model with respect to MAR client's CPU frequency, computation model size, allocated bandwidth, and camera FPS. Each measured data is the average of the per frame energy consumption of 1000 image frames. The calculated MAPE of these four cases are $6.1\%\pm3.4\%$, $7.6\%\pm4.9\%$, $6.9\% \pm 3.9\%$, and $3.7\% \pm 2.6\%$, respectively. Therefore, our proposed energy model can estimate the MAR per frame energy consumption very well.

TABLE II
Power and Duration of Promotion & Tail Phases.

P_{pro} (W)	t_{pro} (s)	P_{tail} (W)	t_{tail} (s)
1.97 ± 0.08	0.034 ± 0.004	1.61 ± 0.15	0.21 ± 0.02

B. Performance Evaluation of LEAF

We simulate an edge-based MAR system with an edge server and multiple MAR clients. Each MAR client may select a different camera FPS, which is obtained randomly in the range of $[1, 30]$ frames. The default user preference is $\lambda_1 = 0.3$ and $\lambda_2 = 1.8$. We compare our proposed LEAF algorithm with two other algorithms summarized as follows:

- **FACT + Interactive:** It uses FACT [20] to select the computation model size, which is optimized for the tradeoff between the service latency and the detection accuracy. As FACT does not consider the MAR client's CPU frequency scaling and radio resource allocation at the edge server, we use *Interactive* to conduct CPU

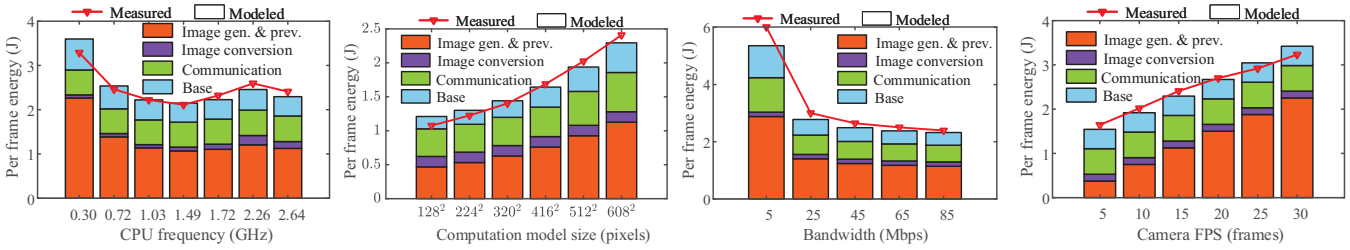
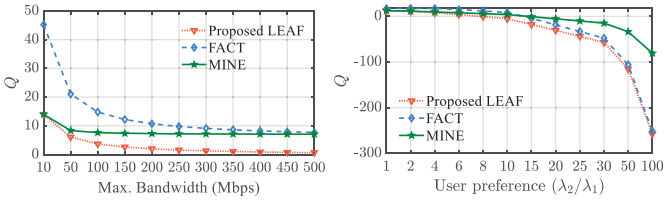
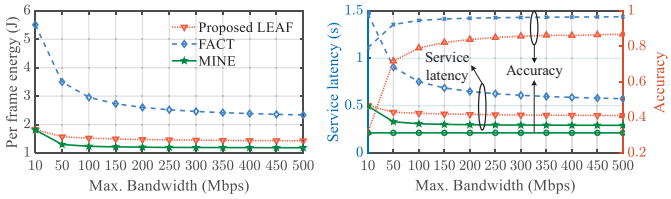


Fig. 9. Measured data vs. estimated data from our proposed analytical model.



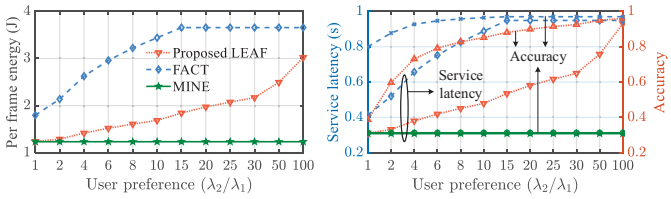
(a) Q vs. Max. bandwidth (b) Q vs. user preference

Fig. 10. Optimality.



(a) (b)

Fig. 11. System performance vs. Max. bandwidth.



(a) (b)

Fig. 12. System performance vs. user preference.

frequency scaling and the radio resource is allocated evenly. Note that FACT does not consider the energy efficiency of MAR clients either.

- **Energy-optimized only solution:** It selects the optimal CPU frequency, computation model size, and bandwidth allocation by minimizing the per frame energy consumption of MAR clients in the system without considering user preferences, which is named as MINE.

Optimality. We first validate the optimality of our proposed LEAF algorithm. As shown in Fig. 10, LEAF always obtains the minimal Q compared to the other two algorithms under variant maximum available bandwidth and user preference.

Comparison under Variant Max. Bandwidth. We then evaluate the impact of the maximum available bandwidth on the performance of the proposed LEAF. As presented in Section V-C, in practical environments, the maximum bandwidth at an edge server for serving its associated MAR clients may vary with the user distribution. For each MAR client, the value of the allocated bandwidth directly impacts not only the service latency and the per frame energy consumption but also the detection accuracy. The evaluation results are depicted in Fig. 11. (i) Compared to FACT, the proposed LEAF decreases up to 40% per frame energy consumption and 35% service latency with less than 9% loss of object detection accuracy when the Max. bandwidth is 300 Mbps. The performance gap between LEAF and FACT is due to the gain derived through optimizing the clients' CPU frequency and the server radio resource allocation. (ii) Compared to MINE, the proposed LEAF significantly improves the detection accuracy at the cost of a slightly increase of the service latency and per frame energy. The performance gap between LEAF and MINE reflects the gain derived through considering the user preference.

Comparison under Variant User Preferences. Finally, we evaluate the impact of the user preference on the performance of the proposed LEAF by varying the value of λ_2/λ_1, as shown in Fig. 12. User preference impacts the tradeoffs among the per frame energy consumption, service latency, and detection accuracy. When λ_2/λ_1 grows, the MAR client emphasizes on the detection accuracy by trading the service latency and per frame energy. Since MINE does not consider the user preference, the variation of λ_2/λ_1 does not change its performance. (i) Compared to FACT, the proposed LEAF reduces over 20% per frame energy consumption while maintaining the same detection accuracy ($\lambda_2/\lambda_1 = 100$). (ii) Compared to MINE, the proposed LEAF is able to enhance over 50% accuracy while ensuring similar per frame energy and service latency ($\lambda_2/\lambda_1 = 2$). Fig. 12 also shows that, as compared to FACT, the proposed LEAF offers more fine-grained and diverse user preference options for MAR clients.

VIII. Conclusion

In this paper, we proposed a user preference based energy-aware edge-based MAR system that can reduce the per frame energy consumption of MAR clients without compromising their user preferences by dynamically selecting the optimal combination of MAR configurations and radio resource allocations according to user preferences, camera FPS, and available radio resources at the edge server. To the best of our knowledge, we built the first analytical energy model for thoroughly investigating the interactions among MAR configuration parameters, user preferences, camera sampling rate, and per frame energy consumption in edge-based MAR systems. Based on the proposed analytical model, we proposed the LEAF optimization algorithm to guide the optimal MAR configurations and resource allocations. The performance of the proposed analytical model is validated against real energy measurements from our testbed and the LEAF algorithm is evaluated through extensive data-driven simulations.

REFERENCES

[1] L. N. Huynh, Y. Lee, and R. K. Balan, "Deepmon: Mobile GPU-based deep learning framework for continuous vision applications," in *Proc. ACM Mobisys*, 2017, pp. 82–95.

[2] H. Wang, B. Kim, J. Xie, and Z. Han, "How is energy consumed in smartphone deep learning apps? Executing locally vs. remotely," in *Proc. IEEE Globecom*, 2019, pp. 1–6.

[3] Y. Xiao, Y. Cui, P. Savolainen, M. Siekkinen, A. Wang, L. Yang, A. Ylä-Jääski, and S. Tarkoma, "Modeling energy consumption of data transmission over Wi-Fi," *IEEE Transactions on Mobile Computing*, vol. 13, no. 8, pp. 1760–1773, 2013.

[4] H. Wang, J. Xie, and X. Liu, "Rethinking mobile devices' energy efficiency in WLAN management services," in *Proc. IEEE SECON*, 2018, pp. 1–9.

[5] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck, "A close examination of performance and power characteristics of 4G LTE networks," in *Proc. ACM MobiSys*, 2012, pp. 225–238.

[6] H. Wang, J. Xie, and T. Han, "V-handoff: A practical energy efficient handoff for 802.11 infrastructure networks," in *Proc. IEEE ICC*, 2017, pp. 1–6.

[7] A. M. Srivatsa and J. Xie, "A performance study of mobile handoff delay in IEEE 802.11-based wireless mesh networks," in *Proc. IEEE ICC*, 2008, pp. 2485–2489.

[8] X. Liu and J. Xie, "A practical self-adaptive rendezvous protocol in cognitive radio ad hoc networks," in *Proc. IEEE INFOCOM*, 2014, pp. 2085–2093.

[9] U. Narayanan and J. Xie, "Signaling cost analysis of handoffs in a mixed IPv4/IPv6 mobile environment," in *Proc. IEEE GLOBECOM*, 2007, pp. 1792–1796.

[10] H. Wang, J. Xie, and T. Han, "A smart service rebuilding scheme across cloudlets via mobile AR frame feature mapping," in *Proc. IEEE ICC*, 2018, pp. 1–6.

[11] A. Shye, B. Scholbrock, and G. Memik, "Into the wild: studying real user activity patterns to guide power optimizations for mobile architectures," in *Proc. IEEE/ACM International Symposium on Microarchitecture*, 2009, pp. 168–178.

[12] M. J. Walker, S. Diestelhorst, A. Hansson, A. K. Das, S. Yang, B. M. Al-Hashimi, and G. V. Merrett, "Accurate and stable run-time power modeling for mobile and embedded CPUs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 1, pp. 106–119, 2016.

[13] K. DeVogeleer, G. Memmi, P. Jouvelot, and F. Coelho, "Modeling the temperature bias of power consumption for nanometer-scale CPUs in application processors," in *Proc. IEEE International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS XIV)*, 2014, pp. 172–180.

[14] F. Xu, Y. Liu, Q. Li, and Y. Zhang, "V-edge: Fast self-constructive power modeling of smartphones based on battery voltage dynamics," in *Proc. USENIX Symposium on Network Systems Design and Implementation (NSDI)*, 2013, pp. 43–55.

[15] W. Hu and G. Cao, "Energy-aware CPU frequency scaling for mobile video streaming." in *Proc. IEEE ICDCS*, 2017, pp. 2314–2321.

[16] ——, "Energy optimization through traffic aggregation in wireless networks," in *Proc. IEEE INFOCOM*, 2014, pp. 916–924.

[17] H. Wang, B. Kim, J. Xie, and Z. Han, "E-auto: A communication scheme for connected vehicles with edge-assisted autonomous driving," in *Proc. IEEE ICC*, 2019, pp. 1–6.

[18] Y. Geng, Y. Yang, and G. Cao, "Energy-efficient computation offloading for multicore-based mobile devices," in *Proc. IEEE INFOCOM*, 2018, pp. 46–54.

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

[20] Q. Liu, S. Huang, J. Opadere, and T. Han, "An edge network orchestrator for mobile augmented reality," in *Proc. IEEE INFOCOM*, 2018, pp. 756–764.

[21] J. Hanhirova, T. Kämäräinen, S. Seppälä, M. Siekkinen, V. Hirvisalo, and A. Ylä-Jääski, "Latency and throughput characterization of convolutional neural networks for mobile computer vision," in *Proc. 9th ACM Multimedia Systems Conference*, 2018, pp. 204–215.

[22] J.-J. Chen, C.-Y. Yang, T.-W. Kuo, and C.-S. Shih, "Energy-efficient real-time task scheduling in multiprocessor DVS systems," in *Proc. IEEE Asia and South Pacific Design Automation Conference*, 2007, pp. 342–349.

[23] J. Kwak, O. Choi, S. Chong, and P. Mohapatra, "Dynamic speed scaling for energy minimization in delay-tolerant smartphone applications," in *Proc. IEEE INFOCOM*, 2014, pp. 2292–2300.

[24] W. Y. Lee, "Energy-saving DVFS scheduling of multiple periodic real-time tasks on multi-core processors," in *Proc. 13th IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications*, 2009, pp. 216–223.

[25] J. Redmon, "Darknet: Open source neural networks in C," http://pjreddie.com/darknet/, 2013–2016.

[26] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," *arXiv*, 2018.

[27] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Proc. European Conference on Computer Vision*, 2014.

[28] "Monsoon power monitor," https://www.msoon.com/.

[29] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.

[30] K. Ha, Z. Chen, W. Hu, W. Richter, P. Pillai, and M. Satyanarayanan, "Towards wearable cognitive assistance," in *Proc. 12th annual international conference on Mobile systems, applications, and services*. ACM, 2014, pp. 68–81.

[31] "SketchAR." [Online]. Available: https://itunes.apple.com/us/app/sketchar-drawing-using-augmented-reality/id1221482822?l=ru&mt=8

[32] C. J. Willmott and K. Matsuura, "Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance," *Climate Research*, vol. 30, no. 1, pp. 79–82, 2005.

[33] K. Deb, *Multi-objective Optimization Using Evolutionary Algorithms*. John Wiley & Sons, 2001, vol. 16.

[34] R. T. Marler and J. S. Arora, "The weighted sum method for multi-objective optimization: new insights," *Structural and Multidisciplinary Optimization*, vol. 41, no. 6, pp. 853–862, 2010.

[35] P. Belotti, C. Kirches, S. Leyffer, J. Linderoth, J. Luedtke, and A. Mahajan, "Mixed-integer nonlinear optimization," *Acta Numerica*, vol. 22, pp. 1–131, 2013.

[36] L. Grippo and M. Sciandrone, "On the convergence of the block nonlinear gauss–seidel method under convex constraints," *Operations Research Letters*, vol. 26, no. 3, pp. 127–136, 2000.

[37] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[38] W. Hu and G. Cao, "Energy-aware video streaming on smartphones," in *Proc. IEEE INFOCOM*, 2015, pp. 1185–1193.