EcoPull

Sustainable IoT Image Retrieval Empowered by TinyML Models

Thorsager, Mathias; Croisfelt, Victor; Shiraishi, Junya; Popovski, Petar

Published in:
GLOBECOM 2024 - 2024 IEEE Global Communications Conference

DOI (link to publication from Publisher):
10.1109/GLOBECOM52923.2024.10901782

Publication date:
2024

Document Version
Accepted author manuscript, peer reviewed version

EcoPull: Sustainable IoT Image Retrieval Empowered by TinyML Models

Mathias Thorsager*, Victor Croisfelt*, Junya Shiraishi*, and Petar Popovski
Department of Electronic Systems, Aalborg University, Denmark
Email: {mdth, vcr, jush, petarp}@es.aau.dk

***Abstract*—This paper introduces EcoPull, a sustainable Internet of Things (IoT) framework empowered by tiny machine learning (TinyML) models for fetching images from wireless visual sensor networks. Two types of learnable TinyML models are installed in the IoT devices: i) a *behavior model* and ii) an *image compressor model*. The first filters out irrelevant images for the current task, reducing unnecessary transmission and resource competition among the devices. The second allows IoT devices to communicate with the receiver via latent representations of images, reducing communication bandwidth usage. However, integrating learnable modules into IoT devices comes at the cost of increased energy consumption due to inference. The numerical results show that the proposed framework can save > 70% energy compared to the baseline while maintaining the quality of the retrieved images at the ES.**

I. Introduction

One of the key requirements for Internet of Things (IoT) devices in the context of 6G is sustainability [1], [2]. The IoT devices will also be affected by the trend of using Artificial Intelligence (AI)/Machine Learning (ML) to support various intelligent tasks at the edge [3]. However, IoT devices are challenged by resource constraints in terms of energy, computation, and memory, which hinders the implementation of learning models on these devices. First, the practical Micro Controller Unit (MCUs) of the IoT devices only have a small amount of memory and storage, *e.g.*, on-chip Static Random Access Memory (SRAM) (< 512 KB) and flash storage (< 2 MB) [4]. Second, running ML models require additional energy cost for IoT devices. Therefore, one must design a communication protocol considering the energy cost associated with the ML model and the main radio interface.

In [5], we introduced Tiny Neural Network transmission over the Air (TinyAirNet), an IoT image retrieval framework that reduces the overall energy consumption of the IoT devices. In TinyAirNet, a TinyML model is transmitted by the edge server (ES) to the IoT devices before they collect new images to facilitate the subsequent semantic query, by which the IoT device selectively transmits relevant images and avoids energy waste. Although this enables the suppression of images likely unrelated to the current task, it still does not prevent energy consumption for sending raw images. One solution is to compress the images so the receiver can reconstruct and exploit its side knowledge. For example, one can deploy a generative model, such as a Generative Adversarial Network (GAN), to compress the raw images, where recent studies [6], [7] show that it can achieve high perceptional quality compared with conventional image compression methods.

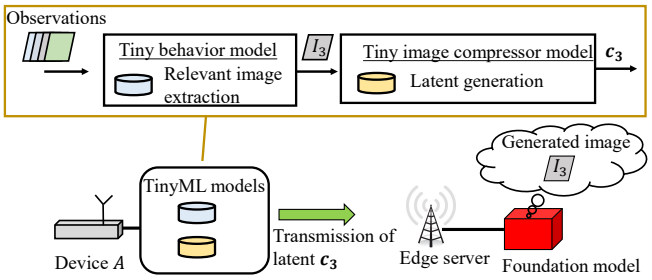

Fig. 1: EcoPull, a sustainable IoT image retrieval framework in which two types of TinyML models are installed into IoT devices: i) a *behavior model* and ii) an *image compressor model.*

Contributions: In this work, we present EcoPull, an IoT image retrieval framework in which the ES pulls that from IoT devices equipped with two types of TinyML models: i) a *behavior model* and ii) an *image compressor model*. Fig. 1 exemplifies the proposed framework, where the ES equipped with a Foundation Model (FM) attempts to obtain the information that an IoT device A observed at a given time. Each IoT device first checks the relevance of the images against the ES's current task by leveraging the installed behavior model to filter out irrelevant images, *e.g.*, by conducting feature extraction and calculating a matching score as in [5], [8]. Next, the IoT device encodes the relevant images, resulting in latent feature vector representations, which we refer to as *latents* [9]. The ES reconstructs the latents using a decoder. We show that the interplay of these two models allows us to reduce the energy consumption of IoT devices to a greater extent while maintaining the quality of the retrieved images at the ES. In particular, we introduce a new performance metric called *Significance and Fidelity (SiFi) that jointly evaluates the significance and the fidelity of the image retrieved at the ES.* We describe our framework from a Medium Access Control (MAC) layer perspective, focusing on the scenario where multiple devices share a common medium.

Related work: We consider that our image compressor model is based on a generative model. Generative AI (G-AI) has been introduced in communication systems, such as in the network layer [10] and the federated learning setup [11]. In

*Mathias T., Victor C., and Junya S. contributed equally to this work. This work was partly supported by the Villum Investigator Grant "WATER" from the Velux Foundation, Denmark, partly by the Horizon Europe SNS "6G-XCEL" project with Grant 101139194, and partly by the Horizon Europe SNS "6G-GOALS" project with grant 101139232.

addition, the concept of Generative Internet of Things (GIoT) was introduced in [12], in which a potential application and its challenges are discussed. In [13], the authors introduced G-AI-aided semantic communication to realize accurate content decoding. Our work relates to these in the sense of only transmitting the intended meaning over the channel thanks to the compression and shared knowledge [14]. However, we focus on reducing and analyzing the overall energy consumption of IoT devices considering the additional energy cost of the ML models, which has been overlooked in the literature.

II. The EcoPull Framework

A. Scenario

We consider a scenario where an ES is interested in retrieving images from K IoT devices, such as mobile robots or fixed-mounted cameras, deployed over a sensing field to fulfill a task requested by a user, such as a mobile user, through the cloud via a *query*. The ES has a FM to *interpret* the user's query. For example, the query could comprise a `prompt`, and the FM could be based on a Generative Pre-trained Transformer (GPT) architecture. We then consider a *pull-based communication system* where the ES retrieves data from the devices based on a piece of *side information* given by the ES' *interpretation* of the query sent by the user. The ES and IoT devices communicate over an idealized slotted multiaccess channel where *packets*, which represent the minimal piece of information, have the same length of a *slot* of duration T_p [s] or equivalently b_p bits [15]. We define a *frame* as a collection of L *slots* whose total number is F. We assume an error-free downlink channel and that the packets are lost in case a collision in the uplink occurs during a slot.

The $i-$th device, where $i \in \mathcal{K} = \{1, \ldots, K\}$, locally stores the collected images in unlabeled datasets $\mathcal{D}^i = \{\mathbf{x}_1^i, \ldots, \mathbf{x}_N^i\}$. The number of images is denoted by $|\mathcal{D}^i| = N^i$ and $\mathbf{x}_n^i \in \mathbb{R}^{M_C \times M_H \times M_W}$ represents the original image with M_C channels, where M_H and M_W are the height and width of the image in pixels, respectively. When referring to a generic image, we drop the indices i and n as $\mathbf{x}$. The index set of images at the i-th device is denoted as $\mathcal{N}^i = \{1, \ldots, N^i\}$. As in [16], we assume each device has fixed-point arithmetic hardware equipped with a two-dimensional processing chip for Convolution Neural Networks (CNNs). This consists of an off-chip Dynamic Random Access Memory (DRAM) that works with $b_{\max}$ quantization bits (full precision), a parallel neuron array with p Multiply-Accumulates (MUACs) units of b_{MUAC} bits, and two on-chip memory levels: a main SRAM buffer that stores, *e.g.*, the weights and activations, and a local SRAM buffer that caches currently used weights and activations. We let b_q be the number of bits for quantization in the SRAM. Each unlabeled dataset $\mathcal{D}^i$ is stored in the DRAM with precision $b_{\max}$. Moreover, each device has stored two TinyML models in the DRAM: i) a *behavior model* and ii) an *image compressor model*, see Sec. II-B.

B. Description of the Scheme

The proposed scheme comprises four distinct phases:

Phase 1. Downlink pulling phase: Triggered by a query from a user, the ES fetches data from the devices. First, the ES transmits the trained TinyML models towards the devices to facilitate the subsequent semantic query, as in [5] – this can be done as the models are updated.[1] Then, the ES broadcasts a b_q-quantized vector semantic feature vector $\mathbf{q} \in \mathbb{R}^M$ obtained by the FM to the devices over the downlink channel.

Phase 2. Behavioral phase: After receiving the semantic query, the devices use their behavior model to compute a set of similarity measures $\{s_n^i\}_{n=1}^{N^i}$ between their images in $\mathcal{D}^i$ and $\mathbf{q}$, where $s_n^i \in [0, 1]$. Let $V_{\text{th}} \in [0, 1]$ be a *relevance threshold*, which can be set by the ES. The $i-$th device extracts the set of relevant images $\mathcal{D}_s^i \subseteq \mathcal{D}^i$ as

$$\mathcal{D}_s^i = \{\mathbf{x}_n^i | s_n^i \geq V_{\text{th}}, \forall n \in \mathcal{N}^i\}. \tag{1}$$

Let $|\mathcal{D}_s^i| = S^i$ be the number of relevant images. If $S^i > 1$, the $i-$th device is a *relevant device* for the given query. The set of relevant devices $\mathcal{K}_s \subseteq \mathcal{K}$ is defined as $\mathcal{K}_s = \{i | S_i > 0, \forall i \in \mathcal{K}\}$, where $|\mathcal{K}_s| = K_s$ is the number of relevant devices.

Phase 3. Uplink compression-then-transmission phase: At the beginning of this phase, the relevant devices compress their relevant images using the image compressor model. Let $C(\cdot)$ denote the tiny encoder and $\mathbf{z} \in \mathbb{R}^{M'_C \times M'_H \times M'_W}$ denote a quantized latent variable, whose quantitation occurs according to $b_{\max}$ and where M'_C is the new number of channels and M'_H and M'_W are the new height and width of the compressed image. Thus, the encoding is represented as $\mathbf{z} = C(\mathbf{x})$. For the $i-$th device, the compressed set of relevant images is:

$$\check{\mathcal{D}}_s^i = \{\mathbf{z}_n^i | \forall \mathbf{x}_n^i \in \mathcal{D}_s^i\}, \tag{2}$$

with $|\check{\mathcal{D}}_s^i| = S^i$. Each device stores the S^i images in a queue $Q^i \leftarrow \check{\mathcal{D}}_s^i$ in the DRAM with precision $b_{\max}$. Next, the relevant devices transmit their compressed relevant images over the $f-$th frame, where we tie the size of a packet to the size of the latent as $b_p = M'_C \cdot M'_H \cdot M'_W \cdot b_{\max}$ bits, where we ignore the header size for simplicity of analysis.

For simplicity, we assume that each relevant device attempts to transmit a single image per frame by choosing a slot at random. With this approach, a number of K_s or less devices compete for the L slots at a given frame. We assume they do so by adopting the slotted ALOHA protocol [15] without re-transmissions to showcase the main benefits of our framework and ease the analysis. After an attempt to transmit an image, the queue is updated as $Q^i \leftarrow Q^i \setminus \mathbf{z}_f^i$, where $\mathbf{z}_f^i$ denotes the image selected at random by the $i-$th device to be transmitted at the $f-$th frame. When a device empties its queue, it becomes idle and stops competing in the next slots.

Phase 4. Data decompression and response to the user: Let $G(\cdot)$ be the tiny decoder stored at the ES. After the F frames are over, the ES collects and decompresses all the non-colliding packets, whose decoding operation is denoted as $\hat{\mathbf{x}} = G(\mathbf{z})$. Then, the ES identifies the top l most relevant

[1] In particular, we assume that the behavior model is sent more frequently than the image compressor since its accuracy depends on the current query. Therefore, we model the energy spent to receive the behavior model in Sec. III, while the cost to receive the image compressor is ignored.

images for the specific query, with, *e.g.*, $l = 3$, to be sent as a reply to the user as a form of text or image by exploiting the FM. To measure the system's performance, we introduce SiFi in Sec. IV.

Based on [5], we assume that the behavior model is a multimodal feature extractor $\zeta : \mathbf{x} \mapsto \mathbf{o}$ based on a FM that can take as input images/texts $\mathbf{x}$ and has as output $\mathbf{o} \in \mathbb{R}^M$. The ES should compress ζ – *e.g.*, by using pruning – to send it to the devices, as described in Phase 1. Because of that, we assume that the devices receive a model $\tilde{\zeta}$ that performs worse than ζ. Given that and the description of Phase 2, the similarity measure at the $i-$th device is computed as $s_n^i = g(\tilde{\zeta}(\mathbf{x}_n^i), \mathbf{q})$, where $g(\cdot)$ is a similarity function, such as cosine similarity.

The fact that $\tilde{\zeta}$ performs worse than ζ is modeled by how much the similarity measure s_n^i deviates from the true similarity measure $\beta_n^i = g(\zeta(\mathbf{x}_n^i), \mathbf{q})$. Specifically:

$$s_n^i = \beta_n^i + w_n^i, \tag{3}$$

where $w_n^i \sim \mathcal{N}(0, \sigma_{\text{ML}}^2)$ is Gaussian distributed and σ_{ML} is a standard deviation representing the model noise. For simplicity, we model the quantization effect from getting $\tilde{\zeta}$ of ζ by setting $\sigma_{\text{ML}} = \frac{1}{b_B}$, where $b_B \leq b_{\max}$ is the number of bits for the quantization of weights of the behavior model for transmission [5].

As for the tiny image compressor model, we adopt the framework presented in [7] that introduces a generative image compression method. Here, we assume that $C(\cdot)$ and $G(\cdot)$ are jointly trained before the model deployment, *e.g.*, at the cloud. We refer the interested reader to [7] for more details about training and architecture. The compression rate r, measured in bits per pixel (bpp), is

$$r = \frac{M_C' \cdot M_H' \cdot M_W' \cdot b_{\max}}{M_H \cdot M_W}. \tag{4}$$

III. Energy Consumption

We consider the energy cost of communication and computation at the IoT devices while ignoring the energy consumed during idle periods. We let ξ_T [W] be the energy cost per uplink transmission and ξ_R [W] be the energy cost for receiving. Next, a general description of the *energy cost per inference of a model*, E_{inf}, follows based on [16].[2] The energy cost per inference of a model is given by:

$$E_{\text{inf}} = E_{\text{DRAM}} + E_{\text{HW}}, \tag{5}$$

comprised of the energy cost of accessing the DRAM where the model and input are stored, denoted as E_{DRAM}, and that of the actual processing for inference denoted as E_{HW}. Then, E_{DRAM} can be expressed as follows:

$$E_{\text{DRAM}} = E_D \cdot [M_C \cdot M_H \cdot M_W] \cdot (b_{\max}/b_q), \tag{6}$$

where E_D is the energy consumed per b_q-bit DRAM access and $M_C \cdot M_H \cdot M_W$ is the input size. While E_{HW} can be modeled as the sum of the computing energy, E_C, and the costs of moving weights and activations from the main to the local SRAM, E_W and E_A, respectively, as described below:

$$E_{\text{HW}} = E_C + E_W + E_A, \tag{7}$$

with $E_C = E_{\text{MUAC}}(U + 3A)$, $E_W = E_M W + E_L U/\sqrt{p}$, and $E_A = 2E_M A + E_L U/\sqrt{p}$, where E_{MUAC} is the energy consumed for a single MUAC operation, $E_L = E_{\text{MUAC}}$ is the energy consumption to read/write from/to the local SRAM, and $E_M = 2E_{\text{MUAC}}$ is the energy consumed for accessing the main SRAM. Other parameters are U, which denotes the network complexity in the number of MUAC operations; W, which denotes the model size in the number of weights and biases; A, which denotes the total number of activations throughout the whole network.

We now instantiate the inference model in (5) for the behavior and the image compressor models. For the behavior model, we let $E_{\text{inf},B}$ denote the energy cost per inference and (U_B, W_B, A_B) be the set of parameters. For the image compressor model, we let $E_{\text{inf},C}$ denote the energy cost per inference and (U_C, W_C, A_C) be the set of parameters.

For the $i-$th device, the *total energy consumed for computation* can be written as:

$$E_{\text{comp}}^i = N^i E_{\text{inf},B} + S^i E_{\text{inf},C} + E_D \cdot (W_B + W_C) \cdot (b_{\max}/b_q), \tag{8}$$

the last term of sum measures the cost of loading the models from the DRAM to the main SRAM. While the *total energy consumed for communication* can be written as:

$$E_{\text{comm}}^i = \frac{(\xi_R \cdot (W_B b_B + M b_q) + \xi_T \cdot S^i b_p)}{R}, \tag{9}$$

where ξ_R multiplies the time of receiving the behavior model and the vector $\mathbf{q}$, ξ_R multiplies the time of transmitting the S^i relevant images, and R is the rate of the downlink and uplink channels in bits per second. *The total energy consumed per device* is then:

$$E_{\text{tot}}^i = E_{\text{comp}}^i + E_{\text{comm}}^i. \tag{10}$$

We note that S^i is a random variable that depends on V_{th} as per (1), so is E_{tot}^i. Let $P_{\text{th}}(V_{\text{th}})$ denote the probability that an image is relevant, *i.e.*, the probability s_n^i in eq. (3) is equal to or higher than V_{th}. We can then derive the *expected total energy consumed* given the model in (3). This can be expressed by considering the distribution of observed similarity measure s given true similarity measure β, $p(s|\beta) = \frac{1}{\sqrt{2\pi\sigma^2}} \int_{-\infty}^{\infty} \exp\left(-\frac{(s-\beta)^2}{2\sigma_{\text{ML}}^2}\right) ds$, as follows:

$$P_{\text{th}}(V_{\text{th}}) = \int_0^1 Q\left(\frac{V_{\text{th}} - \beta}{\sigma_{\text{ML}}}\right) g_T(\beta)\, d\beta, \tag{11}$$

where $Q(x)$ denotes the Q-function defined as $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^{\infty} \exp\left(-\frac{u^2}{2}\right) du$ and $g_T(\beta)$ is the distribution of true similarity measure β. We let $B(p, n, k)$ be the binomial distribution, with probability p and total number of elements n with $B(p, n, k) = \binom{n}{k} p^k (1-p)^{n-k}$. Then, the probability that a device has ν relevant images out of N^i images can

[2] We assume that the main SRAM of the devices is large enough to store feature maps and to hold the entire TinyML model after being loaded from the off-chip DRAM; unused feature maps are erased from the memory.

be expressed as $P_{\text{Rel}}(\nu) = B(P_{\text{th}}(V_{\text{th}}), N^i, \nu)$. Finally, the *expected total energy consumed per device* can be expressed as follows:

$$\mathbb{E}[E^i_{\text{tot}}] = \sum_{\nu=0}^{N^i} \nu \left(E_{\text{inf},C} + \frac{\xi_T b_p}{R} \right) P_{\text{Rel}}(\nu) + \epsilon, \tag{12}$$

where $\epsilon = N^i E_{\text{inf},B} + E_D \cdot (W_B + W_C) \cdot (b_{\max}/b_q) + \xi_R (W_B b_B + M b_q)/R$.

IV. Performance Analysis

A. Measuring the significance and fidelity of retrieved images

To measure the performance of the retrieved image at Phase 4, we introduce SiFi, a metric that can track two distinct objectives simultaneously: *significance* and *fidelity*. On the one hand, the significance represents how many *actual relevant images* can be successfully collected, where the image whose true similarity measure β is equal to or higher than δ is referred to as an *actual relevant image*. On the other hand, the fidelity represents how similar the reconstructed image is to the original one. The fidelity can be measured over the reconstructed image, denoted as $k_d(\mathbf{x}, \hat{\mathbf{x}}) \in [0, 1]$, where k_d is a perceptual quality measure [17], such as Fréchet Inception Distance (FID) [17], which evaluates to 0 when the reconstructed image is the same compared to the one observed by the IoT device. We note that $k_d(\mathbf{x}, \hat{\mathbf{x}})$ varies according to the compression rate r, whose dependency can be denoted as $k_d(\mathbf{x}, \hat{\mathbf{x}}; r)$, as found in [10]. Finally, we can define the SiFi of the proposed framework, $\vartheta \in [0, 1]$. Let $T^i_n \in \{0, 1\}$ be a Bernoulli random variable which models if the transmission of the n-th image of the i-th device was successful, and Ω be the subset of index pairs (i, n) for the actual relevant images, *i.e.*, $\Omega = \{(i, n) \mid \beta^i_n \geq \delta, \forall n \in \mathcal{N}^i, \forall i \in \mathcal{K}\}$ and $|\Omega|$ its cardinality. Then, the SiFi can be defined as:

$$\vartheta = \begin{cases} 1 - \frac{1}{|\Omega|} \sum_{(i,n) \in \Omega} k_d(\mathbf{x}^i_n, \hat{\mathbf{x}}^i_n; r)^{T^i_n} \Gamma^{1-T^i_n}, & \text{if } |\Omega| \geq 1, \\ 1, & \text{otherwise,} \end{cases} \tag{13}$$

where $\Gamma \in [0, 1]$ is a penalization term if the data transmission of an actual relevant image has failed. Here, Γ should be chosen to be greater than $k_d(\mathbf{x}^i_n, \hat{\mathbf{x}}^i_n; r)$. A higher SiFi, ϑ, implies that the ES retrieves the useful data that represents the current/accurate views of the observations of IoT devices regarding the user's query. SiFi can be improved by collecting more data at the expense of increased energy consumption.

B. Expected SiFi

We derive the *proposed framework's expected SiFi*. Without loss of generality, we assume that $N^i = N$, $\forall i \in \mathcal{K}$. We start by deriving the average transmission success probability, defined as the probability that a packet is delivered to the ES without collision. This probability depends on the number of relevant images each device has, S^i, which we model as follows. Let $\{X_\nu\}_{\nu=0}^N$ denote the random variables representing the number of devices having ν relevant images. Then, $\mathbf{X} = \{X_0, \ldots, X_N\}$ follows a multinomial distribution, whose Probability Mass Function (PMF) of $\psi = \{q_0, \ldots, q_N\}$ is:

$$\mathbb{P}(\mathbf{X} = \psi) = \begin{cases} K! \prod_{\nu=0}^N \frac{P_{\text{Rel}(\nu)}^{q_\nu}}{q_\nu!}, & (\sum_{\nu=0}^N q_\nu = K), \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

Given a realization sample $\psi_u = \{q^u_0, q^u_1, \ldots, q^u_N\}$, the number of devices which attempts data transmission in the f-th frame can be expressed as follows:

$$W^u_f = \sum_{j=f}^{N} q^u_j. \tag{15}$$

Using eq. (15), the total number of frames required to collect all relevant images, denoted as n^u_w, can be described as:

$$n^u_w = \arg\min_{J \in [0,N]} \left(q_0 + \sum_{f=1}^{J} q^u_f = K \right). \tag{16}$$

This can be related to the number of frames F that the ES sets. Ideally, we would like to have $F \leq n^u_w$. Based on the eqs. (15) and (16), we can derive the average transmission success probability per relevant image as:

$$P_s(\psi_u) = \begin{cases} \frac{1}{n^u_w} \sum_{f=1}^{n^u_w} \left(1 - \frac{1}{L}\right)^{W^u_f - 1}, & \text{if } n_w \geq 1, \\ 1, & \text{otherwise.} \end{cases} \tag{17}$$

Then, we must consider how many relevant images succeed in data transmission. Recall from Sec. III that the actual relevant image is defined as the image whose true similarity measure β is higher than the threshold δ. Here, the probability of an image is actually relevant, denoted as P_δ, can be expressed as $P_\delta = \int_\delta^1 g_T(\beta) d\beta$. For the actual relevant data to be successfully delivered to the ES given ψ_u, it needs to satisfy two conditions: 1) the observed similarity measure s in eq. (3) should be higher than V_{th}, and 2) the transmission of the data should be successful. Accordingly, the probability that the ES can successfully collect an actual relevant image can be expressed with the conditional probability:

$$P_A(\psi_u) = \frac{P_s(\psi_u)}{P_\delta} \int_\delta^1 Q\left(\frac{V_{\text{th}} - \beta}{\sigma_{\text{ML}}} \right) g_T(\beta) d\beta. \tag{18}$$

Now we can derive the expected SiFi in eq. (13). First, we focus on the case where $|\Omega| \geq 1$, which happens with the probability $P_\Omega = 1 - (1 - P_\delta)^{KN}$. Let $\kappa > 0$ denote the random variable representing the total number of actual relevant images in the sensing field and Z^u_ι be the random variable for ι-th actual relevant image given ϕ_u, which takes the value of $1 - k_d(\mathbf{x}^i_n, \hat{\mathbf{x}}^i_n | r)$ if its image is delivered to the ES without collisions; otherwise, Γ. Then, its expectation is:

$$\mathbb{E}[Z^u_\iota] = (1 - k_d(\mathbf{x}^i_n, \hat{\mathbf{x}}^i_n | r)) P_A(\psi_u) + (1 - P_A(\psi_u))(1 - \Gamma). \tag{19}$$

Using this, the expected SiFi for $|\Omega| \geq 1$, given ψ_u can be calculated as $\frac{1}{\kappa} \mathbb{E}[\sum_{\iota=1}^\kappa Z^u_\iota] = \frac{1}{\kappa} \sum_{\iota=1}^\kappa \mathbb{E}[Z^u_\iota] = \mathbb{E}[Z^u_\iota]$, where we apply the linearity of expectation for the independent random variables $Z^u_1, Z^u_2, \ldots, Z^u_\kappa$. On the other hand, the case of $|\Omega| = 0$ happens with probability $1 - P_\Omega$, in which case the

SiFi is 1, as in eq. (13). Finally, from the above description, the expected value of SiFi can be expressed as follows:

$$\mathcal{V} = \sum_{u=1}^{|\psi|} \mathbb{P}(\psi_u) \left(\mathbb{E}[Z_t^u] P_\Omega + (1 - P_\Omega) \right). \tag{20}$$

C. Getting approximation of expected SiFi

As the calculation of SiFi in eq. (20) requires consideration of all $\binom{K+N}{K}$ realizations of multinomial distribution given in eq. (14), we resort to the Markov Chain Monte Carlo (MCMC) method [18] to obtain approximate results. Specifically, we employ the Metropolis algorithm [18] that generates a sequence of samples (realizations) $Y^{(t)}$ ($t \in [0, T]$) based on probabilistic rules. Here, we denote the number of devices that have ν relevant images at the t-th realization sample as $q_\nu^{(t)}$, which needs to satisfy the constraint, $\sum_{\nu=0}^{N} q_\nu^{(t)} = K$ for all t. Here, an initial state $Y^{(0)}$ is set to fairly distribute the number of devices observing ν images. Then, the sequence of samples is generated as follows:

Step 1. Generate new sample: Create a new sample Y' from the current state $Y^{(t)}$. First, we randomly select one value $a \in [0, N]$ out of $N+1$ values, satisfying $q_a^{(t)} \geq 1$ and decrease $q_a^{(t)}$ by 1. Then, we randomly select another value b satisfying $q_b^{(t)} < N$ and increase $q_b^{(t)}$ by 1. With these operations, a new sample Y' can be generated.

Step 2. Calculate transition cost: Calculate transition cost as $\mathbb{P}(Y')/\mathbb{P}(Y^{(t)})$, by using PMF given by eq. (14). More details on transition cost can be found in [18].

Step 3. Update state: Generate a random number $r' \in [0, 1]$, following uniform distribution, and decide the next state as:

$$Y^{(t+1)} = \begin{cases} Y', & (\text{if } r' \leq \frac{\mathbb{P}(Y')}{\mathbb{P}(Y^{(t)})}), \\ Y^{(t)}, & \text{otherwise.} \end{cases} \tag{21}$$

The above operation is repeated over T times, and for each $Y^{(t)}$, the SiFi is calculated through the equations derived in Sec. IV-B and stored as $u^{(z)}$. Finally, the approximate SiFi is:

$$\hat{\mathcal{V}} = \frac{1}{T} \sum_{t=1}^{T} u^{(t)}. \tag{22}$$

V. Numerical Evaluation

This section investigates the performance of EcoPull. We consider images as $(M_C, M_H, M_W) = (3, 640, 480)$ [7]. The values of E_{MUAC} and E_D are: $E_{\text{MUAC}} = 3.7 \cdot (b_q/b_{\text{MUAC}})^{1.25}$ [pJ] and $E_D = 128 \cdot 3.7 \cdot (b_q/b_{\text{MUAC}})$ [pJ], respectively [16]. Moreover, we use $b_{\max} = 16$, $b_{\text{MUAC}} = 16$, $b_B = 8$, $b_q = 8$, and $p = 64 \cdot b_{\text{MUAC}}/b_q$ for the behavior model based on [16] and $b_q = 16$ for the image compressor model [7]. Using [19], we set $\xi_T = 108$ [mW], $\xi_R = 66.9$ [mW]. For the behavior model, $(U_B, W_B, A_B) = (117\text{ M}, 0.976\text{ M}, 4.309\text{ M})$ [20]. For the image compressor model, $(U_C, W_C, A_C) = (477\text{ M}, 0.0184\text{ M}, 3.54\text{ M})$ [7]. We set $R = 100$ [kbps], $\delta = 0.9$, and number of MCMC rounds $T = 10^4$. For simplicity, we generate $g_T(\beta)$ based on uniform distribution within $[0, 1]$ and assume that $N^i = N$, $\forall i \in \mathcal{K}$. We conduct the Monte Carlo simulation over 10^4 rounds to obtain the averaged results.

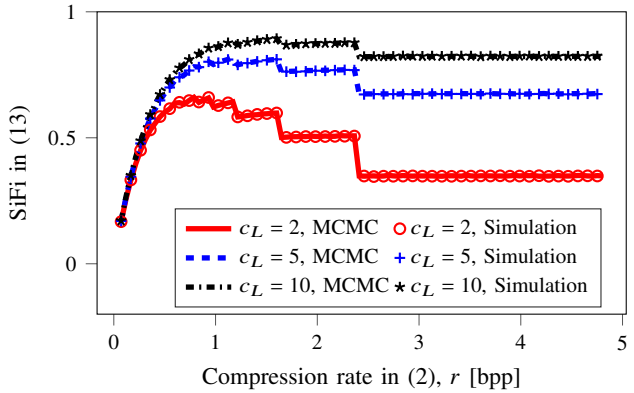

Fig. 2: SiFi in (13) against the compression rate r in [bpp]. 'MCMC' refers to the approximation of the expected SiFi in (20), while 'Simulation' refers to the one obtained by averaging (13). The number of available transmission slots L is calculated as $L = c_L \lceil r^{\max}/r \rceil$, where $r^{\max} = 4.86$ is the average bpp for the PNG compressed images and c_L is a coefficient value [10].

The values of $k_d(\mathbf{x}, \hat{\mathbf{x}})$ in eq. (13) is set as $k_d(\mathbf{x}_n^i, \hat{\mathbf{x}}_n^i | r) = 0.0725^r$ based on the results of our curve fitting [10] in terms of normalized FID using High Fidelity Compression (HiFiC), in which images are taken from the COCO2017 [21], and we set $\Gamma = 1$. We argue these results are representative of the performance of the tiny image compressor in [7] based on the FID results for the nonquantized network, which is shown to be similar in performance to the original image compressor of HiFiC [6].

As benchmarks, we use the conventional compression method referred to as *baseline* and *TinyAirNet* [5]. For the baseline, each device first compresses its image with PNG and transmits it in predetermined slots *without collision*. For the TinyAirNet, each device only uses a behavior model and transmits PNG-compressed images. For brevity, we omit the analysis of both schemes.

Fig. 2 shows the SiFi against the compression rate r, where we set $V_{\text{th}} = 0.6$, $K = 5$, $c_L \in \{2, 5, 10\}$ and $N = 100$. From this figure, first, we can see that the results for EcoPull obtained by our approximate analysis using MCMC coincide with that of simulation results, which validates our approach using MCMC. In the figure, we can see the saw-tooth pattern because of the discrete nature of available transmission slots L for each r, *e.g.*, for $r \in [1.204, 1.608]$, SiFi slightly increases against r, in which L is a constant. Next, the figure also shows an optimal compression rate in terms of the SiFi. Let us denote the optimal value of r as r^{opt}. For $r^{\text{opt}} > r$, the transmission success probability becomes larger thanks to the lower compression rate, but the FID is high, leading to the poor SiFi. For $r^{\text{opt}} < r$, the FID of the retrieved image is low, but the transmission success probability of the relevant image becomes small because of the smaller number of available transmission slots. Further, the SiFi becomes larger as c_L increases. Each device with the relevant image can choose the transmission slot from the larger number of available slots L,

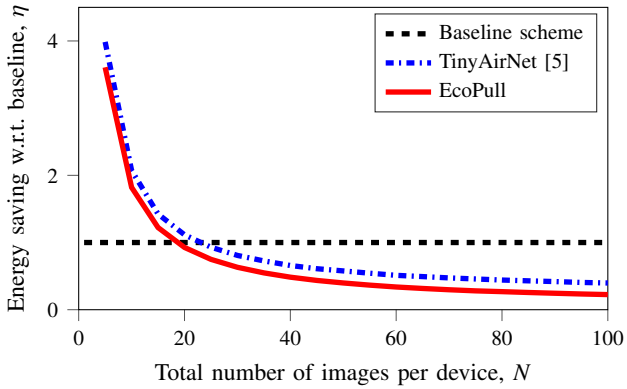

Fig. 3: Energy saving w.r.t. the baseline η as a function of the total number of images per device, N.

increasing the transmission success probability. Finally, from this result, we can see the importance of the choice of the compression rate in accordance with the available transmission slots to maximize the SiFi.

The total energy consumption and SiFi of EcoPull depends on the value of V_{th}, as well as r, as mentioned above. For example, the smaller (higher) V_{th} increases (decreases) the SiFi, but total energy consumption becomes larger (smaller). To realize a fair comparison among the different schemes, we investigate the minimum energy consumption under the constraint of a SiFi value γ_{th}. To this end, we first obtain the subsets of parameters $\{V_{\text{th}}, r\}$ for EcoPull, denoted as $\mathcal{P}(N)$ satisfying the constraint for given the value of N:

$$\mathcal{P}(N) = \min_{V_{\text{th}}, r} \mathbb{E}[E^i_{\text{tot}}](N) \quad \text{s.t. } \hat{\mathcal{V}} \geq \gamma_{\text{th}}. \tag{23}$$

We solved this numerically through a grid search, varying the value of V_{th} within a range of [0.50, 0.80] with a step of 0.01 and that of r within a range of [1, 2] with a step of 0.667.

Fig. 3 shows the energy saving w.r.t. the baseline η as a function of the number of images, N, where we set $K = 5$, $c_L = 5$, and $\gamma_{\text{th}} = 0.8$. Here, η is defined as the ratio between the total amount of energy consumed by the given scheme compared with the baseline scheme. Note that η becomes smaller than 1 only if the total energy consumption of the given method is smaller than the baseline scheme. The figure shows that η of EcoPull and TinyAirNet becomes smaller as N becomes larger. This is because as N increases, the gain from filtering out irrelevant images becomes larger than the additional energy cost brought by the TinyML model. Finally, we can see that EcoPull can realize a smaller η than no latent transmission scheme while satisfying the utility requirements because of the reduction of transmission data size for the relevant images. These results indicate that our proposed scheme can realize high energy efficiency while maintaining higher SiFi, especially when the number of images is high.

VI. Conclusions

In this paper, we have proposed EcoPull, an IoT data collection framework empowered by TinyML models, which aims to reduce the total energy consumed by IoT devices. We have introduced two types of TinyML models to IoT devices: the behavior model and the image compressor model. In our framework, each device first suppresses irrelevant data transmission using the behavior model and then transmits compressed images by applying an image compressor model. We have derived equations expressing total energy consumption and the SiFi, to evaluate the proposed framework. Numerical results have revealed that our proposed framework achieves high energy efficiency while maintaining high retrieved performance, especially when the number of images is large.

References

[1] E. C. Strinati *et al.*, "6G networks: Beyond shannon towards semantic and goal-oriented communications," *Comput. Netw.*, vol. 190, p. 107930, 2021.

[2] K. Sheth *et al.*, "A taxonomy of AI techniques for 6G communication networks," *Comput. Commun.*, vol. 161, pp. 279–303, 2020.

[3] E. C. Strinati *et al.*, "Goal-oriented and semantic communication in 6g ai-native networks: The 6g-goals approach," *arXiv preprint arXiv:2402.07573*, 2024.

[4] J. Lin *et al.*, "MCUNet: Tiny deep learning on IoT devices," *Advances Neural Inf. Proc. Syst.*, vol. 33, pp. 11 711–11 722, 2020.

[5] J. Shiraishi *et al.*, "TinyAirNet: TinyML model transmission for energy-efficient wireless IoT image retrieval," *arXiv preprint arXiv:2311.04788v2*, 2024.

[6] F. Mentzer *et al.*, "High-fidelity generative image compression," *Adv. Neural Inf. Process. Syst.*, vol. 33, pp. 11 913–11 924, 2020.

[7] N. Körber *et al.*, "Tiny generative image compression for bandwidth-constrained sensor applications," in *2021 20th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*. IEEE, 2021, pp. 564–569.

[8] K. Huang *et al.*, "Semantic data sourcing for 6G edge intelligence," *IEEE Commun. Mag.*, vol. 61, no. 12, pp. 70–76, 2023.

[9] A. Ramesh *et al.*, "Hierarchical text-conditional image generation with clip latents," *arXiv preprint arXiv:2204.06125*, vol. 1, no. 2, p. 3, 2022.

[10] M. Thorsager *et al.*, "Generative network layer for communication systems with artificial intelligence," *IEEE Netw. Lett.*, 2024.

[11] X. Huang *et al.*, "Federated learning-empowered AI-generated content in wireless networks," *IEEE Netw.*, 2024.

[12] J. Wen *et al.*, "From generative AI to generative internet of things: Fundamentals, framework, and outlooks," *arXiv preprint arXiv:2310.18382*, 2023.

[13] H. Du *et al.*, "Generative AI-aided joint training-free secure semantic communications via multi-modal prompts," in *ICASSP 2024-2024 IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*. IEEE, 2024, pp. 12 896–12 900.

[14] S. Barbarossa *et al.*, "Semantic communications based on adaptive generative models and information bottleneck," *IEEE Commun. Mag.*, vol. 61, no. 11, pp. 36–41, 2023.

[15] D. Bertsekas and R. Gallager, *Data Networks*, 2nd ed. Prentice Hall, 1996.

[16] B. Moons *et al.*, "Minimum energy quantized neural networks," in *Proc. 2017 51st Asilomar Conf. Signals, Syst., Comput.* IEEE, 2017, pp. 1921–1925.

[17] M. Heusel *et al.*, "GANs trained by a two time-scale update rule converge to a local Nash equilibrium," *Adv. neural inf. process. syst.*, vol. 30, 2017.

[18] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[19] F. Vázquez-Gallego *et al.*, "Modeling and analysis of reservation frame slotted-aloha in wireless machine-to-machine area networks for data collection," *Sensors*, vol. 15, no. 2, pp. 3911–3931, 2015.

[20] K. Xu *et al.*, "EtinyNet: extremely tiny network for TinyML," in *Proc. AAAI conf. artif. intell.*, vol. 36, no. 4, 2022, pp. 4628–4636.

[21] T.-Y. Lin *et al.*, "Microsoft COCO: Common objects in context," in *Comput. Vision–ECCV 2014: 13th Eur. Conf., Zurich, Switzerland, Sep. 6-12, 2014, Proc., Part V 13*. Springer, 2014, pp. 740–755.