Edge-Enhanced Streaming: Distributed Video Up-scaling in Constrained Environments

Ibrahim M. Amer[1], Sharief M. A. Oteafy[1,2], and Hossam S. Hassanein[1]
[1]School of Computing, Queen's University, Kingston, ON, Canada
[2]School of Computing, DePaul University, Chicago, IL, USA
ibrahim.amer@queensu.ca, soteafy@depaul.edu, hossam@cs.queensu.ca

GLOBECOM 2024 - 2024 IEEE Global Communications Conference | 979-8-3503-5125-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/GLOBECOM52923.2024.10901131

***Abstract*—In the face of growing demands for high-quality digital media, enhancing video streaming quality remains a significant challenge, particularly in environments with diverse internet connectivity and limited bandwidth. This paper proposes the Edge-enhanced Streaming (EES) scheme, which leverages edge computing and machine learning to upscale low-bitrate video frames to higher resolutions. Utilizing the distributed computational power of edge devices such as smartphones and laptops, our methodology involves segmenting each video frame into smaller sub-frames. These sub-frames are then processed using a Super-Resolution (SR) machine learning model across available edge devices within the network. This approach optimizes underutilized computational resources, improves processing times, and reduces energy consumption, making it a highly suitable approach for real-time video streaming applications. Furthermore, to address the challenge of device reliability, we incorporate a task replication strategy, ensuring consistent quality improvements even with potential fluctuations in device availability. We evaluate our proposed scheme using the PRIM dataset from the PIRM-SR Challenge. Extensive simulations demonstrate significant enhancements in video quality, confirming the effectiveness of our distributed SR technique in overcoming bandwidth constraints and improving user experience.**

***Index Terms*—Edge Computing, Resources Provisioning, Task Replication, Super-resolution, Reliability, Bandwidth Limitations, Quality of Service**

I. Introduction

As the Internet of Things (IoT) and digital media consumption continue to surge globally, the demand for high-quality video streaming has become increasingly pronounced. By 2025, the number of IoT devices is expected to exceed 75 billion [1], substantially increasing the demands on network and computing resources. This proliferation of devices and data-intensive applications, such as high-definition video streaming, presents significant challenges in maintaining quality of service (QoS) amid varying internet connectivity and bandwidth limitations [2].

Edge Computing (EC) has emerged as a solution to these challenges. By decentralizing data processing to devices closer to data sources, such as smartphones, tablets, and laptops. EC reduces dependency on distant data centers [3], potentially reducing latency and network congestion. This paradigm shift is particularly relevant in scenarios where traditional video streaming solutions falter, such as during periods of low bandwidth availability. This reduces the video bitrate, significantly degrading the viewing experience. [4].

The ever-increasing digital media traffic necessitates a paradigm shift in processing and delivering video content. EC enables content processing directly at the network's edge, closer to users, helping manage bandwidth more effectively while enhancing user experience by providing higher video quality [5].

Our research capitalizes on the capabilities of EC to enhance video streaming quality using machine learning-based Super-Resolution (SR) techniques [6]. Unlike traditional approaches that dynamically adjust video quality, our method aims to upscale the resolution of video frames under constrained bandwidth conditions by leveraging the computational power of edge devices within the network. This distributed approach not only improves the quality of streamed video but also democratizes access to high-quality content across diverse network environments.

A critical issue in deploying EC solutions is the reliability of the user-owned devices, which can vary dramatically in their availability and computational capacity. These devices are subject to the dynamic behavior of their users, who may engage them in other demanding applications, thus affecting their ability to consistently contribute resources for edge computing tasks. To maintain consistent service amidst variability, our approach incorporates a replication strategy. By duplicating tasks across multiple devices, we can mitigate the impact of any single device's failure or unavailability, thereby maintaining the overall reliability of the system.

Furthermore, our approach is designed to accommodate the dynamic nature of network conditions and device availability without compromising the end-user experience. By intelligently managing the distribution and replication of tasks, we can handle fluctuations in device performance and network conditions more effectively.

The main contributions of this paper are:

1) We leverage the underutilized resources, such as smartphones, laptops, and connected vehicles at the edge to improve video quality, focusing on enhancing the resolution of low-bitrate video frames using SR techniques.
2) We employ advanced machine learning models, particularly Generative Adversarial Networks (GANs), to effec-

tively upscale video quality in a distributed computing environment.
3) We introduce a segmented processing strategy, dividing video frames into smaller sub-frames that are enhanced independently on different edge devices, promoting efficient resource utilization and reducing the latency typically associated with centralized processing.
4) We implement task replication to enhance the reliability of our edge computing framework, ensuring robust video streaming even when individual devices are unavailable or underperforming.

In the following sections, we provide a comprehensive review of related work that sets the foundation for our research, followed by a detailed description of our proposed framework. The evaluation section offers an assessment of our approach, employing a variety of metrics to quantify improvements in video quality and user experience. Finally, we discuss the broader implications of our findings and conclude with reflections on the future of video streaming in the age of EC.

Our research highlights the potential of distributed SR techniques in enhancing video streaming quality in a scalable and efficient manner.

II. Related Work and Motivation

Enhancing video quality in bandwidth-constrained environments has been a subject of considerable research interest, leading to the exploration of various techniques and methodologies. This section reviews relevant literature in the fields of super-resolution, machine learning applications in video enhancement, and the utilization of EC for video processing.

The field of SR has significantly advanced with the advent of deep learning techniques, providing a foundation for enhancing low-resolution video frames.

Xintao Wang et al. [7] present an advancement in the field of single-image super-resolution (SR) with their Enhanced Super-Resolution Generative Adversarial Networks (ESR-GAN). Building upon the Super-Resolution Generative Adversarial Networks (SRGAN) model, they introduced several key innovations, including a new network architecture, improvements in adversarial loss, and optimizations in perceptual loss. Notably, they developed Residual-in-Residual Dense Blocks (RRDB), employed a relativistic GAN approach, and optimized the perceptual loss using features before activation. These modifications significantly enhance texture detail and visual realism while reducing artifacts. The effectiveness of ESR-GAN is evident in its performance, achieving first place in the PIRM2018-SR Challenge, marking a notable advancement in SR imaging techniques.

The work in [8] explores enhancing high-definition video streaming over mobile networks through a framework called LiveSR, which uses EC to improve streaming quality and reduce backhaul network load. Employing deep neural network-based SR at the edge, LiveSR significantly reduced backhaul traffic and increased the Quality of Experience (QoE) in real 5G tests. This approach demonstrates a scalable solution to manage bandwidth demands and alleviates network congestion.

Michelini et al. [9] present edge-SR (eSR), an SR framework optimized for edge devices like smartphones. eSR introduces lightweight one-layer architectures that balance image quality with computational efficiency, which are suitable for real-time applications on devices with limited processing power. This method narrows the gap between traditional upscaling techniques and advanced deep learning-based SR approaches. Through extensive testing, eSR demonstrates effective speed-quality trade-offs, offering a scalable SR solution for EC. This work enhances real-time image upscaling on edge devices and sheds light on the mechanisms behind SR, contributing to future developments in edge-based image processing.

Existing studies have significantly advanced SR techniques using deep-learning and EC, yet there is a gap in addressing the distributed processing of these tasks across multiple edge devices, especially the underutilized resources. Current research does not fully address enhancing video quality by distributing the workload among various devices. This approach is promising for real-time, high-quality video enhancement in bandwidth-limited settings. Our work proposes a framework that uses the collective power of edge devices for a scalable improvement in video streaming quality without needing high-speed internet.

III. Edge-Enhanced Streaming (EES)

Our proposed Edge-enhanced Streaming (EES) scheme strategically allocates and replicates the workload of a single SR task across multiple edge devices. This approach involves processing a low-resolution frame to generate a high-resolution image, thereby enhancing video quality in environments with poor network conditions. By distributing the computational load, EES leverages the collective processing capabilities of edge devices, optimizing the SR process for real-time performance and minimizing the impact of bandwidth constraints on video streaming quality. This method not only facilitates a more efficient use of edge computational resources but also ensures that high-quality video content is accessible even in areas with limited internet connectivity to enhance the video quality under poor network conditions.

A. System Model and Overview

Consider a video v streamed in a poor network environment, which comprises x number of frames denoted $\mathcal{F} = \{f_1, \cdots, f_x\}$. Each frame $f_k \forall k \in \mathcal{F}$ will be split into multiple sub-frames of size s, denoted $\hat{f}$, with the splitting criteria depending on the availability and conditions of the edge workers within the area. Consider a set of M SR tasks denoted $\Gamma = \{\gamma_1, \cdots, \gamma_M\}$ and a set of N edge workers within the area denoted $\mathcal{W} = \{w_1, \cdots, w_N\}$. Each task $\gamma_j \in \Gamma$ is defined by: a data size γ_j^{data} in bits, a processing density $\gamma_j^{\text{density}}$ in CPU cycles/bit, that is, the number of CPU cycles required to process a single bit of task's data, and a specific computation

delay deadline $\gamma_j^{\text{deadline}}$ which is the maximum acceptable computation delay as specified by the task requester. Each task γ_j is assigned to an edge worker and replicated as needed to mitigate potential failures due to uncertain availability and limited computational capacity. Each worker w_i has a CPU capacity w_i^{CPU} in cycles/seconds, a CPU utilization $w_i^{\text{utilization}}$ indicating the amount of load handled by the worker's CPU, and a maximum number of tasks w_i^{tasks} that can be carried on to avoid overloading the worker.

Task requests are dispatched to a centralized controller c, which scans the vicinity for workers. The decision regarding task allocation and replication is made by the controller c.

Our goal is to enhance video streaming in bandwidth-limited environments by optimizing task distribution on edge devices. We replicate tasks across multiple devices to ensure consistent quality. This strategy maximizes resource utilization and adapts to network fluctuations.

B. System Architecture and Implementation Details

We employ Docker containerization technology [10] to simulate and leverage edge workers for executing SR tasks. Docker containers offer a lightweight method for deploying applications, encapsulating all necessary dependencies without relying on the host operating system or underlying hardware. This capability ensures Docker containers can operate across nearly any platform, making them an optimal choice for SR tasks, especially given the heterogeneity and limited processing capabilities of the workers. The orchestrator c receives low-resolution video frames $\mathcal{F}$, divides each frame f into multiple smaller sub-frames $\hat{f}$, and for each sub-frame $\hat{f}_{kl}$, initiates a Docker container on a worker w_i. Each container runs the SR task with all necessary dependencies. The results from all workers are aggregated, and the synthesized high-resolution image is broadcast back to the video playback device, thereby enhancing video quality under bandwidth constraints.

Fig. (1) depicts a sophisticated mechanism that allows the orchestrator to assess the availability and capabilities of the edge workers before distributing the workload. Docker containers, running on the workers, are each tasked with processing assigned sub-frames. This ensures efficient use of computational resources across the network and facilitates the rapid processing and reassembly of super-resolved sub-frames by the orchestrator into the final high-resolution video frame. This approach minimizes latency, optimizes energy consumption, and maximizes streaming quality. Task replication enhances worker reliability by distributing multiple SR tasks across different workers, thereby reducing the impact of individual worker failures. The number of replicas for each SR task is determined based on the number of available workers, the required cycles for task γ_j, and the worker's CPU utilization over a specified time period.

First, we calculate the number of cycles required by task γ_j using Eq. (1),

$$\gamma_j^{\text{cycles}} = \gamma_j^{\text{density}} \gamma_j^{\text{data}} \; \forall j \in \Gamma \tag{1}$$

Then, we determine the available CPU cycles for worker w_i using Eq. (2),

$$w_i^{\text{cycles}} = \left(\frac{100 - w_i^{\text{utilization}}}{100} \right) w_i^{\text{CPU}} w_i^{\text{cores}} t \tag{2}$$

where w_i^{cores} denotes the number of CPU cores of w_i, and t represents the time period, in seconds, over which we estimate the free cycles of worker w_i.

Finally, the number of replicas for each task is determined using Eq. (3).

$$\gamma_j^{\text{replicas}} = \gamma_j^{\text{cycles}} \gamma_j^{\text{data}} \tag{3}$$

where the task cycles γ_j^{cycles} are multiplied by the task data γ_j^{data} to adjust the number of replicas according to the size of the task data.

We use the RTAR-H scheme [11], which is based on a bipartite graph game-theoretic matching algorithm, to allocate workers to SR tasks. The steps we use to allocate and replicate the set of tasks Γ to the set of workers $\mathcal{W}$ are illustrated in Algorithm 1.

Algorithm 1 EES at the Orchestrator

Input:
N: number of workers
f: low-resolution frame to be processed
v^{width}: sub-frame window width
v^{height}: sub-frame window height
$\mathcal{W}^{\text{CPU}}$: a vector of workers' CPU capacities
$\mathcal{W}^{\text{cores}}$: a vector of workers' number of cores
$\mathcal{W}^{\text{utilization}}$: a vector of workers' CPU utilization over time t
$\mathcal{W}^{\text{tasks}}$: a vector representing the maximum number of tasks each worker can carry on
Output:
$\overline{f}$: the super-resolved frame after processing

1: **function** EES(N,Γ^{CPU},$\mathcal{W}^{\text{utilization}}$, $\mathcal{W}^{\text{tasks}}$)
2: $\hat{f} \leftarrow$ SPLIT_INTO_SUB_FRAMES($f, v^{\text{width}}, v^{\text{height}}$)
3: ($\Gamma^{\text{density}}, \Gamma^{\text{data}}$) $\leftarrow$ GET_TASKS_SPECS($\hat{f}$)
4: $\Gamma^{\text{cycles}} \leftarrow$ CALCULATE_TASKS_CYCLES($\Gamma^{\text{density}}, \Gamma^{\text{data}}$) // from Eq. (1)
5: $\Gamma^{\text{replicas}} \leftarrow$ GET_TASK_REPLICAS_COUNT($\Gamma^{\text{cycles}}, \Gamma^{\text{data}}$) // from Eq. (3)
6: $\mathcal{W}^{\text{cycles}} \leftarrow$ CALCULATE_WORKERS_AVAIL_CYCLES($\mathcal{W}^{\text{utilization}}$,
7: $\mathcal{W}^{\text{CPU}}, \mathcal{W}^{\text{cores}}, t$) // from Eq. (2)
8: $\mathcal{W}^{\text{rep}} \leftarrow \{100\}$ // repeated for all elements
9: $X \leftarrow$ RTAR_HEURISTIC($\mathcal{W}^{\text{rep}}, \Gamma^{\text{replicas}}, \mathcal{W}^{\text{cycles}}, \mathcal{W}^{\text{tasks}}$)
10: $\tilde{f} \leftarrow$ SEND_TASKS_TO_WORKERS($X, \Gamma^{\text{data}}, \mathcal{W}$)
11: $\overline{f} \leftarrow$ COMBINE_SUB_FRAMES($\tilde{f}$)
12: **return** $\overline{f}$
13: **end function**

In Algorithm 1, we first split the input frame f, as described in line (2), into multiple sub-frames. The size of each sub-frame is determined by the input window's width v^{width} and height v^{height}. The number of generated sub-frames depends on the window dimensions and the size of the frame. In line (3), we calculate the task specifications using the pixels information. In line (4), the CPU cycles required for the tasks are calculated according to Eq. (1). The estimated number of workers for the tasks, denoted as Γ^{replicas}, is calculated in line (5) based on Eq. (3). The available CPU cycles of the workers are calculated using Eq. (2) in line (7). The RTAR-H algorithm [11] recruits workers based on the reputations of the workers;

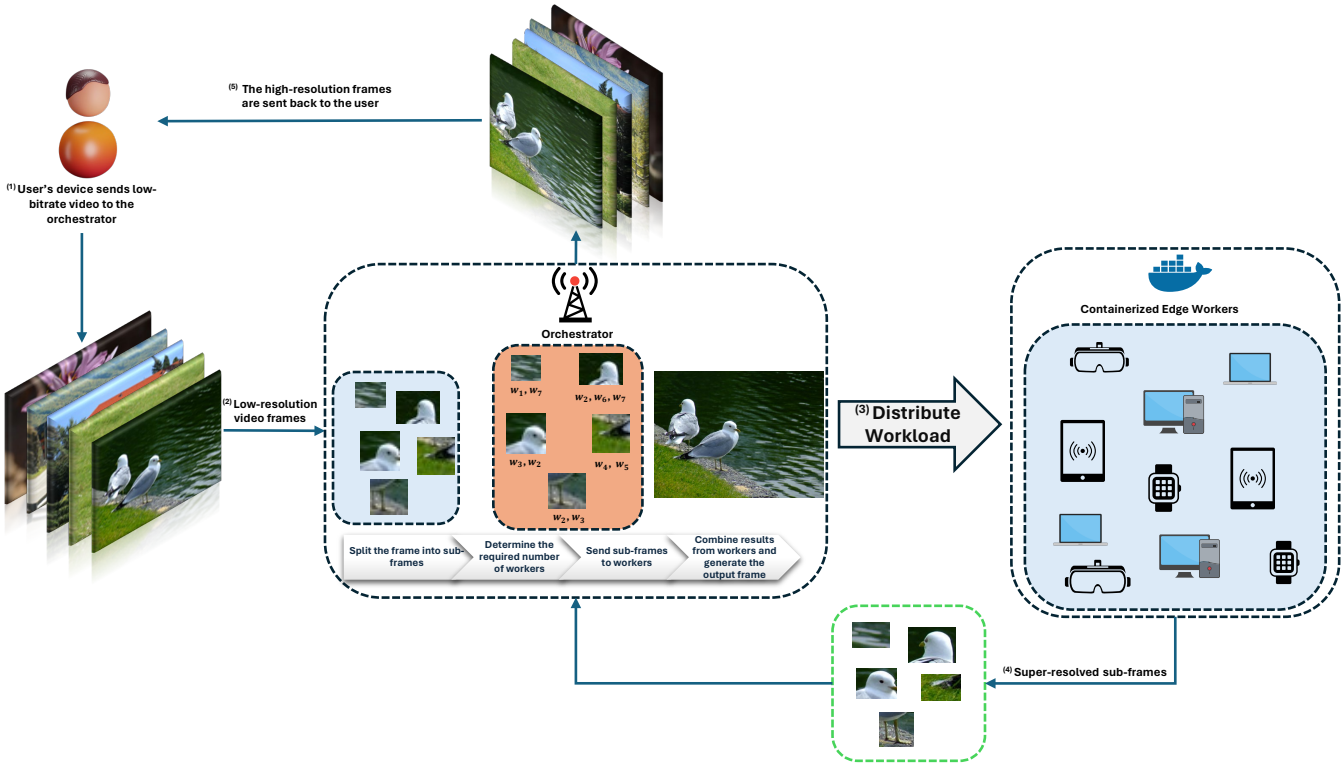


Figure 1: A system overview of the proposed scheme.

however, since our scheme does not focus on workers' reputations we set the reputations of all workers to the maximum value in line (8). Line (9) finds the optimal task assignments to workers; we use Γ^{replicas} instead of task budget in the original scheme [11], and $\mathcal{W}^{\text{cycles}}$ instead of workers costs. In line (10), the allocation of sub-frames to workers, along with the task data, is sent to the workers for processing; subsequently, the function returns the super-resolved sub-frames $\tilde{f}$. In line (11), the sub-frames are recombined into the output image $\overline{f}$ in the same order they were originally sent to the workers because there is no guarantee that the sub-frames will be returned in the same order as they appear in the original frame f. Due to replication, the result of a sub-frame might be returned multiple times to the orchestrator; in such cases, we take the first returned result and disregard the others. For each SR task, we use the ESRGAN scheme [7] that won first place in region three in the PIRM2018-Challenge [12].

IV. Performance Evaluation

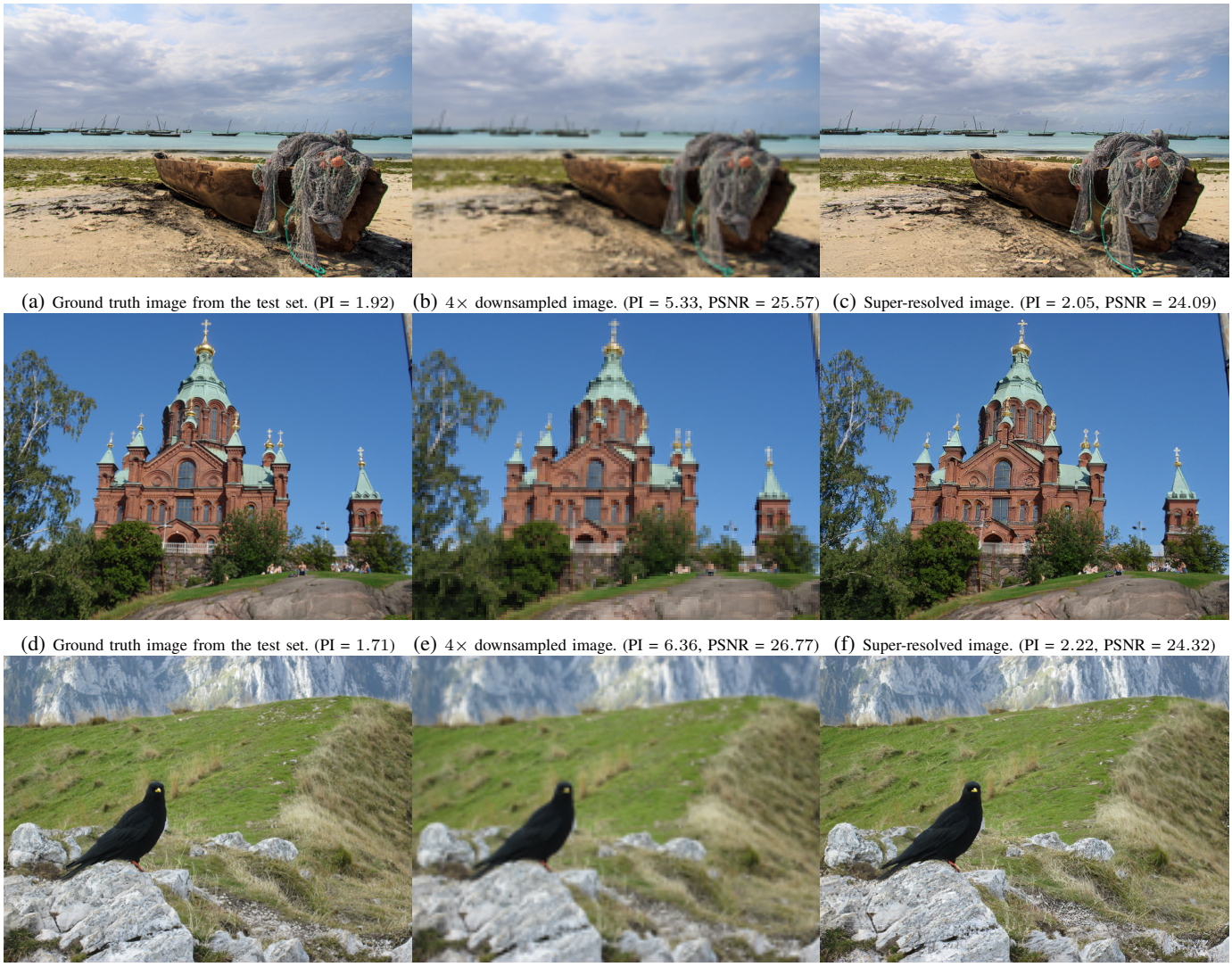
In this section, we evaluate the performance of the EES scheme across various environments and settings. Initially, we assess the quality of the super-resolved frames to ensure that the frame-splitting process does not adversely affect their quality. This assessment is conducted both visually and quantitatively using two specific performance metrics. Furthermore, we evaluate the efficiency gains from distributing the workload across edge workers instead of performing the SR tasks solely on the video playback device. To this end, we employ the following performance metrics: 1) **Peak Signal-to-Noise Ratio (PSNR) [13]**: Due to the lack of a standard and effective metric for perceptual quality, we assess the quality of super-resolved images by calculating the PSNR value between the super-resolved image and its ground truth. 2) **Perceptual Index (PI)**: Introduced by the PIRM challenge [12], this metric assesses the quality of images where no reference (ground truth) exists. 3) **Average Processing Time (APT)**: Measures the mean duration taken to complete the super-resolution tasks, from task initiation on the edge devices to the aggregation of results.

A. Dataset

We evaluate the proposed EES scheme on the prominent PIRM-SR dataset [12], using the test set. This dataset established the first benchmark for SR algorithms focused on perceptual quality. The innovative evaluation methodology introduced in [12] allowed for the evaluation and ranking of perceptual SR techniques together with those aimed at PSNR maximization. We assess the PSNR and PI scores for 300 low-resolution images using the EES scheme.

B. Simulation Setup

We implement the EES scheme using Python and containerization using Docker [10]. Each container is equipped with a pre-configured light-weight web app using Fast API [14].



(a) Ground truth image from the test set. (PI = 1.92) (b) 4× downsampled image. (PI = 5.33, PSNR = 25.57) (c) Super-resolved image. (PI = 2.05, PSNR = 24.09)

(d) Ground truth image from the test set. (PI = 1.71) (e) 4× downsampled image. (PI = 6.36, PSNR = 26.77) (f) Super-resolved image. (PI = 2.22, PSNR = 24.32)

(g) Ground truth image from the test set. (PI = 1.94) (h) 4× downsampled image. (PI = 2.76, PSNR = 27.14) (i) Super-resolved image. (PI = 2.08, PSNR = 24.44)

Figure 2: Qualitative results of SR tasks of three different images from the PRIM dataset [12]. The resulting images can be evaluated visually by comparing it with the ground truth image. The PI an PSNR scores, used in PIRM-SR challenge [12], are used to assess the quality of the super-resolved images. The results show more natural textures, e.g., bird's feather, building structure and grass texture.

The maximum number of workers tested is 100. The workload intensity of tasks $\gamma_j^{\text{density}}$ is in the range of $[1 - 5]\times 10^2$ cycle/bit.

C. Simulation Results and Analysis

Fig. 2 presents the qualitative outcomes of the SR tasks applied to test set images from the PRIM dataset. The analysis compares the original ground truth images with their downsampled and subsequently super-resolved counterparts.

Figs. 2a, 2d, and 2g depict the ground truth images, providing a benchmark for visual quality using PI. Figs. 2b, 2e, and 2h illustrate the 4× downsampled images, which show significant degradation in detail and clarity due to lower resolution. Finally, Figs. 2c, 2f, and 2i showcase the super-resolved images, where enhanced clarity and restored details are evident. The super-resolved images closely approximate the ground truth images' visual quality, demonstrating our scheme's effectiveness. This is quantitatively supported by the improved PI and PSNR scores observed in the super-resolved images compared to the downsampled versions. For instance, the PI improves significantly in the super-resolved images, indicating a closer match to the perceptual quality of the ground truth images. This substantial enhancement in both PI and PSNR scores validates our approach, highlighting its potential to effectively upscale video quality in bandwidth-

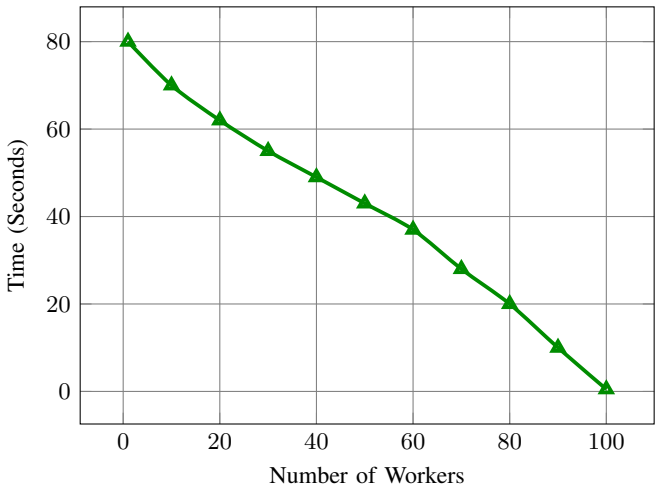


Figure 3: Average Processing Time (APT)

constrained environments, even when the image is split into sub-frames.

Fig. 3 illustrates the relationship between the number of edge workers and the Average Processing Time (APT) for SR tasks, demonstrating a downward trend in processing times as more workers are added. This indicates that the EES system efficiently utilizes parallel computation capabilities of distributed edge workers, thus reducing total processing time and showcasing the system's scalability. The graph suggests that with an increase in edge workers, the system can handle larger workloads more efficiently by distributing tasks among available resources. These results are from an experiment where we used a large image of 6776 in width and 5656 in height.

V. Conclusion

This paper introduced the Edge-enhanced Streaming (EES) scheme, leveraging edge computing and advanced machine learning to enhance video streaming in bandwidth-constrained environments. Our experiments, utilizing the PRIM dataset, demonstrated significant improvements in video quality, confirming the effectiveness of distributed Super-Resolution (SR) techniques. The approach notably benefits from a segmented processing strategy and task replication, which enhances reliability and service consistency. Future work will focus on improving scalability, integrating newer machine learning models to further optimize video upscaling, developing more sophisticated techniques for accurately calculating the number of required replicas based on various parameters, and incorporating more complex scenarios to better evaluate the robustness and applicability of the proposed scheme in diverse settings.

Acknowledgment

This research is supported by a grant from the Natural Sciences and Engineering Research Council of Canada (NSERC) under grant number ALLRP 549919-20, and a grant from Distributive, Ltd.

References

[1] P. K. Danso, S. Dadkhah, E. C. Pinto Neto, A. Zohourian, H. Molyneaux, R. Lu, and A. A. Ghorbani, "Transferability of Machine Learning Algorithm for IoT Device Profiling and Identification," *IEEE Internet of Things Journal*, vol. 11, no. 2, pp. 2322–2335, 1 2024.

[2] X. K. Zou, J. Erman, V. Gopalakrishnan, E. Halepovic, R. Jana, X. Jin, J. Rexford, and R. K. Sinha, "Can accurate predictions improve video streaming in cellular networks?" *HotMobile - 16th International Workshop on Mobile Computing Systems and Applications*, pp. 57–62, 2 2015.

[3] R. Singh and S. S. Gill, "Edge AI: A survey," *Internet of Things and Cyber-Physical Systems*, vol. 3, pp. 71–92, 1 2023.

[4] J. Kua, G. Armitage, and P. Branch, "A Survey of Rate Adaptation Techniques for Dynamic Adaptive Streaming over HTTP," *IEEE Communications Surveys and Tutorials*, vol. 19, no. 3, pp. 1842–1866, 7 2017.

[5] Q. He, S. Tan, F. Chen, X. Xu, L. Qi, X. Hei, H. Jin, and Y. Yang, "EDIndex: Enabling Fast Data Queries in Edge Storage Systems," *SIGIR - Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 675–685, 7 2023.

[6] D. C. Lepcha, B. Goyal, A. Dogra, and V. Goyal, "Image super-resolution: A comprehensive review, recent trends, challenges and applications," *Information Fusion*, vol. 91, pp. 230–260, 3 2023.

[7] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. C. Loy, "ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11133 LNCS, pp. 63–79, 9 2018.

[8] J. D. M. Liborio Filho, M. De Souza Coelho, and C. A. Melo, "Super-resolution on Edge Computing for Improved Adaptive HTTP Live Streaming Delivery," *IEEE 10th International Conference on Cloud Networking, CloudNet*, pp. 104–110, 2021.

[9] P. N. Michelini, Y. Lu, and X. Jiang, "Edge-SR: Super-Resolution for the Masses," *Proceedings - IEEE/CVF Winter Conference on Applications of Computer Vision, WACV*, pp. 4019–4028, 2022.

[10] "What is a Container? — Docker." [Online]. Available: https://www.docker.com/resources/what-container/.[Accessed:2024-04-26]

[11] I. M. Amer, S. Oteafy, S. A. Elsayed, and H. S. Hassanein, "Task Provisioning in Unreliable Edge Networks: Inferring Utility," in *IEEE Global Communications Conference*, 12 2023, p. 5.88.

[12] Y. Blau, R. Mechrez, R. Timofte, T. Michaeli, and L. Zelnik-Manor, "The 2018 PIRM Challenge on Perceptual Image Super-resolution," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11133 LNCS, pp. 334–355, 9 2018.

[13] J. Kim, J. K. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1646–1654, 12 2016.

[14] "tiangolo/fastapi: FastAPI framework, high performance, easy to learn, fast to code, ready for production." [Online]. Available: https://github.com/tiangolo/fastapi.[Accessed:2024-04-26]